Linear Algebra: Foundations to Frontiers

Notes to LAFF With

MATLAB Version

Margaret E. Myers
Pierce M. van de Geijn
Robert A. van de Geijn

Release Date December 21, 2021

This is a work in progress

10 9 8 7 6 5 4 3 2 1

Library of Congress Cataloging-in-Publication Data not yet available
Draft Edition,
This "Draft Edition" allows this material to be used while we sort out through what mechanism we will publish the book.

Contents

Preface

Linear Algebra: Foundations to Frontiers (LAFF) is an experiment in a number of different dimensions.

- It is a resource that integrates a text, a large number of videos (more than 270 by last count), and hands-on activities.
- It connects hand calculations, mathematical abstractions, and computer programming.
- In encourages you to develop the mathematical theory of linear algebra by posing questions rather than outright stating theorems and their proofs.
- It introduces you to the frontier of linear algebra software development.

Our hope is that this will enable you to master all the standard topics that are taught in a typical introductory undergraduate linear algebra course.

Who should LAFF? From our experience offering LAFF as a Massive Open Online Course (MOOC) on ☛ edX, it has become clear that there are a number of audiences for LAFF.

The Independent Beginner. There were MOOC participants for whom LAFF was their first introduction to linear algebra beyond the matrix manipulation that is taught in high school. These were individuals who possess a rare talent for self-learning that is unusual at an early stage in one's schooling. For them, LAFF was a wonderful playground. Others like them may similarly benefit from these materials.

The Guide. What we also hope to deliver with LAFF is a resource for someone who is an effective facilitator of learning (what some would call an instructor) to be used, for example, in a small to medium size classroom setting. While this individual may or may not have our level of expertise in the domain of linear algebra, what is important is that she/he knows how to guide and inspire.

The Refresher. At some point, a student or practitioner of mathematics (in engineering, the physical sciences, the social sciences, business, and many other subjects) realizes that linear algebra is as fundamental as is calculus. This often happens after the individual has already completed the introductory course on the subject and now he/she realizes it is time for a refresher. From our experience with the MOOC, LAFF seems to delight this category of learner. We sequence the material differently from how a typical course on "matrix computations" presents the subject. We focus on fundamental topics that have practical importance and on raising the participant's ability to think more abstractly. We link the material to how one should translate theory into algorithms and implementations. This seemed to appeal even to MOOC participants who had already taken multiple linear algebra courses and/or already had advanced degrees.

The Graduate Student. This is a subcategory of The Refresher. The material that is incorporated in LAFF are meant in part to provide the foundation for a more advanced study of linear algebra. The feedback from those MOOC participants who had already taken linear algebra suggests that LAFF is a good choice for those who want to prepare for a more advanced course. Robert expects the students who take his graduate course in Numerical Linear Algebra to have the material covered by LAFF as a background, but not more. A graduate student may also want to study these undergraduate materials hand-in-hand with Robert's notes for Linear Algebra: Foundations to Frontiers - Notes on Numerical Linear algebra, also available from ☛ `http://www.ulaff.net`.

If you are still trying to decide whether LAFF is for you, you may want to read some of the ☛ Reviews of LAFF (The MOOC) on CourseTalk.

A typical college or university offers at least three undergraduate linear algebra courses: Introduction to Linear Algebra; Linear Algebra and Its Applications; and Numerical Linear Algebra. LAFF aims to be that first course. After mastering this fundamental knowledge, you will be ready for the other courses, or a graduate course on numerical linear algebra.

Acknowledgments

LAFF was first introduced as a Massive Open Online Course (MOOC) offered by edX, a non-profit founded by Harvard University and the Massachusetts Institute of Technology. It was funded by the University of Texas System, an edX partner, and sponsored by a number of entities of The University of Texas at Austin (UT-Austin): the Department of Computer Science (UTCS); the Division of Statistics and Scientific Computation (SSC); the Institute for Computational Engineering and Sciences (ICES); the Texas Advanced Computing Center (TACC); the College of Natural Sciences; and the Office of the Provost. It was also partially sponsored by the National Science Foundation Award ACI-1148125 titled "SI2-SSI: A Linear Algebra Software Infrastructure for Sustained Innovation in Computational Chemistry and other Sciences"[1], which also supports our research on how to develop linear algebra software libraries. The course was and is designed and developed by Dr. Maggie Myers and Prof. Robert van de Geijn based on an introductory undergraduate course, Practical Linear Algebra, offered at UT-Austin.

The Team

Dr. Maggie Myers is a lecturer for the Department of Computer Science and Division of Statistics and Scientific Computing. She currently teaches undergraduate and graduate courses in Bayesian Statistics. Her research activities range from informal learning opportunities in mathematics education to formal derivation of linear algebra algorithms. Earlier in her career she was a senior research scientist with the Charles A. Dana Center and consultant to the Southwest Educational Development Lab (SEDL). Her partnerships (in marriage and research) with Robert have lasted for decades and seems to have survived the development of LAFF.

Dr. Robert van de Geijn is a professor of Computer Science and a member of the Institute for Computational Engineering and Sciences. Prof. van de Geijn is a leading expert in the areas of high-performance computing, linear algebra libraries, parallel processing, and formal derivation of algorithms. He is the recipient of the 2007-2008 President's Associates Teaching Excellence Award from The University of Texas at Austin.

Pierce van de Geijn is one of Robert and Maggie's three sons. He took a year off from college to help launch the course, as a full-time volunteer. His motto: "If I weren't a slave, I wouldn't even get room and board!"

David R. Rosa Tamsen is an undergraduate research assistant to the project. He developed the `laff` application that launches the IPython Notebook server. His technical skills allows this application to execute on a wide range of operating systems. His gentle bed side manners helped MOOC participants overcome complications as they became familiar with the software.

Josh Blair was our social media director, posting regular updates on Twitter and Facebook. He regularly reviewed progress, alerting us of missing pieces. (This was a daunting task, given that we were often still writing material as the hour of release approached.) He also tracked down a miriad of linking errors in this document.

Dr. Erin Byrne and **Dr. Grace Kennedy** from MathWorks provided invaluable support for MATLAB. ☛ MathWorks graciously provided free licenses for the partipants during the offering of the course on the edX platform.

Dr. Tze Meng Low developed PictureFLAME and the online Gaussian elimination exercises. He is now a Systems Scientist at Carnegie Mellon University.

Dr. Ardavan Pedram created the animation of how data moves between memory layers during a high-performance matrix-matrix multiplication.

[1] *Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).*

Sean Cunningham created the LAFF “sizzler” (introductory video). Sean is the multimedia producer at the Texas Advanced Computing Center (TACC).

The music for the sizzler was composed by **Dr. Victor Eijkhout**. Victor is a Research Scientist at TACC. Check him out at ☛ MacJams.

Chase van de Geijn, Maggie and Robert’s youngest son, produced the “Tour of UTCS” video. He faced a real challenge: we were running out of time, Robert had laryngitis, and we still didn’t have the right equipment. Yet, as of this writing, the video already got more than 9.500 views on YouTube!

Sejal Shah of UT-Austin’s Center for Teaching Learning tirelessly helped resolved technical questions, as did **Emily Watson** and **Jennifer Akana** of edX.

Cayetana Garcia and **Julie Heiland** provided invaluable administrative assistants. When a piece of equipment had to be ordered, to be used “yesterday”, they managed to make it happen!

It is important to acknowledge the students in Robert’s SSC 329C classes of Fall 2013 and Spring 2014. They were willing guinnea pigs in this cruel experiment.

Additional support for the first offering (Spring 2014) of LAFF on edX

Jianyu Huang was both a teaching assistant for Robert’s class that ran concurrent with the MOOC and the MOOC itself. Once Jianyu took charge of the tracking of videos, transcripts, and other materials, our task was greatly lightened.

Woody Austin was a teaching assistant in Fall 2013 for Robert’s class that helped develop many of the materials that became part of the LAFF. He was also an assistant for the MOOC, contributing IPython Notebooks, proofing materials, and monitoring the discussion board.

Graduate students **Martin Schatz** and **Tyler Smith** helped monitor the discussion board.

In addition to David and Josh, three undergraduate assistants helped with the many tasks that faced us in Spring 2014: **Ben Holder**, who implemented “Timmy” based on an exercise by our colleague Prof. Alan Cline for the IPython Notebook in Week 3. **Farhan Daya**, **Adam Hopkins**, **Michael Lu**, and **Nabeel Viran**, who monitored the discussion boards and checked materials.

Thank you all!

Finally, we would like to thank the participants for their enthusiasm and valuable comments. Some enthusiastically forged ahead as soon as a new week was launched and gave us early feedback. Without them many glitches would have plagued all participants. Others posed uplifting messages on the discussion board that kept us going. Their patience with our occasional shortcomings were most appreciated!

Week 0

Getting Started

0.1 Opening Remarks

0.1.1 Welcome to LAFF

☛ **YouTube**

Wish You Were Here

☛ **YouTube**

Since most of you are not The University of Texas at Austin students, we thought we'd give you a tour of our new building: the Gates Dell Complex. Want to see more of The University of Texas at Austin? Take the ☛ Virtual Campus Tour.

0.1.2 Outline

Following the "opener" we give the outline for the week:

0.1.3 What You Will Learn

The third unit of the week informs you of what you will learn. This describes the knowledge and skills that you can expect to acquire. In addition, this provides an opportunity for you to self-assess upon completion of the week.

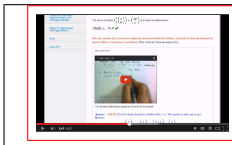
Upon completion of this week, you should be able to

- Navigate through LAFF on the edX platform.
- Keep track of your homework and progress through LAFF.
- Download and start MATLAB.
- Recognize the structure of a typical week.

You may want to print out Appendix B to track your progress throughout the course.

0.2 How to LAFF

0.2.1 How to Navigate LAFF



☛ YouTube

0.2.2 Setting Up to LAFF

It helps if we all set up our environment in a consistent fashion. Go to

☛ http://www.ulaff.net/downloads.html

and follow the instructions on how to download and unzip the file `LAFF-2.0xM.zip`. Once you unzip the file, you will find a directory `LAFF-2.0xM`, with subdirectories. I did this in my home directory, yielding the directory structure in Figure 1. **Notice that this document is now also in that directory!** You will want to start using it, opening it with ☛ Acrobat Reader.

You can optionally download the videos for the various weeks by following the instructions at

☛ http://www.ulaff.net/downloads.html.

0.3 Software to LAFF

0.3.1 Why MATLAB

We use MATLAB as a tool because it was invented to support learning about matrix computations. You will find that the syntax of the language used by MATLAB very closely resembles the mathematical expressions in linear algebra.

Those who do not have access to MATLAB and are not willing to invest in MATLAB will want to consider ☛ GNU Octave instead.

0.3.2 Installing MATLAB

Instructions can be found at ☛ http://www.mathworks.org.

0.3.3 MATLAB Basics

Below you find a few short videos that introduce you to MATLAB. For a more comprehensive tutorial, you may want to visit ☛ MATLAB Tutorials at MathWorks and clicking "Launch Tutorial".

HOWEVER, you need very little familiarity with MATLAB in order to learn what we want you to learn about how abstraction in mathematics is linked to abstraction in algorithms. So, you could just skip these tutorials altogether, and come back to them if you find you want to know more about MATLAB and its programming language (M-script).

What is MATLAB?

☛ YouTube

The MATLAB Environment

☛ YouTube

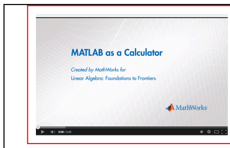
- Users
  - rvdg
    - LAFF-2.0xM
    - LAFF-2.0xM.pdf .... This document! The “Notes to LAFF With” in PDF format that allow you to access most course materials.
    - 1521Instructions.pdf .... Additional instructions for Homework 1.5.2.1.
      - Programming
        - laff .... Subdirectory in which a small library that we will use resides.
        - matvec
        - matmat
        - util
        - vecvec
      - Week01 .... Subdirectory for the coding assignments for Week 01.
        - laff_copy.m .... Implementation of a routine that copies one vector to another that you will write.
        - test_copy.m .... Script for testing your implementation of laff_copy.m.
      - ⋮
      - Week12 .... Subdirectory for the coding assignments for Week 12.
      - Videos
        - Videos-Week00.zip .... Place this file here before unzipping (if you choose to download videos).
        - Videos-Week00 .... Subdirectory created when you unzip Video-Week00.zip.
        - ⋮
        - Videos-Week12.zip .... Place this file here before unzipping (if you choose to download videos).
        - Videos-Week12 .... Subdirectory created when you unzip Video-Week12.zip.
      - Practice .... Various activities that allow you to practice.
      - PictureFLAME .... Web tool that allows you to visualize an implementation as it executes.
      - Spark .... Web tool with which you will translate algorithms into code using the FLAME@lab API.
      - Answers .... Answers to various programming assignments .

Figure 1: Directory structure for your LAFF materials. Items in red will be placed into the materials by you. In this example, we placed LAFF-2.0xM.zip in the home directory Users/rvdg before unzipping. You may want to place it on your account’s “Desktop” instead.

MATLAB Variables

☛ YouTube

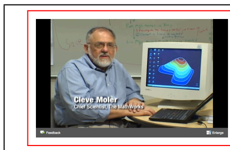
MATLAB as a Calculator



☛ YouTube

0.4 Enrichments

0.4.1 The Origins of MATLAB



☛ to MathWorks.com Video

0.5 Wrap Up

0.5.1 Additional Homework

For a typical week, additional assignments may be given in this unit.

0.5.2 Summary

You will see that we develop a lot of the theory behind the various topics in linear algebra via a sequence of homework exercises. At the end of each week, we summarize theorems and insights for easy reference.

Week 1

Vectors in Linear Algebra

1.1 Opening Remarks

1.1.1 Take Off

"Co-Pilot Roger Murdock (to Captain Clarence Oveur): We have clearance, Clarence.

Captain Oveur: Roger, Roger. What's our vector, Victor?"

From Airplane. Dir. David Zucker, Jim Abrahams, and Jerry Zucker. Perf. Robert Hays, Julie Hagerty, Leslie Nielsen, Robert Stack, Lloyd Bridges, Peter Graves, Kareem Abdul-Jabbar, and Lorna Patterson. Paramount Pictures, 1980. Film.

You can find a video clip by searching "What's our vector Victor?"

Vectors have direction and length. Vectors are commonly used in aviation where they are routinely provided by air traffic control to set the course of the plane, providing efficient paths that avoid weather and other aviation traffic as well as assist disoriented pilots.

Let's begin with vectors to set our course.

1.1.2 Outline Week 1

1.1.3 What You Will Learn

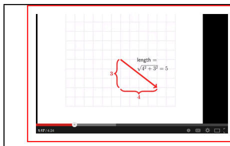
Upon completion of this week, you should be able to

- Represent quantities that have a magnitude and a direction as vectors.
- Read, write, and interpret vector notations.
- Visualize vectors in $\mathbb{R}^2$.
- Perform the vector operations of scaling, addition, dot (inner) product.
- Reason and develop arguments about properties of vectors and operations defined on them.
- Compute the (Euclidean) length of a vector.
- Express the length of a vector in terms of the dot product of that vector with itself.
- Evaluate a vector function.
- Solve simple problems that can be represented with vectors.
- Create code for various vector operations and determine their cost functions in terms of the size of the vectors.
- Gain an awareness of how linear algebra software evolved over time and how our programming assignments fit into this (enrichment).
- Become aware of overflow and underflow in computer arithmetic (enrichment).

Track your progress in Appendix B.

1.2 What is a Vector?

1.2.1 Notation



☛ YouTube

Definition

Definition 1.1 *We will call a one-dimensional array of n numbers a vector of size n:*

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}.$$

- This is an *ordered* array. The position in the array is important.
- We will call the ith number the ith *component* or *element*.
- We denote the ith component of x by χ_i. Here χ is the lower case Greek letter pronounced as "kı". (Learn more about our notational conventions in Section 1.7.1.)

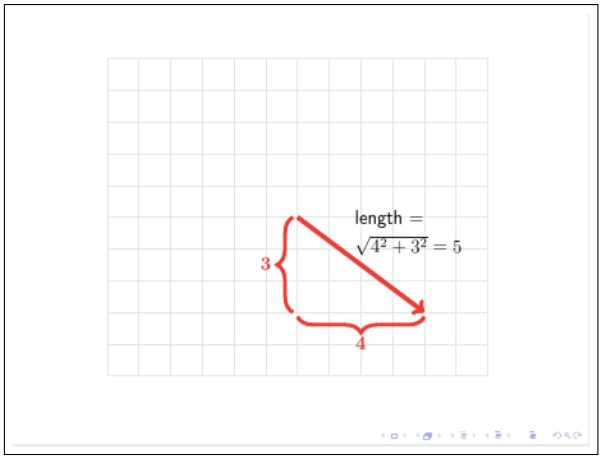
  As a rule, we will use lower case letters to name vectors (e.g., $x, y, \ldots$). The "corresponding" Greek lower case letters are used to name their components.
- We start indexing at 0, as computer scientists do. MATLAB, the tool we will be using to implement our libraries, naturally starts indexing at 1, as do most mathematicians and physical scientists. You'll have to get use to this...
- Each number is, at least for now, a real number, which in math notation is written as $\chi_i \in \mathbb{R}$ (read: "ki sub i (is) in r" or "ki sub i is an element of the set of all real numbers").
- The *size* of the vector is n, the number of components. (Sometimes, people use the words "length" and "size" interchangeably. We will see that length also has another meaning and will try to be consistent.)
- We will write $x \in \mathbb{R}^n$ (read: "x" in "r" "n") to denote that x is a vector of size n with components in the real numbers, denoted by the symbol: $\mathbb{R}$. Thus, $\mathbb{R}^n$ denotes the set of all vectors of size n with components in $\mathbb{R}$. (Later we will talk about vectors with components that are complex valued.)
- A vector has a direction and a length:
  - Its direction is often visualized by drawing an arrow from the origin to the point $(\chi_0, \chi_1, \ldots, \chi_{n-1})$, but the arrow does not necessarily need to start at the origin.
  - Its *length* is given by the Euclidean length of this arrow,

    $$\sqrt{\chi_0^2 + \chi_1^2 + \cdots + \chi_{n-1}^2},$$

    It is denoted by $\|x\|_2$ called the *two-norm*. Some people also call this the *magnitude* of the vector.
- A vector does *not* have a location. Sometimes we will show it starting at the origin, but that is only for convenience. It will often be more convenient to locate it elsewhere or to move it.

Examples

Example 1.2

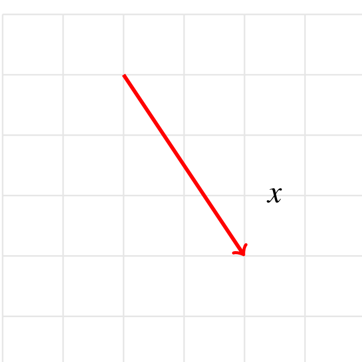


Consider $x = \begin{pmatrix} 4 \\ -3 \end{pmatrix}$. Then

- Components 4 and -3 are the first and second component, respectively.
- $\chi_0 = 4$, $\chi_1 = -3$ so that 4 is the component indexed with 0 and -3 the component indexed with 1.
- The vector is of size 2, so $x \in \mathbb{R}^2$.

Exercises

Homework 1.2.1.1 Consider the following picture:



Using the grid for units,

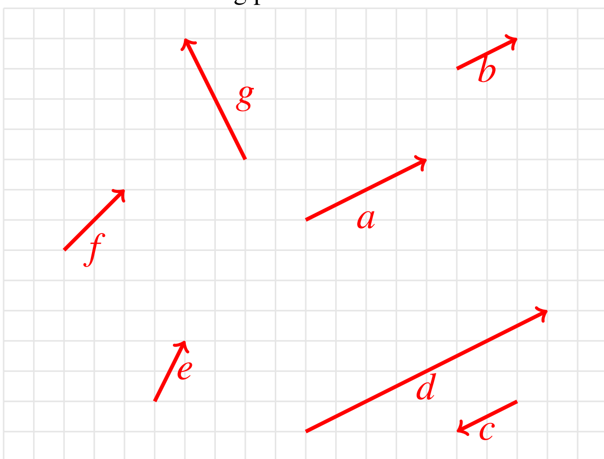
(a) $x = \begin{pmatrix} -2 \\ -3 \end{pmatrix}$ (b) $x = \begin{pmatrix} 3 \\ -2 \end{pmatrix}$

(c) $x = \begin{pmatrix} 2 \\ -3 \end{pmatrix}$ (d) $x = \begin{pmatrix} -3 \\ -2 \end{pmatrix}$

(e) None of these

☛ SEE ANSWER

Homework 1.2.1.2
Consider the following picture:



Using the grid for units,

(a) $a = \begin{pmatrix} \\ \\ \end{pmatrix}$

(b) $b = \begin{pmatrix} \\ \\ \end{pmatrix}$

(c) $c = \begin{pmatrix} \\ \\ \end{pmatrix}$

(d) $d = \begin{pmatrix} \\ \\ \end{pmatrix}$

(e) $e = \begin{pmatrix} \\ \\ \end{pmatrix}$

(f) $f = \begin{pmatrix} \\ \\ \end{pmatrix}$

(g) $g = \begin{pmatrix} \\ \\ \end{pmatrix}$

☛ SEE ANSWER

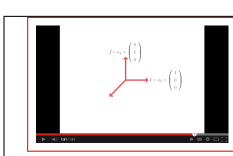
While a vector does not have a location, but has direction and length, vectors are often used to show the direction and length of movement from one location to another. For example, the vector from point $(1,-2)$ to point $(5,1)$ is the vector $\begin{pmatrix} 4 \\ 3 \end{pmatrix}$. We might geometrically represent the vector $\begin{pmatrix} 4 \\ 3 \end{pmatrix}$ by an arrow from point $(1,-2)$ to point $(5,1)$.

Homework 1.2.1.3 Write each of the following as a vector:

- The vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(-1,2)$ to point $(0,0)$.
- The vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(0,0)$ to point $(-1,2)$.
- The vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(-1,2,4)$ to point $(0,0,1)$.
- The vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(1,0,0)$ to point $(4,2,-1)$.

☛ SEE ANSWER

1.2.2 Unit Basis Vectors



☛ **YouTube**

Definition

Definition 1.3 *An important set of vectors is the set of unit basis vectors given by*

$$e_j = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \begin{array}{l} \left.\vphantom{\begin{array}{c}0\\ \vdots\\ 0\end{array}}\right\} j \textit{ zeroes} \\ \longleftarrow \textit{ component indexed by } j \\ \left.\vphantom{\begin{array}{c}0\\ \vdots\\ 0\end{array}}\right\} n-j-1 \textit{ zeroes} \end{array}$$

where the "1" appears as the component indexed by j. Thus, we get the set $\{e_0, e_1, \ldots, e_{n-1}\} \subset \mathbb{R}^n$ given by

$$e_0 = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \quad e_1 = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \quad \cdots, \quad e_{n-1} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

In our presentations, any time you encounter the symbol e_j, it *always* refers to the unit basis vector with the "1" in the component indexed by j.

These vectors are also referred to as the **standard basis vectors**. Other terms used for these vectors are natural basis and canonical basis. Indeed, "unit basis vector" appears to be less commonly used. But we will use it anyway!

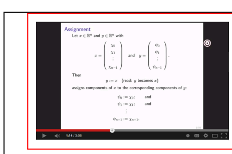
Homework 1.2.2.1 Which of the following is not a unit basis vector?

(a) $\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ (b) $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ (c) $\begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}$ (d) $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ (e) None of these are unit basis vectors.

☛ SEE ANSWER

1.3 Simple Vector Operations

1.3.1 Equality (=), Assignment (:=), and Copy



☛ YouTube

Definition

Definition 1.4 *Two vectors $x, y \in \mathbb{R}^n$ are equal if all their components are element-wise equal:*

$$x = y \textit{ if and only if } \chi_i = \psi_i, \textit{ for all } 0 \le i < n.$$

This means that two vectors are equal if they point in the same direction and are of the same length. They don't, however, need to have the same location.

The *assignment* or *copy* operation assigns the content of one vector to another vector. In our mathematical notation, we will denote this by the symbol := (pronounce: *becomes*). After the assignment, the two vectors are equal to each other.

Algorithm

The following algorithm copies vector $x \in \mathbb{R}^n$ into vector $y \in \mathbb{R}^n$, performing the operation $y := x$:

$$\begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} := \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}$$

for $i = 0, \ldots, n-1$
  $\psi_i := \chi_i$
endfor

Cost

(Notice: we will cost of various operations in more detail in the future.)

Copying one vector to another vector requires $2n$ memory operations (memops).

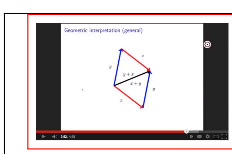
- The vector x of length n must be read, requiring n memops and
- the vector y must be written, which accounts for the other n memops.

Homework 1.3.1.1 Decide if the two vectors are equal.

- The vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(-1,2)$ to point $(0,0)$ and the vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(1,-2)$ to point $(2,-1)$ are equal.
True/False
- The vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(1,-1,2)$ to point $(0,0,0)$ and the vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(1,1,-2)$ to point $(0,2,-4)$ are equal.
True/False

☛ SEE ANSWER

1.3.2 Vector Addition (ADD)



☛ YouTube

Definition

Definition 1.5 *Vector addition $x+y$ (sum of vectors) is defined by*

$$x+y=\begin{pmatrix}\chi_0\\\chi_1\\\vdots\\\chi_{n-1}\end{pmatrix}+\begin{pmatrix}\psi_0\\\psi_1\\\vdots\\\psi_{n-1}\end{pmatrix}=\begin{pmatrix}\chi_0+\psi_0\\\chi_1+\psi_1\\\vdots\\\chi_{n-1}+\psi_{n-1}\end{pmatrix}.$$

In other words, the vectors are added element-wise, yielding a new vector of the same size.

Exercises

Homework 1.3.2.1 $\begin{pmatrix}-1\\2\end{pmatrix}+\begin{pmatrix}-3\\-2\end{pmatrix}=$

☛ SEE ANSWER

Homework 1.3.2.2 $\begin{pmatrix}-3\\-2\end{pmatrix}+\begin{pmatrix}-1\\2\end{pmatrix}=$

☛ SEE ANSWER

Homework 1.3.2.3 For $x,y\in\mathbb{R}^n$,

$$x+y=y+x.$$

Always/Sometimes/Never

☛ SEE ANSWER

Homework 1.3.2.4 $\begin{pmatrix}-1\\2\end{pmatrix}+\left(\begin{pmatrix}-3\\-2\end{pmatrix}+\begin{pmatrix}1\\2\end{pmatrix}\right)=$

☛ SEE ANSWER

Homework 1.3.2.5 $\left(\begin{pmatrix}-1\\2\end{pmatrix}+\begin{pmatrix}-3\\-2\end{pmatrix}\right)+\begin{pmatrix}1\\2\end{pmatrix}=$

☛ SEE ANSWER

Homework 1.3.2.6 For $x,y,z\in\mathbb{R}^n$, $(x+y)+z=x+(y+z)$. Always/Sometimes/Never

☛ SEE ANSWER

Homework 1.3.2.7 $\begin{pmatrix}-1\\2\end{pmatrix}+\begin{pmatrix}0\\0\end{pmatrix}=$

☛ SEE ANSWER

Homework 1.3.2.8 For $x\in\mathbb{R}^n$, $x+0=x$, where 0 is the zero vector of appropriate size.

Always/Sometimes/Never

☛ SEE ANSWER

Algorithm

The following algorithm assigns the sum of vectors x and y (of size n and stored in arrays x and y) to vector z (of size n and stored in array z), computing $z := x + y$:

$$\begin{pmatrix} \zeta_0 \\ \zeta_1 \\ \vdots \\ \zeta_{n-1} \end{pmatrix} := \begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \\ \vdots \\ \chi_{n-1} + \psi_{n-1} \end{pmatrix}.$$

$$\begin{array}{l} \textbf{for } i = 0, \ldots, n-1 \\ \quad \zeta_i := \chi_i + \psi_i \\ \textbf{endfor} \end{array}$$

Cost

On a computer, real numbers are stored as floating point numbers, and real arithmetic is approximated with floating point arithmetic. Thus, we count floating point operations (flops): a multiplication or addition each cost one flop.

Vector addition requires $3n$ memops (x is read, y is read, and the resulting vector is written) and n flops (floating point additions).

For those who understand "Big-O" notation, the cost of the SCAL operation, which is seen in the next section, is $O(n)$. However, we tend to want to be more exact than just saying $O(n)$. To us, the coefficient in front of n is important.

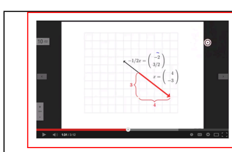
Vector addition in sports

View the following video and find out how the "parallelogram method" for vector addition is useful in sports:

https://www.nsf.gov/news/mmg/mmg_disp.jsp?med_id=69233

Discussion: Can you find other examples of how vector addition is used in sports?

1.3.3 Scaling (SCAL)



☛ YouTube

Definition

Definition 1.6 *Multiplying vector x by scalar α yields a new vector, αx, in the same direction as x, but scaled by a factor α. Scaling a vector by α means each of its components, χ_i, is multiplied by α:*

$$\alpha x = \alpha \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \\ \vdots \\ \alpha\chi_{n-1} \end{pmatrix}.$$

Exercises

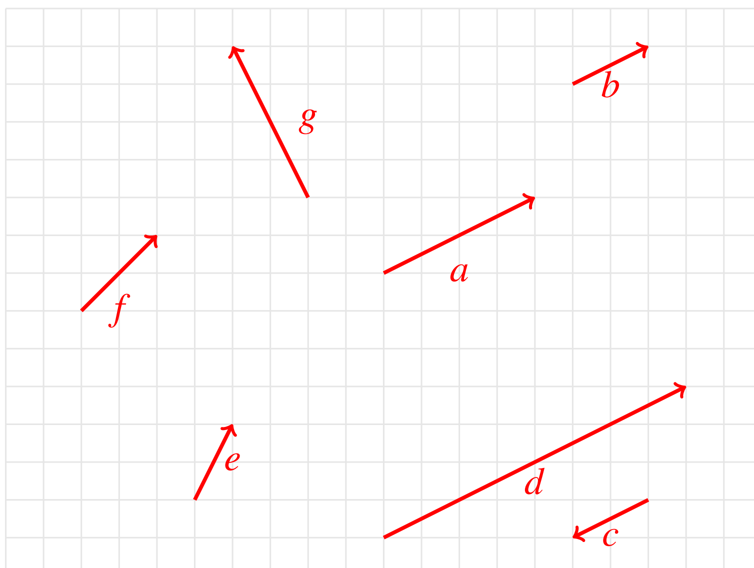
Homework 1.3.3.1 $\left(\left(\begin{array}{c}-1\\2\end{array}\right)+\left(\begin{array}{c}-1\\2\end{array}\right)\right)+\left(\begin{array}{c}-1\\2\end{array}\right)=$

☛ SEE ANSWER

Homework 1.3.3.2 $3\left(\begin{array}{c}-1\\2\end{array}\right)=$

☛ SEE ANSWER

Homework 1.3.3.3 Consider the following picture:



Which vector equals $2a$?; $(1/2)a$? ; and $-(1/2)a$?

☛ SEE ANSWER

Algorithm

The following algorithm scales a vector $x \in \mathbb{R}^n$ by α, overwriting x with the result αx:

$$\left(\begin{array}{c}\chi_0\\\chi_1\\\vdots\\\chi_{n-1}\end{array}\right) := \left(\begin{array}{c}\alpha\chi_0\\\alpha\chi_1\\\vdots\\\alpha\chi_{n-1}\end{array}\right).$$

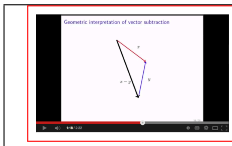
for $i = 0, \ldots, n-1$
  $\chi_i := \alpha\chi_i$
endfor

Cost

Scaling a vector requires n flops and $2n+1$ memops. Here, α is only brought in from memory once and kept in a register for reuse. To fully understand this, you need to know a little bit about computer architecture.

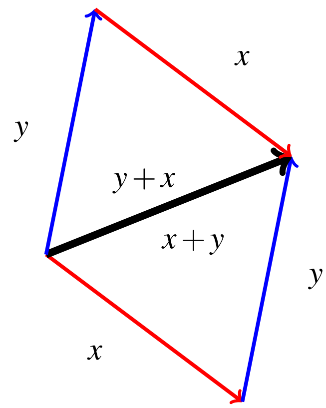
"Among friends" we will simply say that the cost is $2n$ memops since the one extra memory operation (to bring α in from memory) is negligible.

1.3.4 Vector Subtraction



☛ YouTube

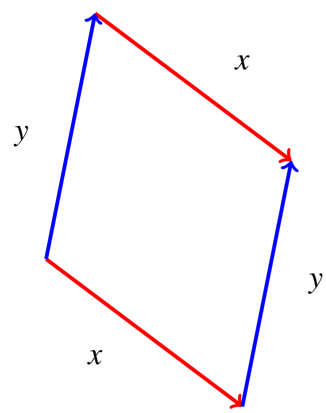
Recall the geometric interpretation for adding two vectors, $x, y \in \mathbb{R}^n$:



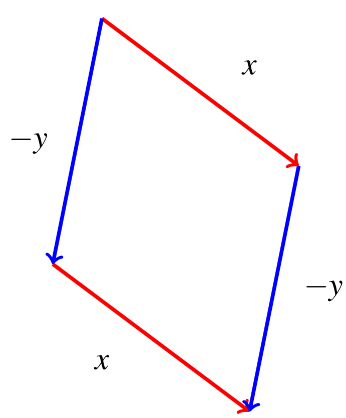
Subtracting y from x is defined as

$$x - y = x + (-y).$$

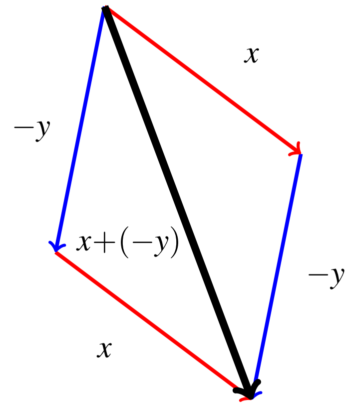
We learned in the last unit that $-y$ is the same as $(-1)y$ which is the same as pointing y in the opposite direction, while keeping it's length the same. This allows us to take the parallelogram that we used to illustrate vector addition



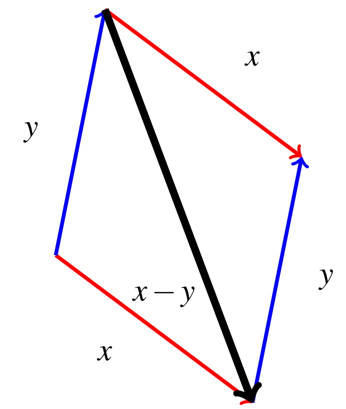
and change it into the equivalent picture



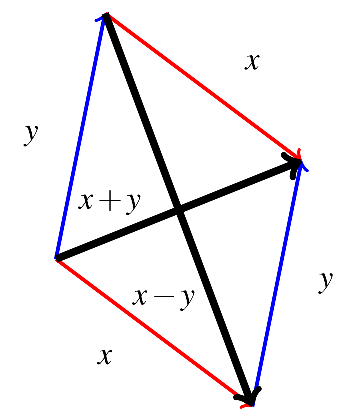
Since we know how to add two vectors, we can now illustrate $x+(-y)$:



Which then means that $x-y$ can be illustrated by



Finally, we note that the parallelogram can be used to simulaneously illustrate vector addition and subtraction:



(Obviously, you need to be careful to point the vectors in the right direction.)

Now computing $x-y$ when $x, y \in \mathbb{R}^n$ is a simple matter of subtracting components of y off the corresponding components of x:

$$x-y = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} - \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} = \begin{pmatrix} \chi_0 - \psi_0 \\ \chi_1 - \psi_1 \\ \vdots \\ \chi_{n-1} - \psi_{n-1} \end{pmatrix}.$$

Homework 1.3.4.1 For $x \in \mathbb{R}^n$, $x - x = 0$.

Always/Sometimes/Never

☛ SEE ANSWER

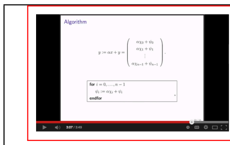
Homework 1.3.4.2 For $x, y \in \mathbb{R}^n$, $x - y = y - x$.

Always/Sometimes/Never

☛ SEE ANSWER

1.4 Advanced Vector Operations

1.4.1 Scaled Vector Addition (AXPY)



☛ YouTube

Definition

Definition 1.7 *One of the most commonly encountered operations when implementing more complex linear algebra operations is the scaled vector addition, which (given $x, y \in \mathbb{R}^n$) computes $y := \alpha x + y$:*

$$\alpha x + y = \alpha \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} = \begin{pmatrix} \alpha\chi_0 + \psi_0 \\ \alpha\chi_1 + \psi_1 \\ \vdots \\ \alpha\chi_{n-1} + \psi_{n-1} \end{pmatrix}.$$

It is often referred to as the AXPY operation, which stands for <u>a</u>lpha times <u>x</u> <u>p</u>lus <u>y</u>. We emphasize that it is typically used in situations where the output vector overwrites the input vector y.

Algorithm

Obviously, one could copy x into another vector, scale it by α, and then add it to y. Usually, however, vector y is simply updated one element at a time:

$$\begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} := \begin{pmatrix} \alpha\chi_0 + \psi_0 \\ \alpha\chi_1 + \psi_1 \\ \vdots \\ \alpha\chi_{n-1} + \psi_{n-1} \end{pmatrix}.$$

for $i = 0, \ldots, n-1$

  $\psi_i := \alpha\chi_i + \psi_i$

endfor

Cost

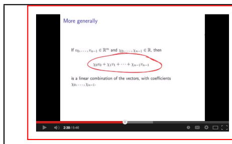
In Section 1.3 for many of the operations we discuss the cost in terms of memory operations (memops) and floating point operations (flops). This is discussed in the text, but not the videos. The reason for this is that we will talk about the cost of various operations later in a larger context, and include these discussions here more for completely.

Homework 1.4.1.1 What is the cost of an axpy operation?

- How many memops?
- How many flops?

☛ SEE ANSWER

1.4.2 Linear Combinations of Vectors



☛ YouTube

Discussion

There are few concepts in linear algebra more fundamental than linear combination of vectors.

Definition

Definition 1.8 *Let $u, v \in \mathbb{R}^m$ and $\alpha, \beta \in \mathbb{R}$. Then $\alpha u + \beta v$ is said to be a linear combination of vectors u and v:*

$$\alpha u + \beta v = \alpha \begin{pmatrix} \upsilon_0 \\ \upsilon_1 \\ \vdots \\ \upsilon_{m-1} \end{pmatrix} + \beta \begin{pmatrix} \nu_0 \\ \nu_1 \\ \vdots \\ \nu_{m-1} \end{pmatrix} = \begin{pmatrix} \alpha\upsilon_0 \\ \alpha\upsilon_1 \\ \vdots \\ \alpha\upsilon_{m-1} \end{pmatrix} + \begin{pmatrix} \beta\nu_0 \\ \beta\nu_1 \\ \vdots \\ \beta\nu_{m-1} \end{pmatrix} = \begin{pmatrix} \alpha\upsilon_0 + \beta\nu_0 \\ \alpha\upsilon_1 + \beta\nu_1 \\ \vdots \\ \alpha\upsilon_{m-1} + \beta\nu_{m-1} \end{pmatrix}.$$

The scalars α and β are the coefficients used in the linear combination.

More generally, if $v_0, \ldots, v_{n-1} \in \mathbb{R}^m$ are n vectors and $\chi_0, \ldots, \chi_{n-1} \in \mathbb{R}$ are n scalars, then $\chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1}$ is a linear combination of the vectors, with coefficients $\chi_0, \ldots, \chi_{n-1}$.

We will often use the summation notation to more concisely write such a linear combination:

$$\chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1} = \sum_{j=0}^{n-1} \chi_j v_j.$$

Homework 1.4.2.1

$$3 \begin{pmatrix} 2 \\ 4 \\ -1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} =$$

☛ SEE ANSWER

Homework 1.4.2.2

$$-3 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + 4 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} =$$

☛ SEE ANSWER

Homework 1.4.2.3 Find α, β, γ such that

$$\alpha \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \gamma \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \\ 3 \end{pmatrix}$$

$\alpha =$ $\beta =$ $\gamma =$

☛ SEE ANSWER

Algorithm

Given $v_0, \ldots, v_{n-1} \in \mathbb{R}^m$ and $\chi_0, \ldots, \chi_{n-1} \in \mathbb{R}$ the linear combination $w = \chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1}$ can be computed by first setting the result vector w to the zero vector of size m, and then performing n AXPY operations:

$w = 0$ (the zero vector of size m)
for $j = 0, \ldots, n-1$
  $w := \chi_j v_j + w$
endfor

The axpy operation computed $y := \alpha x + y$. In our algorithm, χ_j takes the place of α, v_j the place of x, and w the place of y.

Cost

We noted that computing $w = \chi_0 v_0 + \chi_1 v_1 + \cdots \chi_{n-1} v_{n-1}$ can be implementated as n AXPY operations. This suggests that the cost is n times the cost of an AXPY operation with vectors of size m: $n \times (2m) = 2mn$ flops and (approximately) $n \times (3m)$ memops.

However, one can actually do better. The vector w is updated repeatedly. If this vector stays in the L1 cache of a computer, then it needs not be repeatedly loaded from memory, and the cost becomes m memops (to load w into the cache) and then for each AXPY operation (approximately) m memops (to read v_j (ignoring the cost of reading χ_j). Then, once w has been completely updated, it can be written back to memory. So, the total cost related to accessing memory becomes $m + n \times m + m = (n+2)m \approx mn$ memops.

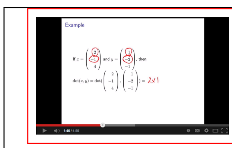
An important example

Example 1.9 Given any $x \in \mathbb{R}^n$ with $x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}$, this vector can always be written as the linear combination of the unit basis vectors given by

$$\begin{aligned} x &= \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} + \chi_1 \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{pmatrix} + \cdots + \chi_{n-1} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \\ &= \chi_0 e_0 + \chi_1 e_1 + \cdots + \chi_{n-1} e_{n-1} = \sum_{i=0}^{n-1} \chi_i e_i. \end{aligned}$$

Shortly, this will become really important as we make the connection between linear combinations of vectors, linear transformations, and matrices.

1.4.3 Dot or Inner Product (DOT)



☛ YouTube

Definition

The other commonly encountered operation is the dot (inner) product. It is defined by

$$\text{dot}(x,y) = \sum_{i=0}^{n-1} \chi_i \psi_i = \chi_0 \psi_0 + \chi_1 \psi_1 + \cdots + \chi_{n-1} \psi_{n-1}.$$

Alternative notation

We will often write

$$\begin{aligned} x^T y &= \text{dot}(x,y) = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}^T \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} \\ &= \begin{pmatrix} \chi_0 & \chi_1 & \cdots & \chi_{n-1} \end{pmatrix} \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} = \chi_0 \psi_0 + \chi_1 \psi_1 + \cdots + \chi_{n-1} \psi_{n-1} \end{aligned}$$

for reasons that will become clear later in the course.

Exercises

Homework 1.4.3.1 $\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} =$

☛ SEE ANSWER

Homework 1.4.3.2 $\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} =$

☛ SEE ANSWER

Homework 1.4.3.3 $\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} =$

☛ SEE ANSWER

Homework 1.4.3.4 For $x, y \in \mathbb{R}^n$, $x^T y = y^T x$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 1.4.3.5 $\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \left(\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \right) =$

☛ SEE ANSWER

Homework 1.4.3.6 $\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} =$

☛ SEE ANSWER

Homework 1.4.3.7 $\left(\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}\right)^T \begin{pmatrix} 1 \\ 0 \\ 0 \\ 2 \end{pmatrix} =$

☛ SEE ANSWER

Homework 1.4.3.8 For $x, y, z \in \mathbb{R}^n$, $x^T(y+z) = x^T y + x^T z$.

Always/Sometimes/Never
☛ SEE ANSWER

Homework 1.4.3.9 For $x, y, z \in \mathbb{R}^n$, $(x+y)^T z = x^T z + y^T z$.

Always/Sometimes/Never
☛ SEE ANSWER

Homework 1.4.3.10 For $x, y \in \mathbb{R}^n$, $(x+y)^T(x+y) = x^T x + 2x^T y + y^T y$.

Always/Sometimes/Never
☛ SEE ANSWER

Homework 1.4.3.11 Let $x, y \in \mathbb{R}^n$. When $x^T y = 0$, x or y is a zero vector.

Always/Sometimes/Never
☛ SEE ANSWER

Homework 1.4.3.12 For $x \in \mathbb{R}^n$, $e_i^T x = x^T e_i = \chi_i$, where χ_i equals the ith component of x.

Always/Sometimes/Never
☛ SEE ANSWER

Algorithm

An algorithm for the DOT operation is given by

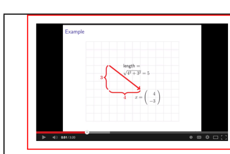
$\alpha := 0$
for $i = 0, \ldots, n-1$
  $\alpha := \chi_i \psi_i + \alpha$
endfor

Cost

Homework 1.4.3.13 What is the cost of a dot product with vectors of size n?

☛ SEE ANSWER

1.4.4 Vector Length (NORM2)



☛ YouTube

Definition

Let $x \in \mathbb{R}^n$. Then the (Euclidean) length of a vector x (the two-norm) is given by

$$\|x\|_2 = \sqrt{\chi_0^2 + \chi_1^2 + \cdots + \chi_{n-1}^2} = \sqrt{\sum_{i=0}^{n-1} \chi_i^2}.$$

Here $\|x\|_2$ notation stands for "the two norm of x", which is another way of saying "the length of x".

A vector of length one is said to be a unit vector.

Exercises

Homework 1.4.4.1 Compute the lengths of the following vectors:

(a) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ (b) $\begin{pmatrix} 1/2 \\ 1/2 \\ 1/2 \\ 1/2 \end{pmatrix}$ (c) $\begin{pmatrix} 1 \\ -2 \\ 2 \end{pmatrix}$ (d) $\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$

☛ SEE ANSWER

Homework 1.4.4.2 Let $x \in \mathbb{R}^n$. The length of x is less than zero: $\|x\|_2 < 0$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 1.4.4.3 If x is a unit vector then x is a unit basis vector.

TRUE/FALSE

☛ SEE ANSWER

Homework 1.4.4.4 If x is a unit basis vector then x is a unit vector.

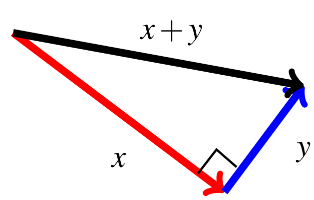
TRUE/FALSE

☛ SEE ANSWER

Homework 1.4.4.5 If x and y are perpendicular (orthogonal) then $x^T y = 0$.

TRUE/FALSE

Hint: Consider the picture



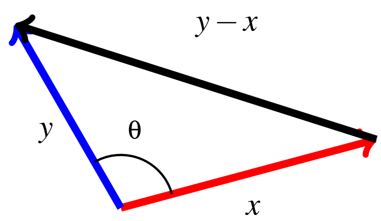
☛ SEE ANSWER

Homework 1.4.4.6 Let $x, y \in \mathbb{R}^n$ be nonzero vectors and let the angle between them equal θ. Then

$$\cos\theta = \frac{x^T y}{\|x\|_2 \|y\|_2}.$$

Always/Sometimes/Never

Hint: Consider the picture and the "Law of Cosines" that you learned in high school. (Or look up this law!)



☛ SEE ANSWER

Homework 1.4.4.7 Let $x, y \in \mathbb{R}^n$ be nonzero vectors. Then $x^T y = 0$ if and only if x and y are orthogonal (perpendicular).

True/False

☛ SEE ANSWER

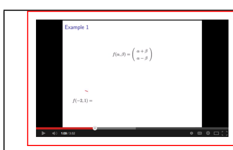
Algorithm

Clearly, $\|x\|_2 = \sqrt{x^T x}$, so that the DOT operation can be used to compute this length.

Cost

If computed with a dot product, it requires approximately n memops and $2n$ flops.

1.4.5 Vector Functions



☛ **YouTube**

Last week, we saw a number of examples where a function, f, takes in one or more scalars and/or vectors, and outputs a vector (where a scalar can be thought of as a special case of a vector, with unit size). These are all examples of vector-valued functions (or vector functions for short).

Definition

A vector(-valued) function is a mathematical functions of one or more scalars and/or vectors whose output is a vector.

Examples

Example 1.10

$$f(\alpha, \beta) = \begin{pmatrix} \alpha + \beta \\ \alpha - \beta \end{pmatrix} \quad \text{so that} \quad f(-2, 1) = \begin{pmatrix} -2 + 1 \\ -2 - 1 \end{pmatrix} = \begin{pmatrix} -1 \\ -3 \end{pmatrix}.$$

Example 1.11

$$f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \alpha \\ \chi_1 + \alpha \\ \chi_2 + \alpha \end{pmatrix} \quad \text{so that} \quad f(-2, \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}) = \begin{pmatrix} 1 + (-2) \\ 2 + (-2) \\ 3 + (-2) \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

Example 1.12 *The* AXPY *and* DOT *vector functions are other functions that we have already encountered.*

Example 1.13

$$f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_1 + \chi_2 \end{pmatrix} \quad \text{so that} \quad f(\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}) = \begin{pmatrix} 1 + 2 \\ 2 + 3 \end{pmatrix} = \begin{pmatrix} 3 \\ 5 \end{pmatrix}.$$

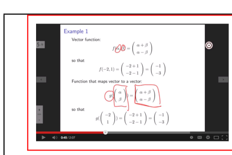
Exercises

Homework 1.4.5.1 If $f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \alpha \\ \chi_1 + \alpha \\ \chi_2 + \alpha \end{pmatrix}$, find

- $f(1, \begin{pmatrix} 6 \\ 2 \\ 3 \end{pmatrix}) =$
- $f(\alpha, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}) =$
- $f(0, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $f(\beta, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $\alpha f(\beta, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $f(\beta, \alpha \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) =$
- $f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + f(\alpha, \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) =$

☛ SEE ANSWER

1.4.6 Vector Functions that Map a Vector to a Vector



☛ YouTube

Now, we can talk about such functions in general as being a function from one vector to another vector. After all, we can take all inputs, make one vector with the separate inputs as the elements or subvectors of that vector, and make that the input for a new function that has the same net effect.

Example 1.14 *Instead of*

$$f(\alpha,\beta) = \begin{pmatrix} \alpha+\beta \\ \alpha-\beta \end{pmatrix} \quad \text{so that} \quad f(-2,1) = \begin{pmatrix} -2+1 \\ -2-1 \end{pmatrix} = \begin{pmatrix} -1 \\ -3 \end{pmatrix}$$

we can define

$$g(\begin{pmatrix} \alpha \\ \beta \end{pmatrix}) = \begin{pmatrix} \alpha+\beta \\ \alpha-\beta \end{pmatrix} \quad \text{so that} \quad g(\begin{pmatrix} -2 \\ 1 \end{pmatrix}) = \begin{pmatrix} -2+1 \\ -2-1 \end{pmatrix} = \begin{pmatrix} -1 \\ -3 \end{pmatrix}$$

Example 1.15 *Instead of*

$$f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+\alpha \\ \chi_1+\alpha \\ \chi_2+\alpha \end{pmatrix} \quad \text{so that} \quad f(-2, \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}) = \begin{pmatrix} 1+(-2) \\ 2+(-2) \\ 3+(-2) \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix},$$

we can define

$$g(\begin{pmatrix} \alpha \\ \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \end{pmatrix}) = g(\begin{pmatrix} \alpha \\ \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+\alpha \\ \chi_1+\alpha \\ \chi_2+\alpha \end{pmatrix} \quad \text{so that} \quad g(\begin{pmatrix} -2 \\ 1 \\ 2 \\ 3 \end{pmatrix}) = \begin{pmatrix} 1+(-2) \\ 2+(-2) \\ 3+(-2) \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

The bottom line is that we can focus on vector functions that map a vector of size n into a vector of size m, which is written as

$$f : \mathbb{R}^n \to \mathbb{R}^m.$$

Exercises

Homework 1.4.6.1 If $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + 1 \\ \chi_1 + 2 \\ \chi_2 + 3 \end{pmatrix}$, evaluate

- $f(\begin{pmatrix} 6 \\ 2 \\ 3 \end{pmatrix}) =$
- $f(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}) =$
- $f(2\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $2f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $f(\alpha\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $\alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) =$
- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) =$

☛ SEE ANSWER

Homework 1.4.6.2 If $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 \\ \chi_0 + \chi_1 \\ \chi_0 + \chi_1 + \chi_2 \end{pmatrix}$, evaluate

- $f(\begin{pmatrix} 6 \\ 2 \\ 3 \end{pmatrix}) =$
- $f(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}) =$
- $f(2\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $2f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $f(\alpha\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $\alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) =$
- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) =$
- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) =$

☛ SEE ANSWER

Homework 1.4.6.3 If $f : \mathbb{R}^n \to \mathbb{R}^m$, then

$$f(0) = 0.$$

Always/Sometimes/Never

☛ SEE ANSWER

Homework 1.4.6.4 If $f : \mathbb{R}^n \to \mathbb{R}^m$, $\lambda \in \mathbb{R}$, and $x \in \mathbb{R}^n$, then

$$f(\lambda x) = \lambda f(x).$$

Always/Sometimes/Never
☛ SEE ANSWER

Homework 1.4.6.5 If $f : \mathbb{R}^n \to \mathbb{R}^m$ and $x, y \in \mathbb{R}^n$, then

$$f(x+y) = f(x) + f(y).$$

Always/Sometimes/Never
☛ SEE ANSWER

1.5 LAFF Package Development: Vectors

1.5.1 Starting the Package

In this course, we will explore and use a rudimentary dense linear algebra software library. The hope is that by linking the abstractions in linear algebra to abstractions (functions) in software, a deeper understanding of the material will be the result.

We will be using the MATLAB interactive environment by MATHWORKS® for our exercises. MATLAB is a high-level language and interactive environment that started as a simple interactive "laboratory" for experimenting with linear algebra. It has since grown into a powerful tool for technical computing that is widely used in academia and industry.

For our Spring 2017 offering of LAFF on the edX platform, MATHWORKS® has again graceously made temporary licenses available for the participants. Instructions on how to install and use MATLAB can be found in Section 0.3.

The way we code can be easily translated into other languages. For example, as part of our FLAME research project we developed a library called `libflame`. Even though we coded it in the C programming language, it still closely resembles the MATLAB code that you will write and the library that you will use.

A library of vector-vector routines

The functionality of the functions that you will write is also part of the "laff" library of routines. What this means will become obvious in subsequent units.

Below is a table of vector functions, and the routines that implement them, that you will be able to use in future weeks. A more complete list of routines is given in Appendix A.

Operation Abbrev.	Definition	Function	MATLAB intrinsic	Approx. cost flops	Approx. cost memops
Vector-vector operations					
Copy (COPY)	$y := x$	`y = laff_copy( x, y )`	`y = x`	0	$2n$
Vector scaling (SCAL)	$x := \alpha x$	`x = laff_scal( alpha, x )`	`x = alpha * x`	n	$2n$
Scaled addition (AXPY)	$y := \alpha x + y$	`y = laff_axpy( alpha, x, y )`	`y = alpha * x + y`	$2n$	$3n$
Dot product (DOT)	$\alpha := x^T y$	`alpha = laff_dot( x, y )`	`alpha = x' * y`	$2n$	$2n$
Length (NORM2)	$\alpha := \|x\|_2$	`alpha = laff_norm2( x )`	`alpha = norm2( x )`	$2n$	n

A couple of comments:

- The operations we will implement are available already in MATLAB. So why do we write them as routines? Because
  1. It helps us connect the abstractions in the mathematics to the abstractions in code; and
  2. Implementations in other languages (e.g. C and Fortran) more closely follow how we will implement the operations as functions/routines.
- In, for example, `laff_copy`, why not make the function

`y = laff_copy( x )`?

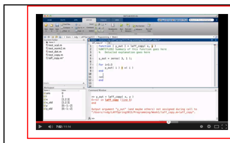
1. Often we will want to copy a column vector to a row vector or a row vector to a column vector. By also passing `y` into the routine, we indicate whether the output should be a row or a column vector.

2. Implementations in other languages (e.g. C and Fortran) more closely follow how we will implement the operations as functions/routines.

The way we will program translates almost directly into equivalent routines for the C or Python programming languages.

Now, let's dive right in! We'll walk you through it in the next units.

1.5.2 A Copy Routine (copy)



☛ YouTube

Homework 1.5.2.1 Implement the function `laff_copy` that copies a vector into another vector. The function is defined as

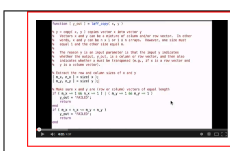
```
function [ y_out ] = laff_copy( x, y )
```

where

- `x` and `y` must each be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- `y_out` must be the same kind of vector as `y` (in other words, if `y` is a column vector, so is `y_out` and if `y` is a row vector, so is `y_out`).
- The function should "transpose" the vector if `x` and `y` do not have the same "shape" (if one is a column vector and the other one is a row vector).
- If `x` and/or `y` are not vectors or if the size of (row or column) vector `x` does not match the size of (row or column) vector `y`, the output should be `'FAILED'`.

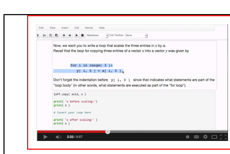
☛ Additional instructions. If link does not work, open `LAFF-2.0xM/1521Instructions.pdf`.

☛ SEE ANSWER



☛ YouTube

1.5.3 A Routine that Scales a Vector (scal)



☛ YouTube

Homework 1.5.3.1 Implement the function `laff_scal` that scales a vector x by a scalar α. The function is defined as

```
function [ x_out ] = laff_scal( alpha, x )
```

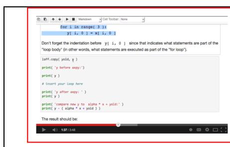
where

- `x` must be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- `x_out` must be the same kind of vector as `x`; and
- If `x` or `alpha` are not a (row or column) vector and scalar, respectively, the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_scal.m`.

☛ SEE ANSWER

1.5.4 A Scaled Vector Addition Routine (axpy)



☛ YouTube

Homework 1.5.4.1 Implement the function `laff_axpy` that computes $\alpha x + y$ given scalar α and vectors x and y. The function is defined as

```
function [ y_out ] = laff_axpy( alpha, x, y )
```

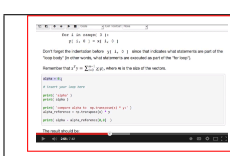
where

- `x` and `y` must each be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- `y_out` must be the same kind of vector as `y`; and
- If `x` and/or `y` are not vectors or if the size of (row or column) vector `x` does not match the size of (row or column) vector `y`, the output should be `'FAILED'`.
- If `alpha` is not a scalar, the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_axpy.m`.

☛ SEE ANSWER

1.5.5 An Inner Product Routine (dot)



☛ YouTube

Homework 1.5.5.1 Implement the function `laff_dot` that computes the dot product of vectors x and y. The function is defined as

```
function [ alpha ] = laff_dot( x, y )
```

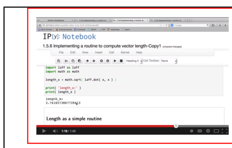
where

- `x` and `y` must each be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- If `x` and/or `y` are not vectors or if the size of (row or column) vector `x` does not match the size of (row or column) vector `y`, the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_dot.m`.

☛ SEE ANSWER

1.5.6 A Vector Length Routine (norm2)



☛ YouTube

Homework 1.5.6.1 Implement the function `laff_norm2` that computes the length of vector x. The function is defined as

```
function [ alpha ] = laff_norm2( x )
```

where

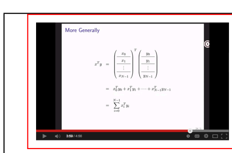
- `x` is an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- If `x` is not a vector the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_norm2.m`..

☛ SEE ANSWER

1.6 Slicing and Dicing

1.6.1 Slicing and Dicing: Dot Product



☛ YouTube

In the video, we justify the following theorem:

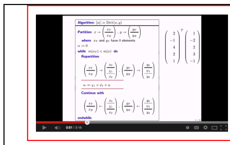
Theorem 1.16 *Let $x, y \in \mathbb{R}^n$ and partition (Slice and Dice) these vectors as*

$$x = \left(\begin{array}{c} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{array} \right) \quad \textit{and} \quad y = \left(\begin{array}{c} y_0 \\ \hline y_1 \\ \hline \vdots \\ \hline y_{N-1} \end{array} \right),$$

where $x_i, y_i \in \mathbb{R}^{n_i}$ *with* $\sum_{i=0}^{N-1} n_i = n$. *Then*

$$x^T y = x_0^T y_0 + x_1^T y_1 + \cdots + x_{N-1}^T y_{N-1} = \sum_{i=0}^{N-1} x_i^T y_i.$$

1.6.2 Algorithms with Slicing and Redicing: Dot Product



☛ YouTube

Algorithm: $[\alpha] := \text{DOT}(x, y)$

Partition $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where x_T and y_T have 0 elements

$\alpha := 0$

while $m(x_T) < m(x)$ **do**

Repartition

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ y_2 \end{array}\right)$$

where χ_1 has 1 row, ψ_1 has 1 row

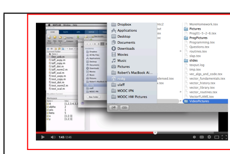
$\alpha := \chi_1 \times \psi_1 + \alpha$

Continue with

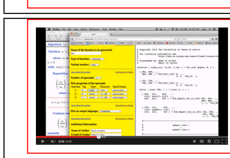
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

1.6.3 Coding with Slicing and Redicing: Dot Product



☛ YouTube



☛ YouTube

There are a number of steps you need to take with MATLAB Online before moving on with this unit. If you do this right, it will save you a lot of grief for the rest of the course:
When you uploaded `LAFF-2.0xM.zip` and unzipped it, that directory and all its subdirectories were automatically placed on the "path". In theory, in Unit 1.5.2, you removed `LAFF-2.0xM` from the path. If not: right-click on that folder, choose "Remove from path" and choose "Selected folder and subfolders". `LAFF-2.0xM` should now turn from black to gray. Next, there is a specific set of functions that we do want on the path. To accomplish this

- Expand folder `LAFF-2.0xM`.
- Expand subfolder `Programming`.
- Right-click on subfolder `laff` , choose "Add to path" and choose "Selected folder and subfolders". `laff` should now turn from gray to black. This should be the last time you need to set the path for this course.

Finally, you will want to make `LAFF-2.0xM -> Programming -> Week01` your current directory for the Command Window. You do this by double clicking on `LAFF-2.0xM -> Programming -> Week01`. To make sure the Command Window views this directory as the current directory, type "`pwd`" in the Command Window.

The video illustrates how to do the exercise using a desktop version of MATLAB. Hopefully it will be intuitively obvious how to do the exercise with MATLAB Online instead. If not, ask questions in the discussion for the unit.

Homework 1.6.3.1 Follow along with the video to implement the routine

`Dot_unb(x, y ).`

The "Spark webpage" can be found at

http://edx-org-utaustinx.s3.amazonaws.com/UT501x/Spark/index.html

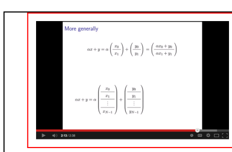
or by opening the file

`LAFF-2.0xM` → `Spark` → `index.html`

that should have been in the `LAFF-2.0xM.zip` file you downloaded and unzipped as described in Week0 (Unit 0.2.7).

☛ SEE ANSWER

1.6.4 Slicing and Dicing: axpy



☛ YouTube

In the video, we justify the following theorem:

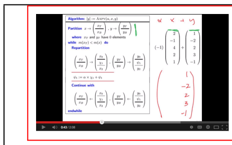
Theorem 1.17 *Let $\alpha \in \mathbb{R}$, $x, y \in \mathbb{R}^n$, and partition (Slice and Dice) these vectors as*

$$x = \left(\begin{array}{c} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{array}\right) \quad \textit{and} \quad y = \left(\begin{array}{c} y_0 \\ \hline y_1 \\ \hline \vdots \\ \hline y_{N-1} \end{array}\right),$$

where $x_i, y_i \in \mathbb{R}^{n_i}$ *with* $\sum_{i=0}^{N-1} n_i = n$. *Then*

$$\alpha x + y = \alpha \left(\begin{array}{c} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{array}\right) + \left(\begin{array}{c} y_0 \\ \hline y_1 \\ \hline \vdots \\ \hline y_{N-1} \end{array}\right) = \left(\begin{array}{c} \alpha x_0 + y_0 \\ \hline \alpha x_1 + y_1 \\ \hline \vdots \\ \hline \alpha x_{N-1} + y_{N-1} \end{array}\right).$$

1.6.5 Algorithms with Slicing and Redicing: axpy



☛ YouTube

Algorithm: $[y] := \text{AXPY}(\alpha, x, y)$

Partition $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where x_T and y_T have 0 elements

while $m(x_T) < m(x)$ **do**

Repartition

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

where χ_1 has 1 row, ψ_1 has 1 row

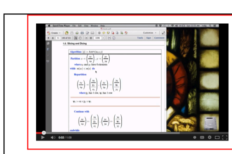
$\psi_1 := \alpha \times \chi_1 + \psi_1$

Continue with

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

1.6.6 Coding with Slicing and Redicing: axpy



☛ YouTube

Homework 1.6.6.1 Implement the routine

```
Axpy_unb( alpha, x, y ).
```

The "Spark webpage" can be found at

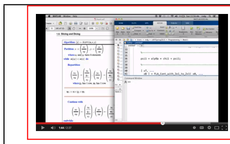
http://edx-org-utaustinx.s3.amazonaws.com/UT501x/Spark/index.html

or by opening the file

```
LAFF-2.0xM → Spark → index.html
```

that should have been in the LAFF-2.0xM.zip file you downloaded and unzipped as described in Week0 (Unit 0.2.7).

☛ SEE ANSWER



☛ YouTube

1.7 Enrichment

1.7.1 Learn the Greek Alphabet

In this course, we try to use the letters and symbols we use in a very consistent way, to help communication. As a general rule

- Lowercase Greek letters (α, β, etc.) are used for scalars.
- Lowercase (Roman) letters (a, b, etc) are used for vectors.
- Uppercase (Roman) letters (A, B, etc) are used for matrices.

Exceptions include the letters i, j, k, l, m, and n, which are typically used for integers.

Typically, if we use a given uppercase letter for a matrix, then we use the corresponding lower case letter for its columns (which can be thought of as vectors) and the corresponding lower case Greek letter for the elements in the matrix. Similarly, as we have already seen in previous sections, if we start with a given letter to denote a vector, then we use the corresponding lower case Greek letter for its elements.

Table 1.1 lists how we will use the various letters.

1.7.2 Other Norms

A norm is a function, in our case of a vector in $\mathbb{R}^n$, that maps every vector to a nonnegative real number. The simplest example is the absolute value of a real number: Given $\alpha \in \mathbb{R}$, the absolute value of α, often written as $|\alpha|$, equals the magnitude of α:

$$|\alpha| = \begin{cases} \alpha & \text{if } \alpha \geq 0 \\ -\alpha & \text{otherwise.} \end{cases}$$

Notice that only $\alpha = 0$ has the property that $|\alpha| = 0$ and that $|\alpha + \beta| \leq |\alpha| + |\beta|$, which is known as the *triangle inequality*.

Similarly, one can find functions, called norms, that measure the magnitude of vectors. One example is the (Euclidean) length of a vector, which we call the 2-norm: for $x \in \mathbb{R}^n$,

$$\|x\|_2 = \sqrt{\sum_{i=0}^{n-1} \chi_i^2}.$$

Clearly, $\|x\|_2 = 0$ if and only if $x = 0$ (the vector of all zeroes). Also, for $x, y \in \mathbb{R}^n$, one can show that $\|x + y\|_2 \leq \|x\|_2 + \|y\|_2$.

A function $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ is a norm if and only if the following properties hold for all $x, y \in \mathbb{R}^n$:

Matrix	Vector	Scalar			Note
		Symbol	LaTeX	Code	
A	a	α	`\alpha`	`alpha`	
B	b	β	`\beta`	`beta`	
C	c	γ	`\gamma`	`gamma`	
D	d	δ	`\delta`	`delta`	
E	e	ε	`\epsilon`	`epsilon`	$e_j = j$th unit basis vector.
F	f	ϕ	`\phi`	`phi`	
G	g	ξ	`\xi`	`xi`	
H	h	η	`\eta`	`eta`	
I					Used for identity matrix.
K	k	κ	`\kappa`	`kappa`	
L	l	λ	`\lambda`	`lambda`	
M	m	μ	`\mu`	`mu`	$m(\cdot) =$ row dimension.
N	n	ν	`\nu`	`nu`	ν is shared with V. $n(\cdot) =$ column dimension.
P	p	π	`\pi`	`pi`	
Q	q	θ	`\theta`	`theta`	
R	r	ρ	`\rho`	`rho`	
S	s	σ	`\sigma`	`sigma`	
T	t	τ	`\tau`	`tau`	
U	u	υ	`\upsilon`	`upsilon`	
V	v	ν	`\nu`	`nu`	ν shared with N.
W	w	ω	`\omega`	`omega`	
X	x	χ	`\chi`	`chi`	
Y	y	ψ	`\psi`	`psi`	
Z	z	ζ	`\zeta`	`zeta`	

Figure 1.1: Correspondence between letters used for matrices (uppercase Roman),vectors (lowercase Roman), and the symbols used to denote their scalar entries (lowercase Greek letters).

- $\|x\| \geq 0$; and
- $\|x\| = 0$ if and only if $x = 0$; and
- $\|x+y\| \leq \|x\| + \|y\|$ (the triangle inequality).

The 2-norm (Euclidean length) is a norm.

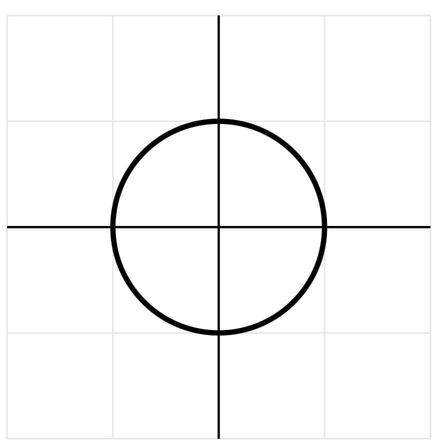
Are there other norms? The answer is yes:

- The taxi-cab norm, also known as the 1-norm:
$$\|x\|_1 = \sum_{i=0}^{n-1} |\chi_i|.$$
It is sometimes called the taxi-cab norm because it is the distance, in blocks, that a taxi would need to drive in a city like New York, where the streets are laid out like a grid.
- For $1 \leq p \leq \infty$, the p-norm:
$$\|x\|_p = \sqrt[p]{\sum_{i=0}^{n-1} |\chi_i|^p} = \left(\sum_{i=0}^{n-1} |\chi_i|^p \right)^{1/p}.$$
Notice that the 1-norm and the 2-norm are special cases.
- The ∞-norm:
$$\|x\|_\infty = \lim_{p\to\infty} \sqrt[p]{\sum_{i=0}^{n-1} |\chi_i|^p} = \max_{i=0}^{n-1} |\chi_i|.$$

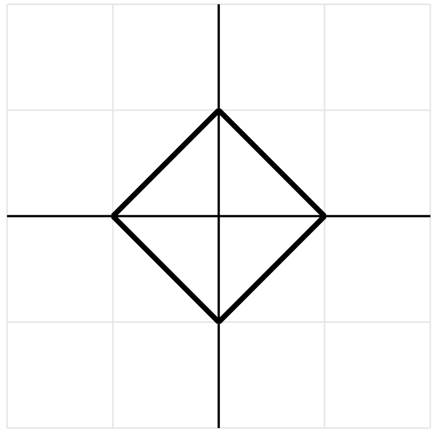
The bottom line is that there are many ways of measuring the length of a vector. In this course, we will only be concerned with the 2-norm.

We will not prove that these are norms, since that, in part, requires one to prove the triangle inequality and then, in turn, requires a theorem known as the Cauchy-Schwarz inequality. Those interested in seeing proofs related to the results in this unit are encouraged to investigate norms further.

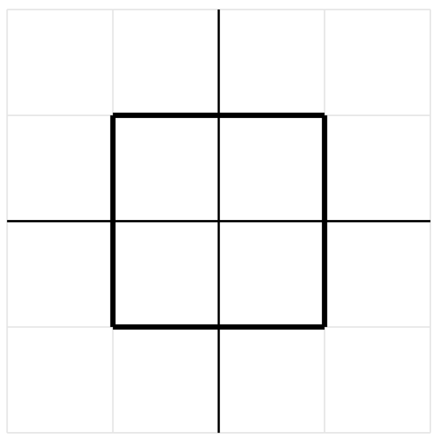
Example 1.18 The vectors with norm equal to one are often of special interest. Below we plot the points to which vectors x with $\|x\|_2 = 1$ point (when those vectors start at the origin, $(0,0)$). (E.g., the vector $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ points to the point $(1,0)$ and that vector has 2-norm equal to one, hence the point is one of the points to be plotted.)



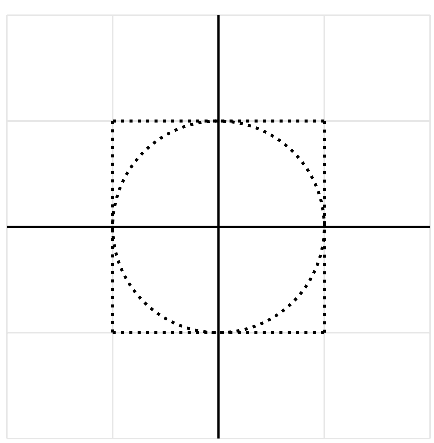
Example 1.19 Similarly, below we plot all points to which vectors x with $\|x\|_1 = 1$ point (starting at the origin).



Example 1.20 Similarly, below we plot all points to which vectors x with $\|x\|_\infty = 1$ point.



Example 1.21 Now consider all points to which vectors x with $\|x\|_p = 1$ point, when $2 < p < \infty$. These form a curve somewhere between the ones corresponding to $\|x\|_2 = 1$ and $\|x\|_\infty = 1$:



1.7.3 Overflow and Underflow

A detailed discussion of how real numbers are actually stored in a computer (approximations called floating point numbers) goes beyond the scope of this course. We will periodically expose some relevant properties of floating point numbers througout the course.

What is import right now is that there is a largest (in magnitude) number that can be stored and a smallest (in magnitude) number not equal to zero, that can be stored. Try to store a number larger in magnitude than this largest number, and you cause what is called an *overflow*. This is often stored as a "Not-A-Number" (NAN). Try to store a number not equal to zero and smaller in magnitude than this smallest number, and you cause what is called an *underflow*. An underflow is often set to zero.

Let us focus on overflow. The problem with computing the length (2-norm) of a vector is that it equals the square root of the sum of the squares of the components. While the answer may not cause an overflow, intermediate results when squaring components could. Specifically, any component greater in magnitude than the square root of the largest number that can be stored will overflow when squared.

The solution is to exploit the following observation: Let $\alpha > 0$. Then

$$\|x\|_2 = \sqrt{\sum_{i=0}^{n-1} \chi_i^2} = \sqrt{\sum_{i=0}^{n-1} \left[\alpha^2 \left(\frac{\chi_i}{\alpha}\right)^2\right]} = \sqrt{\alpha^2 \sum_{i=0}^{n-1} \left(\frac{\chi_i}{\alpha}\right)^2} = \alpha \sqrt{\left(\frac{1}{\alpha}x\right)^T \left(\frac{1}{\alpha}x\right)}$$

Now, we can use the following algorithm to compute the length of vector x:

- Choose $\alpha = \max_{i=0}^{n-1} |\chi_i|$.
- Scale $x := x/\alpha$.
- Compute $\|x\|_2 = \alpha\sqrt{x^T x}$.

Notice that no overflow for intermediate results (when squaring) will happen because all elements are of magnitude less than or equal to one. Similarly, only values that are very small relative to the final results will underflow because at least one of the components of x/α equals one.

1.7.4 A Bit of History

The functions that you developed as part of your LAFF library are very similar in functionality to Fortran routines known as the (level-1) Basic Linear Algebra Subprograms (BLAS) that are commonly used in scientific computing libraries. These were first proposed in the 1970s and were used in the development of one of the first linear algebra libraries, LINPACK. Classic references for that work are

- C. Lawson, R. Hanson, D. Kincaid, and F. Krogh, "Basic Linear Algebra Subprograms for Fortran Usage," *ACM Transactions on Mathematical Software*, 5 (1979) 305–325.
- J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart, *LINPACK Users' Guide*, SIAM, Philadelphia, 1979.

The style of coding that we use is at the core of our FLAME project and was first published in

- John A. Gunnels, Fred G. Gustavson, Greg M. Henry, and Robert A. van de Geijn, "FLAME: Formal Linear Algebra Methods Environment," *ACM Transactions on Mathematical Software*, 27 (2001) 422–455.
- Paolo Bientinesi, Enrique S. Quintana-Orti, and Robert A. van de Geijn, "Representing linear algebra algorithms in code: the FLAME application program interfaces," *ACM Transactions on Mathematical Software*, 31 (2005) 27–59.

1.8 Wrap Up

1.8.1 Homework

Homework 1.8.1.1 Let

$$x = \begin{pmatrix} 2 \\ -1 \end{pmatrix}, \quad y = \begin{pmatrix} \alpha \\ \beta - \alpha \end{pmatrix}, \quad \text{and} \quad x = y.$$

Indicate which of the following must be true (there may be multiple correct answers):

(a) $\alpha = 2$

(b) $\beta = (\beta - \alpha) + \alpha = (-1) + 2 = 1$

(c) $\beta - \alpha = -1$

(d) $\beta - 2 = -1$

(e) $x = 2e_0 - e_1$

☛ SEE ANSWER

Homework 1.8.1.2 A displacement vector represents the length and direction of an imaginary, shortest, straight path between two locations. To illustrate this as well as to emphasize the difference between ordered pairs that represent positions and vectors, we ask you to map a trip we made.

In 2012, we went on a journey to share our research in linear algebra. Below are some displacement vectors to describe parts of this journey using longitude and latitude. For example, we began our trip in Austin, TX and landed in San Jose, CA. Austin has coordinates $30^\circ\ 15'$ N(orth), $97^\circ\ 45'$ W(est) and San Jose's are $37^\circ\ 20'$ N, $121^\circ\ 54'$ W. *(Notice that convention is to report first longitude and then latitude.)* If we think of using longitude and latitude as coordinates in a plane where the first coordinate is position E (positive) or W (negative) and the second coordinate is position N (positive) or S (negative), then Austin's location is $(-97^\circ\ 45', 30^\circ\ 15')$ and San Jose's are $(-121^\circ\ 54', 37^\circ\ 20')$. *(Here, notice the switch in the order in which the coordinates are given because we now want to think of E/W as the x coordinate and N/S as the y coordinate.)* For our displacement vector for this, our first component will correspond to the change in the x coordinate, and the second component will be the change in the second coordinate. For convenience, we extend the notion of vectors so that the components include units as well as real numbers. Notice that for convenience, we extend the notion of vectors so that the components include units as well as real numbers (60 minutes ($'$)= 1 degree($^\circ$). Hence our displacement vector for Austin to San Jose is $\begin{pmatrix} -24^\circ\ 09' \\ 7^\circ\ 05' \end{pmatrix}$.

After visiting San Jose, we returned to Austin before embarking on a multi-legged excursion. That is, from Austin we flew to the first city and then from that city to the next, and so forth. In the end, we returned to Austin.

The following is a table of cities and their coordinates:

City	Coordinates	City	Coordinates
London	$00^\circ\ 08'$ W, $51^\circ\ 30'$ N	Austin	$-97^\circ\ 45'$ E, $30^\circ\ 15'$ N
Pisa	$10^\circ\ 21'$ E, $43^\circ\ 43'$ N	Brussels	$04^\circ\ 21'$ E, $50^\circ\ 51'$ N
Valencia	$00^\circ\ 23'$ E, $39^\circ\ 28'$ N	Darmstadt	$08^\circ\ 39'$ E, $49^\circ\ 52'$ N
Zürich	$08^\circ\ 33'$ E, $47^\circ\ 22'$ N	Krakow	$19^\circ\ 56'$ E, $50^\circ\ 4'$ N

Determine the order in which cities were visited, starting in Austin, given that the legs of the trip (given in order) had the following displacement vectors:

$$\begin{pmatrix} 102^\circ\ 06' \\ 20^\circ\ 36' \end{pmatrix} \rightarrow \begin{pmatrix} 04^\circ\ 18' \\ -00^\circ\ 59' \end{pmatrix} \rightarrow \begin{pmatrix} -00^\circ\ 06' \\ -02^\circ\ 30' \end{pmatrix} \rightarrow \begin{pmatrix} 01^\circ\ 48' \\ -03^\circ\ 39' \end{pmatrix} \rightarrow$$

$$\begin{pmatrix} 09^\circ\ 35' \\ 06^\circ\ 21' \end{pmatrix} \rightarrow \begin{pmatrix} -20^\circ\ 04' \\ 01^\circ\ 26' \end{pmatrix} \rightarrow \begin{pmatrix} 00^\circ\ 31' \\ -12^\circ\ 02' \end{pmatrix} \rightarrow \begin{pmatrix} -98^\circ\ 08' \\ -09^\circ\ 13' \end{pmatrix}$$

☛ SEE ANSWER

Homework 1.8.1.3 These days, high performance computers are called clusters and consist of many compute nodes, connected via a communication network. Each node of the cluster is basically equipped with a central processing unit (CPU), memory chips, a hard disk, and a network card. The nodes can be monitored for average power consumption (via power sensors) and application activity.

A system administrator monitors the power consumption of a node of such a cluster for an application that executes for two hours. This yields the following data:

Component	Average power (W)	Time in use (in hours)	Fraction of time in use
CPU	90	1.4	0.7
Memory	30	1.2	0.6
Disk	10	0.6	0.3
Network	15	0.2	0.1
Sensors	5	2.0	1.0

The energy, often measured in KWh, is equal to power times time. Notice that the total energy consumption can be found using the dot product of the vector of components' average power and the vector of corresponding time in use. What is the total energy consumed by this node in KWh? (The power is in Watts (W), so you will want to convert to Kilowatts (KW).)

Now, let's set this up as two vectors, x and y. The first records the power consumption for each of the components and the other for the total time that each of the components is in use:

$$x = \begin{pmatrix} 90 \\ 30 \\ 10 \\ 15 \\ 5 \end{pmatrix} \quad \text{and} \quad y = 2 \begin{pmatrix} 0.7 \\ 0.6 \\ 0.3 \\ 0.1 \\ 1.0 \end{pmatrix}.$$

Instead, compute $x^T y$. Think: How do the two ways of computing the answer relate?

☛ SEE ANSWER

Homework 1.8.1.4 (Examples from statistics) Linear algebra shows up often when computing with data sets. In this homework, you find out how dot products can be used to define various sums of values that are often encountered in statistics.
Assume you observe a random variable and you let those sampled values be represented by $\chi_i, i = 0, 1, 2, 3, \cdots, n-1$. We can let x be the vector with components χ_i and $\vec{1}$ be a vector of size n with components all ones:

$$x = \begin{pmatrix} \chi_0 \\ \vdots \\ \chi_{n-1} \end{pmatrix}, \quad \text{and} \quad \vec{1} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}.$$

For any x, the sum of the values of x can be computed using the dot product operation as

- $x^T x$
- $\vec{1}^T x$
- $x^T \vec{1}$

The sample mean of a random variable is the sum of the values the random variable takes on divided by the number of values, n. In other words, if the values the random variable takes on are stored in vector x, then $\bar{x} = \frac{1}{n} \sum_{i=0}^{n-1} \chi_i$. Using a dot product operation, for all x this can be computed as

- $\frac{1}{n} x^T x$
- $\frac{1}{n} \vec{1}^T x$
- $(\vec{1}^T \vec{1})^{-1} (x^T \vec{1})$

For any x, the sum of the squares of observations stored in (the elements of) a vector, x, can be computed using a dot product operation as

- $x^T x$
- $\vec{1}^T x$
- $x^T \vec{1}$

☛ SEE ANSWER

1.8.2 Summary of Vector Operations

Vector scaling	$\alpha x = \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \\ \vdots \\ \alpha\chi_{n-1} \end{pmatrix}$
Vector addition	$x + y = \begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \\ \vdots \\ \chi_{n-1} + \psi_{n-1} \end{pmatrix}$
Vector subtraction	$x - y = \begin{pmatrix} \chi_0 - \psi_0 \\ \chi_1 - \psi_1 \\ \vdots \\ \chi_{n-1} - \psi_{n-1} \end{pmatrix}$
AXPY	$\alpha x + y = \begin{pmatrix} \alpha\chi_0 + \psi_0 \\ \alpha\chi_1 + \psi_1 \\ \vdots \\ \alpha\chi_{n-1} + \psi_{n-1} \end{pmatrix}$
dot (inner) product	$x^T y = \sum_{i=0}^{n-1} \chi_i \psi_i$
vector length	$\|x\|_2 = \sqrt{x^T x} = \sqrt{\sum_{i=0}^{n-1} \chi_i \chi_i}$

1.8.3 Summary of the Properties of Vector Operations

Vector Addition

- Is commutative. That is, for all vectors $x, y \in \mathbb{R}^n, x + y = y + x$.
- Is associative. That is, for all vectors $x, y, z \in \mathbb{R}^n, (x + y) + z = x + (y + z)$.
- Has the zero vector as an identity.
- For all vectors $x \in \mathbb{R}^n, x + 0 = 0 + x = x$ where 0 is the vector of size n with 0 for each component.
- Has an inverse, $-x$. That is $x + (-x) = 0$.

The Dot Product of Vectors

- Is commutative. That is, for all vectors $x, y \in \mathbb{R}^n, x^T y = y^T x$.
- Distributes over vector addition. That is, for all vectors $x, y, z \in \mathbb{R}^n, x^T (y + z) = x^T y + x^T z$ and $(x + y)^T z = x^T z + y^T z$.

Partitioned vector operations

For (sub)vectors of appropriate size

- $\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{pmatrix} + \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{pmatrix} = \begin{pmatrix} x_0 + y_0 \\ x_1 + y_1 \\ \vdots \\ x_{N-1} + y_{N-1} \end{pmatrix}.$

- $\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{pmatrix}^T \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{pmatrix} = x_0^T y_0 + x_1^T y_1 + \cdots + x_{N-1}^T y_{N-1} = \sum_{i=0}^{N-1} x_i^T y_i.$

Other Properties

- For $x, y \in \mathbb{R}^n$, $(x+y)^T(x+y) = x^T x + 2x^T y + y^T y$.
- For $x, y \in \mathbb{R}^n$, $x^T y = 0$ if and only if x and y are orthogonal.
- Let $x, y \in \mathbb{R}^n$ be nonzero vectors and let the angle between them equal θ. Then $cos(\theta) = x^T y / \|x\|_2 \|y\|_2$.
- For $x \in \mathbb{R}^n$, $x^T e_i = e_i^T x = \chi_i$ where χ_i equals the ith component of x.

1.8.4 Summary of the Routines for Vector Operations

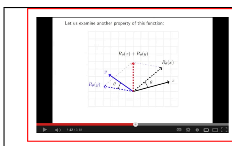
Operation Abbrev.	Definition	Function	Approx. cost	
			flops	memops
Vector-vector operations				
Copy (COPY)	$y := x$	`laff.copy( x, y )`	0	$2n$
Vector scaling (SCAL)	$x := \alpha x$	`laff.scal( alpha, x )`	n	$2n$
Scaled addition (AXPY)	$y := \alpha x + y$	`laff.axpy( alpha, x, y )`	$2n$	$3n$
Dot product (DOT)	$\alpha := x^T y$	`alpha = laff.dot( x, y )`	$2n$	$2n$
Length (NORM2)	$\alpha := \|x\|_2$	`alpha = laff.norm2( x )`	$2n$	n

Week 2

Linear Transformations and Matrices

2.1 Opening Remarks

2.1.1 Rotating in 2D



☛ YouTube

Let $R_\theta : \mathbb{R}^2 \to \mathbb{R}^2$ be the function that rotates an input vector through an angle θ:

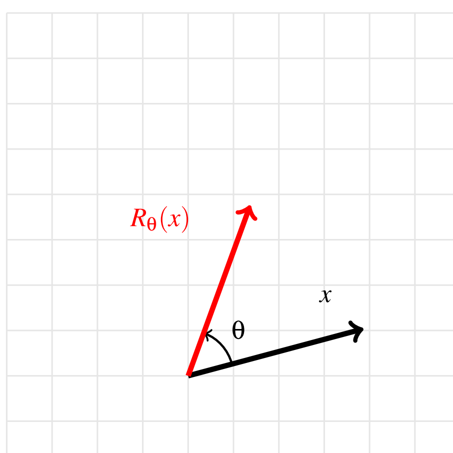


Figure 2.1 illustrates some special properties of the rotation. Functions with these properties are called called linear transformations. Thus, the illustrated rotation in 2D is an example of a linear transformation.

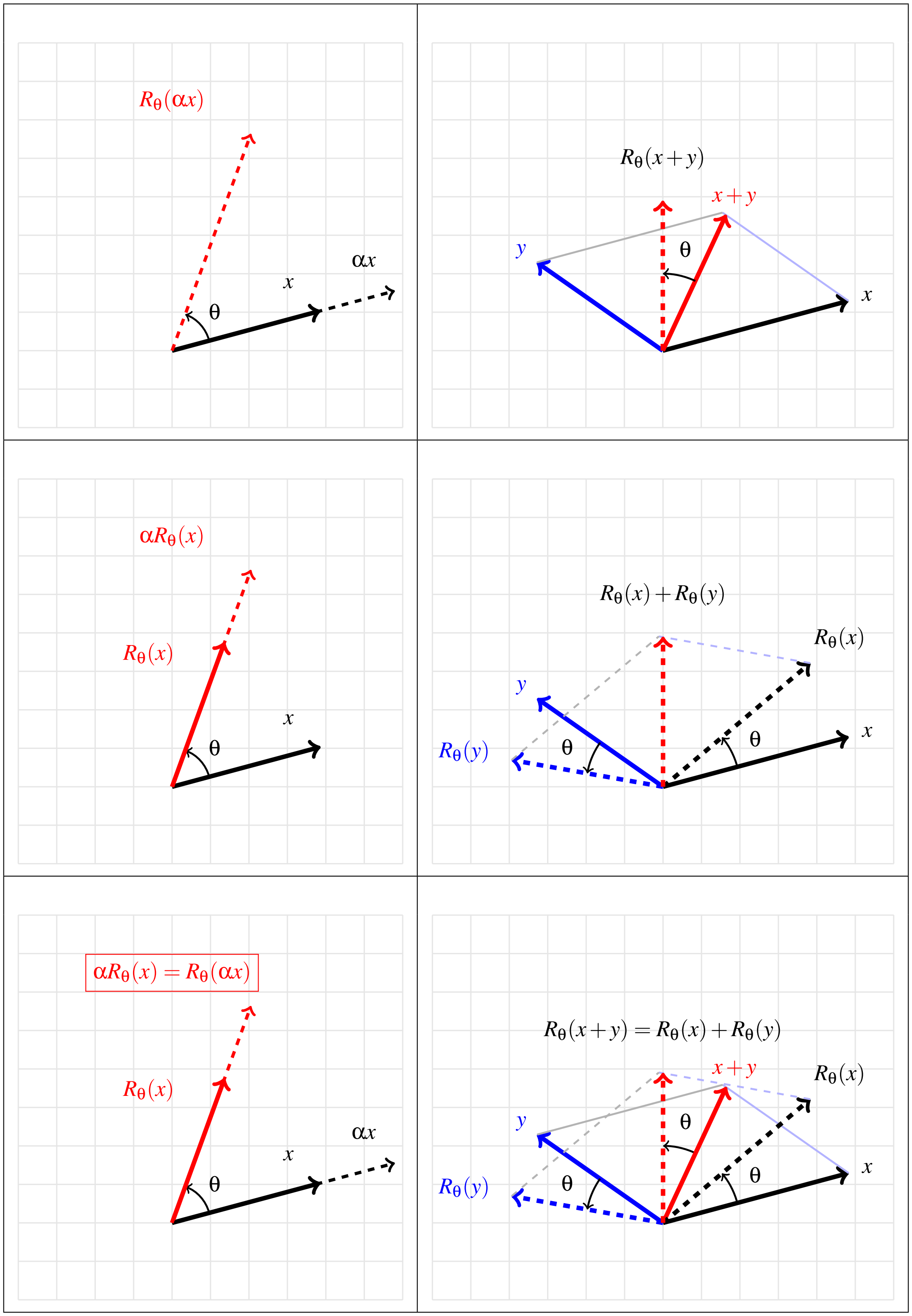
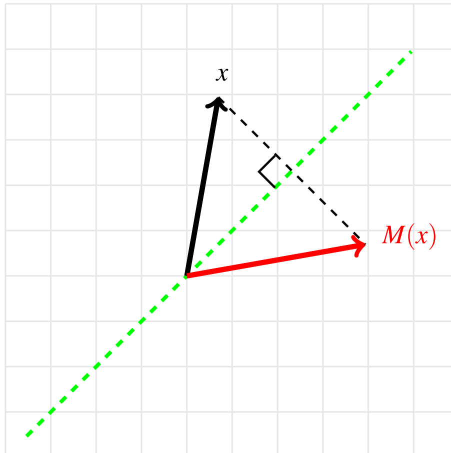


Figure 2.1: The three pictures on the left show that one can scale a vector first and then rotate, or rotate that vector first and then scale and obtain the same result. The three pictures on the right show that one can add two vectors first and then rotate, or rotate the two vectors first and then add and obtain the same result.

Homework 2.1.1.1 A reflection with respect to a 45 degree line is illustrated by



Think of the dashed green line as a mirror and $M : \mathbb{R}^2 \to \mathbb{R}^2$ as the vector function that maps a vector to its mirror image. If $x, y \in \mathbb{R}^2$ and $\alpha \in \mathbb{R}$, then $M(\alpha x) = \alpha M(x)$ and $M(x+y) = M(x) + M(y)$ (in other words, M is a linear transformation).

True/False

☛ SEE ANSWER

2.1.2 Outline

2.1.3 What You Will Learn

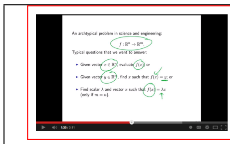
Upon completion of this unit, you should be able to

- Determine if a given vector function is a linear transformation.
- Identify, visualize, and interpret linear transformations.
- Recognize rotations and reflections in 2D as linear transformations of vectors.
- Relate linear transformations and matrix-vector multiplication.
- Understand and exploit how a linear transformation is completely described by how it transforms the unit basis vectors.
- Find the matrix that represents a linear transformation based on how it transforms unit basis vectors.
- Perform matrix-vector multiplication.
- Reason and develop arguments about properties of linear transformations and matrix vector multiplication.
- Read, appreciate, understand, and develop inductive proofs.
  (Ideally you will fall in love with them! They are beautiful. They don't deceive you. You can count on them. You can build on them. The perfect life companion! But it may not be love at first sight.)
- Make conjectures, understand proofs, and develop arguments about linear transformations.
- Understand the connection between linear transformations and matrix-vector multiplication.
- Solve simple problems related to linear transformations.

Track your progress in Appendix B.

2.2 Linear Transformations

2.2.1 What Makes Linear Transformations so Special?



☛ YouTube

Many problems in science and engineering involve vector functions such as: $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$. Given such a function, one often wishes to do the following:

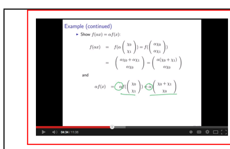
- Given vector $x \in \mathbb{R}^n$, evaluate $f(x)$; or
- Given vector $y \in \mathbb{R}^m$, find x such that $f(x) = y$; or
- Find scalar λ and vector x such that $f(x) = \lambda x$ (only if $m = n$).

For general vector functions, the last two problems are often especially difficult to solve. As we will see in this course, these problems become a lot easier for a special class of functions called linear transformations.

For those of you who have taken calculus (especially multivariate calculus), you learned that general functions that map vectors to vectors and have special properties can locally be approximated with a linear function. Now, we are not going to discuss what make a function linear, but will just say "it involves linear transformations." (When $m = n = 1$ you have likely seen this when you were taught about "Newton's Method") Thus, even when $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is not a linear transformation, linear transformations still come into play. This makes understanding linear transformations fundamental to almost all computational problems in science and engineering, just like calculus is.

But calculus is not a prerequisite for this course, so we won't talk about this... :-(

2.2.2 What is a Linear Transformation?



☛ YouTube

Definition

Definition 2.1 *A vector function $L : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is said to be a linear transformation, if for all $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$*

- *Transforming a scaled vector is the same as scaling the transformed vector:*

$$L(\alpha x) = \alpha L(x)$$

- *Transforming the sum of two vectors is the same as summing the two transformed vectors:*

$$L(x + y) = L(x) + L(y)$$

Examples

Example 2.2 *The transformation* $f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}\right) = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix}$ *is a linear transformation.*

The way we prove this is to pick arbitrary $\alpha \in \mathbb{R}$, $x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}$, *and* $y = \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}$ *for which we then show that* $f(\alpha x) = \alpha f(x)$ *and* $f(x + y) = f(x) + f(y)$*:*

- *Show* $f(\alpha x) = \alpha f(x)$*:*

$$f(\alpha x) = f(\alpha \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = f(\begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_0 + \alpha\chi_1 \\ \alpha\chi_0 \end{pmatrix} = \begin{pmatrix} \alpha(\chi_0 + \chi_1) \\ \alpha\chi_0 \end{pmatrix}$$

and

$$\alpha f(x) = \alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \alpha \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix} = \begin{pmatrix} \alpha(\chi_0 + \chi_1) \\ \alpha\chi_0 \end{pmatrix}$$

Both $f(\alpha x)$ *and* $\alpha f(x)$ *evaluate to the same expression. One can then make this into one continuous sequence of equivalences by rewriting the above as*

$$\begin{aligned} f(\alpha x) &= f(\alpha \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = f(\begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_0 + \alpha\chi_1 \\ \alpha\chi_0 \end{pmatrix} \\ &= \begin{pmatrix} \alpha(\chi_0 + \chi_1) \\ \alpha\chi_0 \end{pmatrix} = \alpha \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix} = \alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \alpha f(x). \end{aligned}$$

- *Show* $f(x+y) = f(x) + f(y)$*:*

$$f(x+y) = f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}) = f(\begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \end{pmatrix}) = \begin{pmatrix} (\chi_0 + \psi_0) + (\chi_1 + \psi_1) \\ \chi_0 + \psi_0 \end{pmatrix}$$

and

$$\begin{aligned} f(x) + f(y). &= f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix} + \begin{pmatrix} \psi_0 + \psi_1 \\ \psi_0 \end{pmatrix} \\ &= \begin{pmatrix} (\chi_0 + \chi_1) + (\psi_0 + \psi_1) \\ \chi_0 + \psi_0 \end{pmatrix}. \end{aligned}$$

Both $f(x+y)$ *and* $f(x) + f(y)$ *evaluate to the same expression since scalar addition is commutative and associative. The above observations can then be rearranged into the sequence of equivalences*

$$\begin{aligned} f(x+y) &= f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}) = f(\begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \end{pmatrix}) \\ &= \begin{pmatrix} (\chi_0 + \psi_0) + (\chi_1 + \psi_1) \\ \chi_0 + \psi_0 \end{pmatrix} = \begin{pmatrix} (\chi_0 + \chi_1) + (\psi_0 + \psi_1) \\ \chi_0 + \psi_0 \end{pmatrix} \\ &= \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix} + \begin{pmatrix} \psi_0 + \psi_1 \\ \psi_0 \end{pmatrix} = f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}) = f(x) + f(y). \end{aligned}$$

Example 2.3 *The transformation* $f(\begin{pmatrix} \chi \\ \psi \end{pmatrix}) = \begin{pmatrix} \chi + \psi \\ \chi + 1 \end{pmatrix}$ *is not a linear transformation.*

We will start by trying a few scalars α *and a few vectors* x *and see whether* $f(\alpha x) = \alpha f(x)$*. If we find even* **one** *example such that* $f(\alpha x) \neq f(\alpha x)$ *then we have proven that* f *is not a linear transformation. Likewise, if we find even* **one** *pair of vectors* x *and* y *such that* $f(x+y) \neq f(x) + f(y)$ *then we have done the same.*

A vector function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a linear transformation if *for all* scalars α and *for all* vectors $x, y \in \mathbb{R}^n$ it is that case that

- $f(\alpha x) = \alpha f(x)$ and
- $f(x+y) = f(x) + f(y)$.

If there is even one scalar α and vector $x \in \mathbb{R}^n$ such that $f(\alpha x) \neq \alpha f(x)$ or if there is even one pair of vectors $x, y \in \mathbb{R}^n$ such that $f(x+y) \neq f(x) + f(y)$, then the vector function f is *not* a linear transformation. Thus, in order to show that a vector function f is *not* a linear transformation, it suffices to find one such counter example.

Now, let us try a few:

- *Let* $\alpha = 1$ *and* $\begin{pmatrix} \chi \\ \psi \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. *Then*

$$f(\alpha \begin{pmatrix} \chi \\ \psi \end{pmatrix}) = f(1 \times \begin{pmatrix} 1 \\ 1 \end{pmatrix}) = f(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = \begin{pmatrix} 1+1 \\ 1+1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

and

$$\alpha f(\begin{pmatrix} \chi \\ \psi \end{pmatrix}) = 1 \times f(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = 1 \times \begin{pmatrix} 1+1 \\ 1+1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}.$$

For this example, $f(\alpha x) = \alpha f(x)$, *but there may still be an example such that* $f(\alpha x) \neq \alpha f(x)$.

- *Let* $\alpha = 0$ *and* $\begin{pmatrix} \chi \\ \psi \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. *Then*

$$f(\alpha \begin{pmatrix} \chi \\ \psi \end{pmatrix}) = f(0 \times \begin{pmatrix} 1 \\ 1 \end{pmatrix}) = f(\begin{pmatrix} 0 \\ 0 \end{pmatrix}) = \begin{pmatrix} 0+0 \\ 0+1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

and

$$\alpha f(\begin{pmatrix} \chi \\ \psi \end{pmatrix}) = 0 \times f(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = 0 \times \begin{pmatrix} 1+1 \\ 1+1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

For this example, we have found a case where $f(\alpha x) \neq \alpha f(x)$. *Hence, the function is not a linear transformation.*

Homework 2.2.2.1 The vector function $f(\begin{pmatrix} \chi \\ \psi \end{pmatrix}) = \begin{pmatrix} \chi\psi \\ \chi \end{pmatrix}$ is a linear transformation.

TRUE/FALSE
☛ SEE ANSWER

Homework 2.2.2.2 The vector function $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+1 \\ \chi_1+2 \\ \chi_2+3 \end{pmatrix}$ is a linear transformation. (This is the same function as in Homework 1.4.6.1.)

TRUE/FALSE
☛ SEE ANSWER

Homework 2.2.2.3 The vector function $f\left(\begin{pmatrix}\chi_0\\ \chi_1\\ \chi_2\end{pmatrix}\right)=\begin{pmatrix}\chi_0\\ \chi_0+\chi_1\\ \chi_0+\chi_1+\chi_2\end{pmatrix}$ is a linear transformation. (This is the same function as in Homework 1.4.6.2.)

TRUE/FALSE

☛ SEE ANSWER

Homework 2.2.2.4 If $L:\mathbb{R}^n\rightarrow\mathbb{R}^m$ is a linear transformation, then $L(0)=0$.
(Recall that 0 equals a vector with zero components of appropriate size.)

Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.2.2.5 Let $f:\mathbb{R}^n\rightarrow\mathbb{R}^m$ and $f(0)\neq 0$. Then f is not a linear transformation.

True/False

☛ SEE ANSWER

Homework 2.2.2.6 Let $f:\mathbb{R}^n\rightarrow\mathbb{R}^m$ and $f(0)=0$. Then f is a linear transformation.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.2.2.7 Find an example of a function f such that $f(\alpha x)=\alpha f(x)$, but for some x,y it is the case that $f(x+y)\neq f(x)+f(y)$. (This is pretty tricky!)

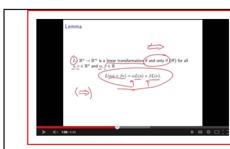
☛ SEE ANSWER

Homework 2.2.2.8 The vector function $f\left(\begin{pmatrix}\chi_0\\ \chi_1\end{pmatrix}\right)=\begin{pmatrix}\chi_1\\ \chi_0\end{pmatrix}$ is a linear transformation.

TRUE/FALSE

☛ SEE ANSWER

2.2.3 Of Linear Transformations and Linear Combinations



☛ YouTube

Now that we know what a linear transformation and a linear combination of vectors are, we are ready to start making the connection between the two with matrix-vector multiplication.

Lemma 2.4 *$L:\mathbb{R}^n\rightarrow\mathbb{R}^m$ is a linear transformation if and only if (iff) for all $u,v\in\mathbb{R}^n$ and $\alpha,\beta\in\mathbb{R}$*

$$L(\alpha u+\beta v)=\alpha L(u)+\beta L(v).$$

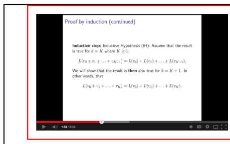
Proof:

($\Rightarrow$) Assume that $L:R^n\rightarrow R^m$ is a linear transformation and let $u,v\in\mathbb{R}^n$ be *arbitrary* vectors and $\alpha,\beta\in\mathbb{R}$ be *arbitrary* scalars. Then

$$\begin{array}{ll} & L(\alpha u+\beta v)\\ = & \quad <\text{since } \alpha u \text{ and } \beta v \text{ are vectors and } L \text{ is a linear transformation} >\\ & L(\alpha u)+L(\beta v)\\ = & \quad <\text{ since } L \text{ is a linear transformation} >\\ & \alpha L(u)+\beta L(v)\end{array}$$

($\Leftarrow$) Assume that for all $u, v \in \mathbb{R}^n$ and all $\alpha, \beta \in \mathbb{R}$ it is the case that $L(\alpha u + \beta v) = \alpha L(u) + \beta L(v)$. We need to show that

- $L(\alpha u) = \alpha L(u)$.
  This follows immediately by setting $\beta = 0$.
- $L(u + v) = L(u) + L(v)$.
  This follows immediately by setting $\alpha = \beta = 1$.



☛ YouTube

Lemma 2.5 *Let $v_0, v_1, \ldots, v_{k-1} \in \mathbb{R}^n$ and let $L : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transformation. Then*

$$L(v_0 + v_1 + \ldots + v_{k-1}) = L(v_0) + L(v_1) + \ldots + L(v_{k-1}). \tag{2.1}$$

While it is tempting to say that this is simply obvious, we are going to prove this rigorously. When one tries to prove a result for a general k, where k is a natural number, one often uses a "proof by induction". We are going to give the proof first, and then we will explain it.

Proof: Proof by induction on k.

Base case: $k = 1$. For this case, we must show that $L(v_0) = L(v_0)$. This is trivially true.

Inductive step: Inductive Hypothesis (IH): Assume that the result is true for $k = K$ where $K \geq 1$:

$$L(v_0 + v_1 + \ldots + v_{K-1}) = L(v_0) + L(v_1) + \ldots + L(v_{K-1}).$$

We will show that the result is **then** also true for $k = K + 1$. In other words, that

$$L(v_0 + v_1 + \ldots + v_{K-1} + v_K) = L(v_0) + L(v_1) + \ldots + L(v_{K-1}) + L(v_K).$$

$$L(v_0 + v_1 + \ldots + v_K)$$

$=$ $\quad < \text{expose extra term -- We know we can do this, since } K \geq 1 >$

$$L(v_0 + v_1 + \ldots + v_{K-1} + v_K)$$

$=$ $\quad < \text{associativity of vector addition} >$

$$L((v_0 + v_1 + \ldots + v_{K-1}) + v_K)$$

$=$ $\quad < L \text{ is a linear transformation)} >$

$$L(v_0 + v_1 + \ldots + v_{K-1}) + L(v_K)$$

$=$ $\quad < \text{Inductive Hypothesis} >$

$$L(v_0) + L(v_1) + \ldots + L(v_{K-1}) + L(v_K)$$

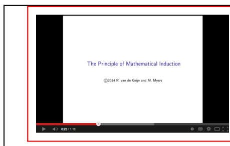
By the Principle of Mathematical Induction the result holds for all k.

The idea is as follows:

- The base case shows that the result is true for $k = 1$: $L(v_0) = L(v_0)$.
- The inductive step shows that if the result is true for $k = 1$, then the result is true for $k = 1 + 1 = 2$ so that $L(v_0 + v_1) = L(v_0) + L(v_1)$.
- Since the result is indeed true for $k = 1$ (as proven by the base case) we now know that the result is also true for $k = 2$.
- The inductive step also implies that if the result is true for $k = 2$, then it is also true for $k = 3$.
- Since we just reasoned that it is true for $k = 2$, we now know it is also true for $k = 3$: $L(v_0 + v_1 + v_2) = L(v_0) + L(v_1) + L(v_2)$.
- And so forth.

2.3 Mathematical Induction

2.3.1 What is the Principle of Mathematical Induction?



☛ YouTube

The **Principle of Mathematical Induction** (weak induction) says that *if* one can show that

- (Base case) a property holds for $k = k_b$; *and*
- (Inductive step) if it holds for $k = K$, where $K \geq k_b$, then it is also holds for $k = K + 1$,

then one can conclude that the property holds for all integers $k \geq k_b$. Often $k_b = 0$ or $k_b = 1$.

If mathematical induction intimidates you, have a look at in the enrichment for this week (Section 2.5.2) :Puzzles and Paradoxes in Mathematical Induction", by Adam Bjorndahl.

Here is Maggie's take on Induction, extending it beyond the proofs we do.

If you want to prove something holds for all members of a set that can be defined inductively, then you would use mathematical induction. You may recall a set is a collection and as such the order of its members is not important. However, some sets do have a natural ordering that can be used to describe the membership. This is especially valuable when the set has an infinite number of members, for example, natural numbers. Sets for which the membership can be described by suggesting there is a first element (or small group of firsts) then from this first you can create another (or others) then more and more by applying a rule to get another element in the set are our focus here. If all elements (members) are in the set because they are either the first (basis) or can be constructed by applying "The" rule to the first (basis) a finite number of times, then the set can be inductively defined.

So for us, the set of natural numbers is inductively defined. As a computer scientist you would say 0 is the first and the rule is to add one to get another element. So $0, 1, 2, 3, \ldots$ are members of the natural numbers. In this way, 10 is a member of natural numbers because you can find it by adding 1 to 0 ten times to get it.

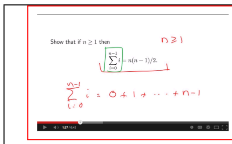
So, the Principle of Mathematical induction proves that something is true for all of the members of a set that can be defined inductively. If this set has an infinite number of members, you couldn't show it is true for each of them individually. The idea is if it is true for the first(s) and it is true for any constructed member(s) no matter where you are in the list, it must be true for all. Why? Since we are proving things about natural numbers, the idea is if it is true for 0 and the next constructed, it must be true for 1 but then its true for 2, and then 3 and 4 and 5 ...and 10 and $\ldots$ and 10000 and 10001 , etc (all natural numbers). This is only because of the special ordering we can put on this set so we can know there is a next one for which it must be true. People often picture this rule by thinking of climbing a ladder or pushing down dominoes. If you know you started and you know where ever you are the next will follow then you must make it through all (even if there are an infinite number).

That is why to prove something using the Principle of Mathematical Induction you must show what you are proving holds at a start and then if it holds (assume it holds up to some point) then it holds for the next constructed element in the set. With these two parts shown, we know it must hold for all members of this inductively defined set.

You can find many examples of how to prove using PMI as well as many examples of when and why this method of proof will fail all over the web. Notice it only works for statements about sets "that can be defined inductively". Also notice subsets of natural numbers can often be defined inductively. For example, if I am a mathematician I may start counting at 1. Or I may decide that the statement holds for natural numbers ≥ 4 so I start my base case at 4.

My last comment in this very long message is that this style of proof extends to other structures that can be defined inductively (such as trees or special graphs in CS).

2.3.2 Examples



☛ YouTube

Later in this course, we will look at the cost of various operations that involve matrices and vectors. In the analyses, we will often encounter a cost that involves the expression $\sum_{i=0}^{n-1} i$. We will now show that

$$\sum_{i=0}^{n-1} i = n(n-1)/2.$$

Proof:

Base case: $n = 1$. For this case, we must show that $\sum_{i=0}^{1-1} i = 1(0)/2$.

$\sum_{i=0}^{1-1} i$

$=$ $\quad$ $<$ Definition of summation $>$

0

$=$ $\quad$ $<$ arithmetic $>$

$1(0)/2$

This proves the base case.

Inductive step: Inductive Hypothesis (IH): Assume that the result is true for $n = k$ where $k \geq 1$:

$$\sum_{i=0}^{k-1} i = k(k-1)/2.$$

We will show that the result is then also true for $n = k+1$:

$$\sum_{i=0}^{(k+1)-1} i = (k+1)((k+1)-1)/2.$$

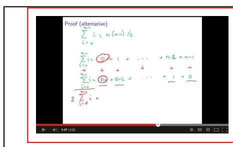
Assume that $k \geq 1$. Then

$$
\begin{aligned}
& \textstyle\sum_{i=0}^{(k+1)-1} i \\
= \quad & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad < \text{arithmetic} > \\
& \textstyle\sum_{i=0}^{k} i \\
= \quad & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad < \text{split off last term} > \\
& \textstyle\sum_{i=0}^{k-1} i + k \\
= \quad & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad < \text{I.H.} > \\
& k(k-1)/2 + k. \\
= \quad & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad < \text{algebra} > \\
& (k^2 - k)/2 + 2k/2. \\
= \quad & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad < \text{algebra} > \\
& (k^2 + k)/2. \\
= \quad & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad < \text{algebra} > \\
& (k+1)k/2. \\
= \quad & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad < \text{arithmetic} > \\
& (k+1)((k+1)-1)/2.
\end{aligned}
$$

This proves the inductive step.

By the Principle of Mathematical Induction the result holds for all n.

As we become more proficient, we will start combining steps. For now, we give lots of detail to make sure everyone stays on board.



☛ YouTube

There is an alternative proof for this result which does not involve mathematical induction. We give this proof now because it is a convenient way to rederive the result should you need it in the future.

Proof:(alternative)

$$
\begin{array}{rcccccccccc}
\sum_{i=0}^{n-1} i & = & 0 & + & 1 & + & \cdots & + & (n-2) & + & (n-1) \\
\sum_{i=0}^{n-1} i & = & (n-1) & + & (n-2) & + & \cdots & + & 1 & + & 0 \\
\hline
2\sum_{i=0}^{n-1} i & = & \multicolumn{9}{c}{\underbrace{(n-1) \; + \; (n-1) \; + \; \cdots \; + \; (n-1) \; + \; (n-1)}_{n \text{ times the term } (n-1)}}
\end{array}
$$

so that $2\sum_{i=0}^{n-1} i = n(n-1)$. Hence $\sum_{i=0}^{n-1} i = n(n-1)/2$.

For those who don't like the "$\cdots$" in the above argument, notice that

$$
\begin{aligned}
2\sum_{i=0}^{n-1} i &= \sum_{i=0}^{n-1} i + \sum_{j=0}^{n-1} j && < \text{algebra} > \\
&= \sum_{i=0}^{n-1} i + \sum_{j=n-1}^{0} j && < \text{reverse the order of the summation} > \\
&= \sum_{i=0}^{n-1} i + \sum_{i=0}^{n-1} (n-i-1) && < \text{substituting } j = n-i-1 > \\
&= \sum_{i=0}^{n-1} (i+n-i-1) && < \text{merge sums} > \\
&= \sum_{i=0}^{n-1} (n-1) && < \text{algebra} > \\
&= n(n-1) && < (n-1) \text{ is summed } n \text{ times} >.
\end{aligned}
$$

Hence $\sum_{i=0}^{n-1} i = n(n-1)/2$.

Homework 2.3.2.1 Let $n \geq 1$. Then $\sum_{i=1}^{n} i = n(n+1)/2$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.3.2.2 Let $n \geq 1$. $\sum_{i=0}^{n-1} 1 = n$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.3.2.3 Let $n \geq 1$ and $x \in \mathbb{R}^m$. Then

$$\sum_{i=0}^{n-1} x = \underbrace{x + x + \cdots + x}_{n \text{ times}} = nx$$

Always/Sometimes/Never

☛ SEE ANSWER

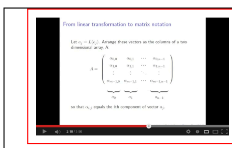
Homework 2.3.2.4 Let $n \geq 1$. $\sum_{i=0}^{n-1} i^2 = (n-1)n(2n-1)/6$.

Always/Sometimes/Never

☛ SEE ANSWER

2.4 Representing Linear Transformations as Matrices

2.4.1 From Linear Transformation to Matrix-Vector Multiplication



☛ YouTube

Theorem 2.6 *Let $v_o, v_1, \ldots, v_{n-1} \in \mathbb{R}^n$, $\alpha_o, \alpha_1, \ldots, \alpha_{n-1} \in \mathbb{R}$, and let $L : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transformation. Then*

$$L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{n-1} v_{n-1}) = \alpha_0 L(v_0) + \alpha_1 L(v_1) + \cdots + \alpha_{n-1} L(v_{n-1}). \tag{2.2}$$

Proof:

$L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{n-1} v_{n-1})$

$=$ < Lemma 2.5: $L(v_0 + \cdots + v_{n-1}) = L(v_0) + \cdots + L(v_{n-1})$ >

$L(\alpha_0 v_0) + L(\alpha_1 v_1) + \cdots + L(\alpha_{n-1} v_{n-1})$

$=$ <Definition of linear transformation, n times >

$\alpha_0 L(v_0) + \alpha_1 L(v_1) + \cdots + \alpha_{k-1} L(v_{k-1}) + \alpha_{n-1} L(v_{n-1})$.

Homework 2.4.1.1 Give an alternative proof for this theorem that mimics the proof by induction for the lemma that states that $L(v_0 + \cdots + v_{n-1}) = L(v_0) + \cdots + L(v_{n-1})$.

☛ SEE ANSWER

Homework 2.4.1.2 Let L be a linear transformation such that

$$L(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 3 \\ 5 \end{pmatrix} \quad \text{and} \quad L(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 2 \\ -1 \end{pmatrix}.$$

Then $L(\begin{pmatrix} 2 \\ 3 \end{pmatrix}) =$

☛ SEE ANSWER

For the next three exercises, let L be a linear transformation such that

$$L(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 3 \\ 5 \end{pmatrix} \quad \text{and} \quad L(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = \begin{pmatrix} 5 \\ 4 \end{pmatrix}.$$

Homework 2.4.1.3 $L(\begin{pmatrix} 3 \\ 3 \end{pmatrix}) =$

☛ SEE ANSWER

Homework 2.4.1.4 $L(\begin{pmatrix} -1 \\ 0 \end{pmatrix}) =$

☛ SEE ANSWER

Homework 2.4.1.5 $L(\begin{pmatrix} 2 \\ 3 \end{pmatrix}) =$

☛ SEE ANSWER

Homework 2.4.1.6 Let L be a linear transformation such that

$$L(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = \begin{pmatrix} 5 \\ 4 \end{pmatrix}.$$

Then $L(\begin{pmatrix} 3 \\ 2 \end{pmatrix}) =$

☛ SEE ANSWER

Homework 2.4.1.7 Let L be a linear transformation such that

$$L(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = \begin{pmatrix} 5 \\ 4 \end{pmatrix} \quad \text{and} \quad L(\begin{pmatrix} 2 \\ 2 \end{pmatrix}) = \begin{pmatrix} 10 \\ 8 \end{pmatrix}.$$

Then $L(\begin{pmatrix} 3 \\ 2 \end{pmatrix}) =$

☛ SEE ANSWER

Now we are ready to link linear transformations to matrices and matrix-vector multiplication.

Recall that any vector $x \in \mathbb{R}^n$ can be written as

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 \underbrace{\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}}_{e_0} + \chi_1 \underbrace{\begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}}_{e_1} + \cdots + \chi_{n-1} \underbrace{\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}}_{e_{n-1}} = \sum_{j=0}^{n-1} \chi_j e_j.$$

Let $L : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transformation. Given $x \in \mathbb{R}^n$, the result of $y = L(x)$ is a vector in $\mathbb{R}^m$. But then

$$y = L(x) = L\left(\sum_{j=0}^{n-1} \chi_j e_j.\right) = \sum_{j=0}^{n-1} \chi_j L(e_j) = \sum_{j=0}^{n-1} \chi_j a_j,$$

where we let $a_j = L(e_j)$.

The Big Idea. The linear transformation L is completely described by the vectors

$$a_0, a_1, \ldots, a_{n-1}, \quad \text{where } a_j = L(e_j)$$

because for any vector x, $L(x) = \sum_{j=0}^{n-1} \chi_j a_j$.

By arranging these vectors as the columns of a two-dimensional array, which we call the matrix A, we arrive at the observation that the matrix is simply a representation of the corresponding linear transformation L.

Homework 2.4.1.8 Give the matrix that corresponds to the linear transformation $f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}\right) = \begin{pmatrix} 3\chi_0 - \chi_1 \\ \chi_1 \end{pmatrix}$.

☛ SEE ANSWER

Homework 2.4.1.9 Give the matrix that corresponds to the linear transformation $f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}\right) = \begin{pmatrix} 3\chi_0 - \chi_1 \\ \chi_2 \end{pmatrix}$.

☛ SEE ANSWER

If we let

$$A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \underbrace{\alpha_{m-1,0}}_{a_0} & \underbrace{\alpha_{m-1,1}}_{a_1} & \cdots & \underbrace{\alpha_{m-1,n-1}}_{a_{n-1}} \end{pmatrix}$$

so that $\alpha_{i,j}$ equals the ith component of vector a_j, then

$$\begin{aligned} L(x) &= L(\sum_{j=0}^{n-1} \chi_j e_j) = \sum_{j=0}^{n-1} L(\chi_j e_j) = \sum_{j=0}^{n-1} \chi_j L(e_j) = \sum_{j=0}^{n-1} \chi_j a_j \\ &= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} \end{aligned}$$

$$
\begin{aligned}
&= \chi_0 \begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \\ \vdots \\ \alpha_{m-1,0} \end{pmatrix} + \chi_1 \begin{pmatrix} \alpha_{0,1} \\ \alpha_{1,1} \\ \vdots \\ \alpha_{m-1,1} \end{pmatrix} + \cdots + \chi_{n-1} \begin{pmatrix} \alpha_{0,n-1} \\ \alpha_{1,n-1} \\ \vdots \\ \alpha_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \chi_0\alpha_{0,0} \\ \chi_0\alpha_{1,0} \\ \vdots \\ \chi_0\alpha_{m-1,0} \end{pmatrix} + \begin{pmatrix} \chi_1\alpha_{0,1} \\ \chi_1\alpha_{1,1} \\ \vdots \\ \chi_1\alpha_{m-1,1} \end{pmatrix} + \cdots + \begin{pmatrix} \chi_{n-1}\alpha_{0,n-1} \\ \chi_{n-1}\alpha_{1,n-1} \\ \vdots \\ \chi_{n-1}\alpha_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \chi_0\alpha_{0,0}+ & \chi_1\alpha_{0,1}+ & \cdots+ & \chi_{n-1}\alpha_{0,n-1} \\ \chi_0\alpha_{1,0}+ & \chi_1\alpha_{1,1}+ & \cdots+ & \chi_{n-1}\alpha_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \chi_0\alpha_{m-1,0}+ & \chi_1\alpha_{m-1,1}+ & \cdots+ & \chi_{n-1}\alpha_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0}\chi_0+ & \alpha_{0,1}\chi_1+ & \cdots+ & \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0+ & \alpha_{1,1}\chi_1+ & \cdots+ & \alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0}\chi_0+ & \alpha_{m-1,1}\chi_1+ & \cdots+ & \alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = Ax.
\end{aligned}
$$

Definition 2.7 **($\mathbb{R}^{m\times n}$)**
The set of all $m \times n$ real valued matrices is denoted by $\mathbb{R}^{m\times n}$.

Thus, $A \in \mathbb{R}^{m\times n}$ means that A is a real valued matrix of size $m \times n$.

Definition 2.8 **(Matrix-vector multiplication or product)**
Let $A \in \mathbb{R}^{m\times n}$ and $x \in \mathbb{R}^n$ with

$$
A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,0} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,0} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,0} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \quad \text{and} \quad x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}.
$$

then

$$
\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}
$$

$$= \begin{pmatrix} \alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + & \cdots + & \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + & \cdots + & \alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + & \cdots + & \alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix}. \quad (2.3)$$

2.4.2 Practice with Matrix-Vector Multiplication

Homework 2.4.2.1 Compute Ax when $A = \begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix}$ and $x = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$.

☛ SEE ANSWER

Homework 2.4.2.2 Compute Ax when $A = \begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix}$ and $x = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$.

☛ SEE ANSWER

Homework 2.4.2.3 If A is a matrix and e_j is a unit basis vector of appropriate length, then $Ae_j = a_j$, where a_j is the jth column of matrix A.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.4.2.4 If x is a vector and e_i is a unit basis vector of appropriate size, then their dot product, $e_i^T x$, equals the ith entry in x, χ_i.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.4.2.5 Compute

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}^T \left(\begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right) = \_\_\_\_\_$$

☛ SEE ANSWER

Homework 2.4.2.6 Compute

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}^T \left(\begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right) = \_\_\_\_\_$$

☛ SEE ANSWER

Homework 2.4.2.7 Let A be a $m \times n$ matrix and $\alpha_{i,j}$ its (i,j) element. Then $\alpha_{i,j} = e_i^T (Ae_j)$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.4.2.8 Compute

- $\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix} \left((-2) \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) =$

- $(-2) \left(\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) =$

- $\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix} \left(\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right) =$

- $\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} =$

☛ SEE ANSWER

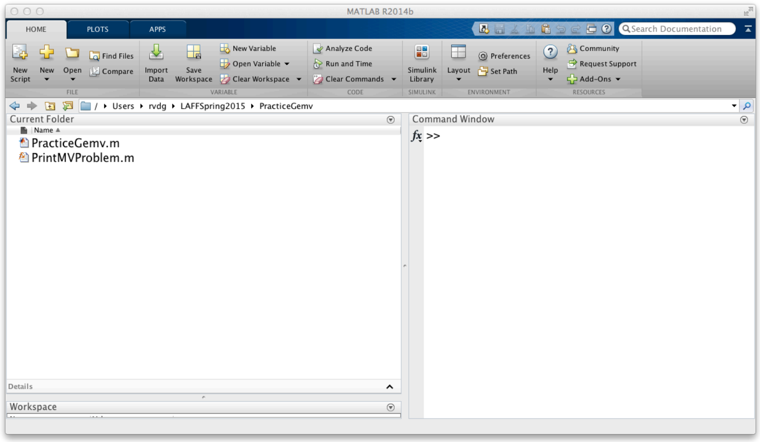
Homework 2.4.2.9 Let $A \in \mathbb{R}^{m \times n}$; $x, y \in \mathbb{R}^n$; and $\alpha \in \mathbb{R}$. Then

- $A(\alpha x) = \alpha(Ax)$.
- $A(x+y) = Ax + Ay$.

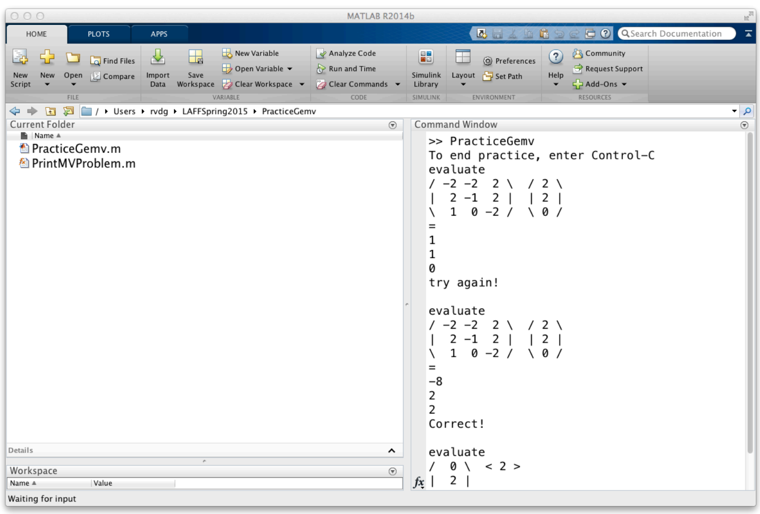
Always/Sometimes/Never

☛ SEE ANSWER

Homework 2.4.2.10 You can practice as little or as much as you want!
Some of the following instructions are for the desktop version of Matlab, but it should be pretty easy to figure out what to do instead with Matlab Online.
Start up Matlab or log on to Matlab Online and change the current directory to Programming/Week02/.



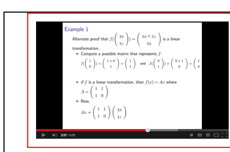
Then type `PracticeGemv` in the command window and you get to practice all the matrix-vector multiplications you want! For example, after a bit of practice my window looks like



Practice all you want!

☛ SEE ANSWER

2.4.3 It Goes Both Ways



☛ YouTube

The last exercise proves that the function that computes matrix-vector multiplication is a linear transformation:

Theorem 2.9 *Let $L : \mathbb{R}^n \to \mathbb{R}^m$ be defined by $L(x) = Ax$ where $A \in \mathbb{R}^{m\times n}$. Then L is a linear transformation.*

A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is a linear transformation if and only if it can be written as a matrix-vector multiplication.

Homework 2.4.3.1 Give the linear transformation that corresponds to the matrix

$$\begin{pmatrix} 2 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

☛ SEE ANSWER

Homework 2.4.3.2 Give the linear transformation that corresponds to the matrix

$$\begin{pmatrix} 2 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

☛ SEE ANSWER

Example 2.10 We showed that the function $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix}$ is a linear transformation in an earlier example. We will now provide an alternate proof of this fact.

We compute a *possible* matrix, A, that represents this linear transformation. We will then show that $f(x) = Ax$, which then means that f *is* a linear transformation since the above theorem states that matrix-vector multiplications are linear transformations.

To compute a possible matrix that represents f consider:

$$f(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1+0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad \text{and} \quad f(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0+1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Thus, *if* f is a linear transformation, then $f(x) = Ax$ where $A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$. Now,

$$Ax = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} g = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix} = f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = f(x).$$

Hence f is a linear transformation since $f(x) = Ax$.

Example 2.11 In Example 2.3 we showed that the transformation $f(\begin{pmatrix} \chi \\ \psi \end{pmatrix}) = \begin{pmatrix} \chi+\psi \\ \chi+1 \end{pmatrix}$ is not a linear transformation. We now show this again, by computing a possible matrix that represents it, and then showing that it does *not* represent it.
To compute a possible matrix that represents f consider:

$$f(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1+0 \\ 1+1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix} \quad \text{and} \quad f(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0+1 \\ 0+1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Thus, *if* f is a linear transformation, then $f(x) = Ax$ where $A = \begin{pmatrix} 1 & 1 \\ 2 & 1 \end{pmatrix}$. Now,

$$Ax = \begin{pmatrix} 1 & 1 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \chi_0+\chi_1 \\ 2\chi_0+\chi_1 \end{pmatrix} \neq \begin{pmatrix} \chi_0+\chi_1 \\ \chi_0+1 \end{pmatrix} = f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = f(x).$$

Hence f is *not* a linear transformation since $f(x) \neq Ax$.

The above observations give us a straight-forward, fool-proof way of checking whether a function is a linear transformation. You compute a possible matrix and then you check if the matrix-vector multiply always yields the same result as evaluating the function.

Homework 2.4.3.3 Let f be a vector function such that $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0^2 \\ \chi_1 \end{pmatrix}$ Then

- (a) f is a linear transformation.
- (b) f is not a linear transformation.
- (c) Not enough information is given to determine whether f is a linear transformation.

How do you know?

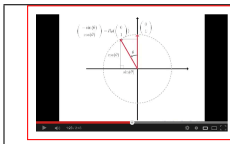
☛ SEE ANSWER

Homework 2.4.3.4 For each of the following, determine whether it is a linear transformation or not:

- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 \\ 0 \\ \chi_2 \end{pmatrix}$.
- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0^2 \\ 0 \end{pmatrix}$.

☛ SEE ANSWER

2.4.4 Rotations and Reflections, Revisited

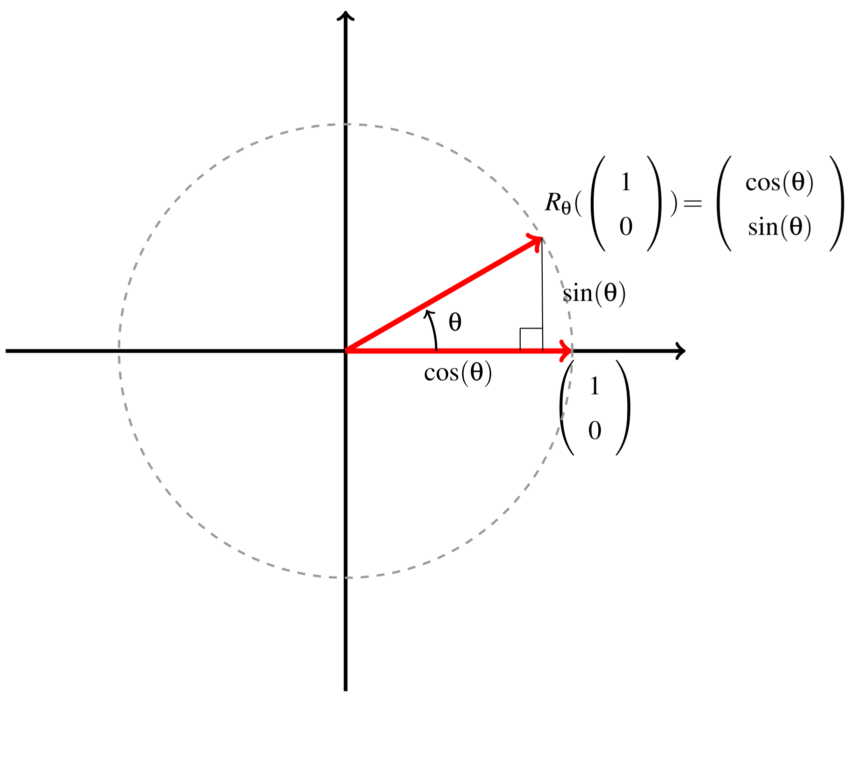


☛ YouTube

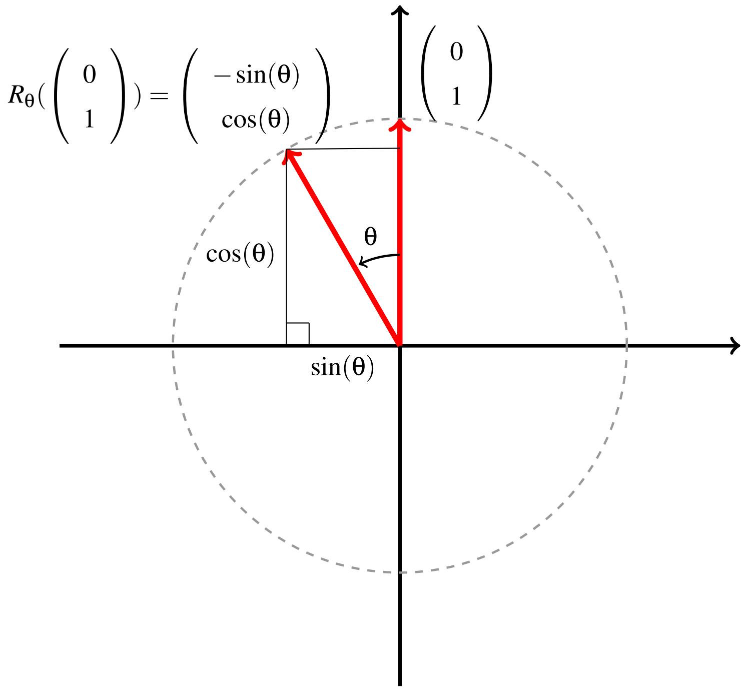
Recall that in the opener for this week we used a geometric argument to conclude that a rotation $R_\theta : \mathbb{R}^2 \to \mathbb{R}^2$ is a linear transformation. We now show how to compute the matrix, A, that represents this rotation.

Given that the transformation is from $\mathbb{R}^2$ to $\mathbb{R}^2$, we know that the matrix will be a 2×2 matrix. It will take vectors of size two as input and will produce vectors of size two. We have also learned that the first column of the matrix A will equal $R_\theta(e_0)$ and the second column will equal $R_\theta(e_1)$.

We first determine what vector results when $e_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ is rotated through an angle θ:



Next, we determine what vector results when $e_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ is rotated through an angle θ:



This shows that

$$R_\theta(e_0) = \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \quad \text{and} \quad R_\theta(e_1) = \begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix}.$$

We conclude that

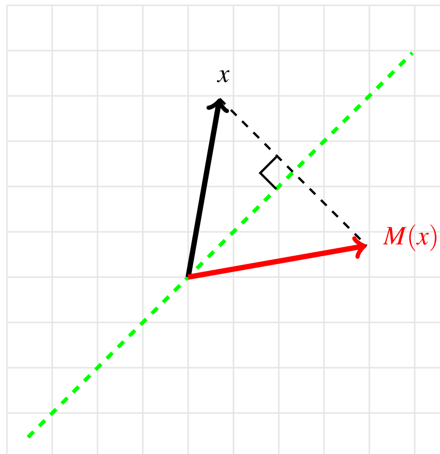
$$A = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}.$$

This means that an arbitrary vector $x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}$ is transformed into

$$R_\theta(x) = Ax = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \cos(\theta)\chi_0 - \sin(\theta)\chi_1 \\ \sin(\theta)\chi_0 + \cos(\theta)\chi_1 \end{pmatrix}.$$

This is a formula very similar to a formula you may have seen in a precalculus or physics course when discussing *change of coordinates*. We will revisit to this later.

Homework 2.4.4.1 A reflection with respect to a 45 degree line is illustrated by

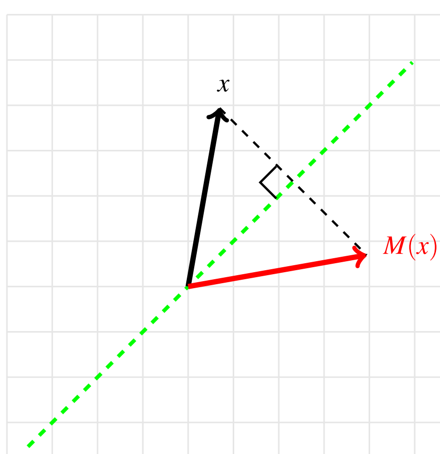


Again, think of the dashed green line as a mirror and let $M : \mathbb{R}^2 \to \mathbb{R}^2$ be the vector function that maps a vector to its mirror image. Evaluate (by examining the picture)

- $M(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) =$.
- $M(\begin{pmatrix} 0 \\ 3 \end{pmatrix}) =$.
- $M(\begin{pmatrix} 1 \\ 2 \end{pmatrix}) =$.

☛ SEE ANSWER

Homework 2.4.4.2 A reflection with respect to a 45 degree line is illustrated by

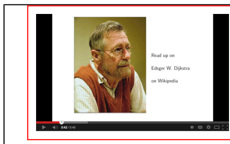


Again, think of the dashed green line as a mirror and let $M : \mathbb{R}^2 \to \mathbb{R}^2$ be the vector function that maps a vector to its mirror image. Compute the matrix that represents M (by examining the picture).

☛ SEE ANSWER

2.5 Enrichment

2.5.1 The Importance of the Principle of Mathematical Induction for Programming



☛ YouTube

Read the ACM Turing Lecture 1972 (Turing Award acceptance speech) by Edsger W. Dijkstra:

The Humble Programmer.

Now, to see how the foundations we teach in this class can take you to the frontier of computer science, I encourage you to download (for free)

The Science of Programming Matrix Computations

Skip the first chapter. Go directly to the second chapter. For now, read ONLY that chapter!

Here are the major points as they relate to this class:

- Last week, we introduced you to a notation for expressing algorithms that builds on slicing and dicing vectors.
- This week, we introduced you to the Principle of Mathematical Induction.
- In Chapter 2 of "The Science of Programming Matrix Computations", we
  - Show how Mathematical Induction is related to computations by a loop.
  - How one can use Mathematical Induction to prove the correctness of a loop.
    (No more debugging! You prove it correct like you prove a theorem to be true.)
  - show how one can systematically derive algorithms to be correct. As Dijkstra said:

    > Today [back in 1972, but still in 2014] a usual technique is to make a program and then to test it. But: program testing can be a very effective way to show the presence of bugs, but is hopelessly inadequate for showing their absence. The only effective way to raise the confidence level of a program significantly is to give a convincing proof of its correctness. But one should not first make the program and then prove its correctness, because then the requirement of providing the proof would only increase the poor programmer's burden. On the contrary: the programmer should let correctness proof and program grow hand in hand.

    To our knowledge, for more complex programs that involve loops, we are unique in having made this comment of Dijkstra's practical. (We have practical libraries with hundreds of thousands of lines of code that have been derived to be correct.)

Teaching you these techniques as part of this course would take the course in a very different direction. So, if this interests you, you should pursue this further on your own.

2.5.2 Puzzles and Paradoxes in Mathematical Induction

Read the article "Puzzles and Paradoxes in Mathematical Induction" by Adam Bjorndahl.

2.6 Wrap Up

2.6.1 Homework

Homework 2.6.1.1 Suppose a professor decides to assign grades based on two exams and a final. Either all three exams (worth 100 points each) are equally weighted or the final is double weighted to replace one of the exams to benefit the student. The records indicate each score on the first exam as χ_0, the score on the second as χ_1, and the score on the final as χ_2. The professor transforms these scores and looks for the maximum entry. The following describes the linear transformation:

$$l(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \chi_1 + \chi_2 \\ \chi_0 + 2\chi_2 \\ \chi_1 + 2\chi_2 \end{pmatrix}$$

What is the matrix that corresponds to this linear transformation?

If a student's scores are $\begin{pmatrix} 68 \\ 80 \\ 95 \end{pmatrix}$, what is the transformed score?

☛ SEE ANSWER

2.6.2 Summary

A linear transformation is a vector function that has the following two properties:

- Transforming a scaled vector is the same as scaling the transformed vector:

$$L(\alpha x) = \alpha L(x)$$

- Transforming the sum of two vectors is the same as summing the two transformed vectors:

$$L(x + y) = L(x) + L(y)$$

$L : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a linear transformation if and only if (iff) for all $u, v \in \mathbb{R}^n$ and $\alpha, \beta \in \mathbb{R}$

$$L(\alpha u + \beta v) = \alpha L(u) + \beta L(v).$$

If $L : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a linear transformation, then

$$L(\beta_0 x_0 + \beta_1 x_1 + \cdots + \beta_{k-1} x_{k-1}) = \beta_0 L(x_0) + \beta_1 L(x_1) + \cdots + \beta_{k-1} L(x_{k-1}).$$

A vector function $L : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a linear transformation if and only if it can be represented by an $m \times n$ **matrix**, which is a very special two dimensional array of numbers (elements).

The **set of all real valued** $m \times n$ **matrices** is denoted by $\mathbb{R}^{m \times n}$.

Let A is the matrix that represents $L : \mathbb{R}^n \to \mathbb{R}^m$, $x \in \mathbb{R}^n$, and let

$$\begin{array}{rcll} A & = & \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) & (a_j \text{ equals the } j\text{th column of } A) \\ & = & \left(\begin{array}{cccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{array} \right) & (\alpha_{i,j} \text{ equals the } (i,j) \text{ element of } A). \\ x & = & \left(\begin{array}{c} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{array} \right) & \end{array}$$

Then

- $A \in \mathbb{R}^{m \times n}$.
- $a_j = L(e_j) = Ae_j$ (the jth column of A is the vector that results from transforming the unit basis vector e_j).
- $L(x) = L(\sum_{j=0}^{n-1} \chi_j e_j) = \sum_{j=0}^{n-1} L(\chi_j e_j) = \sum_{j=0}^{n-1} \chi_j L(e_j) = \sum_{j=0}^{n-1} \chi_j a_j$.
- $$\begin{array}{rcl} Ax & = & L(x) \\ & = & \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) \left(\begin{array}{c} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{array} \right) \\ & = & \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} \\ & = & \chi_0 \left(\begin{array}{c} \alpha_{0,0} \\ \alpha_{1,0} \\ \vdots \\ \alpha_{m-1,0} \end{array} \right) + \chi_1 \left(\begin{array}{c} \alpha_{0,1} \\ \alpha_{1,1} \\ \vdots \\ \alpha_{m-1,1} \end{array} \right) + \cdots + \chi_{n-1} \left(\begin{array}{c} \alpha_{0,n-1} \\ \alpha_{1,n-1} \\ \vdots \\ \alpha_{m-1,n-1} \end{array} \right) \\ & = & \left(\begin{array}{c} \chi_0 \alpha_{0,0} + \chi_1 \alpha_{0,1} + \cdots + \chi_{n-1} \alpha_{0,n-1} \\ \chi_0 \alpha_{1,0} + \chi_1 \alpha_{1,1} + \cdots + \chi_{n-1} \alpha_{1,n-1} \\ \vdots \\ \chi_0 \alpha_{m-1,0} + \chi_1 \alpha_{m-1,1} + \cdots + \chi_{n-1} \alpha_{m-1,n-1} \end{array} \right) \\ & = & \left(\begin{array}{cccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{array} \right) \left(\begin{array}{c} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{array} \right). \end{array}$$

How to check if a vector function is a linear transformation:

- Check if $f(0) = 0$. If it isn't, it is **not** a linear transformation.
- If $f(0) = 0$ then *either*:

- Prove it is or isn't a linear transformation from the definition:
  - Find an example where $f(\alpha x) \neq \alpha f(x)$ or $f(x+y) \neq f(x)+f(y)$. In this case the function is *not* a linear transformation; or
  - Prove that $f(\alpha x) = \alpha f(x)$ and $f(x+y) = f(x)+f(y)$ for all α, x, y.

  or
- Compute the *possible* matrix A that represents it and see if $f(x) = Ax$. If it is equal, it is a linear transformation. If it is not, it is not a linear transformation.

Mathematical induction is a powerful proof technique about natural numbers. (There are more general forms of mathematical induction that we will not need in our course.)

The following results about summations will be used in future weeks:

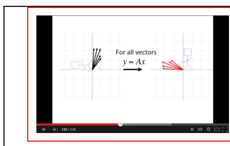
- $\sum_{i=0}^{n-1} i = n(n-1)/2 \approx n^2/2$.
- $\sum_{i=1}^{n} i = n(n+1)/2 \approx n^2/2$.
- $\sum_{i=0}^{n-1} i^2 = (n-1)n(2n-1)/6 \approx \frac{1}{3}n^3$.

Week 3

Matrix-Vector Operations

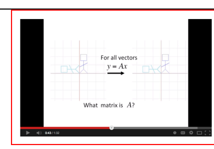
3.1 Opening Remarks

3.1.1 Timmy Two Space



☛ YouTube

Homework 3.1.1.1 Click on the below link to open a browser window with the "Timmy Two Space" exercise. This exercise was suggested to us by our colleague Prof. Alan Cline. It was first implemented using an IPython Notebook by Ben Holder. During the Spring 2014 offering of LAFF on the edX platform, one of the participants, Ed McCardell, rewrote the activity as ☛ Local copy. (If this link does not work, open `LAFF-2.0xM/Timmy/index.html`).
If you get really frustrated, here is a hint:



☛ YouTube

☛ SEE ANSWER

3.1.2 Outline Week 3

3.1.3 What You Will Learn

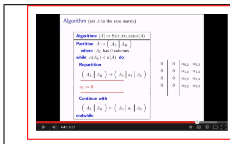
Upon completion of this unit, you should be able to

- Recognize matrix-vector multiplication as a linear combination of the columns of the matrix.
- Given a linear transformation, determine the matrix that represents it.
- Given a matrix, determine the linear transformation that it represents.
- Connect special linear transformations to special matrices.
- Identify special matrices such as the zero matrix, the identity matrix, diagonal matrices, triangular matrices, and symmetric matrices.
- Transpose a matrix.
- Scale and add matrices.
- Exploit properties of special matrices.
- Extrapolate from concrete computation to algorithms for matrix-vector multiplication.
- Partition (slice and dice) matrices with and without special properties.
- Use partitioned matrices and vectors to represent algorithms for matrix-vector multiplication.
- Use partitioned matrices and vectors to represent algorithms in code.

Track your progress in Appendix B.

3.2 Special Matrices

3.2.1 The Zero Matrix



☛ YouTube

Homework 3.2.1.1 Let $L_0 : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be the function defined for every $x \in \mathbb{R}^n$ as $L_0(x) = 0$, where 0 denotes the zero vector "of appropriate size". L_0 is a linear transformation.

True/False

☛ SEE ANSWER

We will denote the matrix that represents L_0 by 0, where we typically know what its row and column sizes are from context (in this case, $0 \in \mathbb{R}^{m \times n}$). If it is not obvious, we may use a subscript ($0_{m \times n}$) to indicate its size, that is, m rows and n columns.

By the definition of a matrix, the jth column of matrix 0 is given by $L_0(e_j) = 0$ (a vector with m zero components). Thus, the matrix that represents L_0, which we will call the zero matrix, is given by the $m \times n$ matrix

$$0 = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}.$$

It is easy to check that for any $x \in \mathbb{R}^n$, $0_{m \times n} x_n = 0_m$.

Definition 3.1 *A matrix $A \in \mathbb{R}^{m \times n}$ equals the $m \times n$ zero matrix if all of its elements equal zero.*

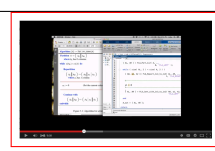
Througout this course, we will use the number 0 to indicate a scalar, vector, or matrix of "appropriate size".

In Figure 3.1, we give an algorithm that, given an $m \times n$ matrix A, sets it to zero. Notice that it exposes columns one at a time, setting the exposed column to zero.

MATLAB provides the function "zeros" that returns a zero matrix of indicated size. Your are going to write your own, to helps you understand the material.

Make sure that the path to the `laff` subdirectory is added in MATLAB, so that the various routines form the laff library that we are about to use will be found by MATLAB. How to do this was discussed in Unit 1.6.3.

Homework 3.2.1.2 With the FLAME API for MATLAB (FLAME@lab) implement the algorithm in Figure 3.1. You will use the function `laff_zerov( x )`, which returns a zero vector of the same size and shape (column or row) as input vector `x`. Since you are still getting used to programming with M-script and FLAME@lab, you may want to follow the instructions in this video:



☛ YouTube

Some links that will come in handy:

- ☛ Spark
  (alternatively, open the file ☛ `LAFF-2.0xM/Spark/index.html`)
- ☛ PictureFLAME
  (alternatively, open the file ☛ `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

You will need these in many future exercises. Bookmark them!

☛ SEE ANSWER

Algorithm: $[A] := \text{SET\_TO\_ZERO}(A)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$
where A_L has 0 columns

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right)$$

where a_1 has 1 column

$a_1 := 0$ (Set the current column to zero)

Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right)$$

endwhile

Figure 3.1: Algorithm for setting matrix A to the zero matrix.

Homework 3.2.1.3 In the MATLAB Command Window, type

```
A = zeros( 5,4 )
```

What is the result?

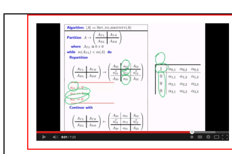
☛ SEE ANSWER

Homework 3.2.1.4 Apply the zero matrix to Timmy Two Space. What happens?

1. Timmy shifts off the grid.
2. Timmy disappears into the origin.
3. Timmy becomes a line on the x-axis.
4. Timmy becomes a line on the y-axis.
5. Timmy doesn't change at all.

☛ SEE ANSWER

3.2.2 The Identity Matrix



☛ YouTube

Homework 3.2.2.1 Let $L_I : \mathbb{R}^n \rightarrow \mathbb{R}^n$ be the function defined for every $x \in \mathbb{R}^n$ as $L_I(x) = x$. L_I is a linear transformation.

True/False

☛ SEE ANSWER

We will denote the matrix that represents L_I by the letter I (capital "I") and call it the identity matrix. Usually, the size of the identity matrix is obvious from context. If not, we may use a subscript, I_n, to indicate the size, that is: a matrix that has n rows and n columns (and is hence a "square matrix").

Again, by the definition of a matrix, the jth column of I is given by $L_I(e_j) = e_j$. Thus, the identity matrix is given by

$$I = \left(\begin{array}{c|c|c|c} e_0 & e_1 & \cdots & e_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{array} \right).$$

Here, and frequently in the future, we use vertical lines to indicate a partitioning of a matrix into its columns. (Slicing and dicing again!) It is easy to check that $Ix = x$.

Definition 3.2 *A matrix $I \in \mathbb{R}^{n \times n}$ equals the $n \times n$ identity matrix if all its elements equal zero, except for the elements on the diagonal, which all equal one.*

The diagonal of a matrix A consists of the entries $\alpha_{0,0}$, $\alpha_{1,1}$, etc. In other words, all elements $\alpha_{i,i}$.

Througout this course, we will use the capital letter I to indicate an identity matrix "of appropriate size".

We now motivate an algorithm that, given an $n \times n$ matrix A, sets it to the identity matrix.

We'll start by trying to closely mirror the `Set_to_zero` algorithm from the previous unit:

Algorithm: $[A] := \text{SET\_TO\_IDENTITY}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_L & A_R \end{array} \right)$
  where A_L has 0 columns

while $n(A_L) < n(A)$ **do**

  Repartition

  $\left(\begin{array}{c|c} A_L & A_R \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array} \right)$
  where a_1 has 1 column

  $a_1 := e_j$ (Set the current column to the correct unit basis vector)

  Continue with

  $\left(\begin{array}{c|c} A_L & A_R \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array} \right)$

endwhile

The problem is that our notation doesn't keep track of the column index, j. Another problem is that we don't have a routine to set a vector to the jth unit basis vector.

To overcome this, we recognize that the jth column of A, which in our algorithm above appears as a_1, and the jth unit basis vector can each be partitioned into three parts:

$$a_1 = a_j = \begin{pmatrix} a_{01} \\ \alpha_{11} \\ a_{21} \end{pmatrix} \quad \text{and} \quad e_j = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix},$$

where the 0's refer to vectors of zeroes of appropriate size. To then set a_1 $(= a_j)$ to the unit basis vector, we can make the assignments

$$\begin{array}{rcl} a_{01} & := & 0 \\ \alpha_{11} & := & 1 \\ a_{21} & := & 0 \end{array}$$

The algorithm in Figure 3.2 very naturally exposes exactly these parts of the current column.

Algorithm: $[A] := \text{SET\_TO\_IDENTITY}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

set current column to appropriate unit basis vector

$a_{01} := 0$	set a_{01}'s components to zero
$\alpha_{11} := 1$	
$a_{21} := 0$	set a_{21}'s components to zero

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 3.2: Algorithm for setting matrix A to the identity matrix.

Why is it guaranteed that α_{11} refers to the diagonal element of the current column?

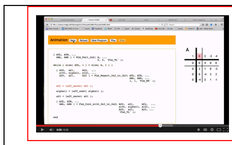
Answer: A_{TL} starts as a 0×0 matrix, and is expanded by a row and a column in every iteration. Hence, it is always square. This guarantees that α_{11} is on the diagonal.

MATLAB provides the routine "eye" that returns an identity matrix of indicated size. But we will write our own.

Homework 3.2.2.2 With the FLAME API for MATLAB (FLAME@lab) implement the algorithm in Figure 3.2. You will use the functions `laff_zerov( x )` and `laff_onev( x )`, which return a zero vector and vector of all ones of the same size and shape (column or row) as input vector `x`, respectively. Try it yourself! (Hint: in Spark, you will want to pick Direction `TL->BR`.) Feel free to look at the below video if you get stuck.
Some links that will come in handy:

- ☛ Spark
  (alternatively, open the file ☛ `LAFF-2.0xM/Spark/index.html`)
- ☛ PictureFLAME
  (alternatively, open the file ☛ `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

You will need these in many future exercises. Bookmark them!



☛ **YouTube**

☛ SEE ANSWER

Homework 3.2.2.3 In the MATLAB Command Window, type

```
A = eye( 4,4 )
```

What is the result?

☛ SEE ANSWER

Homework 3.2.2.4 Apply the identity matrix to Timmy Two Space. What happens?

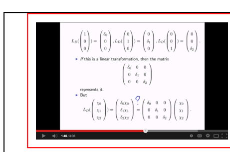
1. Timmy shifts off the grid.
2. Timmy disappears into the origin.
3. Timmy becomes a line on the x-axis.
4. Timmy becomes a line on the y-axis.
5. Timmy doesn't change at all.

☛ SEE ANSWER

Homework 3.2.2.5 The trace of a matrix equals the sum of the diagonal elements. What is the trace of the identity $I \in \mathbb{R}^{n\times n}$?

☛ SEE ANSWER

3.2.3 Diagonal Matrices



☛ **YouTube**

Let $L_D : \mathbb{R}^n \to \mathbb{R}^n$ be the function defined for every $x \in \mathbb{R}^n$ as

$$L\left(\begin{pmatrix}\chi_0\\ \chi_1\\ \vdots\\ \chi_{n-1}\end{pmatrix}\right) = \left(\begin{pmatrix}\delta_0\chi_0\\ \delta_1\chi_1\\ \vdots\\ \delta_{n-1}\chi_{n-1}\end{pmatrix}\right),$$

where $\delta_0, \ldots, \delta_{n-1}$ are constants.

Here, we will denote the matrix that represents L_D by the letter D. Once again, by the definition of a matrix, the jth column of D is given by

$$L_D(e_j) = L_D(\begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}) = \begin{pmatrix} \delta_0 \times 0 \\ \vdots \\ \delta_{j-1} \times 0 \\ \delta_j \times 1 \\ \delta_{j+1} \times 0 \\ \vdots \\ \delta_{n-1} \times 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \delta_j \times 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \delta_j \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \delta_j e_j.$$

This means that

$$D = \left(\begin{array}{c|c|c|c} \delta_0 e_0 & \delta_1 e_1 & \cdots & \delta_{n-1} e_{n-1} \end{array} \right) = \begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}.$$

Definition 3.3 *A matrix $A \in \mathbb{R}^{n \times n}$ is said to be diagonal if $\alpha_{i,j} = 0$ for all $i \neq j$ so that*

$$A = \begin{pmatrix} \alpha_{0,0} & 0 & \cdots & 0 \\ 0 & \alpha_{1,1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_{n-1,n-1} \end{pmatrix}.$$

Homework 3.2.3.1 Let $A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix}$ and $x = \begin{pmatrix} 2 \\ 1 \\ -2 \end{pmatrix}$. Evaluate Ax.

☛ SEE ANSWER

Homework 3.2.3.2 Let $D = \begin{pmatrix} 2 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & -1 \end{pmatrix}$. What linear transformation, L, does this matrix represent? In particular, answer the following questions:

- $L : \mathbb{R}^n \to \mathbb{R}^m$. What are m and n?
- A linear transformation can be described by how it transforms the unit basis vectors:

$$L(e_0) = \begin{pmatrix} \\ \\ \\ \end{pmatrix}; L(e_1) = \begin{pmatrix} \\ \\ \\ \end{pmatrix}; L(e_2) = \begin{pmatrix} \\ \\ \\ \end{pmatrix}$$

- $L(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \\ \\ \\ \end{pmatrix}$

☛ SEE ANSWER

Algorithm: $[A] := \text{SET\_TO\_DIAGONAL\_MATRIX}(A, x)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$

where A_{TL} is 0×0, x_T has 0 elements

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right)$$

where α_{11} is 1×1, χ_1 is a scalar

$a_{01} := 0$

$\alpha_{11} := \chi_1$

$a_{21} := 0$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right)$$

endwhile

Figure 3.3: Algorithm that sets A to a diagonal matrix with the entries of x on its diagonal.

An algorithm that sets a given square matrix A to a diagonal matrix that has as its ith diagonal entry the ith entry of vector x ig given in Figure 3.3.

Homework 3.2.3.3 Implement a function

```
[ A_out ] = DiagonalMatrix_unb( A, x )
```

based on Figure 3.3.

☛ SEE ANSWER

Homework 3.2.3.4 In the MATLAB Command Window, type

```
x = [ -1; 2; -3 ]
A = diag( x )
```

What is the result?

☛ SEE ANSWER

In linear algebra an element-wise vector-vector product is not a meaningful operation: when $x, y \in \mathbb{R}^n$ the product xy has no meaning. However, MATLAB has an "element-wise multiplication" operator "`.*`". Try

```
x = [-1; 2; -3]
y = [1; -1; 2]
x .* y
diag( x ) * y
```

Conclude that element-wise multiplication by a vector is the same as multiplication by a diagonal matrix with diagonal elements equal to the elements of that vector.

Homework 3.2.3.5 Apply the diagonal matrix $\begin{pmatrix} -1 & 0 \\ 0 & 2 \end{pmatrix}$ to Timmy Two Space. What happens?

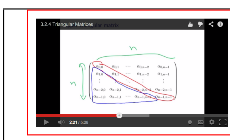
1. Timmy shifts off the grid.
2. Timmy is rotated.
3. Timmy doesn't change at all.
4. Timmy is flipped with respect to the vertical axis.
5. Timmy is stretched by a factor two in the vertical direction.

☛ SEE ANSWER

Homework 3.2.3.6 Compute the trace of $\begin{pmatrix} -1 & 0 \\ 0 & 2 \end{pmatrix}$.

☛ SEE ANSWER

3.2.4 Triangular Matrices



☛ YouTube

Homework 3.2.4.1 Let $L_U : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ be defined as $L_U(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} 2\chi_0 - \chi_1 + \chi_2 \\ 3\chi_1 - \chi_2 \\ -2\chi_2 \end{pmatrix}$. We have proven for similar functions that they are linear transformations, so we will skip that part. What matrix, U, represents this linear transformation?

☛ SEE ANSWER

A matrix like U in the above practice is called a triangular matrix. In particular, it is an *upper* triangular matrix.

The following defines a number of different special cases of triangular matrices:

Definition 3.4 (Triangular matrix)
A matrix $A \in \mathbb{R}^{n\times n}$ is said to be

lower triangular	$\alpha_{i,j} = 0$ *if* $i < j$	$\left(\begin{array}{ccccc} \alpha_{0,0} & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{array}\right)$
strictly lower triangular	$\alpha_{i,j} = 0$ *if* $i \leq j$	$\left(\begin{array}{ccccc} 0 & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & 0 & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 0 \end{array}\right)$
unit lower triangular	$\alpha_{i,j} = \begin{cases} 0 & \text{if } i < j \\ 1 & \text{if } i = j \end{cases}$	$\left(\begin{array}{ccccc} 1 & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & 1 & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 1 \end{array}\right)$
upper triangular	$\alpha_{i,j} = 0$ *if* $i > j$	$\left(\begin{array}{ccccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \alpha_{n-2,n-2} & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & \alpha_{n-1,n-1} \end{array}\right)$
strictly upper triangular	$\alpha_{i,j} = 0$ *if* $i \geq j$	$\left(\begin{array}{ccccc} 0 & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & 0 & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & 0 \end{array}\right)$
unit upper triangular	$\alpha_{i,j} = \begin{cases} 0 & \text{if } i > j \\ 1 & \text{if } i = j \end{cases}$	$\left(\begin{array}{ccccc} 1 & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & 1 & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & 1 \end{array}\right)$

If a matrix is either lower or upper triangular, it is said to be triangular.

Homework 3.2.4.2 A matrix that is both lower and upper triangular is, in fact, a diagonal matrix.
Always/Sometimes/Never
☛ SEE ANSWER

Homework 3.2.4.3 A matrix that is both strictly lower and strictly upper triangular is, in fact, a zero matrix.
Always/Sometimes/Never
☛ SEE ANSWER

The algorithm in Figure 3.4 sets a given matrix $A \in \mathbb{R}^{n\times n}$ to its lower triangular part (zeroing the elements above the diagonal).

Homework 3.2.4.4 In the above algorithm you could have replaced $a_{01} := 0$ with $a_{12}^T := 0$.
Always/Sometimes/Never
☛ SEE ANSWER

Algorithm: $[A] := \text{SET\_TO\_LOWER\_TRIANGULAR\_MATRIX}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

set the elements of the current column above the diagonal to zero

$a_{01} := 0$ set a_{01}'s components to zero

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 3.4: Algorithm for making a matrix A a lower triangular matrix by setting the entries above the diagonal to zero.

Homework 3.2.4.5 Consider the following algorithm.

Algorithm: $[A] := \text{SET\_TO\_???\_TRIANGULAR\_MATRIX}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

?????

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Change the ????? in the above algorithm so that it sets A to its

The MATLAB functions `tril` and `triu`, when given an $n \times n$ matrix `A`, return the lower and upper triangular parts of A, respectively. The strictly lower and strictly upper triangular parts of `A` can be extracted by the calls `tril( A, -1 )` and `triu( A, 1 )`, respectively. We now write our own routines that sets the appropriate entries in a matrix to zero.

Homework 3.2.4.6 Implement functions for each of the algorithms from the last homework. In other words, implement functions that, given a matrix A, return a matrix equal to

- the upper triangular part. (`Set_to_upper_triangular_matrix`)
- the strictly upper triangular part. (`Set_to_strictly_upper_triangular_matrix`)
- the unit upper triangular part. (`Set_to_unit_upper_triangular_matrix`)
- strictly lower triangular part. (`Set_to_strictly_lower_triangular_matrix`)
- unit lower triangular part. (`Set_to_unit_lower_triangular_matrix`)

(Implement as many as you enjoy implementing. Then move on.)

☛ SEE ANSWER

Homework 3.2.4.7 In MATLAB try this:

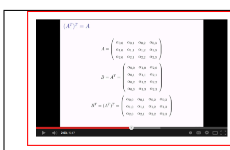
```
A = [ 1,2,3;4,5,6;7,8,9 ]
tril( A )
tril( A, -1 )
tril( A, -1 ) + eye( size( A ) )
triu( A )
triu( A, 1 )
triu( A, 1 ) + eye( size( A ) )
```

Homework 3.2.4.8 Apply $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ to Timmy Two Space. What happens to Timmy?

1. Timmy shifts off the grid.
2. Timmy becomes a line on the x-axis.
3. Timmy becomes a line on the y-axis.
4. Timmy is skewed to the right.
5. Timmy doesn't change at all.

☛ SEE ANSWER

3.2.5 Transpose Matrix



☛ YouTube

Definition 3.5 *Let $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times m}$. Then B is said to be the transpose of A if, for $0 \leq i < m$ and $0 \leq j < n$, $\beta_{j,i} = \alpha_{i,j}$. The transpose of a matrix A is denoted by A^T so that $B = A^T$.*

We have already used ^{T} to indicate a row vector, which is consistent with the above definition: it is a column vector that has been transposed.

Homework 3.2.5.1 Let $A = \begin{pmatrix} -1 & 0 & 2 & 1 \\ 2 & -1 & 1 & 2 \\ 3 & 1 & -1 & 3 \end{pmatrix}$ and $x = \begin{pmatrix} -1 \\ 2 \\ 4 \end{pmatrix}$. What are A^T and x^T?

☛ SEE ANSWER

Clearly, $(A^T)^T = A$.

Notice that the columns of matrix A become the rows of matrix A^T. Similarly, the rows of matrix A become the columns of matrix A^T.

The following algorithm sets a given matrix $B \in \mathbb{R}^{n \times m}$ to the transpose of a given matrix $A \in \mathbb{R}^{m \times n}$:

Algorithm: $[B] := \text{TRANSPOSE}(A, B)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right)$

where A_L has 0 columns, B_T has 0 rows

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ B_2 \end{array} \right)$$

where a_1 has 1 column, b_1 has 1 row

$b_1^T := a_1^T$ (Set the current row of B to the current column of A)

Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,|\, a_1 \,\middle|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \leftarrow \left(\begin{array}{c} B_0 \\ b_1^T \\ \hline B_2 \end{array} \right)$$

endwhile

The ^{T} in b_1^T is part of indicating that b_1^T is a row. The ^{T} in a_1^T in the assignment changes the column vector a_1 into a row vector so that it can be assigned to b_1^T.

Homework 3.2.5.2 Consider the following algorithm.

Algorithm: $[B] := \text{TRANSPOSE\_ALTERNATIVE}(A, B)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $B \rightarrow \left(\begin{array}{c|c} B_L & B_R \end{array}\right)$

where A_T has 0 rows, B_L has 0 columns

while $m(A_T) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c|c} B_L & B_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} B_0 & b_1 & B_2 \end{array}\right)$$

where a_1 has 1 row, b_1 has 1 column

Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c|c} B_L & B_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} B_0 & b_1 & B_2 \end{array}\right)$$

endwhile

Modify the above algorithm so that it copies rows of A into columns of B.

☛ SEE ANSWER

Homework 3.2.5.3 Implement functions

- `Transpose_unb( A, B )`
- `Transpose_alternative_unb( A, B )`

☛ SEE ANSWER

Homework 3.2.5.4 The transpose of a lower triangular matrix is an upper triangular matrix.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.2.5.5 The transpose of a strictly upper triangular matrix is a strictly lower triangular matrix.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.2.5.6 The transpose of the identity is the identity.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.2.5.7 Evaluate

- $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}^T =$
- $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}^T =$

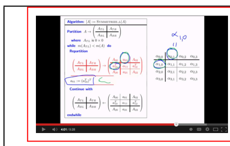
☛ SEE ANSWER

Homework 3.2.5.8 If $A = A^T$ then $A = I$ (the identity).

True/False

☛ SEE ANSWER

3.2.6 Symmetric Matrices



☛ YouTube

A matrix $A \in \mathbb{R}^{n\times n}$ is said to be symmetric if $A = A^T$.

In other words, if $A \in \mathbb{R}^{n\times n}$ is symmetric, then $\alpha_{i,j} = \alpha_{j,i}$ for all $0 \leq i, j < n$. Another way of expressing this is that

$$A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{0,n-2} & \alpha_{1,n-2} & \cdots & \alpha_{n-2,n-2} & \alpha_{n-2,n-1} \\ \alpha_{0,n-1} & \alpha_{1,n-1} & \cdots & \alpha_{n-2,n-1} & \alpha_{n-1,n-1} \end{pmatrix}$$

and

$$A = \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{n-2,0} & \alpha_{n-1,0} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{n-2,1} & \alpha_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & \alpha_{n-1,n-2} \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{pmatrix}.$$

Homework 3.2.6.1 Assume the below matrices are symmetric. Fill in the remaining elements.

$$\begin{pmatrix} 2 & \square & -1 \\ -2 & 1 & -3 \\ \square & \square & -1 \end{pmatrix}; \quad \begin{pmatrix} 2 & \square & \square \\ -2 & 1 & \square \\ -1 & 3 & -1 \end{pmatrix}; \quad \begin{pmatrix} 2 & 1 & -1 \\ \square & 1 & -3 \\ \square & \square & -1 \end{pmatrix}.$$

☛ SEE ANSWER

Homework 3.2.6.2 A triangular matrix that is also symmetric is, in fact, a diagonal matrix. Always/Sometimes/Never

☛ SEE ANSWER

The nice thing about symmetric matrices is that only approximately half of the entries need to be stored. Often, only the lower triangular or only the upper triangular part of a symmetric matrix is stored. Indeed: Let A be symmetric, let L be the lower triangular matrix stored in the lower triangular part of A, and let $\widetilde{L}$ is the strictly lower triangular matrix stored in the strictly lower triangular part of A. Then $A = L + \widetilde{L}^T$:

$$
\begin{aligned}
A &= \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{n-2,0} & \alpha_{n-1,0} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{n-2,1} & \alpha_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & \alpha_{n-1,n-2} \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0} & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{pmatrix} + \begin{pmatrix} 0 & \alpha_{1,0} & \cdots & \alpha_{n-2,0} & \alpha_{n-1,0} \\ 0 & 0 & \cdots & \alpha_{n-2,1} & \alpha_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \alpha_{n-1,n-2} \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0} & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{pmatrix} + \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & 0 & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 0 \end{pmatrix}^T .
\end{aligned}
$$

Let A be symmetric and assume that $A = L + \widetilde{L}^T$ as discussed above. Assume that only L is stored in A and that we would like to also set the upper triangular parts of A to their correct values (in other words, set the strictly upper triangular part of A to $\widetilde{L}$). The following algorithm performs this operation, which we will call "symmetrizing" A:

Algorithm: $[A] := \text{SYMMETRIZE\_FROM\_LOWER\_TRIANGLE}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

(set a_{01}'s components to their symmetric parts below the diagonal)

$a_{01} := (a_{10}^T)^T$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Homework 3.2.6.3 In the above algorithm one can replace $a_{01} := a_{10}^T$ by $a_{12}^T = a_{21}$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.2.6.4 Consider the following algorithm.

Algorithm: $[A] := \text{SYMMETRIZE\_FROM\_UPPER\_TRIANGLE}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

?????

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

What commands need to be introduced between the lines in order to "symmetrize" A assuming that only its upper triangular part is stored initially.

☛ SEE ANSWER

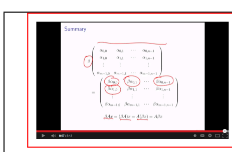
Homework 3.2.6.5 Implement functions

- `Symmetrize_from_lower_triangle_unb( A, B )`
- `Symmetrize_from_upper_triangle_unb( A, B )`

☛ SEE ANSWER

3.3 Operations with Matrices

3.3.1 Scaling a Matrix



☛ **YouTube**

Theorem 3.6 *Let $L_A : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a linear transformation and, for all $x \in \mathbb{R}^n$, define the function $L_B : \mathbb{R}^n \rightarrow \mathbb{R}^m$ by $L_B(x) = \beta L_A(x)$, where β is a scalar. Then $L_B(x)$ is a linear transformation.*

Homework 3.3.1.1 Prove the above theorem.

☛ SEE ANSWER

Let A be the matrix that represents L_A. Then, for all $x \in \mathbb{R}^n$, $\beta(Ax) = \beta L_A(x) = L_B(x)$. Since L_B is a linear transformation, there should be a matrix B such that, for all $x \in \mathbb{R}^n$, $Bx = L_B(x) = \beta(Ax)$. Recall that $b_j = Be_j$, the jth column of B. Thus, $b_j = Be_j = \beta(Ae_j) = \beta a_j$, where a_j equals the jth column of A. We conclude that B is computed from A by scaling each column by β. But that simply means that each element of B is scaled by β.

The above motivates the following definition.

If $A \in \mathbb{R}^{m\times n}$ and $\beta \in \mathbb{R}$, then

$$\beta \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} = \begin{pmatrix} \beta\alpha_{0,0} & \beta\alpha_{0,1} & \cdots & \beta\alpha_{0,n-1} \\ \beta\alpha_{1,0} & \beta\alpha_{1,1} & \cdots & \beta\alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta\alpha_{m-1,0} & \beta\alpha_{m-1,1} & \cdots & \beta\alpha_{m-1,n-1} \end{pmatrix}.$$

An alternative motivation for this definition is to consider

$$\begin{aligned}
\beta(Ax) &= \beta \begin{pmatrix} \alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + & \cdots + & \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + & \cdots + & \alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + & \cdots + & \alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \beta(\alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + & \cdots + & \alpha_{0,n-1}\chi_{n-1}) \\ \beta(\alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + & \cdots + & \alpha_{1,n-1}\chi_{n-1}) \\ \vdots & \vdots & \vdots & \vdots \\ \beta(\alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + & \cdots + & \alpha_{m-1,n-1}\chi_{n-1}) \end{pmatrix} \\
&= \begin{pmatrix} \beta\alpha_{0,0}\chi_0 + & \beta\alpha_{0,1}\chi_1 + & \cdots + & \beta\alpha_{0,n-1}\chi_{n-1} \\ \beta\alpha_{1,0}\chi_0 + & \beta\alpha_{1,1}\chi_1 + & \cdots + & \beta\alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \beta\alpha_{m-1,0}\chi_0 + & \beta\alpha_{m-1,1}\chi_1 + & \cdots + & \beta\alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \beta\alpha_{0,0} & \beta\alpha_{0,1} & \cdots & \beta\alpha_{0,n-1} \\ \beta\alpha_{1,0} & \beta\alpha_{1,1} & \cdots & \beta\alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \beta\alpha_{m-1,0} & \beta\alpha_{m-1,1} & \cdots & \beta\alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = (\beta A)x.
\end{aligned}$$

Since, by design, $\beta(Ax) = (\beta A)x$ we can drop the parentheses and write βAx (which also equals $A(\beta x)$ since $L(x) = Ax$ is a linear transformation).

Given matrices $\beta \in \mathbb{R}$ and $A \in \mathbb{R}^{m\times n}$, the following algorithm scales A by β.

Algorithm: $[A] := \text{SCALE\_MATRIX}(\beta, A)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$
  where A_L has 0 columns

while $n(A_L) < n(A)$ **do**

  Repartition

  $\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right)$
  where a_1 has 1 column

$a_1 := \beta a_1$ (Scale the current column of A)

  Continue with

  $\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right)$

endwhile

Homework 3.3.1.2 Consider the following algorithm.

Algorithm: $[A] := \text{SCALE\_MATRIX\_ALTERNATIVE}(\beta, A)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right)$
  where A_T has 0 rows

while $m(A_T) < m(A)$ **do**

  Repartition

  $\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array} \right)$
  where a_1 has 1 row

?????

  Continue with

  $\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \leftarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array} \right)$

endwhile

What update will scale A one row at a time?

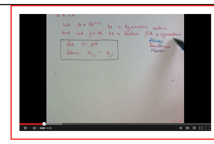
☛ SEE ANSWER

With MATLAB, when `beta` is a scalar and `A` is a matrix, the simple command `beta * A` will scale A by beta.

Homework 3.3.1.3 Implement function `Scale_matrix_unb( beta, A )`.

☛ SEE ANSWER

Homework 3.3.1.4



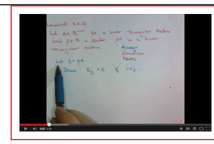
☛ YouTube

Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix and $\beta \in \mathbb{R}$ a scalar, βA is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.1.5



☛ YouTube

Let $A \in \mathbb{R}^{n\times n}$ be a lower triangular matrix and $\beta \in \mathbb{R}$ a scalar, βA is a lower triangular matrix.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.1.6 Let $A \in \mathbb{R}^{n\times n}$ be a diagonal matrix and $\beta \in \mathbb{R}$ a scalar, βA is a diagonal matrix.

Always/Sometimes/Never

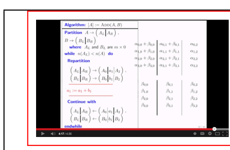
☛ SEE ANSWER

Homework 3.3.1.7 Let $A \in \mathbb{R}^{m\times n}$ be a matrix and $\beta \in \mathbb{R}$ a scalar, $(\beta A)^T = \beta A^T$.

Always/Sometimes/Never

☛ SEE ANSWER

3.3.2 Adding Matrices



☛ YouTube

Homework 3.3.2.1 The sum of two linear transformations is a linear transformation. More formally: Let $L_A : \mathbb{R}^n \to \mathbb{R}^m$ and $L_B : \mathbb{R}^n \to \mathbb{R}^m$ be two linear transformations. Let $L_C : \mathbb{R}^n \to \mathbb{R}^m$ be defined by $L_C(x) = L_A(x) + L_B(x)$. L_C is a linear transformation.

Always/Sometimes/Never

☛ SEE ANSWER

Now, let A, B, and C be the matrices that represent L_A, L_B, and L_C in the above theorem, respectively. Then, for all $x \in \mathbb{R}^n$, $Cx = L_C(x) = L_A(x) + L_B(x)$. What does c_j, the jth column of C, equal?

$$c_j = Ce_j = L_C(e_j) = L_A(e_j) + L_B(e_j) = Ae_j + Be_j = a_j + b_j,$$

where a_j, b_j, and c_j equal the jth columns of A, B, and C, respectively. Thus, the jth column of C equals the sum of the corresponding columns of A and B. That simply means that each element of C equals the sum of the corresponding elements of A and B.

If $A, B \in \mathbb{R}^{m \times n}$, then

$$A + B = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} + \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0} & \beta_{m-1,1} & \cdots & \beta_{m-1,n-1} \end{pmatrix}$$
$$= \begin{pmatrix} \alpha_{0,0}+\beta_{0,0} & \alpha_{0,1}+\beta_{0,1} & \cdots & \alpha_{0,n-1}+\beta_{0,n-1} \\ \alpha_{1,0}+\beta_{1,0} & \alpha_{1,1}+\beta_{1,1} & \cdots & \alpha_{1,n-1}+\beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0}+\beta_{m-1,0} & \alpha_{m-1,1}+\beta_{m-1,1} & \cdots & \alpha_{m-1,n-1}+\beta_{m-1,n-1} \end{pmatrix}.$$

Given matrices $A, B \in \mathbb{R}^{m \times n}$, the following algorithm adds B to A.

Algorithm: $[A] := \text{ADD\_MATRICES}(A, B)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$, $B \rightarrow \left(B_L \,\middle|\, B_R \right)$
  where A_L has 0 columns, B_L has 0 columns

while $n(A_L) < n(A)$ **do**

  Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right), \left(B_L \,\middle|\, B_R \right) \rightarrow \left(B_0 \,\middle|\, b_1 \,\middle|\, B_2 \right)$$

  where a_1 has 1 column, b_1 has 1 column

  $a_1 := a_1 + b_1$ (Add the current column of B to the current column of A)

  Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right), \left(B_L \,\middle|\, B_R \right) \leftarrow \left(B_0 \,\middle|\, b_1 \,\middle|\, B_2 \right)$$

endwhile

Homework 3.3.2.2 Consider the following algorithm.

Algorithm: $[A] := \text{ADD\_MATRICES\_ALTERNATIVE}(A, B)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right)$

where A_T has 0 rows, B_T has 0 rows

while $m(A_T) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array}\right)$$

where a_1 has 1 row, b_1 has 1 row

Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array}\right)$$

endwhile

What update will add B to A one row at a time, overwriting A with the result?

☛ SEE ANSWER

When `A` and `B` are created as matrices of the same size, MATLAB adds two matrices with the simple command `A + B` . We'll just use that when we need it!

Try this! In MATLAB execute

```
A = [ 1,2;3,4;5,6 ]
B = [ -1,2;3,-4;5,6 ]
C = A + B
```

Homework 3.3.2.3 Let $A, B \in \mathbb{R}^{m \times n}$. $A + B = B + A$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.4 Let $A, B, C \in \mathbb{R}^{m \times n}$. $(A + B) + C = A + (B + C)$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.5 Let $A, B \in \mathbb{R}^{m \times n}$ and $\gamma \in \mathbb{R}$. $\gamma(A + B) = \gamma A + \gamma B$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.6 Let $A \in \mathbb{R}^{m \times n}$ and $\beta, \gamma \in \mathbb{R}$. $(\beta + \gamma)A = \beta A + \gamma A$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.7 Let $A, B \in \mathbb{R}^{n \times n}$. $(A + B)^T = A^T + B^T$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.8 Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric matrices. $A+B$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.9 Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric matrices. $A-B$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.10 Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric matrices and $\alpha, \beta \in \mathbb{R}$. $\alpha A+\beta B$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 3.3.2.11 Let $A, B \in \mathbb{R}^{n\times n}$.

If A and B are lower triangular matrices then $A+B$ is lower triangular.

True/False

If A and B are strictly lower triangular matrices then $A+B$ is strictly lower triangular.

True/False

If A and B are unit lower triangular matrices then $A+B$ is unit lower triangular.

True/False

If A and B are upper triangular matrices then $A+B$ is upper triangular.

True/False

If A and B are strictly upper triangular matrices then $A+B$ is strictly upper triangular.

True/False

If A and B are unit upper triangular matrices then $A+B$ is unit upper triangular.

True/False

☛ SEE ANSWER

Homework 3.3.2.12 Let $A, B \in \mathbb{R}^{n\times n}$.

If A and B are lower triangular matrices then $A-B$ is lower triangular.

True/False

If A and B are strictly lower triangular matrices then $A-B$ is strictly lower triangular. True/False

If A and B are unit lower triangular matrices then $A-B$ is *strictly* lower triangular. True/False

If A and B are upper triangular matrices then $A-B$ is upper triangular. True/False

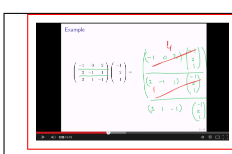
If A and B are strictly upper triangular matrices then $A-B$ is strictly upper triangular. True/False

If A and B are unit upper triangular matrices then $A-B$ is unit upper triangular. True/False

☛ SEE ANSWER

3.4 Matrix-Vector Multiplication Algorithms

3.4.1 Via Dot Products



☛ YouTube

Motivation

Recall that if $y = Ax$, where $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, and $y \in \mathbb{R}^m$, then

$$y = \left(\begin{array}{c}\psi_0\\ \psi_1\\ \vdots\\ \psi_{m-1}\end{array}\right) = \left(\begin{array}{rrrr}\alpha_{0,0}\chi_0+ & \alpha_{0,1}\chi_1+ & \cdots+ & \alpha_{0,n-1}\chi_{n-1}\\ \alpha_{1,0}\chi_0+ & \alpha_{1,1}\chi_1+ & \cdots+ & \alpha_{1,n-1}\chi_{n-1}\\ \vdots & \vdots & \vdots & \vdots\\ \alpha_{m-1,0}\chi_0+ & \alpha_{m-1,1}\chi_1+ & \cdots+ & \alpha_{m-1,n-1}\chi_{n-1}\end{array}\right).$$

If one looks at a typical row,

$$\alpha_{i,0}\chi_0+ \quad \alpha_{i,1}\chi_1+ \quad \cdots+ \quad \alpha_{i,n-1}\chi_{n-1}$$

one notices that this is just the dot product of vectors

$$\widetilde{a}_i = \left(\begin{array}{c}\alpha_{i,0}\\ \alpha_{i,1}\\ \vdots\\ \alpha_{i,n-1}\end{array}\right) \quad \text{and} \quad x = \left(\begin{array}{c}\chi_0\\ \chi_1\\ \vdots\\ \chi_n - 1\end{array}\right).$$

In other words, the dot product of the ith row of A, viewed as a column vector, with the vector x, which one can visualize as

$$\left(\begin{array}{c}\psi_0\\ \vdots\\ \boxed{\psi_i}\\ \vdots\\ \psi_{m-1}\end{array}\right) = \left(\begin{array}{cccc}\alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1}\\ \vdots & \vdots & & \vdots\\ \hline \alpha_{i,0} & \alpha_{i,1} & \cdots & \alpha_{i,n-1}\\ \hline \vdots & \vdots & & \vdots\\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1}\end{array}\right)\left(\boxed{\begin{array}{c}\chi_0\\ \chi_1\\ \vdots\\ \chi_{n-1}\end{array}}\right)$$

The above argument starts to expain why we write the dot product of vectors x and y as $x^T y$.

Example 3.7 Let $A = \begin{pmatrix} -1 & 0 & 2 \\ 2 & -1 & 1 \\ 3 & 1 & -1 \end{pmatrix}$ and $x = \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}$. Then

$$
\begin{aligned}
Ax &= \begin{pmatrix} -1 & 0 & 2 \\ 2 & -1 & 1 \\ 3 & 1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} = \left(\begin{array}{c} \begin{pmatrix} -1 & 0 & 2 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} \\ \hline \begin{pmatrix} 2 & -1 & 1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} \\ \hline \begin{pmatrix} 3 & 1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} \end{array} \right) \\
&= \left(\begin{array}{c} \begin{pmatrix} -1 \\ 0 \\ 2 \end{pmatrix}^T \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} \\ \hline \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix}^T \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} \\ \hline \begin{pmatrix} 3 \\ 1 \\ -1 \end{pmatrix}^T \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} \end{array} \right) = \begin{pmatrix} (-1)(-1)+(0)(2)+(2)(1) \\ (2)(-1)+(-1)(2)+(1)(1) \\ (3)(-1)+(1)(2)+(-1)(1) \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ -2 \end{pmatrix}
\end{aligned}
$$

Algorithm (traditional notation)

An algorithm for computing $y := Ax + y$ (notice that we add the result of Ax to y) via dot products is given by

$$
\begin{array}{l}
\textbf{for } i = 0, \ldots, m-1 \\
\quad \textbf{for } j = 0, \ldots, n-1 \\
\quad\quad \psi_i := \psi_i + \alpha_{i,j}\chi_j \\
\quad \textbf{endfor} \\
\textbf{endfor}
\end{array}
$$

If initially $y = 0$, then it computes $y := Ax$.

Now, let us revisit the fact that the matrix-vector multiply can be computed as dot products of the rows of A with the vector x. Think of the matrix A as individual rows:

$$
A = \begin{pmatrix} \widetilde{a}_0^T \\ \widetilde{a}_1^T \\ \vdots \\ \widetilde{a}_{m-1}^T \end{pmatrix},
$$

where $\widetilde{a}_i$ is the (column) vector which, when transposed, becomes the ith row of the matrix. Then

$$Ax = \begin{pmatrix} \widetilde{a}_0^T \\ \widetilde{a}_1^T \\ \vdots \\ \widetilde{a}_{m-1}^T \end{pmatrix} x = \begin{pmatrix} \widetilde{a}_0^T x \\ \widetilde{a}_1^T x \\ \vdots \\ \widetilde{a}_{m-1}^T x \end{pmatrix},$$

which is exactly what we reasoned before. To emphasize this, the algorithm can then be annotated as follows:

$$\begin{array}{l}
\textbf{for } i = 0, \ldots, m-1 \\
\quad \left.\begin{array}{l}
\textbf{for } j = 0, \ldots, n-1 \\
\quad \psi_i := \psi_i + \alpha_{i,j}\chi_j \\
\textbf{endfor}
\end{array}\right\} \psi_i := \psi_i + \tilde{a}_i^T x \\
\textbf{endfor}
\end{array}$$

Algorithm (FLAME notation)

We now present the algorithm that casts matrix-vector multiplication in terms of dot products using the FLAME notation with which you became familiar earlier this week:

Algorithm: $y := \text{MVMULT\_N\_UNB\_VAR1}(A, x, y)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where A_T is $0 \times n$ and y_T is 0×1

while $m(A_T) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ A_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ y_2 \end{array}\right)$$

where a_1 is a row

$\psi_1 := a_1^T x + \psi_1$

Continue with

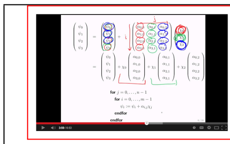
$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Homework 3.4.1.1 Implement function `Mvmult_n_unb_var1( A, x, y )`.

☛ SEE ANSWER

3.4.2 Via AXPY Operations



☛ YouTube

Motivation

Note that, by definition,

$$Ax = \begin{pmatrix} \alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + & \cdots + & \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + & \cdots + & \alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + & \cdots + & \alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} =$$

$$\chi_0 \begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \\ \vdots \\ \alpha_{m-1,0} \end{pmatrix} + \chi_1 \begin{pmatrix} \alpha_{0,1} \\ \alpha_{1,1} \\ \vdots \\ \alpha_{m-1,1} \end{pmatrix} + \cdots + \chi_{n-1} \begin{pmatrix} \alpha_{0,n-1} \\ \alpha_{1,n-1} \\ \vdots \\ \alpha_{m-1,n-1} \end{pmatrix}.$$

Example 3.8 Let $A = \begin{pmatrix} -1 & 0 & 2 \\ 2 & -1 & 1 \\ 3 & 1 & -1 \end{pmatrix}$ and $x = \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}$. Then

$$\begin{aligned} Ax &= \begin{pmatrix} -1 & 0 & 2 \\ 2 & -1 & 1 \\ 3 & 1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} = (-1)\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} + (2)\begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} + (1)\begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} (-1)(-1) \\ (-1)(2) \\ (-1)(3) \end{pmatrix} + \begin{pmatrix} (2)(0) \\ (2)(-1) \\ (2)(1) \end{pmatrix} + \begin{pmatrix} (1)(2) \\ (1)(1) \\ (1)(-1) \end{pmatrix} \\ &= \begin{pmatrix} (-1)(-1)+(0)(2)+(2)(1) \\ (2)(-1)+(-1)(2)+(1)(1) \\ (3)(-1)+(1)(2)+(-1)(1) \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ -2 \end{pmatrix} \end{aligned}$$

Algorithm (traditional notation)

The above suggests the alternative algorithm for computing $y := Ax + y$ given by

$$\begin{array}{l} \textbf{for } j = 0, \ldots, n-1 \\ \quad \textbf{for } i = 0, \ldots, m-1 \\ \quad\quad \psi_i := \psi_i + \alpha_{i,j}\chi_j \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}$$

If we let a_j denote the vector that equals the jth column of A, then

$$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$$

and

$$\begin{aligned} Ax &= \chi_0 \underbrace{\begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \\ \vdots \\ \alpha_{m-1,0} \end{pmatrix}}_{a_0} + \chi_1 \underbrace{\begin{pmatrix} \alpha_{0,1} \\ \alpha_{1,1} \\ \vdots \\ \alpha_{m-1,1} \end{pmatrix}}_{a_1} + \cdots + \chi_{n-1} \underbrace{\begin{pmatrix} \alpha_{0,n-1} \\ \alpha_{1,n-1} \\ \vdots \\ \alpha_{m-1,n-1} \end{pmatrix}}_{a_{n-1}} \\ &= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}. \end{aligned}$$

This is emphasized by annotating the algorithm as follows:

$$\begin{array}{l} \textbf{for } j = 0, \ldots, n-1 \\ \quad \left. \begin{array}{l} \textbf{for } i = 0, \ldots, m-1 \\ \quad \psi_i := \psi_i + \alpha_{i,j}\chi_j \\ \textbf{endfor} \end{array} \right\} y := \chi_j a_j + y \\ \textbf{endfor} \end{array}$$

Algorithm (FLAME notation)

Here is the algorithm that casts matrix-vector multiplication in terms of AXPYs using the FLAME notation:

Algorithm: $y := \text{MVMULT\_N\_UNB\_VAR2}(A, x, y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_L & A_R \end{array} \right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right)$

  where A_L is $m \times 0$ and x_T is 0×1

while $m(x_T) < m(x)$ **do**

  Repartition

$$\left(\begin{array}{c|c} A_L & A_R \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array} \right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array} \right)$$

  where a_1 is a column

$y := \chi_1 a_1 + y$

  Continue with

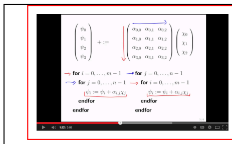
$$\left(\begin{array}{c|c} A_L & A_R \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array} \right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array} \right)$$

endwhile

Homework 3.4.2.1 Implement function `Mvmult_n_unb_var2( A, x, y )`.
(Hint: use the function `laff_dots( x, y, alpha )` that updates $\alpha := x^T y + \alpha$.)

☛ SEE ANSWER

3.4.3 Compare and Contrast



☛ YouTube

Motivation

It is always useful to compare and contrast different algorithms for the same operation.

Algorithms (traditional notation)

Let us put the two algorithms that compute $y := Ax + y$ via "double nested loops" next to each other:

for $j = 0, \ldots, n-1$	**for** $i = 0, \ldots, m-1$
for $i = 0, \ldots, m-1$	**for** $j = 0, \ldots, n-1$
$\psi_i := \psi_i + \alpha_{i,j}\chi_j$	$\psi_i := \psi_i + \alpha_{i,j}\chi_j$
endfor	**endfor**
endfor	**endfor**

On the left is the algorithm based on the AXPY operation and on the right the one based on the dot product. Notice that these loops differ only in that the order of the two loops are interchanged. This is known as "interchanging loops" and is sometimes used by compilers to optimize nested loops. In the enrichment section of this week we will discuss why you may prefer one ordering of the loops over another.

The above explains, in part, why we chose to look at $y := Ax + y$ rather than $y := Ax$. For $y := Ax + y$, the two algorithms differ only in the order in which the loops appear. To compute $y := Ax$, one would have to initialize each component of y to zero, $\psi_i := 0$. Depending on where in the algorithm that happens, transforming an algorithm that computes $y := Ax$ elements of y at a time (the inner loop implements a dot product) into an algorithm that computes with columns of A (the inner loop implements an AXPY operation) gets trickier.

Algorithms (FLAME notation)

Now let us place the two algorithms presented using the FLAME notation next to each other:

Algorithm: $y := \text{MVMULT\_N\_UNB\_VAR1}(A, x, y)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

  where A_T is $0 \times n$ and y_T is 0×1

while $m(A_T) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

  $\psi_1 := a_1^T x + \psi_1$

  Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{MVMULT\_N\_UNB\_VAR2}(A, x, y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_L & A_R \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$

  where A_L is $m \times 0$ and x_T is 0×1

while $m(x_T) < m(x)$ **do**

  Repartition

$$\left(\begin{array}{c|c} A_L & A_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right)$$

  $y := \chi_1 a_1 + y$

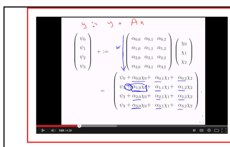
  Continue with

$$\left(\begin{array}{c|c} A_L & A_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \chi_1 \\ \hline x_2 \end{array}\right)$$

endwhile

The algorithm on the left clearly accesses the matrix by rows while the algorithm on the right accesses it by columns. Again, this is important to note, and will be discussed in enrichment for this week.

3.4.4 Cost of Matrix-Vector Multiplication



☛ YouTube

Consider $y := Ax + y$, where $A \in \mathbb{R}^{m \times n}$:

$$y = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{m-1} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + & \cdots + & \alpha_{0,n-1}\chi_{n-1} + & \psi_0 \\ \alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + & \cdots + & \alpha_{1,n-1}\chi_{n-1} + & \psi_2 \\ \vdots & \vdots & \vdots & & \vdots \\ \alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + & \cdots + & \alpha_{m-1,n-1}\chi_{n-1} + & \psi_{m-1} \end{pmatrix}.$$

Notice that there is a multiply and an add for every element of A. Since A has $m \times n = mn$ elements, $y := Ax + y$, requires mn multiplies and mn adds, for a total of $2mn$ floating point operations (flops). This count is the same regardless of the order of the loops (i.e., regardless of whether the matrix-vector multiply is organized by computing dot operations with the rows or axpy operations with the columns).

3.5 Wrap Up

3.5.1 Homework

No additional homework this week. You have done enough...

3.5.2 Summary

Special Matrices

Name	Represents linear transformation	Has entries
Zero matrix, $0_{m\times n} \in \mathbb{R}^{m\times n}$	$L_0 : \mathbb{R}^n \to \mathbb{R}^m$ $L_0(x) = 0$ for all x	$0 = 0_{m\times n} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$
Identity matrix, $I \in \mathbb{R}^{n\times n}$	$L_I : \mathbb{R}^n \to \mathbb{R}^n$ $L_I(x) = x$ for all x	$I = I_{n\times n} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$
Diagonal matrix, $D \in \mathbb{R}^{n\times n}$	$L_D : \mathbb{R}^n \to \mathbb{R}^n$ if $y = L_D(x)$ then $\psi_i = \delta_i \chi_i$	$D = \begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}$

Triangular matrices

$A \in \mathbb{R}^{n\times n}$ is said to be...	if ...	
lower triangular	$\alpha_{i,j} = 0$ if $i < j$	$\begin{pmatrix} \alpha_{0,0} & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{pmatrix}$
strictly lower triangular	$\alpha_{i,j} = 0$ if $i \leq j$	$\begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & 0 & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 0 \end{pmatrix}$
unit lower triangular	$\alpha_{i,j} = \begin{cases} 0 & \text{if } i < j \\ 1 & \text{if } i = j \end{cases}$	$\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & 1 & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 1 \end{pmatrix}$
upper triangular	$\alpha_{i,j} = 0$ if $i > j$	$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \alpha_{n-2,n-2} & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & \alpha_{n-1,n-1} \end{pmatrix}$
strictly upper triangular	$\alpha_{i,j} = 0$ if $i \geq j$	$\begin{pmatrix} 0 & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & 0 & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$
unit upper triangular	$\alpha_{i,j} = \begin{cases} 0 & \text{if } i > j \\ 1 & \text{if } i = j \end{cases}$	$\begin{pmatrix} 1 & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & 1 & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$

Transpose matrix

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_{m-2,0} & \alpha_{m-2,1} & \cdots & \alpha_{m-2,n-2} & \alpha_{m-2,n-1} \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-2} & \alpha_{m-1,n-1} \end{pmatrix}^T = \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{m-2,0} & \alpha_{m-1,0} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{m-2,1} & \alpha_{m-1,1} \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_{0,n-2} & \alpha_{1,n-2} & \cdots & \alpha_{m-2,n-2} & \alpha_{m-1,n-2} \\ \alpha_{0,n-1} & \alpha_{1,n-1} & \cdots & \alpha_{m-2,n-1} & \alpha_{m-1,n-1} \end{pmatrix}$$

Symmetric matrix

Matrix $A \in \mathbb{R}^{n\times n}$ is symmetric if and only if $A = A^T$:

$$A = \left(\begin{array}{ccccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & \alpha_{n-2,n-1} \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{array}\right) = \left(\begin{array}{ccccc} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{n-2,0} & \alpha_{n-1,0} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{n-2,1} & \alpha_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{0,n-2} & \alpha_{1,n-2} & \cdots & \alpha_{n-2,n-2} & \alpha_{n-1,n-2} \\ \alpha_{0,n-1} & \alpha_{1,n-1} & \cdots & \alpha_{n-2,n-1} & \alpha_{n-1,n-1} \end{array}\right) = A^T$$

Scaling a matrix

Let $\beta \in \mathbb{R}$ and $A \in \mathbb{R}^{m\times n}$. Then

$$\begin{array}{rcl} \beta A & = & \beta \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array}\right) = \left(\begin{array}{c|c|c|c} \beta a_0 & \beta a_1 & \cdots & \beta a_{n-1} \end{array}\right) \\ & = & \beta \left(\begin{array}{cccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{array}\right) = \left(\begin{array}{cccc} \beta\alpha_{0,0} & \beta\alpha_{0,1} & \cdots & \beta\alpha_{0,n-1} \\ \beta\alpha_{1,0} & \beta\alpha_{1,1} & \cdots & \beta\alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta\alpha_{m-1,0} & \beta\alpha_{m-1,1} & \cdots & \beta\alpha_{m-1,n-1} \end{array}\right) \end{array}$$

Adding matrices

Let $A, B \in \mathbb{R}^{m\times n}$. Then

$$\begin{array}{rcl} A + B & = & \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array}\right) + \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right) = \left(\begin{array}{c|c|c|c} a_0 + b_0 & a_1 + b_1 & \cdots & a_{n-1} + b_{n-1} \end{array}\right) \\ & = & \left(\begin{array}{cccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{array}\right) + \left(\begin{array}{cccc} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0} & \beta_{m-1,1} & \cdots & \beta_{m-1,n-1} \end{array}\right) \\ & = & \left(\begin{array}{cccc} \alpha_{0,0} + \beta_{0,0} & \alpha_{0,1} + \beta_{0,1} & \cdots & \alpha_{0,n-1} + \beta_{0,n-1} \\ \alpha_{1,0} + \beta_{1,0} & \alpha_{1,1} + \beta_{1,1} & \cdots & \alpha_{1,n-1} + \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} + \beta_{m-1,0} & \alpha_{m-1,1} + \beta_{m-1,1} & \cdots & \alpha_{m-1,n-1} + \beta_{m-1,n-1} \end{array}\right) \end{array}$$

- Matrix addition commutes: $A + B = B + A$.
- Matrix addition is associative: $(A + B) + C = A + (B + C)$.
- $(A + B)^T = A^T + B^T$.

Matrix-vector multiplication

$$Ax = \left(\begin{array}{cccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{array}\right) \left(\begin{array}{c} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{array}\right) = \left(\begin{array}{cccc} \alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + & \cdots + & \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + & \cdots + & \alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + & \cdots + & \alpha_{m-1,n-1}\chi_{n-1} \end{array}\right)$$

$$
\begin{aligned}
&= \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} \\
&= \begin{pmatrix} \widetilde{a}_0^T \\ \widetilde{a}_1^T \\ \vdots \\ \widetilde{a}_{m-1}^T \end{pmatrix} x = \begin{pmatrix} \widetilde{a}_0^T x \\ \widetilde{a}_1^T x \\ \vdots \\ \widetilde{a}_{m-1}^T x \end{pmatrix}
\end{aligned}
$$

Week 4

From Matrix-Vector Multiplication to Matrix-Matrix Multiplication

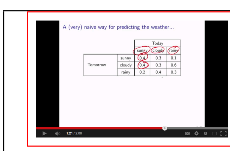
There are a LOT of programming assignments this week.

- They are meant to help clarify “slicing and dicing”.
- They show that the right abstractions in the mathematics, when reflected in how we program, allow one to implement algorithms very quickly.
- They help you understand special properties of matrices.

Practice as much as you think will benefit your understanding of the material. There is no need to do them all!

4.1 Opening Remarks

4.1.1 Predicting the Weather



☛ YouTube

The following table tells us how the weather for any day (e.g., today) predicts the weather for the next day (e.g., tomorrow):

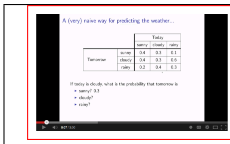
		Today		
		sunny	cloudy	rainy
Tomorrow	sunny	0.4	0.3	0.1
	cloudy	0.4	0.3	0.6
	rainy	0.2	0.4	0.3

This table is interpreted as follows: If today is rainy, then the probability that it will be cloudy tomorrow is 0.6, etc.

Homework 4.1.1.1 If today is cloudy, what is the probability that tomorrow is

- sunny?
- cloudy?
- rainy?

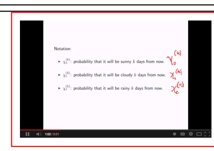
☛ SEE ANSWER



☛ YouTube

Homework 4.1.1.2 If today is sunny, what is the probability that the day after tomorrow is sunny? cloudy? rainy?
☛ SEE ANSWER

Try this! If today is cloudy, what is the probability that a week from today it is sunny? cloudy? rainy?
Think about this for at most two minutes, and then look at the answer.



☛ YouTube

When things get messy, it helps to introduce some notation.

- Let $\chi_s^{(k)}$ denote the probability that it will be sunny k days from now (on day k).
- Let $\chi_c^{(k)}$ denote the probability that it will be cloudy k days from now.
- Let $\chi_r^{(k)}$ denote the probability that it will be rainy k days from now.

The discussion so far motivate the equations

$$\begin{array}{rcrcrcr}
\chi_s^{(k+1)} & = & 0.4 \times \chi_s^{(k)} & + & 0.3 \times \chi_c^{(k)} & + & 0.1 \times \chi_r^{(k)} \\
\chi_c^{(k+1)} & = & 0.4 \times \chi_s^{(k)} & + & 0.3 \times \chi_c^{(k)} & + & 0.6 \times \chi_r^{(k)} \\
\chi_r^{(k+1)} & = & 0.2 \times \chi_s^{(k)} & + & 0.4 \times \chi_c^{(k)} & + & 0.3 \times \chi_r^{(k)}.
\end{array}$$

The probabilities that denote what the weather may be on day k and the table that summarizes the probabilities are often represented as a *(state) vector*, $x^{(k)}$, and *(transition) matrix*, P, respectively:

$$x^{(k)} = \begin{pmatrix} \chi_s^{(k)} \\ \chi_c^{(k)} \\ \chi_r^{(k)} \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix}.$$

The transition from day k to day $k+1$ is then written as the matrix-vector product (multiplication)

$$\begin{pmatrix} \chi_s^{(k+1)} \\ \chi_c^{(k+1)} \\ \chi_r^{(k+1)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(k)} \\ \chi_c^{(k)} \\ \chi_r^{(k)} \end{pmatrix}$$

or $x^{(k+1)} = Px^{(k)}$, which is simply a more compact representation (way of writing) the system of linear equations.

What this demonstrates is that matrix-vector multiplication can also be used to compactly write a set of simultaneous linear equations.

Assume again that today is cloudy so that the probability that it is sunny, cloudy, or rainy today is 0, 1, and 0, respectively:

$$x^{(0)} = \begin{pmatrix} \chi_s^{(0)} \\ \chi_c^{(0)} \\ \chi_r^{(0)} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

(If we KNOW today is cloudy, then the probability that is is sunny today is zero, etc.)

> Ah! Our friend the unit basis vector reappears!

Then the vector of probabilities for tomorrow's weather, $x^{(1)}$, is given by

$$\begin{aligned} \begin{pmatrix} \chi_s^{(1)} \\ \chi_c^{(1)} \\ \chi_r^{(1)} \end{pmatrix} &= \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(0)} \\ \chi_c^{(0)} \\ \chi_r^{(0)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 0.4 \times 0 & + & 0.3 \times 1 & + & 0.1 \times 0 \\ 0.4 \times 0 & + & 0.3 \times 1 & + & 0.6 \times 0 \\ 0.2 \times 0 & + & 0.4 \times 1 & + & 0.3 \times 0 \end{pmatrix} = \begin{pmatrix} 0.3 \\ 0.3 \\ 0.4 \end{pmatrix}. \end{aligned}$$

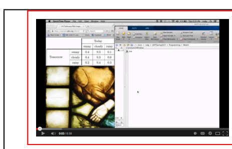
> Ah! $Pe_1 = p_1$, where p_1 is the second column in matrix P. You should not be surprised!

The vector of probabilities for the day after tomorrow, $x^{(2)}$, is given by

$$\begin{aligned} \begin{pmatrix} \chi_s^{(2)} \\ \chi_c^{(2)} \\ \chi_r^{(2)} \end{pmatrix} &= \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(1)} \\ \chi_c^{(1)} \\ \chi_r^{(1)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 0.3 \\ 0.3 \\ 0.4 \end{pmatrix} \\ &= \begin{pmatrix} 0.4 \times 0.3 & + & 0.3 \times 0.3 & + & 0.1 \times 0.4 \\ 0.4 \times 0.3 & + & 0.3 \times 0.3 & + & 0.6 \times 0.4 \\ 0.2 \times 0.3 & + & 0.4 \times 0.3 & + & 0.3 \times 0.4 \end{pmatrix} = \begin{pmatrix} 0.25 \\ 0.45 \\ 0.30 \end{pmatrix}. \end{aligned}$$

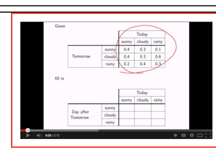
Repeating this process (preferrably using Python rather than by hand), we can find the probabilities for the weather for the next seven days, under the assumption that today is cloudy:

	k							
	0	1	2	3	4	5	6	7
$x^{(k)} =$	$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0.3 \\ 0.3 \\ 0.4 \end{pmatrix}$	$\begin{pmatrix} 0.25 \\ 0.45 \\ 0.30 \end{pmatrix}$	$\begin{pmatrix} 0.265 \\ 0.415 \\ 0.320 \end{pmatrix}$	$\begin{pmatrix} 0.2625 \\ 0.4225 \\ 0.3150 \end{pmatrix}$	$\begin{pmatrix} 0.26325 \\ 0.42075 \\ 0.31600 \end{pmatrix}$	$\begin{pmatrix} 0.26312 \\ 0.42112 \\ 0.31575 \end{pmatrix}$	$\begin{pmatrix} 0.26316 \\ 0.42104 \\ 0.31580 \end{pmatrix}$



☛ **YouTube**

> **Homework 4.1.1.3** Follow the instructions in the above video
>
> ☛ SEE ANSWER



☛ **YouTube**

We could build a table that tells us how to predict the weather for the day after tomorrow from the weather today:

		Today		
		sunny	cloudy	rainy
Day after Tomorrow	sunny			
	cloudy			
	rainy			

One way you can do this is to observe that

$$\begin{aligned}
\begin{pmatrix} \chi_s^{(2)} \\ \chi_c^{(2)} \\ \chi_r^{(2)} \end{pmatrix} &= \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(1)} \\ \chi_c^{(1)} \\ \chi_r^{(1)} \end{pmatrix} \\
&= \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \left(\begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(0)} \\ \chi_c^{(0)} \\ \chi_r^{(0)} \end{pmatrix} \right) = Q \begin{pmatrix} \chi_s^{(0)} \\ \chi_c^{(0)} \\ \chi_r^{(0)} \end{pmatrix},
\end{aligned}$$

where Q is the transition matrix that tells us how the weather today predicts the weather the day after tomorrow. (Well, actually, we don't yet know that applying a matrix to a vector twice is a linear transformation... We'll learn that later this week.)

Now, just like P is simply the matrix of values from the original table that showed how the weather tomorrow is predicted from today's weather, Q is the matrix of values for the above table.

Homework 4.1.1.4 Given

		Today		
		sunny	cloudy	rainy
Tomorrow	sunny	0.4	0.3	0.1
	cloudy	0.4	0.3	0.6
	rainy	0.2	0.4	0.3

fill in the following table, which predicts the weather the day after tomorrow given the weather today:

		Today		
		sunny	cloudy	rainy
Day after Tomorrow	sunny			
	cloudy			
	rainy			

Now here is the hard part: Do so without using your knowledge about how to perform a matrix-matrix multiplication, since you won't learn about that until later this week... May we suggest that you instead use MATLAB to perform the necessary calculations.

☛ SEE ANSWER

4.1.2 Outline

4.1.3 What You Will Learn

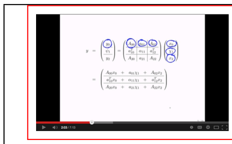
Upon completion of this unit, you should be able to

- Apply matrix vector multiplication to predict the probability of future states in a Markov process.
- Make use of partitioning to perform matrix vector multiplication.
- Transpose a partitioned matrix.
- Partition conformally, ensuring that the size of the matrices and vectors match so that matrix-vector multiplication works.
- Take advantage of special structures to perform matrix-vector multiplication with triangular and symmetric matrices.
- Express and implement various matrix-vector multiplication algorithms using the FLAME notation and FlamePy.
- Make connections between the composition of linear transformations and matrix-matrix multiplication.
- Compute a matrix-matrix multiplication.
- Recognize scalars and column/row vectors as special cases of matrices.
- Compute common vector-vector and matrix-vector operations as special cases of matrix-matrix multiplication.
- Compute an outer product xy^T as a special case of matrix-matrix multiplication and recognize that
  - The rows of the resulting matrix are scalar multiples of y^T.
  - The columns of the resulting matrix are scalar multiples of x.

Track your progress in Appendix B.

4.2 Preparation

4.2.1 Partitioned Matrix-Vector Multiplication



☛ YouTube

Motivation

Consider

$$A=\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)=\left(\begin{array}{rr|r|rr} -1 & 2 & 4 & 1 & 0 \\ 1 & 0 & -1 & -2 & 1 \\ \hline 2 & -1 & 3 & 1 & 2 \\ \hline 1 & 2 & 3 & 4 & 3 \\ -1 & -2 & 0 & 1 & 2 \end{array}\right),$$

$$x=\left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right)=\left(\begin{array}{c} 1 \\ 2 \\ \hline 3 \\ \hline 4 \\ 5 \end{array}\right), \quad \text{and} \quad y=\left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right),$$

where $y_0, y_2 \in \mathbb{R}^2$. Then $y = Ax$ means that

$$\begin{aligned} y &= \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)=\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)\left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right)=\left(\begin{array}{ccccc} A_{00}x_0 & + & a_{01}\chi_1 & + & A_{02}x_2 \\ \hline a_{10}^T x_0 & + & \alpha_{11}\chi_1 & + & a_{12}^T x_2 \\ \hline A_{20}x_0 & + & a_{21}\chi_1 & + & A_{22}x_2 \end{array}\right) \\ &= \left(\begin{array}{c} \begin{pmatrix} -1 & 2 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 2 \end{pmatrix} + \begin{pmatrix} 4 \\ -1 \end{pmatrix} 3 + \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix}\begin{pmatrix} 4 \\ 5 \end{pmatrix} \\ \hline \begin{pmatrix} 2 & -1 \end{pmatrix}\begin{pmatrix} 1 \\ 2 \end{pmatrix} + \begin{pmatrix} 3 \end{pmatrix} 3 + \begin{pmatrix} 1 & 2 \end{pmatrix}\begin{pmatrix} 4 \\ 5 \end{pmatrix} \\ \hline \begin{pmatrix} 1 & 2 \\ -1 & -2 \end{pmatrix}\begin{pmatrix} 1 \\ 2 \end{pmatrix} + \begin{pmatrix} 3 \\ 0 \end{pmatrix} 3 + \begin{pmatrix} 4 & 3 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 4 \\ 5 \end{pmatrix} \end{array}\right) = \\ & \left(\begin{array}{c} \begin{pmatrix} (-1)\times(1)+(2)\times(2) \\ (1)\times(1)+(0)\times(2) \end{pmatrix} + \begin{pmatrix} (4)\times(3) \\ (-1)\times(3) \end{pmatrix} + \begin{pmatrix} (1)\times(4)+(0)\times(5) \\ (-2)\times(4)+(1)\times(5) \end{pmatrix} \\ \hline (2)\times(1)+(-1)\times(2)+ \quad (3)\times(3)+ \quad (1)\times(4)+(2)\times(5) \\ \hline \begin{pmatrix} (1)\times(1)+(2)\times(2) \\ (-1)\times(1)+(-2)\times(2) \end{pmatrix} + \begin{pmatrix} (3)\times 3 \\ (0)\times 3 \end{pmatrix} + \begin{pmatrix} (4)\times(4)+(3)\times(5) \\ (1)\times(4)+(2)\times(5) \end{pmatrix} \end{array}\right) = \end{aligned}$$

$$\left(\begin{array}{c} (-1)\times(1)+(2)\times(2)+(4)\times(3)+(1)\times(4)+(0)\times(5) \\ (1)\times(1)+(0)\times(2)+(-1)\times(3)+(-2)\times(4)+(1)\times(5) \\ \hline (2)\times(1)+(-1)\times(2)+(3)\times(3)+(1)\times(4)+(2)\times(5) \\ \hline (1)\times(1)+(2)\times(2)+(3)\times(3)+(4)\times(4)+(3)\times(5) \\ (-1)\times(1)+(-2)\times(2)+(0)\times(3)+(1)\times(4)+(2)\times(5) \end{array}\right) = \left(\begin{array}{c} 19 \\ -5 \\ \hline 23 \\ \hline 45 \\ 9 \end{array}\right)$$

Homework 4.2.1.1 Consider

$$A = \left(\begin{array}{rrrrr} -1 & 2 & 4 & 1 & 0 \\ 1 & 0 & -1 & -2 & 1 \\ 2 & -1 & 3 & 1 & 2 \\ 1 & 2 & 3 & 4 & 3 \\ -1 & -2 & 0 & 1 & 2 \end{array}\right) \quad \text{and} \quad x = \left(\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\right),$$

and partition these into submatrices (regions) as follows:

$$\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \quad \text{and} \quad \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right),$$

where $A_{00} \in \mathbb{R}^{3x3}$, $x_0 \in \mathbb{R}^3$, α_{11} is a scalar, and χ_1 is a scalar. Show with lines how A and x are partitioned:

$$\left(\begin{array}{rrrrr} -1 & 2 & 4 & 1 & 0 \\ 1 & 0 & -1 & -2 & 1 \\ 2 & -1 & 3 & 1 & 2 \\ 1 & 2 & 3 & 4 & 3 \\ -1 & -2 & 0 & 1 & 2 \end{array}\right) \quad \left(\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\right).$$

☛ SEE ANSWER

Homework 4.2.1.2 With the partitioning of matrices A and x in the above exercise, repeat the partitioned matrix-vector multiplication, similar to how this unit started.

☛ SEE ANSWER

Theory

Let $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, and $y \in \mathbb{R}^m$. Partition

$$A = \left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{array}\right), \quad x = \left(\begin{array}{c} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{array}\right), \quad \text{and} \quad y = \left(\begin{array}{c} y_0 \\ \hline y_1 \\ \hline \vdots \\ \hline y_{M-1} \end{array}\right)$$

where

- $m = m_0 + m_1 + \cdots + m_{M-1}$,
- $m_i \geq 0$ for $i = 0, \ldots, M-1$,

- $n = n_0 + n_1 + \cdots + n_{N-1}$,
- $n_j \geq 0$ for $j = 0, \ldots, N-1$, and
- $A_{i,j} \in \mathbb{R}^{m_i \times n_j}$, $x_j \in \mathbb{R}^{n_j}$, and $y_i \in \mathbb{R}^{m_i}$.

If $y = Ax$ then

$$\left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{array}\right)\left(\begin{array}{c} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{array}\right)$$

$$= \left(\begin{array}{c} A_{0,0}x_0 + A_{0,1}x_1 + \cdots + A_{0,N-1}x_{N-1} \\ \hline A_{1,0}x_0 + A_{1,1}x_1 + \cdots + A_{1,N-1}x_{N-1} \\ \hline \vdots \\ \hline A_{M-1,0}x_0 + A_{M-1,1}x_1 + \cdots + A_{M-1,N-1}x_{N-1} \end{array}\right).$$

In other words,

$$y_i = \sum_{j=0}^{N-1} A_{i,j} x_j.$$

This is intuitively true and messy to prove carefully. Therefore we will not give its proof, relying on the many examples we will encounter in subsequent units instead.

If one partitions matrix A, vector x, and vector y into blocks, **and** one makes sure the dimensions match up, **then** blocked matrix-vector multiplication proceeds exactly as does a regular matrix-vector multiplication **except** that individual multiplications of scalars commute while (in general) individual multiplications with matrix and vector blocks (submatrices and subvectors) do not.

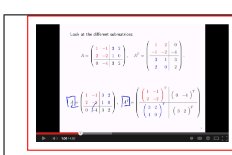
The labeling of the submatrices and subvectors in this unit was carefully chosen to convey information. Consider

$$A = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

The letters that are used convey information about the shapes. For example, for a_{01} and a_{21} the use of a lowercase Roman letter indicates they are column vectors while the ^{T}s in a_{10}^T and a_{12}^T indicate that they are row vectors. Symbols α_{11} and χ_1 indicate these are scalars. We will use these conventions consistently to enhance readability.

Notice that the partitioning of matrix A and vectors x and y has to be "conformal". The simplest way to understand this is that matrix-vector multiplication only works if the sizes of matrices and vectors being multiply match. So, a partitioning of A, x, and y, when performing a given operation, is conformal if the suboperations with submatrices and subvectors that are encountered make sense.

4.2.2 Transposing a Partitioned Matrix



☛ YouTube

Motivation

Consider

$$\left(\begin{array}{ccc|c} 1 & -1 & 3 & 2 \\ 2 & -2 & 1 & 0 \\ \hline 0 & -4 & 3 & 2 \end{array}\right)^T = \left(\begin{array}{c|c} \begin{pmatrix} 1 & -1 & 3 \\ 2 & -2 & 1 \end{pmatrix} & \begin{pmatrix} 2 \\ 0 \end{pmatrix} \\ \hline \begin{pmatrix} 0 & -4 & 3 \end{pmatrix} & \begin{pmatrix} 2 \end{pmatrix} \end{array}\right)^T$$

$$= \left(\begin{array}{c|c} \begin{pmatrix} 1 & -1 & 3 \\ 2 & -2 & 1 \end{pmatrix}^T & \begin{pmatrix} 0 & -4 & 3 \end{pmatrix}^T \\ \hline \begin{pmatrix} 2 \\ 0 \end{pmatrix}^T & \begin{pmatrix} 2 \end{pmatrix}^T \end{array}\right)$$

$$= \left(\begin{array}{c|c} \begin{pmatrix} 1 & 2 \\ -1 & -2 \\ 3 & 1 \end{pmatrix} & \begin{pmatrix} 0 \\ -4 \\ 3 \end{pmatrix} \\ \hline \begin{pmatrix} 2 & 0 \end{pmatrix} & \begin{pmatrix} 2 \end{pmatrix} \end{array}\right) = \left(\begin{array}{cc|c} 1 & 2 & 0 \\ -1 & -2 & -4 \\ 3 & 1 & 3 \\ \hline 2 & 0 & 2 \end{array}\right).$$

This example illustrates a general rule: When transposing a partitioned matrix (matrix partitioned into submatrices), you transpose the matrix of blocks, and then you transpose each block.

Homework 4.2.2.1 Show, step-by-step, how to transpose

$$\left(\begin{array}{cc|cc} 1 & -1 & 3 & 2 \\ 2 & -2 & 1 & 0 \\ \hline 0 & -4 & 3 & 2 \end{array}\right)$$

☛ SEE ANSWER

Theory

Let $A \in \mathbb{R}^{m \times n}$ be partitioned as follows:

$$A = \left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,N-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{array}\right),$$

where $A_{i,j} \in \mathbb{R}^{m_i \times n_j}$. Then

$$A^T = \left(\begin{array}{c|c|c|c} A_{0,0}^T & A_{1,0}^T & \cdots & A_{M-1,0}^T \\ \hline A_{0,1}^T & A_{1,1}^T & \cdots & A_{M-1,1}^T \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{0,N-1}^T & A_{1,N-1}^T & \cdots & A_{M-1,N-1}^T \end{array}\right).$$

Transposing a partitioned matrix means that you view each submatrix as if it is a scalar, and you then transpose the matrix as if it is a matrix of scalars. But then you recognize that each of those scalars is actually a submatrix and you also transpose that submatrix.

Special cases

We now discuss a number of special cases that you may encounter.

Each submatrix is a scalar. If

$$A = \left(\begin{array}{c|c|c|c} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,N-1} \\ \hline \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,N-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline \alpha_{M-1,0} & \alpha_{M-1,1} & \cdots & \alpha_{M-1,N-1} \end{array} \right)$$

then

$$A^T = \left(\begin{array}{c|c|c|c} \alpha_{0,0}^T & \alpha_{1,0}^T & \cdots & \alpha_{M-1,0}^T \\ \hline \alpha_{0,1}^T & \alpha_{1,1}^T & \cdots & \alpha_{M-1,1}^T \\ \hline \vdots & \vdots & & \vdots \\ \hline \alpha_{0,N-1}^T & \alpha_{1,N-1}^T & \cdots & \alpha_{M-1N-1}^T \end{array} \right) = \left(\begin{array}{cccc} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{M-1,0} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{M-1,1} \\ \vdots & \vdots & & \vdots \\ \alpha_{0,N-1} & \alpha_{1,N-1} & \cdots & \alpha_{M-1,N-1} \end{array} \right).$$

This is because the transpose of a scalar is just that scalar.

The matrix is partitioned by rows. If

$$A = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array} \right),$$

where each $\widetilde{a}_i^T$ is a row of A, then

$$A^T = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array} \right)^T = \left(\begin{array}{c|c|c|c} (\widetilde{a}_0^T)^T & (\widetilde{a}_1^T)^T & \cdots & (\widetilde{a}_{m-1}^T)^T \end{array} \right) = \left(\begin{array}{c|c|c|c} \widetilde{a}_0 & \widetilde{a}_1 & \cdots & \widetilde{a}_{m-1} \end{array} \right).$$

This shows that rows of A, $\widetilde{a}_i^T$, become columns of A^T: $\widetilde{a}_i$.

The matrix is partitioned by columns. If

$$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right),$$

where each a_j is a column of A, then

$$A^T = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)^T = \left(\begin{array}{c} a_0^T \\ \hline a_1^T \\ \hline \vdots \\ \hline a_{n-1}^T \end{array} \right).$$

This shows that columns of A, a_j, become rows of A^T: a_j^T.

2×2 **blocked partitioning.** If

$$A = \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array} \right),$$

then

$$A^T = \left(\begin{array}{c|c} A_{TL}^T & A_{BL}^T \\ \hline A_{TR}^T & A_{BR}^T \end{array} \right).$$

3×3 **blocked partitioning.** If

$$A = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array} \right),$$

then

$$A^T = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array} \right)^T = \left(\begin{array}{c|c|c} A_{00}^T & \left(a_{10}^T\right)^T & A_{20}^T \\ \hline a_{01}^T & \alpha_{11}^T & a_{21}^T \\ \hline A_{02}^T & \left(a_{12}^T\right)^T & A_{22}^T \end{array} \right) = \left(\begin{array}{c|c|c} A_{00}^T & a_{10} & A_{20}^T \\ \hline a_{01}^T & \alpha_{11} & a_{21}^T \\ \hline A_{02}^T & a_{12} & A_{22}^T \end{array} \right).$$

Anyway, you get the idea!!!

Homework 4.2.2.2 Transpose the following matrices:

1. $\left(\begin{array}{c} 3 \end{array}\right)$

2. $\left(\begin{array}{c} 3 \\ 1 \\ \hline 1 \\ \hline 8 \end{array}\right)$

3. $\left(\begin{array}{cc|c|c} 3 & 1 & 1 & 8 \end{array}\right)$

4. $\left(\begin{array}{c|c|c|c} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{array}\right)$

5. $\left(\begin{array}{c|c|c} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{array}\right)$

6. $\left(\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \end{array}\right)$

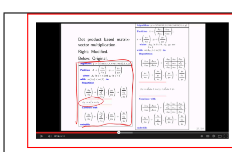
7. $\left(\left(\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \end{array}\right)^T\right)^T$

☛ SEE ANSWER

For any matrix $A \in \mathbb{R}^{m \times n}$,

$$A^{T^T} = (A^T)^T = A$$

4.2.3 Matrix-Vector Multiplication, Again



☛ YouTube

Motivation

In the next few units, we will modify the matrix-vector multiplication algorithms from last week so that they can take advantage of matrices with special structure (e.g., triangular or symmetric matrices).

Now, what makes a triangular or symmetric matrix special? For one thing, it is square. For another, it only requires one triangle of a matrix to be stored. It was for this reason that we ended up with "algorithm skeletons" that looked like the one in

Algorithm: $[A] := \text{ALGORITHM\_SKELETON}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

  where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

  where α_{11} is 1×1

  Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 4.1: Code skeleton for algorithms when matrices are triangular or symmetric.

Figure 4.1 when we presented algorithms for "triangularizing" or "symmetrizing" a matrix.

Now, consider a typical partitioning of a matrix that is encountered in such an algorithm:

$$\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{01} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) = \left(\begin{array}{cc|c|ccc} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \hline \times & \times & \times & \times & \times & \times \\ \hline \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \end{array}\right),$$

where each $\times$ represents an entry in the matrix (in this case 6×6). If, for example, the matrix is lower triangular,

$$\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) = \left(\begin{array}{cc|c|ccc} \times & 0 & 0 & 0 & 0 & 0 \\ \times & \times & 0 & 0 & 0 & 0 \\ \hline \times & \times & \times & 0 & 0 & 0 \\ \hline \times & \times & \times & \times & 0 & 0 \\ \times & \times & \times & \times & \times & 0 \\ \times & \times & \times & \times & \times & \times \end{array}\right),$$

Algorithm: $y := \textsc{Mvmult\_n\_unb\_var1b}(A, x, y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$,
$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$
where A_{TL} is 0×0, x_T, y_T are 0×1
while $m(A_{TL}) < m(A)$ **do**
Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$\psi_1 := a_{10}^T x_0 + \alpha_{11} \chi_1 + a_{12}^T x_2 + \psi_1$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \textsc{Mvmult\_n\_unb\_var2b}(A, x, y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$,
$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$
where A_{TL} is 0×0, x_T, y_T are 0×1
while $m(A_{TL}) < m(A)$ **do**
Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$y_0 := \chi_1 a_{01} + y_0$
$\psi_1 := \chi_1 \alpha_{11} + \psi_1$
$y_2 := \chi_1 a_{21} + y_2$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Figure 4.2: Alternative algorithms for matrix-vector multiplication.

then $a_{01} = 0$, $A_{02} = 0$, and $a_{12}^T = 0$. (Remember: the "0" is a matrix or vector "of appropriate size".) If instead the matrix is symmetric with only the lower triangular part stored, then $a_{01} = \left(a_{10}^T\right)^T = a_{10}$, $A_{02} = A_{20}^T$, and $a_{12}^T = a_{21}^T$.

The above observation leads us to express the matrix-vector multiplication algorithms for computing $y := Ax + y$ given in Figure 4.2. Note:

- For the left algorithm, what was previously the "current" row in matrix A, a_1^T, is now viewed as consisting of three parts:

$$a_1^T = \left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \end{array}\right)$$

while the vector x is now also partitioned into three parts:

$$x = \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_1 \end{array}\right).$$

As we saw in the first week, the partitioned dot product becomes

$$a_1^T x = \left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \end{array}\right) \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_1 \end{array}\right) = a_{10}^T x_0 + \alpha_{11} \chi_1 + a_{12}^T x_2,$$

which explains why the update

$$\psi_1 := a_1^T x + \psi_1$$

is now

$$\psi_1 := a_{10}^T x_0 + \alpha_{11}\chi_1 + a_{12}^T x_2 + \psi_1.$$

- Similar, for the algorithm on the right, based on the matrix-vector multiplication algorithm that uses the AXPY operations, we note that

$$y := \chi_1 a_1 + y$$

is replaced by

$$\left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right) := \chi_1 \left(\begin{array}{c} a_{01} \\ \hline \alpha_{11} \\ \hline a_{21} \end{array}\right) + \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

which equals

$$\left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right) := \left(\begin{array}{c} \chi_1 a_{01} + y_0 \\ \hline \chi_1 \alpha_{11} + \psi_1 \\ \hline \chi_1 a_{21} + y_2 \end{array}\right).$$

This explains the update

$$\begin{array}{rcl} y_0 & := & \chi_1 a_{01} + y_0 \\ \psi_1 & := & \chi_1 \alpha_{11} + \psi_1 \\ y_2 & := & \chi_1 a_{21} + y_2. \end{array}$$

Now, for matrix-vector multiplication $y := Ax + y$, it is not beneficial to break the computation up in this way. Typically, a dot product is more efficient than multiple operations with the subvectors. Similarly, typically one AXPY is more efficient then multiple AXPYs. But the observations in this unit lay the foundation for modifying the algorithms to take advantage of special structure in the matrix, later this week.

Homework 4.2.3.1 Implement routines

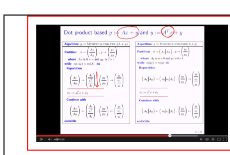
- `[ y_out ] = Mvmult_n_unb_var1B( A, x, y )`; and
- `[ y_out ] = Mvmult_n_unb_var2B( A, x, y )`

that compute $y := Ax + y$ via the algorithms in Figure 4.2.

☛ SEE ANSWER

4.3 Matrix-Vector Multiplication with Special Matrices

4.3.1 Transpose Matrix-Vector Multiplication



☛ YouTube

Algorithm: $y := \text{MVMULT\_T\_UNB\_VAR1}(A, x, y)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right)$

  where A_L is $m \times 0$ and y_T is 0×1

while $m(y_T) < m(y)$ **do**

  Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,|\, A_2 \right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ y_2 \end{array} \right)$$

$$\psi_1 := a_1^T x + \psi_1$$

  Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,|\, a_1 \,\middle|\, A_2 \right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right) \leftarrow \left(\begin{array}{c} y_0 \\ \psi_1 \\ \hline y_2 \end{array} \right)$$

endwhile

Algorithm: $y := \text{MVMULT\_T\_UNB\_VAR2}(A, x, y)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right)$

  where A_T is $0 \times n$ and x_T is 0×1

while $m(A_T) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ A_2 \end{array} \right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ x_2 \end{array} \right)$$

$$y := \chi_1 a_1 + y$$

  Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \leftarrow \left(\begin{array}{c} A_0 \\ a_1^T \\ \hline A_2 \end{array} \right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \leftarrow \left(\begin{array}{c} x_0 \\ \chi_1 \\ \hline x_2 \end{array} \right)$$

endwhile

Figure 4.3: Algorithms for computing $y := A^T x + y$.

Motivation

Let $A = \begin{pmatrix} 1 & -2 & 0 \\ 2 & -1 & 1 \\ 1 & 2 & 3 \end{pmatrix}$ and $x = \begin{pmatrix} -1 \\ 2 \\ -3 \end{pmatrix}$. Then

$$A^T x = \left(\begin{array}{c|c|c} 1 & -2 & 0 \\ 2 & -1 & 1 \\ 1 & 2 & 3 \end{array} \right)^T \begin{pmatrix} -1 \\ 2 \\ -3 \end{pmatrix} = \left(\begin{array}{ccc} 1 & 2 & 1 \\ \hline -2 & -1 & 2 \\ \hline 0 & 1 & 3 \end{array} \right) \begin{pmatrix} -1 \\ 2 \\ -3 \end{pmatrix} = \left(\begin{array}{c} 0 \\ \hline -6 \\ \hline -7 \end{array} \right).$$

The thing to notice is that what was a column in A becomes a row in A^T.

Algorithms

Let us consider how to compute $y := A^T x + y$.

It would be possible to explicitly transpose matrix A into a new matrix B (using, for example, the transpose function you wrote in Week 3) and to then compute $y := Bx + y$. This approach has at least two drawbacks:

- You will need space for the matrix B. Computational scientists tend to push the limits of available memory, and hence are always hesitant to use large amounts of space that isn't absolutely necessary.
- Transposing A into B takes time. A matrix-vector multiplication requires $2mn$ flops. Transposing a matrix requires $2mn$ memops (mn reads from memory and mn writes to memory). Memory operations are very slow relative to floating point operations... So, you will spend all your time transposing the matrix.

Now, the motivation for this unit suggest that we can simply use columns of A for the dot products in the dot product based algorithm for $y := Ax + y$. This suggests the algorithm in FLAME notation in Figure 4.3 (left). Alternatively, one can exploit the fact that columns in A become rows of A^T to change the algorithm for computing $y := Ax + y$ that is based on AXPY operations into an algorithm for computing $y := A^T x + y$, as shown in Figure 4.3 (right).

Implementation

Homework 4.3.1.1 Implement the routines

- `[ y_out ] = Mvmult_t_unb_var1( A, x, y )`; and
- `[ y_out ] = Mvmult_t_unb_var2( A, x, y )`

that compute $y := A^T x + y$ via the algorithms in Figure 4.3.

☛ SEE ANSWER

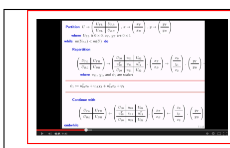
Homework 4.3.1.2 Implementations achieve better performance (finish faster) if one accesses data consecutively in memory. Now, most scientific computing codes store matrices in "column-major order" which means that the first column of a matrix is stored consecutively in memory, then the second column, and so forth. Now, this means that an algorithm that accesses a matrix by columns tends to be faster than an algorithm that accesses a matrix by rows. That, in turn, means that when one is presented with more than one algorithm, one should pick the algorithm that accesses the matrix by columns.
Our FLAME notation makes it easy to recognize algorithms that access the matrix by columns.

- For the matrix-vector multiplication $y := Ax + y$, would you recommend the algorithm that uses dot products or the algorithm that uses `axpy` operations?
- For the matrix-vector multiplication $y := A^T x + y$, would you recommend the algorithm that uses dot products or the algorithm that uses `axpy` operations?

☛ SEE ANSWER

The point of this last exercise is to make you aware of the fact that knowing more than one algorithm can give you a performance edge. (Useful if you pay \$30 million for a supercomputer and you want to get the most out of its use.)

4.3.2 Triangular Matrix-Vector Multiplication



☛ YouTube

Motivation

Let $U \in \mathbb{R}^{n \times n}$ be an upper triangular matrix and $x \in \mathbb{R}^n$ be a vector. Consider

$$
Ux = \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right)\left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right) = \left(\begin{array}{cc|c|cc} -1 & 2 & 4 & 1 & 0 \\ 0 & 0 & -1 & -2 & 1 \\ \hline 0 & 0 & 3 & 1 & 2 \\ \hline 0 & 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 0 & 2 \end{array}\right)\left(\begin{array}{c} 1 \\ 2 \\ \hline 3 \\ \hline 4 \\ 5 \end{array}\right)
$$

$$= \left(\begin{array}{c} \star \\ \star \\ \hline \begin{pmatrix} 0 \\ 0 \end{pmatrix}^T \begin{pmatrix} 1 \\ 2 \end{pmatrix} + (3)(3) + \begin{pmatrix} 1 \\ 2 \end{pmatrix}^T \begin{pmatrix} 4 \\ 5 \end{pmatrix} \\ \hline \star \\ \star \end{array} \right) = \left(\begin{array}{c} \star \\ \star \\ \hline (3)(3) + \begin{pmatrix} 1 \\ 2 \end{pmatrix}^T \begin{pmatrix} 4 \\ 5 \end{pmatrix} \\ \hline \star \\ \star \end{array} \right),$$

where $\star$s indicate components of the result that aren't important in our discussion right now. We notice that $u_{10}^T = 0$ (a vector of two zeroes) and hence we need not compute with it.

Theory

If

$$U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array} \right) = \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array} \right),$$

where U_{TL} and U_{00} are square matrices. Then

- $U_{BL} = 0$, $u_{10}^T = 0$, $U_{20} = 0$, and $u_{21} = 0$, where 0 indicates a matrix or vector of the appropriate dimensions.
- U_{TL} and U_{BR} are upper triangular matrices.

We will just state this as "intuitively obvious".

Similarly, if

$$L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array} \right) = \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array} \right),$$

where L_{TL} and L_{00} are square matrices, then

- $L_{TR} = 0$, $l_{01} = 0$, $L_{02} = 0$, and $l_{12}^T = 0$, where 0 indicates a matrix or vector of the appropriate dimensions.
- L_{TL} and L_{BR} are lower triangular matrices.

Algorithms

Let us start by focusing on $y := Ux + y$, where U is upper triangular. The algorithms from the previous section can be restated as in Figure 4.4, replacing A by U. Now, notice the parts in gray. Since $u_{10}^T = 0$ and $u_{21} = 0$, those computations need not be performed! Bingo, we have two algorithms that take advantage of the zeroes below the diagonal. We probably should explain the names of the routines:

> TRMVP_UN_UNB_VAR1: Triangular matrix-vector multiply plus (y), with upper triangular matrix that is not transposed, unblocked variant 1.

(Yes, a bit convoluted, but such is life.)

Homework 4.3.2.1 Write routines

- `[ y_out ] = Trmvp_un_unb_var1 ( U, x, y )`; and
- `[ y_out ] = Trmvp_un_unb_var2( U, x, y )`

that implement the algorithms in Figure 4.4 that compute $y := Ux + y$.

☛ SEE ANSWER

Algorithm: $y := \text{TRMVP\_UN\_UNB\_VAR1}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where U_{TL} is 0×0, x_T, y_T are 0×1

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$\psi_1 := u_{10}^T x_0 + \upsilon_{11}\chi_1 + u_{12}^T x_2 + \psi_1$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{TRMVP\_UN\_UNB\_VAR2}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where U_{TL} is 0×0, x_T, y_T are 0×1

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$y_0 := \chi_1 u_{01} + y_0$

$\psi_1 := \chi_1 \upsilon_{11} + \psi_1$

$y_2 := \chi_1 u_{21} + y_2$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Figure 4.4: Algorithms for computing $y := Ux + y$, where U is upper triangular.

Homework 4.3.2.2 Modify the algorithms in Figure 4.5 so that they compute $y := Lx + y$, where L is a lower triangular matrix: (Just strike out the parts that evaluate to zero. We suggest you do this homework in conjunction with the next one.)

☛ SEE ANSWER

Homework 4.3.2.3 Write the functions

- `[ y_out ] = Trmvp_ln_unb_var1 ( L, x, y );` and
- `[ y_out ] = Trmvp_ln_unb_var2( L, x, y )`

that implement thenalgorithms for computing $y := Lx + y$ from Homework 4.3.2.2.

☛ SEE ANSWER

Homework 4.3.2.4 Modify the algorithms in Figure 4.6 to compute $x := Ux$, where U is an upper triangular matrix. **You may not use** y**. You have to overwrite** x **without using work space.** Hint: Think carefully about the order in which elements of x are computed and overwritten. You may want to do this exercise hand-in-hand with the implementation in the next homework.

☛ SEE ANSWER

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR1}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$\psi_1 := l_{10}^T x_0 + \lambda_{11}\chi_1 + l_{12}^T x_2 + \psi_1$$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR2}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$y_0 := \chi_1 l_{01} + y_0$$
$$\psi_1 := \chi_1 \lambda_{11} + \psi_1$$
$$y_2 := \chi_1 l_{21} + y_2$$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Figure 4.5: Algorithms to be used in Homework 4.3.2.2.

Homework 4.3.2.5 Write routines

- `[ x_out ] = Trmv_un_unb_var1 ( U, x )`; and
- `[ x_out ] = Trmv_un_unb_var2( U, x )`

that implement the algorithms for computing $x := Ux$ from Homework 4.3.2.4.

☛ SEE ANSWER

Homework 4.3.2.6 Modify the algorithms in Figure 4.7 to compute $x := Lx$, where L is a lower triangular matrix. **You may not use** y. **You have to overwrite** x **without using work space.** Hint: Think carefully about the order in which elements of x are computed and overwritten. This question is VERY tricky... You may want to do this exercise hand-in-hand with the implementation in the next homework.

☛ SEE ANSWER

Homework 4.3.2.7 Write routines

- `[ y_out ] = Trmv_ln_unb_var1 ( L, x )`; and
- `[ y_out ] = Trmv_ln_unb_var2( L, x )`

that implement the algorithms from Homework 4.3.2.6 for computing $x := Lx$.

☛ SEE ANSWER

Algorithm: $y := \text{TRMVP\_UN\_UNB\_VAR1}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$,
$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$
where U_{TL} is 0×0, x_T, y_T are 0×1
while $m(U_{TL}) < m(U)$ **do**
Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$\psi_1 := u_{10}^T x_0 + \upsilon_{11}\chi_1 + u_{12}^T x_2 + \psi_1$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{TRMVP\_UN\_UNB\_VAR2}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$,
$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$
where U_{TL} is 0×0, x_T, y_T are 0×1
while $m(U_{TL}) < m(U)$ **do**
Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$y_0 := \chi_1 u_{01} + y_0$
$\psi_1 := \chi_1 \upsilon_{11} + \psi_1$
$y_2 := \chi_1 u_{21} + y_2$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array}\right),$$
$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Figure 4.6: Algorithms to be used in Homework 4.3.2.4.

Homework 4.3.2.8 Develop algorithms for computing $y := U^T x + y$ and $y := L^T x + y$, where U and L are respectively upper triangular and lower triangular. Do not explicitly transpose matrices U and L. Write routines

- `[ y_out ] = Trmvp_ut_unb_var1 ( U, x, y )`; and
- `[ y_out ] = Trmvp_ut_unb_var2( U, x, y )`
- `[ y_out ] = Trmvp_lt_unb_var1 ( L, x, y )`; and
- `[ y_out ] = Trmvp_ln_unb_var2( L, x, y )`

that implement these algorithms.

☛ SEE ANSWER

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR1}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$\psi_1 := l_{10}^T x_0 + \lambda_{11}\chi_1 + l_{12}^T x_2 + \psi_1$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR2}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$y_0 := \chi_1 l_{01} + y_0$

$\psi_1 := \chi_1 \lambda_{11} + \psi_1$

$y_2 := \chi_1 l_{21} + y_2$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Figure 4.7: Algorithms to be used in Homework 4.3.2.6.

Homework 4.3.2.9 Develop algorithms for computing $x := U^T x$ and $x := L^T x$, where U and L are respectively upper triangular and lower triangular. Do not explicitly transpose matrices U and L. Write routines

- `[ y_out ] = Trmv_ut_unb_var1 ( U, x )`; and
- `[ y_out ] = Trmv_ut_unb_var2( U, x )`
- `[ y_out ] = Trmv_lt_unb_var1 ( L, x )`; and
- `[ y_out ] = Trmv_ln_unb_var2( L, x )`

that implement these algorithms.

☛ SEE ANSWER

Cost

Let us analyze the algorithms for computing $y := Ux + y$. (The analysis of all the other algorithms is very similar.)

For the dot product based algorithm, the cost is in the update $\psi_1 := \upsilon_{11}\chi_1 + u_{12}^T x_2 + \psi_1$ which is typically computed in two steps:

- $\psi_1 := \upsilon_{11}\chi_1 + \psi_1$; followed by
- a dot product $\psi_1 := u_{12}^T x_2 + \psi_1$.

Now, during the first iteration, u_{12}^T and x_2 are of length $n-1$, so that that iteration requires $2(n-1)+2 = 2n$ flops for the first step. During the kth iteration (starting with $k=0$), u_{12}^T and x_2 are of length $(n-k-1)$ so that the cost of that iteration is $2(n-k)$ flops. Thus, if A is an $n \times n$ matrix, then the total cost is given by

$$\sum_{k=0}^{n-1} [2(n-k)] = 2\sum_{k=0}^{n-1}(n-k) = 2(n+(n-1)+\cdots+1) = 2\sum_{k=1}^{n} k = 2(n+1)n/2.$$

flops. (Recall that we proved in the second week that $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.)

Homework 4.3.2.10 Compute the cost, in flops, of the algorithm for computing $y := Lx + y$ that uses AXPY s.

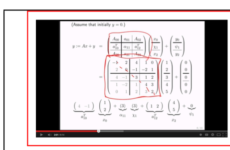
☛ SEE ANSWER

Homework 4.3.2.11 As hinted at before: Implementations achieve better performance (finish faster) if one accesses data consecutively in memory. Now, most scientific computing codes store matrices in "column-major order" which means that the first column of a matrix is stored consecutively in memory, then the second column, and so forth. Now, this means that an algorithm that accesses a matrix by columns tends to be faster than an algorithm that accesses a matrix by rows. That, in turn, means that when one is presented with more than one algorithm, one should pick the algorithm that accesses the matrix by columns.
Our FLAME notation makes it easy to recognize algorithms that access the matrix by columns. For example, in this unit, if the algorithm accesses submatrix a_{01} or a_{21} then it accesses columns. If it accesses submatrix a_{10}^T or a_{12}^T, then it accesses the matrix by rows.
For each of these, which algorithm accesses the matrix by columns:

- For $y := Ux + y$, TRSVP_UN_UNB_VAR1 or TRSVP_UN_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?
- For $y := Lx + y$, TRSVP_LN_UNB_VAR1 or TRSVP_LN_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?
- For $y := U^T x + y$, TRSVP_UT_UNB_VAR1 or TRSVP_UT_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?
- For $y := L^T x + y$, TRSVP_LT_UNB_VAR1 or TRSVP_LT_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?

☛ SEE ANSWER

4.3.3 Symmetric Matrix-Vector Multiplication



☛ YouTube

Motivation

Consider

$$\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), = \left(\begin{array}{rr|r|rr} -1 & 2 & 4 & 1 & 0 \\ 2 & 0 & -1 & -2 & 1 \\ \hline 4 & -1 & 3 & 1 & 2 \\ \hline 1 & -2 & 1 & 4 & 3 \\ 0 & 1 & 2 & 3 & 2 \end{array}\right).$$

Here we purposely chose the matrix on the right to be symmetric. We notice that $a_{10}^T = a_{01}$, $A_{20}^T = A_{02}$, and $a_{12}^T = a_{21}$. A moment of reflection will convince you that this is a general principle, when A_{00} is square. Moreover, notice that A_{00} and A_{22} are then symmetric as well.

Algorithm: $y := \text{SYMV\_U\_UNB\_VAR1}(A,x,y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where A_{TL} is 0×0, x_T, y_T are 0×1

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$\psi_1 := \overbrace{a_{10}^T}^{a_{01}^T} x_0 + \alpha_{11}\chi_1 + a_{12}^T x_2 + \psi_1$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{SYMV\_U\_UNB\_VAR2}(A,x,y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where A_{TL} is 0×0, x_T, y_T are 0×1

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$y_0 := \chi_1 a_{01} + y_0$$
$$\psi_1 := \chi_1 \alpha_{11} + \psi_1$$
$$y_2 := \chi_1 \underbrace{a_{21}}_{a_{12}} + y_2$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Figure 4.8: Algorithms for computing $y := Ax + y$ where A is symmetric, where only the upper triangular part of A is stored.

Theory

Consider

$$A = \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$

where A_{TL} and A_{00} are square matrices. If A is symmetric then

- A_{TL}, A_{BR}, A_{00}, and A_{22} are symmetric;
- $a_{10}^T = a_{01}^T$ and $a_{12}^T = a_{21}^T$; and
- $A_{20} = A_{02}^T$.

We will just state this as "intuitively obvious".

Algorithms

Consider computing $y := Ax + y$ where A is a symmetric matrix. Since the upper and lower triangular part of a symmetric matrix are simply the transpose of each other, it is only necessary to store half the matrix: only the upper triangular part or only the

Algorithm: $y := \text{SYMV\_L\_UNB\_VAR1}(A,x,y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$,
$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where A_{TL} is 0×0, x_T, y_T are 0×1

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$\psi_1 := a_{10}^T x_0 + \alpha_{11}\chi_1 + a_{12}^T x_2 + \psi_1$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{SYMV\_L\_UNB\_VAR2}(A,x,y)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$,
$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where A_{TL} is 0×0, x_T, y_T are 0×1

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$y_0 := \chi_1 a_{01} + y_0$

$\psi_1 := \chi_1 \alpha_{11} + \psi_1$

$y_2 := \chi_1 a_{21} + y_2$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Figure 4.9: Algorithms for Homework 4.3.3.2

lower triangular part. In Figure 4.8 we repeat the algorithms for matrix-vector multiplication from an earlier unit, and annotate them for the case where A is symmetric and only stored in the upper triangle. The change is simple: a_{10} and a_{21} are not stored and thus

- For the left algorithm, the update $\psi_1 := a_{10}^T x_0 + \alpha_{11}\chi_1 + a_{12}^T x_2 + \psi_1$ must be changed to $\psi_1 := a_{01}^T x_0 + \alpha_{11}\chi_1 + a_{12}^T x_2 + \psi_1$.

- For the algorithm on the right, the update $y_2 := \chi_1 a_{21} + y_2$ must be changed to $y_2 := \chi_1 a_{12} + y_2$ (or, more precisely, $y_2 := \chi_1 (a_{12}^T)^T + y_2$ since a_{12}^T is the label for part of a row).

Homework 4.3.3.1 Write routines

- `[ y_out ] = Symv_u_unb_var1 ( A, x, y )`; and
- `[ y_out ] = Symv_u_unb_var2( A, x, y )`

that implement the algorithms in Figure 4.8.

☛ SEE ANSWER

Homework 4.3.3.2 Modify the algorithms in Figure 4.9 to compute $y := Ax + y$, where A is symmetric and stored in the lower triangular part of matrix. You may want to do this in conjunction with the next exercise.

☛ SEE ANSWER

Homework 4.3.3.3 Write routines

- `[ y_out ] = Symv_l_unb_var1 ( A, x, y )`; and
- `[ y_out ] = Symv_l_unb_var2( A, x, y )`

that implement the algorithms from the previous homework.

☛ SEE ANSWER

Homework 4.3.3.4 Challenge question! As hinted at before: Implementations achieve better performance (finish faster) if one accesses data consecutively in memory. Now, most scientific computing codes store matrices in "column-major order" which means that the first column of a matrix is stored consecutively in memory, then the second column, and so forth. Now, this means that an algorithm that accesses a matrix by columns tends to be faster than an algorithm that accesses a matrix by rows. That, in turn, means that when one is presented with more than one algorithm, one should pick the algorithm that accesses the matrix by columns. Our FLAME notation makes it easy to recognize algorithms that access the matrix by columns.

The problem with the algorithms in this unit is that all of them access both part of a row AND part of a column. So, your challenge is to devise an algorithm for computing $y := Ax + y$ where A is symmetric and only stored in one half of the matrix that only accesses parts of columns. We will call these "variant 3". Then, write routines

- `[ y_out ] = Symv_u_unb_var3 ( A, x, y );` and
- `[ y_out ] = Symv_l_unb_var3( A, x, y )`

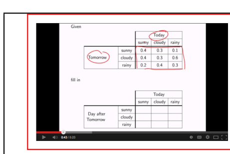
Hint: (Let's focus on the case where only the lower triangular part of A is stored.)

- If A is symmetric, then $A = L + \widehat{L}^T$ where L is the lower triangular part of A and $\widehat{L}$ is the strictly lower triangular part of A.
- Identify an algorithm for $y := Lx + y$ that accesses matrix A by columns.
- Identify an algorithm for $y := \widehat{L}^T x + y$ that accesses matrix A by columns.
- You now have two loops that together compute $y := Ax + y = (L + \widehat{L}^T)x + y = Lx + \widehat{L}^T x + y$.
- Can you "merge" the loops into one loop?

☛ SEE ANSWER

4.4 Matrix-Matrix Multiplication (Product)

4.4.1 Motivation



☛ YouTube

The first unit of the week, in which we discussed a simple model for prediction the weather, finished with the following exercise:

Given

		Today		
		sunny	cloudy	rainy
Tomorrow	sunny	0.4	0.3	0.1
	cloudy	0.4	0.3	0.6
	rainy	0.2	0.4	0.3

fill in the following table, which predicts the weather the day after tomorrow given the weather today:

		Today		
		sunny	cloudy	rainy
Day after Tomorrow	sunny			
	cloudy			
	rainy			

Now here is the hard part: Do so without using your knowledge about how to perform a matrix-matrix multiplication, since you won't learn about that until later this week...

The entries in the table turn out to be the entries in the transition matrix Q that was described just above the exercise:

$$\begin{aligned}\begin{pmatrix}\chi_s^{(2)}\\ \chi_c^{(2)}\\ \chi_r^{(2)}\end{pmatrix} &= \begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\begin{pmatrix}\chi_s^{(1)}\\ \chi_c^{(1)}\\ \chi_r^{(1)}\end{pmatrix}\\ &= \begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\left(\begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\begin{pmatrix}\chi_s^{(0)}\\ \chi_c^{(0)}\\ \chi_r^{(0)}\end{pmatrix}\right) = Q\begin{pmatrix}\chi_s^{(0)}\\ \chi_c^{(0)}\\ \chi_r^{(0)}\end{pmatrix},\end{aligned}$$

Now, those of you who remembered from, for example, some other course that

$$\begin{aligned}&\begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\left(\begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\begin{pmatrix}\chi_s^{(0)}\\ \chi_c^{(0)}\\ \chi_r^{(0)}\end{pmatrix}\right)\\ &\qquad = \left(\begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\right)\begin{pmatrix}\chi_s^{(0)}\\ \chi_c^{(0)}\\ \chi_r^{(0)}\end{pmatrix}\end{aligned}$$

would recognize that

$$Q = \begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}\begin{pmatrix}0.4 & 0.3 & 0.1\\ 0.4 & 0.3 & 0.6\\ 0.2 & 0.4 & 0.3\end{pmatrix}.$$

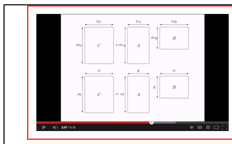
And, if you then remembered how to perform a matrix-matrix multiplication (or you did `P * P` in Python), you would have deduced that

$$Q = \begin{pmatrix}0.3 & 0.25 & 0.25\\ 0.4 & 0.45 & 0.4\\ 0.3 & 0.3 & 0.35\end{pmatrix}.$$

These then become the entries in the table. If you knew all the above, well, GOOD FOR YOU!

However, there are all kinds of issues that one really should discuss. How do you know such a matrix exists? Why is matrix-matrix multiplication defined this way? We answer that in the next few units.

4.4.2 From Composing Linear Transformations to Matrix-Matrix Multiplication



☛ YouTube

Homework 4.4.2.1 Let $L_A : \mathbb{R}^k \to \mathbb{R}^m$ and $L_B : \mathbb{R}^n \to \mathbb{R}^k$ both be linear transformations and, for all $x \in \mathbb{R}^n$, define the function $L_C : \mathbb{R}^n \to \mathbb{R}^m$ by $L_C(x) = L_A(L_B(x))$. $L_C(x)$ is a linear transformations.

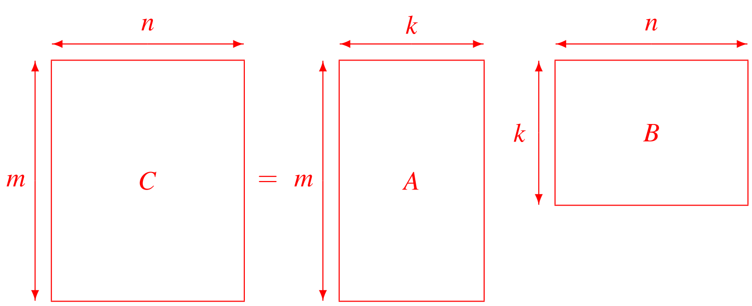
Always/Sometimes/Never

☛ SEE ANSWER

Now, let linear transformations L_A, L_B, and L_C be represented by matrices $A \in \mathbb{R}^{m\times k}$, $B \in \mathbb{R}^{k\times n}$, and $C \in \mathbb{R}^{m\times n}$, respectively. (You know such matrices exist since L_A, L_B, and L_C are linear transformations.) Then $Cx = L_C(x) = L_A(L_B(x)) = A(Bx)$.

The matrix-matrix multiplication (product) is defined as the matrix C such that, for all vectors x, $Cx = A(B(x))$. The notation used to denote that matrix is $C = A \times B$ or, equivalently, $C = AB$. The operation AB is called a matrix-matrix multiplication or product.

If A is $m_A \times n_A$ matrix, B is $m_B \times n_B$ matrix, and C is $m_C \times n_C$ matrix, then for $C = AB$ to hold it must be the case that $m_C = m_A$, $n_C = n_B$, and $n_A = m_B$. Usually, the integers m and n are used for the sizes of C: $C \in \mathbb{R}^{m\times n}$ and k is used for the "other size": $A \in \mathbb{R}^{m\times k}$ and $B \in \mathbb{R}^{k\times n}$:



Homework 4.4.2.2 Let $A \in \mathbb{R}^{m\times n}$. A^TA is well-defined. (By well-defined we mean that A^TA makes sense. In this particular case this means that the dimensions of A^T and A are such that A^TA can be computed.)

Always/Sometimes/Never

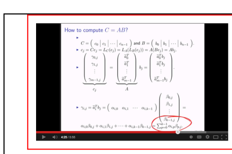
☛ SEE ANSWER

Homework 4.4.2.3 Let $A \in \mathbb{R}^{m\times n}$. AA^T is well-defined.

Always/Sometimes/Never

☛ SEE ANSWER

4.4.3 Computing the Matrix-Matrix Product



☛ YouTube

The question now becomes how to compute C given matrices A and B. For this, we are going to use and abuse the unit basis vectors e_j.

Consider the following. Let

- $C \in \mathbb{R}^{m\times n}$, $A \in \mathbb{R}^{m\times k}$, and $B \in \mathbb{R}^{k\times n}$; and
- $C = AB$; and
- $L_C : \mathbb{R}^n \to \mathbb{R}^m$ equal the linear transformation such that $L_C(x) = Cx$; and
- $L_A : \mathbb{R}^k \to \mathbb{R}^m$ equal the linear transformation such that $L_A(x) = Ax$.
- $L_B : \mathbb{R}^n \to \mathbb{R}^k$ equal the linear transformation such that $L_B(x) = Bx$; and
- e_j denote the jth unit basis vector; and
- c_j denote the jth column of C; and
- b_j denote the jth column of B.

Then

$$c_j = Ce_j = L_C(e_j) = L_A(L_B(e_j)) = L_A(Be_j) = L_A(b_j) = Ab_j.$$

From this we learn that

If $C = AB$ then the jth column of C, c_j, equals Ab_j, where b_j is the jth column of B.

Since by now you should be very comfortable with partitioning matrices by columns, we can summarize this as

$$\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array} \right) = C = AB = A \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} Ab_0 & Ab_1 & \cdots & Ab_{n-1} \end{array} \right).$$

Now, let's expose the elements of C, A, and B.

$$C = \begin{pmatrix} \gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1} \\ \gamma_{1,0} & \gamma_{1,1} & \cdots & \gamma_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \gamma_{m-1,0} & \gamma_{m-1,1} & \cdots & \gamma_{m-1,n-1} \end{pmatrix}, \quad A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,k-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,k-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,k-1} \end{pmatrix},$$

$$\text{and} \quad B = \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \beta_{k-1,0} & \beta_{k-1,1} & \cdots & \beta_{k-1,n-1} \end{pmatrix}.$$

We are going to show that

$$\gamma_{i,j} = \sum_{p=0}^{k-1} \alpha_{i,p}\beta_{p,j},$$

which you may have learned in a high school algebra course.

We reasoned that $c_j = Ab_j$:

$$\begin{pmatrix} \gamma_{0,j} \\ \gamma_{1,j} \\ \vdots \\ \boxed{\gamma_{i,j}} \\ \vdots \\ \gamma_{m-1,j} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,k-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,k-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{i,0} & \alpha_{i,1} & \cdots & \alpha_{i,k-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,k-1} \end{pmatrix} \left(\boxed{\begin{matrix} \beta_{0,j} \\ \beta_{1,j} \\ \vdots \\ \beta_{k-1,j} \end{matrix}}\right).$$

Here we highlight the ith element of c_j, $\gamma_{i,j}$, and the ith row of A. We recall that the ith element of Ax equals the dot product of the ith row of A with the vector x. Thus, $\gamma_{i,j}$ equals the dot product of the ith row of A with the vector b_j:

$$\gamma_{i,j} = \sum_{p=0}^{k-1} \alpha_{i,p}\beta_{p,j}.$$

Let $A \in \mathbb{R}^{m\times k}$, $B \in \mathbb{R}^{k\times n}$, and $C \in \mathbb{R}^{m\times n}$. Then the matrix-matrix multiplication (product) $C = AB$ is computed by

$$\gamma_{i,j} = \sum_{p=0}^{k-1} \alpha_{i,p}\beta_{p,j} = \alpha_{i,0}\beta_{0,j} + \alpha_{i,1}\beta_{1,j} + \cdots + \alpha_{i,k-1}\beta_{k-1,j}.$$

As a result of this definition $Cx = A(Bx) = (AB)x$ and can drop the parentheses, unless they are useful for clarity: $Cx = ABx$ and $C = AB$.

Homework 4.4.3.1 Compute

$$Q = P \times P = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix}$$

☛ SEE ANSWER

We emphasize that for matrix-matrix multiplication to be a legal operations, the row and column dimensions of the matrices must obey certain constraints. Whenever we talk about dimensions being *conformal*, we mean that the dimensions are such that the encountered matrix multiplications are valid operations.

Homework 4.4.3.2 Let $A = \begin{pmatrix} 2 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 3 & 1 \\ -1 & 1 & 1 \end{pmatrix}$ and $B = \begin{pmatrix} 2 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$. Compute

- $AB =$
- $BA =$

☛ SEE ANSWER

Homework 4.4.3.3 Let $A \in \mathbb{R}^{m\times k}$ and $B \in \mathbb{R}^{k\times n}$ and $AB = BA$. A and B are square matrices.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 4.4.3.4 Let $A \in \mathbb{R}^{m\times k}$ and $B \in \mathbb{R}^{k\times n}$.

$$AB = BA.$$

Always/Sometimes/Never
☛ SEE ANSWER

Homework 4.4.3.5 Let $A, B \in \mathbb{R}^{n\times n}$. $AB = BA$.

Always/Sometimes/Never
☛ SEE ANSWER

Homework 4.4.3.6 A^2 is defined as AA. Similarly $A^k = \underbrace{AA\cdots A}_{k \text{ occurrences of } A}$. Consistent with this, $A^0 = I$ so that

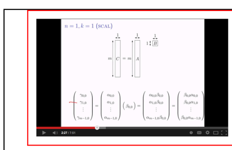
$A^k = A^{k-1}A$ for $k > 0$.
A^k is well-defined only if A is a square matrix.

True/False
☛ SEE ANSWER

Homework 4.4.3.7 Let A, B, C be matrix "of appropriate size" so that $(AB)C$ is well defined. $A(BC)$ is well defined.

Always/Sometimes/Never
☛ SEE ANSWER

4.4.4 Special Shapes



☛ YouTube

We now show that if one treats scalars, column vectors, and row vectors as special cases of matrices, then many (all?) operations we encountered previously become simply special cases of matrix-matrix multiplication. In the below discussion, consider $C = AB$ where $C \in \mathbb{R}^{m\times n}$, $A \in \mathbb{R}^{m\times k}$, and $B \in \mathbb{R}^{k\times n}$.

$m = n = k = 1$ **(scalar multiplication)**

$$1 \updownarrow \overset{\overset{1}{\leftrightarrow}}{\boxed{C}} = 1 \updownarrow \overset{\overset{1}{\leftrightarrow}}{\boxed{A}} \quad 1 \updownarrow \overset{\overset{1}{\leftrightarrow}}{\boxed{B}}$$

In this case, all three matrices are actually scalars:

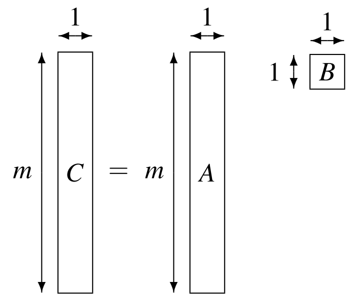
$$\left(\begin{array}{c} \gamma_{0,0} \end{array}\right) = \left(\begin{array}{c} \alpha_{0,0} \end{array}\right)\left(\begin{array}{c} \beta_{0,0} \end{array}\right) = \left(\begin{array}{c} \alpha_{0,0}\beta_{0,0} \end{array}\right)$$

so that matrix-matrix multiplication becomes scalar multiplication.

Homework 4.4.4.1 Let $A = \left(\begin{array}{c} 4 \end{array}\right)$ and $B = \left(\begin{array}{c} 3 \end{array}\right)$. Then $AB =$ ___.

☛ SEE ANSWER

$n = 1, k = 1$ (SCAL)



Now the matrices look like

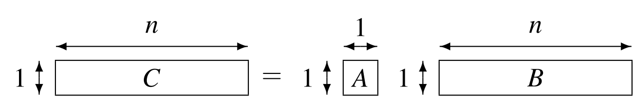
$$\begin{pmatrix} \gamma_{0,0} \\ \gamma_{1,0} \\ \vdots \\ \gamma_{m-1,0} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \\ \vdots \\ \alpha_{m-1,0} \end{pmatrix} \begin{pmatrix} \beta_{0,0} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0}\beta_{0,0} \\ \alpha_{1,0}\beta_{0,0} \\ \vdots \\ \alpha_{m-1,0}\beta_{0,0} \end{pmatrix} = \begin{pmatrix} \beta_{0,0}\alpha_{0,0} \\ \beta_{0,0}\alpha_{1,0} \\ \vdots \\ \beta_{0,0}\alpha_{m-1,0} \end{pmatrix} = \beta_{0,0} \begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \\ \vdots \\ \alpha_{m-1,0} \end{pmatrix}.$$

In other words, C and A are vectors, B is a scalar, and the matrix-matrix multiplication becomes scaling of a vector.

Homework 4.4.4.2 Let $A = \begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix}$ and $B = \begin{pmatrix} 4 \end{pmatrix}$. Then $AB =$.

☛ SEE ANSWER

$m = 1, k = 1$ (SCAL)



Now the matrices look like

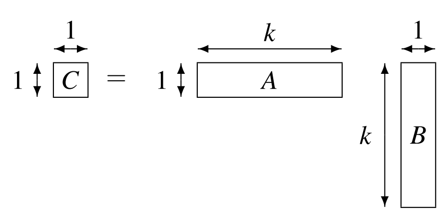
$$\begin{aligned} \begin{pmatrix} \gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1} \end{pmatrix} &= \begin{pmatrix} \alpha_{0,0} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \end{pmatrix} \\ &= \alpha_{0,0} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \end{pmatrix} \\ &= \begin{pmatrix} \alpha_{0,0}\beta_{0,0} & \alpha_{0,0}\beta_{0,1} & \cdots & \alpha_{0,0}\beta_{0,n-1} \end{pmatrix}. \end{aligned}$$

In other words, C and B are just row vectors and A is a scalar. The vector C is computed by scaling the row vector B by the scalar A.

Homework 4.4.4.3 Let $A = \begin{pmatrix} 4 \end{pmatrix}$ and $B = \begin{pmatrix} 1 & -3 & 2 \end{pmatrix}$. Then $AB =$.

☛ SEE ANSWER

$m = 1, n = 1$ (DOT)



The matrices look like

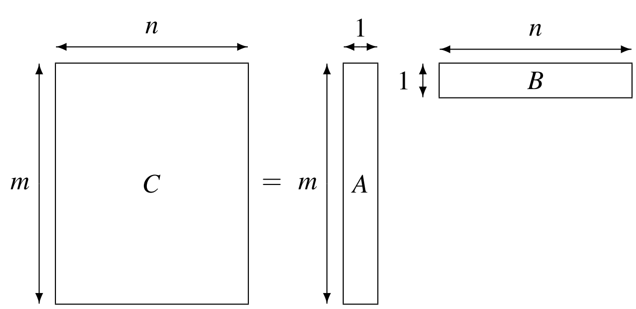
$$\left(\begin{array}{c}\gamma_{0,0}\end{array}\right) = \left(\begin{array}{cccc}\alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,k-1}\end{array}\right)\left(\begin{array}{c}\beta_{0,0}\\ \beta_{1,0}\\ \vdots\\ \beta_{k-1,0}\end{array}\right) = \sum_{p=0}^{k-1}\alpha_{0,p}\beta_{p,0}.$$

In other words, C is a scalar that is computed by taking the dot product of the one row that is A and the one column that is B.

Homework 4.4.4.4 Let $A = \left(\begin{array}{ccc}1 & -3 & 2\end{array}\right)$ and $B = \left(\begin{array}{c}2\\ -1\\ 0\end{array}\right)$. Then $AB =$

☛ SEE ANSWER

$k = 1$ **(outer product)**



$$\left(\begin{array}{cccc}\gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1}\\ \gamma_{1,0} & \gamma_{1,1} & \cdots & \gamma_{1,n-1}\\ \vdots & \vdots & \ddots & \vdots\\ \gamma_{m-1,0} & \gamma_{m-1,1} & \cdots & \gamma_{m-1,n-1}\end{array}\right) = \left(\begin{array}{c}\alpha_{0,0}\\ \alpha_{1,0}\\ \vdots\\ \alpha_{m-1,0}\end{array}\right)\left(\begin{array}{cccc}\beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1}\end{array}\right)$$

$$= \left(\begin{array}{cccc}\alpha_{0,0}\beta_{0,0} & \alpha_{0,0}\beta_{0,1} & \cdots & \alpha_{0,0}\beta_{0,n-1}\\ \alpha_{1,0}\beta_{0,0} & \alpha_{1,0}\beta_{0,1} & \cdots & \alpha_{1,0}\beta_{0,n-1}\\ \vdots & \vdots & \ddots & \vdots\\ \alpha_{m-1,0}\beta_{0,0} & \alpha_{m-1,0}\beta_{0,1} & \cdots & \alpha_{m-1,0}\beta_{0,n-1}\end{array}\right)$$

Homework 4.4.4.5 Let $A = \left(\begin{array}{c}1\\ -3\\ 2\end{array}\right)$ and $B = \left(\begin{array}{cc}-1 & -2\end{array}\right)$. Then $AB =$

☛ SEE ANSWER

Homework 4.4.4.6 Let $a = \begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix}$ and $b^T = \begin{pmatrix} -1 & -2 \end{pmatrix}$ and $C = ab^T$. Partition C by columns and by rows:

$$C = \left(c_0 \mid c_1 \right) \quad \text{and} \quad C = \begin{pmatrix} \widetilde{c}_0^T \\ \widetilde{c}_1^T \\ \widetilde{c}_2^T \end{pmatrix}$$

Then

- $c_0 = (-1)\begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix} = \begin{pmatrix} (-1)\times(1) \\ (-1)\times(-3) \\ (-1)\times(2) \end{pmatrix}$ True/False
- $c_1 = (-2)\begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix} = \begin{pmatrix} (-2)\times(1) \\ (-2)\times(-3) \\ (-2)\times(2) \end{pmatrix}$ True/False
- $C = \left(\begin{array}{c|c} (-1)\times(1) & (-2)\times(1) \\ (-1)\times(-3) & (-2)\times(-3) \\ (-1)\times(2) & (-2)\times(2) \end{array}\right)$ True/False
- $\widetilde{c}_0^T = (1)\begin{pmatrix} -1 & -2 \end{pmatrix} = \begin{pmatrix} (1)\times(-1) & (1)\times(-2) \end{pmatrix}$ True/False
- $\widetilde{c}_1^T = (-3)\begin{pmatrix} -1 & -2 \end{pmatrix} = \begin{pmatrix} (-3)\times(-1) & (-3)\times(-2) \end{pmatrix}$ True/False
- $\widetilde{c}_2^T = (2)\begin{pmatrix} -1 & -2 \end{pmatrix} = \begin{pmatrix} (2)\times(-1) & (2)\times(-2) \end{pmatrix}$ True/False
- $C = \left(\begin{array}{cc} (-1)\times(1) & (-2)\times(1) \\ \hline (-1)\times(-3) & (-2)\times(-3) \\ \hline (-1)\times(2) & (-2)\times(2) \end{array}\right)$ True/False

☛ SEE ANSWER

Homework 4.4.4.7 Fill in the boxes:

$$\begin{pmatrix} \square \\ \square \\ \square \\ \square \end{pmatrix} \begin{pmatrix} 2 & -1 & 3 \end{pmatrix} = \begin{pmatrix} 4 & \square & \square \\ -2 & \square & \square \\ 2 & \square & \square \\ 6 & \square & \square \end{pmatrix}$$

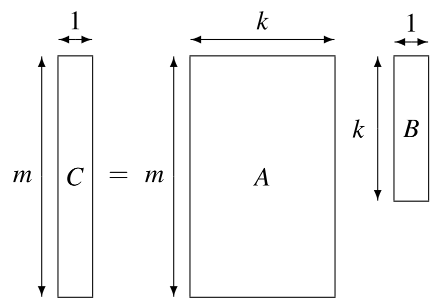
☛ SEE ANSWER

Homework 4.4.4.8 Fill in the boxes:

$$\begin{pmatrix} 2 \\ -1 \\ 1 \\ 3 \end{pmatrix} \begin{pmatrix} \square & \square & \square \end{pmatrix} = \begin{pmatrix} 4 & -2 & 6 \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \end{pmatrix}$$

☛ SEE ANSWER

$n = 1$ **(matrix-vector product)**

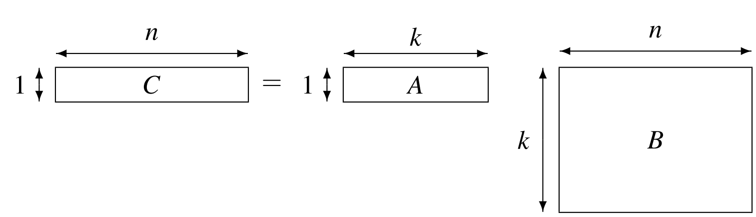


$$\begin{pmatrix} \gamma_{0,0} \\ \gamma_{1,0} \\ \vdots \\ \gamma_{m-1,0} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,k-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,k-1} \end{pmatrix} \begin{pmatrix} \beta_{0,0} \\ \beta_{1,0} \\ \vdots \\ \beta_{k-1,0} \end{pmatrix}$$

We have studied this special case in great detail. To emphasize how it relates to have matrix-matrix multiplication is computed, consider the following:

$$\begin{pmatrix} \gamma_{0,0} \\ \vdots \\ \boxed{\gamma_{i,0}} \\ \vdots \\ \gamma_{m-1,0} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{i,0} & \alpha_{i,1} & \cdots & \alpha_{i,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,k-1} \end{pmatrix} \begin{pmatrix} \boxed{\begin{matrix} \beta_{0,0} \\ \beta_{1,0} \\ \vdots \\ \beta_{k-1,0} \end{matrix}} \end{pmatrix}$$

$m = 1$ **(row vector-matrix product)**



$$\begin{pmatrix} \gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,k-1} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{k-1,0} & \beta_{k-1,1} & \cdots & \beta_{k-1,n-1} \end{pmatrix}$$

so that $\gamma_{0,j} = \sum_{p=0}^{k-1} \alpha_{0,p}\beta_{p,j}$. To emphasize how it relates to have matrix-matrix multiplication is computed, consider the following:

$$\begin{pmatrix} \gamma_{0,0} & \cdots & \boxed{\gamma_{0,j}} & \cdots & \gamma_{0,n-1} \end{pmatrix}$$
$$= \left(\boxed{\alpha_{0,0} \quad \alpha_{0,1} \quad \cdots \quad \alpha_{0,k-1}} \right) \begin{pmatrix} \beta_{0,0} & \cdots & \beta_{0,j} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \cdots & \beta_{1,j} & \cdots & \beta_{1,n-1} \\ \vdots & & \vdots & & \vdots \\ \beta_{k-1,0} & \cdots & \beta_{k-1,j} & \cdots & \beta_{k-1,n-1} \end{pmatrix}.$$

Homework 4.4.4.9 Let $A = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$ and $B = \begin{pmatrix} 1 & -2 & 2 \\ 4 & 2 & 0 \\ 1 & 2 & 3 \end{pmatrix}$. Then $AB =$

☛ SEE ANSWER

Homework 4.4.4.10 Let $e_i \in \mathbb{R}^m$ equal the ith unit basis vector and $A \in \mathbb{R}^{m\times n}$. Then $e_i^T A = \check{a}_i^T$, the ith row of A.

Always/Sometimes/Never

☛ SEE ANSWER

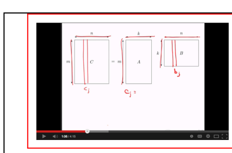
Homework 4.4.4.11 Get as much practice as you want with the MATLAB script in

```
LAFF-2.0xM/Programming/Week04/PracticeGemm.m
```

☛ SEE ANSWER

If you understand how to perform a matrix-matrix multiplication, then you know how to perform all other operations with matrices and vectors that we have encountered so far.

4.4.5 Cost



☛ YouTube

Consider the matrix-matrix multiplication $C = AB$ where $C \in \mathbb{R}^{m\times n}$, $A \in \mathbb{R}^{m\times k}$, and $B \in \mathbb{R}^{k\times n}$. Let us examine what the cost of this operation is:

- We argued that, by definition, the jth column of C, c_j, is computed by the matrix-vector multiplication Ab_j, where b_j is the jth column of B.

- Last week we learned that a matrix-vector multiplication of a $m \times k$ matrix times a vector of size k requires $2mk$ floating point operations (flops).

- C has n columns (since it is a $m \times n$ matrix.).

Putting all these observations together yields a cost of

$$n \times (2mk) = 2mnk \text{ flops.}$$

Try this! Recall that the dot product of two vectors of size k requires (approximately) $2k$ flops. We learned in the previous units that if $C = AB$ then $\gamma_{i,j}$ equals the dot product of the ith row of A and the jth column of B. Use this to give an alternative justification that a matrix multiplication requires $2mnk$ flops.

4.5 Enrichment

4.5.1 Markov Chains: Their Application

Matrices have many "real world" applications. As we have seen this week, one noteworthy use is connected to Markov chains. There are many, many examples of the use of Markov chains. You can find a brief look at some significant applications in THE FIVE GREATEST APPLICATIONS OF MARKOV CHAINS by Philipp von Hilgers and Amy N. Langville. (http://langvillea.people.cofc.edu/MCapps7.pdf).

4.6 Wrap Up

4.6.1 Homework

Homework 4.6.1.1 Let $A \in \mathbb{R}^{m\times n}$ and $x \in \mathbb{R}^n$. Then $(Ax)^T = x^T A^T$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 4.6.1.2 Our `laff` library has a routine

```
laff_gemv( trans, alpha, A, x, beta, y )
```

that has the following property

- `laff_gemv( 'No transpose', alpha, A, x, beta, y )` computes $y := \alpha Ax + \beta y$.

- `laff_gemv( 'Transpose', alpha, A, x, beta, y )` computes $y := \alpha A^T x + \beta y$.

The routine works regardless of whether x and/or y are column and/or row vectors.
Our library does NOT include a routine to compute $y^T := x^T A$. What call could you use to compute $y^T := x^T A$ if y^T is stored in `yt` and x^T in `xt`?

- `laff_gemv( 'No transpose', 1.0, A, xt, 0.0, yt ).`

- `laff_gemv( 'No transpose', 1.0, A, xt, 1.0, yt ).`

- `laff_gemv( 'Transpose', 1.0, A, xt, 1.0, yt ).`

- `laff_gemv( 'Transpose', 1.0, A, xt, 0.0, yt ).`

☛ SEE ANSWER

Homework 4.6.1.3 Let $A = \begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix}$. Compute

- $A^2 =$
- $A^3 =$
- For $k > 1$, $A^k =$

☛ SEE ANSWER

Homework 4.6.1.4 Let $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

- $A^2 =$
- $A^3 =$
- For $n \geq 0$, $A^{2n} =$
- For $n \geq 0$, $A^{2n+1} =$

☛ SEE ANSWER

Homework 4.6.1.5 Let $A = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$.

- $A^2 =$
- $A^3 =$
- For $n \geq 0$, $A^{4n} =$
- For $n \geq 0$, $A^{4n+1} =$

☛ SEE ANSWER

Homework 4.6.1.6 Let A be a square matrix. If $AA = 0$ (the zero matrix) then A is a zero matrix. (AA is often written as A^2.)

True/False

☛ SEE ANSWER

Homework 4.6.1.7 There exists a real valued matrix A such that $A^2 = -I$. (Recall: I is the identity)

True/False

☛ SEE ANSWER

Homework 4.6.1.8 There exists a matrix A that is not diagonal such that $A^2 = I$.

True/False

☛ SEE ANSWER

4.6.2 Summary

Partitioned matrix-vector multiplication

$$\left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{array}\right)\left(\begin{array}{c} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{array}\right) = \left(\begin{array}{c} A_{0,0}x_0 + A_{0,1}x_1 + \cdots + A_{0,N-1}x_{N-1} \\ \hline A_{1,0}x_0 + A_{1,1}x_1 + \cdots + A_{1,N-1}x_{N-1} \\ \hline \vdots \\ \hline A_{M-1,0}x_0 + A_{M-1,1}x_1 + \cdots + A_{M-1,N-1}x_{N-1} \end{array}\right).$$

Transposing a partitioned matrix

$$\left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,N-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{array}\right)^T = \left(\begin{array}{c|c|c|c} A_{0,0}^T & A_{1,0}^T & \cdots & A_{M-1,0}^T \\ \hline A_{0,1}^T & A_{1,1}^T & \cdots & A_{M-1,1}^T \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{0,N-1}^T & A_{1,N-1}^T & \cdots & A_{M-1,N-1}^T \end{array}\right).$$

Composing linear transformations

Let $L_A : \mathbb{R}^k \to \mathbb{R}^m$ and $L_B : \mathbb{R}^n \to \mathbb{R}^k$ both be linear transformations and, for all $x \in \mathbb{R}^n$, define the function $L_C : \mathbb{R}^n \to \mathbb{R}^m$ by $L_C(x) = L_A(L_B(x))$. Then $L_C(x)$ is a linear transformations.

Matrix-matrix multiplication

$$AB = A\left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right) = \left(\begin{array}{c|c|c|c} Ab_0 & Ab_1 & \cdots & Ab_{n-1} \end{array}\right).$$

If

$$C = \begin{pmatrix} \gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1} \\ \gamma_{1,0} & \gamma_{1,1} & \cdots & \gamma_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \gamma_{m-1,0} & \gamma_{m-1,1} & \cdots & \gamma_{m-1,n-1} \end{pmatrix}, \quad A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,k-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,k-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,k-1} \end{pmatrix},$$

$$\text{and} \quad B = \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \beta_{k-1,0} & \beta_{k-1,1} & \cdots & \beta_{k-1,n-1} \end{pmatrix}.$$

then $C = AB$ means that $\gamma_{i,j} = \sum_{p=0}^{k-1} \alpha_{i,p}\beta_{p,j}$.

A table of matrix-matrix multiplications with matrices of special shape is given at the end of this week.

Outer product

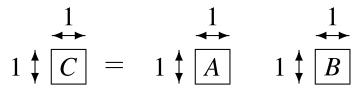
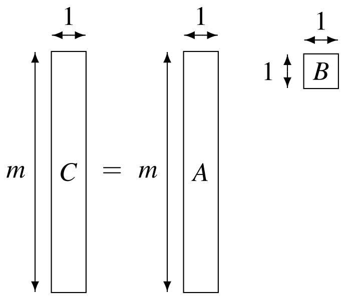
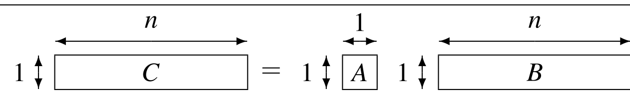
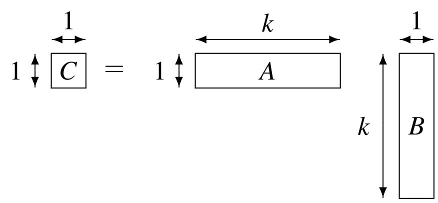
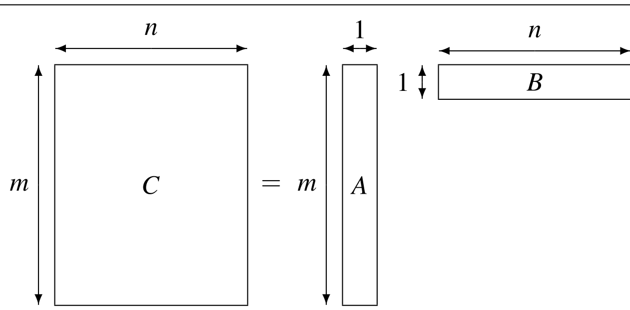
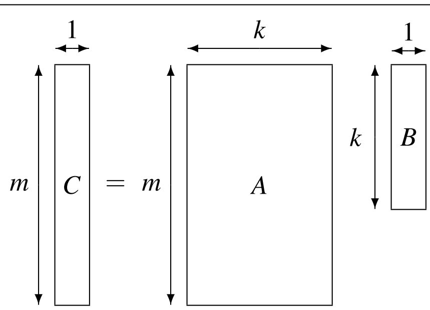
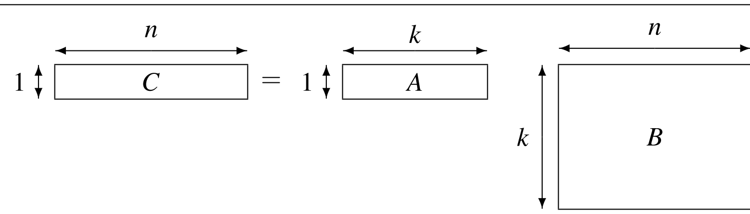
Let $x \in \mathbb{R}^m$ and $y \in \mathbb{R}^n$. Then the *outer product* of x and y is given by xy^T. Notice that this yields an $m \times n$ matrix:

$$xy^T = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{m-1} \end{pmatrix}\begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix}^T = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{m-1} \end{pmatrix}\begin{pmatrix} \psi_0 & \psi_1 & \cdots & \psi_{n-1} \end{pmatrix}$$

$$= \begin{pmatrix} \chi_0\psi_0 & \chi_0\psi_1 & \cdots & \chi_0\psi_{n-1} \\ \chi_1\psi_0 & \chi_1\psi_1 & \cdots & \chi_1\psi_{n-1} \\ \vdots & \vdots & & \vdots \\ \chi_{m-1}\psi_0 & \chi_{m-1}\psi_1 & \cdots & \chi_{m-1}\psi_{n-1} \end{pmatrix}.$$

LAFF routines

Operation Abbrev.	Definition	Function	Approx. cost	
		`laff_`	flops	memops
Vector-vector operations				
Copy (COPY)	$y := x$	`copy( x, y )`	0	$2n$
Vector scaling (SCAL)	$x := \alpha x$	`scal( alpha, x )`	n	$2n$
Vector scaling (SCAL)	$x := x/\alpha$	`invscal( alpha, x )`	n	$2n$
Scaled addition (AXPY)	$y := \alpha x + y$	`axpy( alpha, x, y )`	$2n$	$3n$
Dot product (DOT)	$\alpha := x^T y$	`alpha = dot( x, y )`	$2n$	$2n$
Dot product (DOTS)	$\alpha := x^T y + \alpha$	`dots( x, y, alpha )`	$2n$	$2n$
Length (NORM2)	$\alpha := \|x\|_2$	`alpha = norm2( x )`	$2n$	n
Matrix-vector operations				
General matrix-vector	$y := \alpha A x + \beta y$	`gemv( 'No transpose', alpha, A, x, beta, y )`	$2mn$	mn
multiplication (GEMV)	$y := \alpha A^T x + \beta y$	`gemv( 'Transpose', alpha, A, x, beta, y )`	$2mn$	mn
Rank-1 update (GER)	$A := \alpha x y^T + A$	`ger( alpha, x, y, A )`	$2mn$	mn

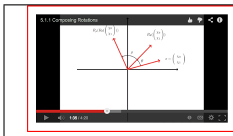
m	n	k	Shape	Comment
1	1	1		Scalar multiplication
m	1	1		Vector times scalar = scalar times vector
1	n	1		Scalar times row vector
1	1	k		Dot product (with row and column)
m	n	1		Outer product
m	1	k		Matrix-vector multiplication
1	n	k		Row vector times matrix multiply

Week 5

Matrix-Matrix Multiplication

5.1 Opening Remarks

5.1.1 Composing Rotations



☛ YouTube

Homework 5.1.1.1 Which of the following statements are *true*:

- $$\left(\begin{array}{c} \cos(\rho+\sigma+\tau) \\ \sin(\rho+\sigma+\tau) \end{array}\right) = \left(\begin{array}{c|c} \cos(\tau) & -\sin(\tau) \\ \hline \sin(\tau) & \cos(\tau) \end{array}\right) \left(\begin{array}{c} \cos(\rho+\sigma) \\ \sin(\rho+\sigma) \end{array}\right)$$

  True/False

- $$\left(\begin{array}{c} \cos(\rho+\sigma+\tau) \\ \sin(\rho+\sigma+\tau) \end{array}\right) = \left(\begin{array}{c|c} \cos(\tau) & -\sin(\tau) \\ \hline \sin(\tau) & \cos(\tau) \end{array}\right) \left(\begin{array}{c} \cos\rho\cos\sigma - \sin\rho\sin\sigma \\ \sin\rho\cos\sigma + \cos\rho\sin\sigma. \end{array}\right)$$

  True/False

- $$\begin{array}{rcl} \cos(\rho+\sigma+\tau) & = & \cos(\tau)(\cos\rho\cos\sigma - \sin\rho\sin\sigma) - \sin(\tau)(\sin\rho\cos\sigma + \cos\rho\sin\sigma) \\ \sin(\rho+\sigma+\tau) & = & \sin(\tau)(\cos\rho\cos\sigma - \sin\rho\sin\sigma) + \cos(\tau)(\sin\rho\cos\sigma + \cos\rho\sin\sigma) \end{array}$$

  True/False

☛ SEE ANSWER

5.1.2 Outline

5.1.3 What You Will Learn

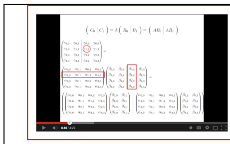
Upon completion of this unit, you should be able to

- **Recognize that matrix-matrix multiplication is not commutative.**
- Relate composing rotations to matrix-matrix multiplication.
- Fluently compute a matrix-matrix multiplication.
- Perform matrix-matrix multiplication with partitioned matrices.
- Identify, apply, and prove properties of matrix-matrix multiplication, such as $(AB)^T = B^T A^T$.
- Exploit special structure of matrices to perform matrix-matrix multiplication with special matrices, such as identity, triangular, and diagonal matrices.
- Identify whether or not matrix-matrix multiplication preserves special properties in matrices, such as symmetric and triangular structure.
- Express a matrix-matrix multiplication in terms of matrix-vector multiplications, row vector times matrix multiplications, and rank-1 updates.
- Appreciate how partitioned matrix-matrix multiplication enables high performance. (Optional, as part of the enrichment.)

Track your progress in Appendix B.

5.2 Observations

5.2.1 Partitioned Matrix-Matrix Multiplication



☛ YouTube

Theorem 5.1 *Let* $C \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{m \times k}$, *and* $B \in \mathbb{R}^{k \times n}$. *Let*

- $m = m_0 + m_1 + \cdots m_{M-1}$, $m_i \geq 0$ *for* $i = 0, \ldots, M-1$;
- $n = n_0 + n_1 + \cdots n_{N-1}$, $n_j \geq 0$ *for* $j = 0, \ldots, N-1$; *and*
- $k = k_0 + k_1 + \cdots k_{K-1}$, $k_p \geq 0$ *for* $p = 0, \ldots, K-1$.

Partition

$$C = \left(\begin{array}{c|c|c|c} C_{0,0} & C_{0,1} & \cdots & C_{0,N-1} \\ \hline C_{1,0} & C_{1,1} & \cdots & C_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline C_{M-1,0} & C_{M-1,1} & \cdots & C_{M-1,N-1} \end{array} \right), A = \left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,K-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,K-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,K-1} \end{array} \right),$$

$$\text{and } B = \left(\begin{array}{c|c|c|c} B_{0,0} & B_{0,1} & \cdots & B_{0,N-1} \\ \hline B_{1,0} & B_{1,1} & \cdots & B_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline B_{K-1,0} & B_{K-1,1} & \cdots & B_{K-1,N-1} \end{array} \right),$$

with $C_{i,j} \in \mathbb{R}^{m_i \times n_j}$, $A_{i,p} \in \mathbb{R}^{m_i \times k_p}$, *and* $B_{p,j} \in \mathbb{R}^{k_p \times n_j}$. *Then* $C_{i,j} = \sum_{p=0}^{K-1} A_{i,p} B_{p,j}$.

If one partitions matrices C, A, and B into blocks, **and** one makes sure the dimensions match up, **then** blocked matrix-matrix multiplication proceeds exactly as does a regular matrix-matrix multiplication **except** that individual multiplications of scalars commute while (in general) individual multiplications with matrix blocks (submatrices) do not.

Example 5.2 Consider

$$A = \left(\begin{array}{rr|rr} -1 & 2 & 4 & 1 \\ 1 & 0 & -1 & -2 \\ 2 & -1 & 3 & 1 \\ 1 & 2 & 3 & 4 \end{array}\right), B = \left(\begin{array}{rrr} -2 & 2 & -3 \\ 0 & 1 & -1 \\ \hline -2 & -1 & 0 \\ 4 & 0 & 1 \end{array}\right), \text{ and } \quad AB = \begin{pmatrix} -2 & -4 & 2 \\ -8 & 3 & -5 \\ -6 & 0 & -4 \\ 8 & 1 & -1 \end{pmatrix}:$$

If

$$A_0 = \begin{pmatrix} -1 & 2 \\ 1 & 0 \\ 2 & -1 \\ 1 & 2 \end{pmatrix}, A_1 = \begin{pmatrix} 4 & 1 \\ -1 & -2 \\ 3 & 1 \\ 3 & 4 \end{pmatrix}, B_0 \begin{pmatrix} -2 & 2 & -3 \\ 0 & 1 & -1 \end{pmatrix}, \text{ and } B_1 = \begin{pmatrix} -2 & -1 & 0 \\ 4 & 0 & 1 \end{pmatrix}.$$

Then

$$AB = \begin{pmatrix} A_0 & A_1 \end{pmatrix} \begin{pmatrix} B_0 \\ B_1 \end{pmatrix} = A_0 B_0 + A_1 B_1 :$$

$$\begin{aligned}
&\underbrace{\left(\begin{array}{rr|rr} -1 & 2 & 4 & 1 \\ 1 & 0 & -1 & -2 \\ 2 & -1 & 3 & 1 \\ 1 & 2 & 3 & 4 \end{array}\right)}_{A} \underbrace{\left(\begin{array}{rrr} -2 & 2 & -3 \\ 0 & 1 & -1 \\ \hline -2 & -1 & 0 \\ 4 & 0 & 1 \end{array}\right)}_{B} \\
= \; & \underbrace{\begin{pmatrix} -1 & 2 \\ 1 & 0 \\ 2 & -1 \\ 1 & 2 \end{pmatrix}}_{A_0} \underbrace{\begin{pmatrix} -2 & 2 & -3 \\ 0 & 1 & -1 \end{pmatrix}}_{B_0} + \underbrace{\begin{pmatrix} 4 & 1 \\ -1 & -2 \\ 3 & 1 \\ 3 & 4 \end{pmatrix}}_{A_1} \underbrace{\begin{pmatrix} -2 & -1 & 0 \\ 4 & 0 & 1 \end{pmatrix}}_{B_1} \\
= \; & \underbrace{\begin{pmatrix} 2 & 0 & 1 \\ -2 & 2 & -3 \\ -4 & 3 & -5 \\ -2 & 4 & -5 \end{pmatrix}}_{A_0 B_0} + \underbrace{\begin{pmatrix} -4 & -4 & 1 \\ -6 & 1 & -2 \\ -2 & -3 & 1 \\ 10 & -3 & 4 \end{pmatrix}}_{A_1 B_1} = \underbrace{\begin{pmatrix} -2 & -4 & 2 \\ -8 & 3 & -5 \\ -6 & 0 & -4 \\ 8 & 1 & -1 \end{pmatrix}}_{AB}.
\end{aligned}$$

5.2.2 Properties

No video for this unit.

Is matrix-matrix multiplication associative?

Homework 5.2.2.1 Let $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $B = \begin{pmatrix} 0 & 2 & -1 \\ 1 & 1 & 0 \end{pmatrix}$, and $C = \begin{pmatrix} 0 & 1 \\ 1 & 2 \\ 1 & -1 \end{pmatrix}$. Compute

- $AB =$
- $(AB)C =$
- $BC =$
- $A(BC) =$

☛ SEE ANSWER

Homework 5.2.2.2 Let $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times k}$, and $C \in \mathbb{R}^{k \times l}$. $(AB)C = A(BC)$.

Always/Sometimes/Never

☛ SEE ANSWER

If you conclude that $(AB)C = A(BC)$, then we can simply write ABC since lack of parenthesis does not cause confusion about the order in which the multiplication needs to be performed.

In a previous week, we argued that $e_i^T(Ae_j)$ equals $\alpha_{i,j}$, the (i, j) element of A. We can now write that as $\alpha_{i,j} = e_i^T Ae_j$, since we can drop parentheses.

Is matrix-matrix multiplication distributive?

Homework 5.2.2.3 Let $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $B = \begin{pmatrix} 2 & -1 \\ 1 & 0 \end{pmatrix}$, and $C = \begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix}$. Compute

- $A(B+C) =$.
- $AB + AC =$.
- $(A+B)C =$.
- $AC + BC =$.

☛ SEE ANSWER

Homework 5.2.2.4 Let $A \in \mathbb{R}^{m \times k}$, $B \in \mathbb{R}^{k \times n}$, and $C \in \mathbb{R}^{k \times n}$. $A(B+C) = AB + AC$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.2.5 If $A \in \mathbb{R}^{m \times k}$, $B \in \mathbb{R}^{m \times k}$, and $C \in \mathbb{R}^{k \times n}$, then $(A+B)C = AC + BC$.

True/False

☛ SEE ANSWER

5.2.3 Transposing a Product of Matrices

No video for this unit.

Homework 5.2.3.1 Let $A = \begin{pmatrix} 2 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 3 & 1 \\ -1 & 1 & 1 \end{pmatrix}$ and $B = \begin{pmatrix} 2 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$. Compute

- $A^T A =$
- $AA^T =$
- $(AB)^T =$
- $A^T B^T =$
- $B^T A^T =$

☛ SEE ANSWER

Homework 5.2.3.2 Let $A \in \mathbb{R}^{m \times k}$ and $B \in \mathbb{R}^{k \times n}$. $(AB)^T = B^T A^T$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.3.3 Let A, B, and C be conformal matrices so that ABC is well-defined. Then $(ABC)^T = C^T B^T A^T$.

Always/Sometimes/Never

☛ SEE ANSWER

5.2.4 Matrix-Matrix Multiplication with Special Matrices

No video for this unit.

Multiplication with an identity matrix

Homework 5.2.4.1 Compute

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} =$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} =$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} =$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} =$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} =$

☛ SEE ANSWER

Homework 5.2.4.2 Compute

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} =$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 0 \\ 3 \end{pmatrix} =$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ -1 \end{pmatrix} =$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix} =$

☛ SEE ANSWER

Homework 5.2.4.3 Let $A \in \mathbb{R}^{m\times n}$ and let I denote the identity matrix of appropriate size. $AI = IA = A$.

Always/Sometimes/Never

☛ SEE ANSWER

Multiplication with a diagonal matrix

Homework 5.2.4.4 Compute

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} =$
- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} =$
- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ -3 \end{pmatrix} =$
- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} =$

☛ SEE ANSWER

Homework 5.2.4.5 Compute

- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} =$
- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} -2 \\ 0 \\ 3 \end{pmatrix} =$
- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ -1 \end{pmatrix} =$
- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix} =$

☛ SEE ANSWER

Homework 5.2.4.6 Let $A \in \mathbb{R}^{m\times n}$ and let D denote the diagonal matrix with diagonal elements $\delta_0, \delta_1, \cdots, \delta_{n-1}$. Partition A by columns: $A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$.

$$AD = \left(\begin{array}{c|c|c|c} \delta_0 a_0 & \delta_1 a_1 & \cdots & \delta_{n-1} a_{n-1} \end{array} \right).$$

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.4.7 Let $A \in \mathbb{R}^{m\times n}$ and let D denote the diagonal matrix with diagonal elements $\delta_0, \delta_1, \cdots, \delta_{m-1}$. Partition A by rows: $A = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array} \right)$.

$$DA = \left(\begin{array}{c} \delta_0 \widetilde{a}_0^T \\ \hline \delta_1 \widetilde{a}_1^T \\ \hline \vdots \\ \hline \delta_{m-1} \widetilde{a}_{m-1}^T \end{array} \right).$$

Always/Sometimes/Never

☛ SEE ANSWER

Triangular matrices

Homework 5.2.4.8 Compute $\left(\begin{array}{rrr} 1 & -1 & -2 \\ 0 & 2 & 3 \\ 0 & 0 & 1 \end{array} \right) \left(\begin{array}{rrr} -2 & 1 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{array} \right) =$

☛ SEE ANSWER

Homework 5.2.4.9 Compute the following, using what you know about partitioned matrix-matrix multiplication:

$\left(\begin{array}{rr|r} 1 & -1 & -2 \\ 0 & 2 & 3 \\ \hline 0 & 0 & 1 \end{array} \right) \left(\begin{array}{rr|r} -2 & 1 & -1 \\ 0 & 1 & 2 \\ \hline 0 & 0 & 1 \end{array} \right) =$

☛ SEE ANSWER

Homework 5.2.4.10 Let $U, R \in \mathbb{R}^{n\times n}$ be upper triangular matrices. UR is an upper triangular matrix.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.4.11 The product of an $n \times n$ lower triangular matrix times an $n \times n$ lower triangular matrix is a lower triangular matrix.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.4.12 The product of an $n \times n$ lower triangular matrix times an $n \times n$ upper triangular matrix is a diagonal matrix.

Always/Sometimes/Never

☛ SEE ANSWER

Symmetric matrices

Homework 5.2.4.13 Let $A \in \mathbb{R}^{m \times n}$. $A^T A$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.4.14 Evaluate

- $\left(\begin{array}{r} -1 \\ 1 \\ 2 \end{array}\right)\left(\begin{array}{rrr} -1 & 1 & 2 \end{array}\right) =$

- $\left(\begin{array}{r} 2 \\ 0 \\ -1 \end{array}\right)\left(\begin{array}{rrr} 2 & 0 & -1 \end{array}\right) =$

- $\left(\begin{array}{r|r} -1 & 2 \\ 1 & 0 \\ 2 & -1 \end{array}\right)\left(\begin{array}{rrr} -1 & 1 & 2 \\ \hline 2 & 0 & -1 \end{array}\right) =$

- $\left(\begin{array}{r} 1 \\ -2 \\ 2 \end{array}\right)\left(\begin{array}{rrr} 1 & -2 & 2 \end{array}\right) =$

- $\left(\begin{array}{rr|r} -1 & 2 & 1 \\ 1 & 0 & -2 \\ 2 & -1 & 2 \end{array}\right)\left(\begin{array}{rrr} -1 & 1 & 2 \\ 2 & 0 & -1 \\ \hline 1 & -2 & 2 \end{array}\right) =$

☛ SEE ANSWER

Homework 5.2.4.15 Let $x \in \mathbb{R}^n$. The outer product xx^T is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.4.16 Let $A \in \mathbb{R}^{n \times n}$ be symmetric and $x \in \mathbb{R}^n$. $A + xx^T$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.4.17 Let $A \in \mathbb{R}^{m \times n}$. Then AA^T is symmetric. (In your reasoning, we want you to use insights from previous homeworks.)

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.2.4.18 Let $A, B \in \mathbb{R}^{n \times n}$ be symmetric matrices. AB is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

A generalization of $A + xx^T$ with symmetric A and vector x, is given by

$$A := \alpha x x^T + A,$$

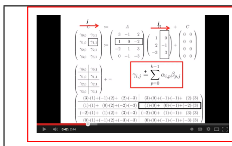
where α is a scalar. This is known as a *symmetric rank-1 update*.
The last exercise motivates the fact that the result itself is symmetric. The reason for the name "rank-1 update" will become clear later in the course, when we will see that a matrix that results from an outer product, yx^T, has rank at most equal to one.
This operation is sufficiently important that it is included in the `laff` library as function

```
[ y_out ] = laff_syr( alpha, x, A )
```

which updates $A := \alpha x x^T + A$.

5.3 Algorithms for Computing Matrix-Matrix Multiplication

5.3.1 Lots of Loops



☛ YouTube

In Theorem 5.1, partition C into elements (scalars), and A and B by rows and columns, respectively. In other words, let $M = m$, $m_i = 1$, $i = 0, \ldots, m-1$; $N = n$, $n_j = 1$, $j = 0, \ldots, n-1$; and $K = 1$, $k_0 = k$. Then

$$\left(\begin{array}{c|c|c|c} \gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1} \\ \hline \gamma_{1,0} & \gamma_{1,1} & \cdots & \gamma_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \gamma_{m-1,0} & \gamma_{m-1,1} & \cdots & \gamma_{m-1,n-1} \end{array}\right), A = \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array}\right), \text{ and } B = \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right)$$

so that

$$\begin{aligned} C &= \left(\begin{array}{c|c|c|c} \gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1} \\ \hline \gamma_{1,0} & \gamma_{1,1} & \cdots & \gamma_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \gamma_{m-1,0} & \gamma_{m-1,1} & \cdots & \gamma_{m-1,n-1} \end{array}\right) = \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array}\right) \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right) \\ &= \left(\begin{array}{c|c|c|c} \tilde{a}_0^T b_0 & \tilde{a}_0^T b_1 & \cdots & \tilde{a}_0^T b_{n-1} \\ \hline \tilde{a}_1^T b_0 & \tilde{a}_1^T b_1 & \cdots & \tilde{a}_1^T b_{n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \tilde{a}_{m-1}^T b_0 & \tilde{a}_{m-1}^T b_1 & \cdots & \tilde{a}_{m-1}^T b_{n-1} \end{array}\right). \end{aligned}$$

As expected, $\gamma_{i,j} = \tilde{a}_i^T b_j$: the dot product of the ith row of A with the jth column of B.

Example 5.3

$$\left(\begin{array}{rrr} -1 & 2 & 4 \\ \hline 1 & 0 & -1 \\ \hline 2 & -1 & 3 \end{array}\right)\left(\begin{array}{r|r} -2 & 2 \\ 0 & 1 \\ -2 & -1 \end{array}\right) = \left(\begin{array}{c|c} \begin{pmatrix} -1 & 2 & 4 \end{pmatrix}\begin{pmatrix} -2 \\ 0 \\ -2 \end{pmatrix} & \begin{pmatrix} -1 & 2 & 4 \end{pmatrix}\begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix} \\ \hline \begin{pmatrix} 1 & 0 & -1 \end{pmatrix}\begin{pmatrix} -2 \\ 0 \\ -2 \end{pmatrix} & \begin{pmatrix} 1 & 0 & -1 \end{pmatrix}\begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix} \\ \hline \begin{pmatrix} 2 & -1 & 3 \end{pmatrix}\begin{pmatrix} -2 \\ 0 \\ -2 \end{pmatrix} & \begin{pmatrix} 2 & -1 & 3 \end{pmatrix}\begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix} \end{array}\right)$$

$$= \left(\begin{array}{r|r} -6 & -4 \\ \hline 0 & 3 \\ \hline -10 & 0 \end{array}\right)$$

This motivates the following two algorithms for computing $C = AB + C$. In both, the outer two loops visit all elements $\gamma_{i,j}$ of C, and the inner loop updates a given $\gamma_{i,j}$ with the dot product of the ith row of A and the jth column of B. They differ in that the first updates C one column at a time (the outer loop is over the columns of C and B) while the second updates C one row at a time (the outer loop is over the rows of C and A).

$$\begin{array}{l} \textbf{for } j = 0, \ldots, n-1 \\ \quad \textbf{for } i = 0, \ldots, m-1 \\ \quad\quad \left.\begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \textbf{endfor} \end{array}\right\} \gamma_{i,j} := \tilde{a}_i^T b_j + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array} \quad \text{or} \quad \begin{array}{l} \textbf{for } i = 0, \ldots, m-1 \\ \quad \textbf{for } j = 0, \ldots, n-1 \\ \quad\quad \left.\begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \textbf{endfor} \end{array}\right\} \gamma_{i,j} := \tilde{a}_i^T b_j + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}$$

Homework 5.3.1.1 Consider the MATLAB function

```
function [ C_out ] = MatMatMult( A, B, C )

[ m, n ] = size( C );
[ m_A, k ] = size( A );
[ m_B, n_B ] = size( B );

for j = 1:n
   for i = 1:m
      for p = 1:k
         C( i,j ) = A( i, p ) * B( p, j ) + C( i, j );
      end
   end
end
```

- Download the files MatMatMult.m and test_MatMatMult.m into, for example,

  `LAFF-2.0xM -> Programming -> Week5`

  (creating the directory if necessary).

- Examine the script `test_MatMatMult.m` and then execute it in the MATLAB Command Window: `test_MatMatMult`.

- Now, exchange the order of the loops:

```
for j = 1:n
   for p = 1:k
      for i = 1:m
         C( i,j ) = A( i, p ) * B( p, j ) + C( i, j );
      end
   end
end
```

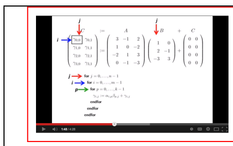
  save the result, and execute `test_MatMatMult` again. What do you notice?

- How may different ways can you order the “triple-nested loop”?

- Try them all and observe how the result of executing `test_MatMatMult` does or does not change.

☛ SEE ANSWER

5.3.2 Matrix-Matrix Multiplication by Columns



☛ YouTube

Homework 5.3.2.1 Let A and B be matrices and AB be well-defined and let B have at least four columns. If the first and fourth columns of B are the same, then the first and fourth columns of AB are the same.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.3.2.2 Let A and B be matrices and AB be well-defined and let A have at least four columns. If the first and fourth columns of A are the same, then the first and fourth columns of AB are the same.

Always/Sometimes/Never

☛ SEE ANSWER

In Theorem 5.1 let us partition C and B by columns and not partition A. In other words, let $M = 1$, $m_0 = m$; $N = n$, $n_j = 1$, $j = 0,\ldots,n-1$; and $K = 1$, $k_0 = k$. Then

$$C = \left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array} \right) \quad \text{and} \quad B = \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right)$$

so that

$$\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array} \right) = C = AB = A \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} Ab_0 & Ab_1 & \cdots & Ab_{n-1} \end{array} \right).$$

Homework 5.3.2.3

- $$\left(\begin{array}{rrr} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{array} \right) \left(\begin{array}{|r|cc} -1 & & \\ 2 & & \\ 1 & & \end{array} \right) =$$
- $$\left(\begin{array}{rrr} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{array} \right) \left(\begin{array}{r|r|c} -1 & 0 & \\ 2 & 1 & \\ 1 & -1 & \end{array} \right) =$$
- $$\left(\begin{array}{rrr} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{array} \right) \left(\begin{array}{rr|r|} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array} \right) =$$

☛ SEE ANSWER

Example 5.4

$$\left(\begin{array}{rrr} -1 & 2 & 4 \\ 1 & 0 & -1 \\ 2 & -1 & 3 \end{array} \right) \left(\begin{array}{r|r} -2 & 2 \\ 0 & 1 \\ -2 & -1 \end{array} \right) = \left(\begin{array}{c|c} \left(\begin{array}{rrr} -1 & 2 & 4 \\ 1 & 0 & -1 \\ 2 & -1 & 3 \end{array} \right) \left(\begin{array}{r} -2 \\ 0 \\ -2 \end{array} \right) & \left(\begin{array}{rrr} -1 & 2 & 4 \\ 1 & 0 & -1 \\ 2 & -1 & 3 \end{array} \right) \left(\begin{array}{r} 2 \\ 1 \\ -1 \end{array} \right) \end{array} \right)$$
$$= \left(\begin{array}{r|r} -6 & -4 \\ 0 & 3 \\ -10 & 0 \end{array} \right)$$

By moving the loop indexed by j to the outside in the algorithm for computing $C = AB + C$ we observe that

$$\begin{array}{l} \textbf{for } j = 0,\ldots,n-1 \\ \quad \left. \begin{array}{l} \textbf{for } i = 0,\ldots,m-1 \\ \quad \textbf{for } p = 0,\ldots,k-1 \\ \quad\quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array} \right\} c_j := Ab_j + c_j \\ \textbf{endfor} \end{array} \quad \text{or} \quad \begin{array}{l} \textbf{for } j = 0,\ldots,n-1 \\ \quad \left. \begin{array}{l} \textbf{for } p = 0,\ldots,k-1 \\ \quad \textbf{for } i = 0,\ldots,m-1 \\ \quad\quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array} \right\} c_j := Ab_j + c_j \\ \textbf{endfor} \end{array}$$

Exchanging the order of the two inner-most loops merely means we are using a different algorithm (dot product vs. AXPY) for the matrix-vector multiplication $c_j := Ab_j + c_j$.

An algorithm that computes $C = AB + C$ one column at a time, represented with FLAME notation, is given in Figure 5.1

Homework 5.3.2.4 Implement the routine

```
[ C_out ] = Gemm_unb_var1( A, B, C )
```

based on the algorithm in Figure 5.1.

☛ SEE ANSWER

Algorithm: $C := \text{GEMM\_UNB\_VAR1}(A, B, C)$

Partition $B \rightarrow \left(B_L \,\middle|\, B_R \right)$, $C \rightarrow \left(C_L \,\middle|\, C_R \right)$
  where B_L has 0 columns, C_L has 0 columns

while $n(B_L) < n(B)$ **do**

  Repartition

$$\left(B_L \,\middle|\, B_R \right) \rightarrow \left(B_0 \,\middle|\, b_1 \,\middle|\, B_2 \right), \left(C_L \,\middle|\, C_R \right) \rightarrow \left(C_0 \,\middle|\, c_1 \,\middle|\, C_2 \right)$$

  where b_1 has 1 column, c_1 has 1 column

$c_1 := Ab_1 + c_1$

  Continue with

$$\left(B_L \,\middle|\, B_R \right) \leftarrow \left(B_0 \,\middle|\, b_1 \,\middle|\, B_2 \right), \left(C_L \,\middle|\, C_R \right) \leftarrow \left(C_0 \,\middle|\, c_1 \,\middle|\, C_2 \right)$$

endwhile

Figure 5.1: Algorithm for $C = AB + C$, computing C one column at a time.

5.3.3 Matrix-Matrix Multiplication by Rows

☛ YouTube

Homework 5.3.3.1 Let A and B be matrices and AB be well-defined and let A have at least four rows. If the first and fourth rows of A are the same, then the first and fourth rows of AB are the same.

Always/Sometimes/Never

☛ SEE ANSWER

In Theorem 5.1 partition C and A by rows and do not partition B. In other words, let $M = m$, $m_i = 1$, $i = 0, \ldots, m-1$; $N = 1$, $n_0 = n$; and $K = 1$, $k_0 = k$. Then

$$C = \left(\begin{array}{c} \tilde{c}_0^T \\ \hline \tilde{c}_1^T \\ \hline \vdots \\ \hline \tilde{c}_{m-1}^T \end{array} \right) \quad \text{and} \quad A = \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right)$$

so that

$$\left(\begin{array}{c} \tilde{c}_0^T \\ \hline \tilde{c}_1^T \\ \hline \vdots \\ \hline \tilde{c}_{m-1}^T \end{array} \right) = C = AB = \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right) B = \left(\begin{array}{c} \tilde{a}_0^T B \\ \hline \tilde{a}_1^T B \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T B \end{array} \right).$$

This shows how C can be computed one row at a time.

Example 5.5

$$
\left(\begin{array}{rrr} -1 & 2 & 4 \\ \hline 1 & 0 & -1 \\ \hline 2 & -1 & 3 \end{array}\right)
\left(\begin{array}{rr} -2 & 2 \\ 0 & 1 \\ -2 & -1 \end{array}\right)
=
\left(\begin{array}{c}
\left(\begin{array}{rrr} -1 & 2 & 4 \end{array}\right)\left(\begin{array}{rr} -2 & 2 \\ 0 & 1 \\ -2 & -1 \end{array}\right) \\ \hline
\left(\begin{array}{rrr} 1 & 0 & -1 \end{array}\right)\left(\begin{array}{rr} -2 & 2 \\ 0 & 1 \\ -2 & -1 \end{array}\right) \\ \hline
\left(\begin{array}{rrr} 2 & -1 & 3 \end{array}\right)\left(\begin{array}{rr} -2 & 2 \\ 0 & 1 \\ -2 & -1 \end{array}\right)
\end{array}\right)
=
\left(\begin{array}{rr} -6 & -4 \\ \hline 0 & 3 \\ \hline -10 & 0 \end{array}\right)
$$

In the algorithm for computing $C = AB + C$ the loop indexed by i can be moved to the outside so that

$$
\begin{array}{l}
\textbf{for } i = 0, \ldots, m-1 \\
\quad \left.\begin{array}{l}
\textbf{for } j = 0, \ldots, n-1 \\
\quad \textbf{for } p = 0, \ldots, k-1 \\
\qquad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\
\quad \textbf{endfor} \\
\textbf{endfor}
\end{array}\right\} \tilde{c}_i^T := \tilde{a}_i^T B + \tilde{c}_i^T \\
\textbf{endfor}
\end{array}
\quad \text{or} \quad
\begin{array}{l}
\textbf{for } i = 0, \ldots, m-1 \\
\quad \left.\begin{array}{l}
\textbf{for } p = 0, \ldots, k-1 \\
\quad \textbf{for } j = 0, \ldots, n-1 \\
\qquad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\
\quad \textbf{endfor} \\
\textbf{endfor}
\end{array}\right\} \tilde{c}_i^T := \tilde{a}_i^T B + \tilde{c}_i^T \\
\textbf{endfor}
\end{array}
$$

An algorithm that computes $C = AB + C$ row at a time, represented with FLAME notation, is given in Figure 5.2.

Homework 5.3.3.2

- $\left(\begin{array}{rrr} \hline 1 & -2 & 2 \\ \hline & & \\ & & \end{array}\right)\left(\begin{array}{rrr} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) =$
- $\left(\begin{array}{rrr} 1 & -2 & 2 \\ \hline -1 & 2 & 1 \\ \hline & & \end{array}\right)\left(\begin{array}{rrr} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) =$
- $\left(\begin{array}{rrr} 1 & -2 & 2 \\ -1 & 2 & 1 \\ \hline 0 & 1 & 2 \\ \hline \end{array}\right)\left(\begin{array}{rrr} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) =$

☛ SEE ANSWER

Homework 5.3.3.3 Implement the routine

```
[ C_out ] = Gemm_unb_var2( A, B, C )
```

based on the algorithm in Figure 5.2.

☛ SEE ANSWER

Algorithm: $C := \text{GEMM\_UNB\_VAR2}(A, B, C)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $C \rightarrow \left(\begin{array}{c} C_T \\ \hline C_B \end{array}\right)$

  where A_T has 0 rows, C_T has 0 rows

while $m(A_T) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ A_2 \end{array}\right), \left(\begin{array}{c} C_T \\ \hline C_B \end{array}\right) \rightarrow \left(\begin{array}{c} C_0 \\ \hline c_1^T \\ C_2 \end{array}\right)$$

  where a_1 has 1 row, c_1 has 1 row

$c_1^T := a_1^T B + c_1^T$

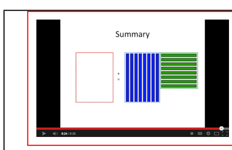
  Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} C_T \\ \hline C_B \end{array}\right) \leftarrow \left(\begin{array}{c} C_0 \\ c_1^T \\ \hline C_2 \end{array}\right)$$

endwhile

Figure 5.2: Algorithm for $C = AB + C$, computing C one row at a time.

5.3.4 Matrix-Matrix Multiplication with Rank-1 Updates



☛ YouTube

In Theorem 5.1 partition A and B by columns and rows, respectively, and do not partition C. In other words, let $M = 1$, $m_0 = m$; $N = 1$, $n_0 = n$; and $K = k$, $k_p = 1$, $p = 0, \ldots, k-1$. Then

$$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{k-1} \end{array}\right) \quad \text{and} \quad B = \left(\begin{array}{c} \tilde{b}_0^T \\ \hline \tilde{b}_1^T \\ \hline \vdots \\ \hline \tilde{b}_{k-1}^T \end{array}\right)$$

so that

$$C = AB = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{k-1} \end{array}\right) \left(\begin{array}{c} \tilde{b}_0^T \\ \hline \tilde{b}_1^T \\ \hline \vdots \\ \hline \tilde{b}_{k-1}^T \end{array}\right) = a_0\tilde{b}_0^T + a_1\tilde{b}_1^T + \cdots + a_{k-1}\tilde{b}_{k-1}^T.$$

Notice that each term $a_p\tilde{b}_p^T$ is an outer product of a_p and $\tilde{b}_p$. Thus, if we start with $C := 0$, the zero matrix, then we can compute

$C := AB + C$ as

$$C := a_{k-1}\tilde{b}_{k-1}^T + (\cdots + (a_p\tilde{b}_p^T + (\cdots + (a_1\tilde{b}_1^T + (a_0\tilde{b}_0^T + C))\cdots))\cdots),$$

which illustrates that $C := AB$ can be computed by first setting C to zero, and then repeatedly updating it with rank-1 updates.

Example 5.6

$$\left(\begin{array}{r|r|r} -1 & 2 & 4 \\ 1 & 0 & -1 \\ 2 & -1 & 3 \end{array}\right)\left(\begin{array}{rr} -2 & 2 \\ \hline 0 & 1 \\ \hline -2 & -1 \end{array}\right)$$

$$= \begin{pmatrix} -1 \\ 1 \\ 2 \end{pmatrix}\begin{pmatrix} -2 & 2 \end{pmatrix} + \begin{pmatrix} 2 \\ 0 \\ -1 \end{pmatrix}\begin{pmatrix} 0 & 1 \end{pmatrix} + \begin{pmatrix} 4 \\ -1 \\ 3 \end{pmatrix}\begin{pmatrix} -2 & -1 \end{pmatrix}$$

$$= \begin{pmatrix} 2 & -2 \\ -2 & 2 \\ -4 & 4 \end{pmatrix} + \begin{pmatrix} 0 & 2 \\ 0 & 0 \\ 0 & -1 \end{pmatrix} + \begin{pmatrix} -8 & -4 \\ 2 & 1 \\ -6 & -3 \end{pmatrix} = \begin{pmatrix} -6 & -4 \\ 0 & 3 \\ -10 & 0 \end{pmatrix}$$

In the algorithm for computing $C := AB + C$ the loop indexed by p can be moved to the outside so that

$$\begin{array}{l} \textbf{for } p = 0,\ldots,k-1 \\ \quad \left.\begin{array}{l} \textbf{for } j = 0,\ldots,n-1 \\ \quad \textbf{for } i = 0,\ldots,m-1 \\ \qquad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}\right\} C := a_p\tilde{b}_p^T + C \\ \textbf{endfor} \end{array} \quad \text{or} \quad \begin{array}{l} \textbf{for } p = 0,\ldots,k-1 \\ \quad \left.\begin{array}{l} \textbf{for } i = 0,\ldots,m-1 \\ \quad \textbf{for } j = 0,\ldots,n-1 \\ \qquad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}\right\} C := a_p\tilde{b}_p^T + C \\ \textbf{endfor} \end{array}$$

An algorithm that computes $C = AB + C$ with rank-1 updates, represented with FLAME notation, is given in Figure 5.3.

Homework 5.3.4.1

- $\left(\begin{array}{|r|ccc} 1 & & & \\ -1 & & & \\ 0 & & & \end{array}\right)\left(\begin{array}{rrr} \hline -1 & 0 & 1 \\ \hline & & \\ & & \end{array}\right) =$

- $\left(\begin{array}{c|r|c} & -2 & \\ & 2 & \\ & 1 & \end{array}\right)\left(\begin{array}{rrr} & & \\ \hline 2 & 1 & -1 \\ \hline & & \end{array}\right) =$

- $\left(\begin{array}{cc|r|} & & 2 \\ & & 1 \\ & & 2 \end{array}\right)\left(\begin{array}{rrr} & & \\ & & \\ \hline 1 & -1 & 2 \\ \hline \end{array}\right) =$

- $\begin{pmatrix} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix}\begin{pmatrix} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{pmatrix} =$

☛ SEE ANSWER

Algorithm: $C := \text{GEMM\_UNB\_VAR3}(A, B, C)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right)$

where A_L has 0 columns, B_T has 0 rows

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ B_2 \end{array} \right)$$

where a_1 has 1 column, b_1 has 1 row

$C := a_1 b_1^T + C$

Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,|\, a_1 \,\middle|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \leftarrow \left(\begin{array}{c} B_0 \\ b_1^T \\ \hline B_2 \end{array} \right)$$

endwhile

Figure 5.3: Algorithm for $C = AB + C$, computing C via rank-1 updates.

Homework 5.3.4.2 Implement the routine

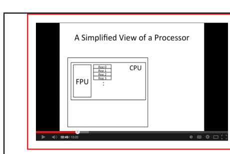
```
[ C_out ] = Gemm_unb_var2( A, B, C )
```

based on the algorithm in Figure 5.3.

☛ SEE ANSWER

5.4 Enrichment

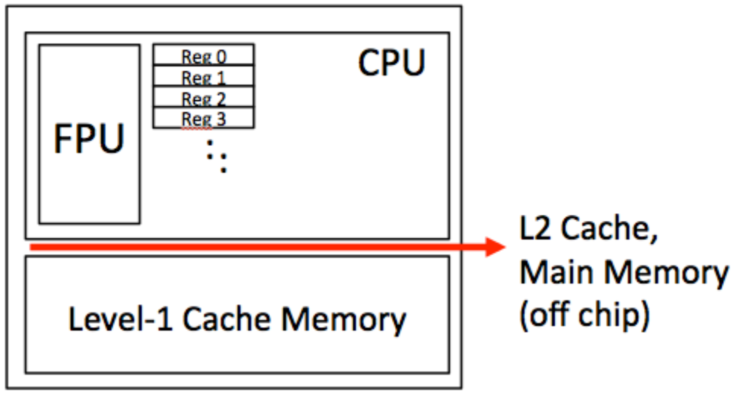
5.4.1 Slicing and Dicing for Performance



☛ YouTube

Computer Architecture (Very) Basics

A highly simplified description of a processor is given below.

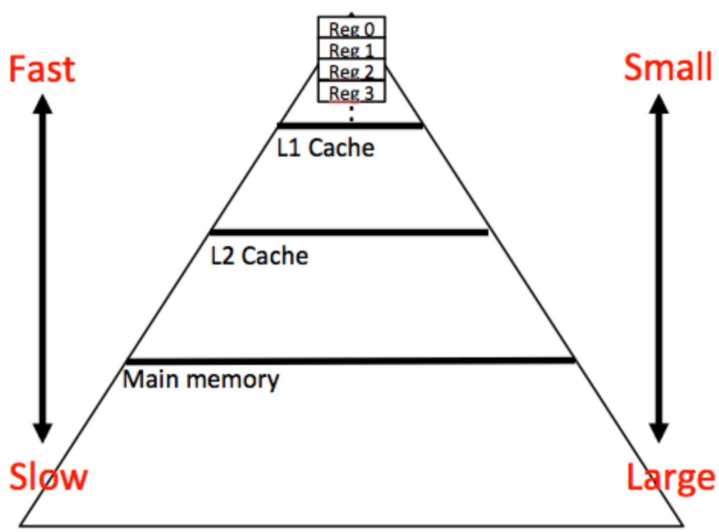


Yes, it is very, very simplified. For example, these days one tends to talk about "cores" and there are multiple cores on a computer chip. But this simple view of what a processor is will serve our purposes just fine.

At the heart of the processor is the Central Processing Unit (CPU). It is where the computing happens. For us, the important parts of the CPU are the Floating Point Unit (FPU), where floating point computations are performed, and the registers, where data with which the FPU computes must reside. A typical processor will have 16-64 registers. In addition to this, a typical processor has a small amount of memory on the chip, called the Level-1 (L1) Cache. The L1 cache can typically hold 16Kbytes (about 16,000 bytes) or 32Kbytes. The L1 cache is fast memory, fast enough to keep up with the FPU as it computes.

Additional memory is available "off chip". There is the Level-2 (L2) Cache and Main Memory. The L2 cache is slower than the L1 cache, but not as slow as main memory. To put things in perspective: in the time it takes to bring a floating point number from main memory onto the processor, the FPU can perform 50-100 floating point computations. **Memory is very slow.** (There might be an L3 cache, but let's not worry about that.) Thus, where in these different layers of the hierarchy of memory data exists greatly affects how fast computation can be performed, since waiting for the data may become the dominating factor. Understanding this memory hierarchy is important.

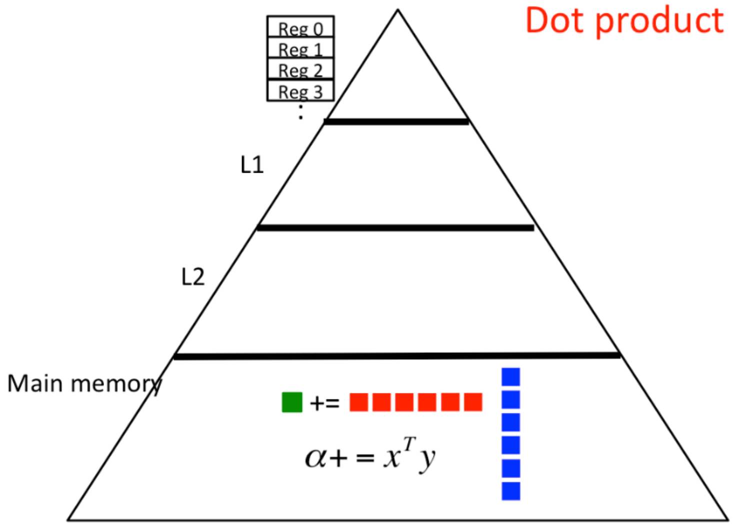
Here is how to view the memory as a pyramid:



At the top, there are the registers. For computation to happen, data must be in registers. Below it are the L1 and L2 caches. At the bottom, main memory. Below that layer, there may be further layers, like disk storage.

Now, the name of the game is to keep data in the faster memory layers to overcome the slowness of main memory. Notice that computation can also hide the "latency" to memory: one can overlap computation and the fetching of data.

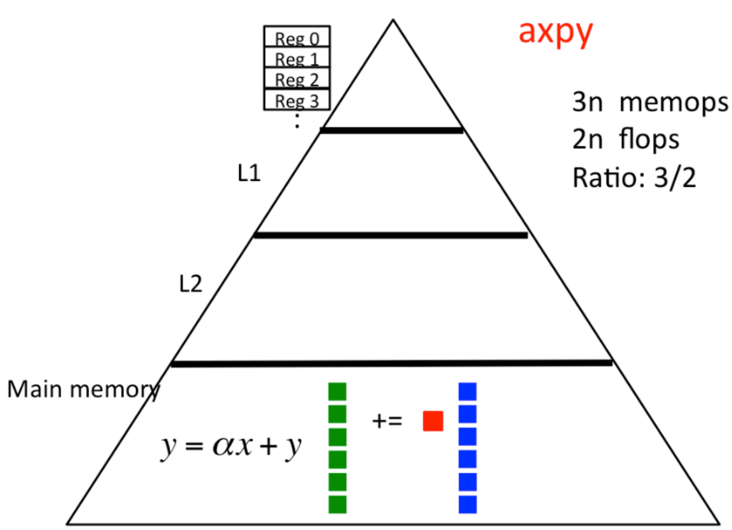
Vector-Vector Computations Let's consider performing the dot product operation $\alpha := x^T y$, with vectors $x, y \in \mathbb{R}^n$ that reside in main memory.



Notice that inherently the components of the vectors must be loaded into registers at some point of the computation, requiring $2n$ memory operations (memops). The scalar α can be stored in a register as the computation proceeds, so that it only needs to be written to main memory once, at the end of the computation. This one memop can be ignored relative to the $2n$ memops required to fetch the vectors. Along the way, (approximately) $2n$ flops are performed: an add and a multiply for each pair of components of x and y.

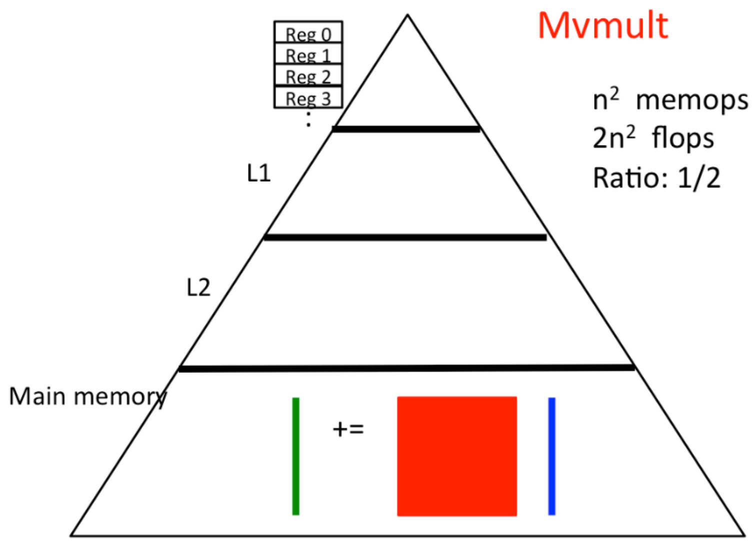
The problem is that the ratio of memops to flops is $2n/2n = 1/1$. Since memops are extremely slow, the cost is in moving the data, not in the actual computation itself. Yes, there is cache memory in between, but if the data starts in main memory, this is of no use: there isn't any reuse of the components of the vectors.

The problem is worse for the AXPY operation, $y := \alpha x + y$:



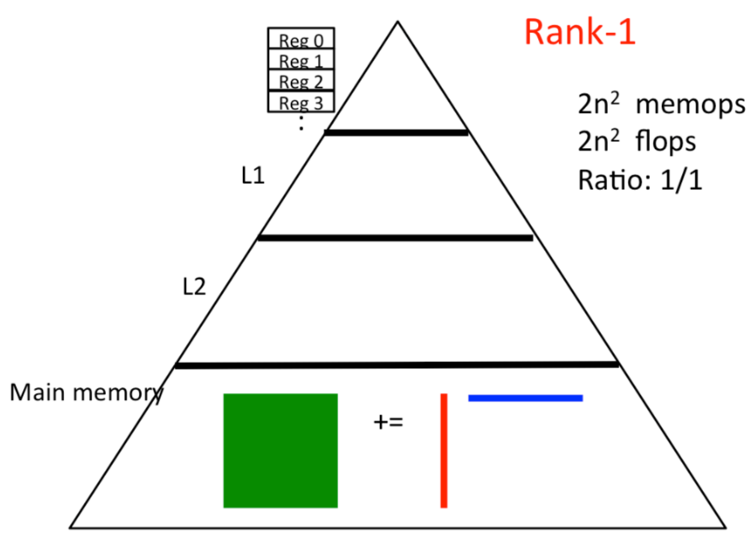
Here the components of the vectors x and y must be read from main memory, and the result y must be written back to main memory, for a total of $3n$ memops. The scalar α can be kept in a register, and therefore reading it from main memory is insignificant. The computation requires $2n$ flops, yielding a ratio of 3 memops for every 2 flops.

Matrix-Vector Computations Now, let's examine how matrix-vector multiplication, $y := Ax + y$, fares. For our analysis, we will assume a square $n \times n$ matrix A. All operands start in main memory.



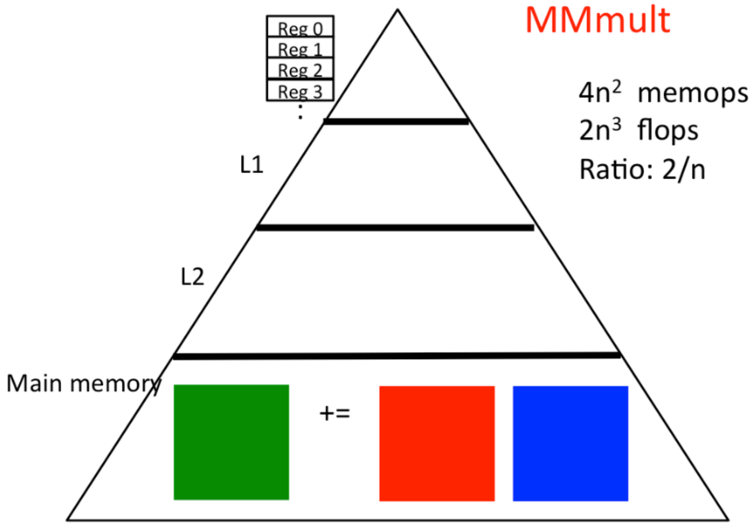
Now, inherently, all $n \times n$ elements of A must be read from main memory, requiring n^2 memops. Inherently, for each element of A only two flops are performed: an add and a multiply, for a total of $2n^2$ flops. There *is* an opportunity to bring components of x and/or y into cache memory and/or registers, and reuse them there for many computations. For example, if y is computed via dot products of rows of A with the vector x, the vector x can be brought into cache memory and reused many times. The component of y being computed can then be kept in a registers during the computation of the dot product. For this reason, we ignore the cost of reading and writing the vectors. Still, the ratio of memops to flops is approximately $n^2/2n^2 = 1/2$. This is only slightly better than the ratio for dot and AXPY.

The story is worse for a rank-1 update, $A := xy^T + A$. Again, for our analysis, we will assume a square $n \times n$ matrix A. All operands start in main memory.



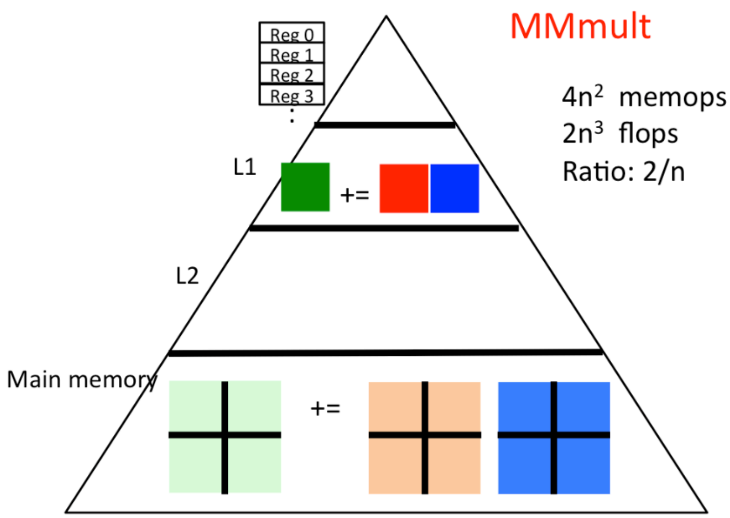
Now, inherently, all $n \times n$ elements of A must be read from main memory, requiring n^2 memops. But now, after having been updated, each element must also be written back to memory, for another n^2 memops. Inherently, for each element of A only two flops are performed: an add and a multiply, for a total of $2n^2$ flops. Again, there *is* an opportunity to bring components of x and/or y into cache memory and/or registers, and reuse them there for many computations. Again, for this reason we ignore the cost of reading the vectors. Still, the ratio of memops to flops is approximately $2n^2/2n^2 = 1/1$.

Matrix-Matrix Computations Finally, let's examine how matrix-matrix multiplication, $C := AB + C$, overcomes the memory bottleneck. For our analysis, we will assume all matrices are square $n \times n$ matrices and all operands start in main memory.



Now, inherently, all elements of the three matrices must be read at least once from main memory, requiring $3n^2$ memops, and C must be written at least once back to main memory, for another n^2 memops. We saw that a matrix-matrix multiplication requires a total of $2n^3$ flops. If this can be achieved, then the ratio of memops to flops becomes $4n^2/2n^3 = 2/n$. If n is large enough, the cost of accessing memory can be overcome. To achieve this, all three matrices must be brought into cache memory, the computation performed while the data is in cache memory, and then the result written out to main memory.

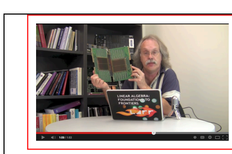
The problem is that the matrices typically are too big to fit in, for example, the L1 cache. To overcome this limitation, we can use our insight that matrices can be partitioned, and matrix-matrix multiplication can be performed with submatrices (blocks).



This way, near-peak performance can be achieved.

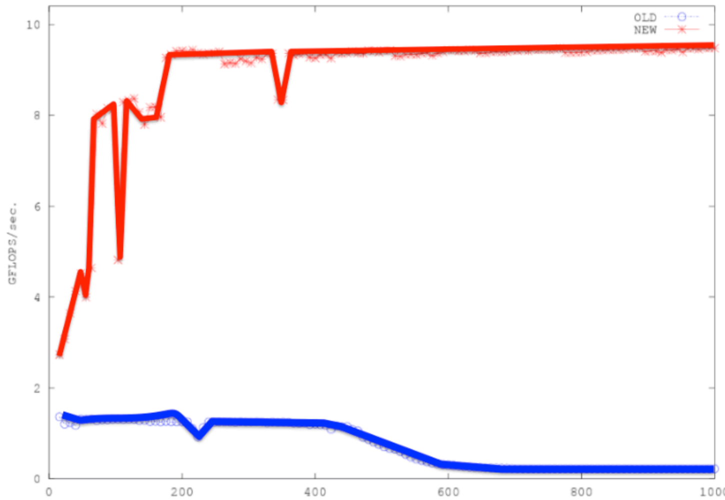
To achieve very high performance, one has to know how to partition the matrices more carefully, and arrange the operations in a very careful order. But the above describes the fundamental ideas.

5.4.2 How It is Really Done



☛ YouTube

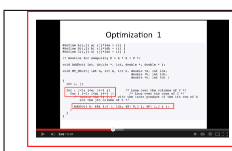
Measuring Performance There are two attributes of a processor that affect the rate at which it can compute: its clock rate, which is typically measured in GHz (billions of cycles per second) and the number of floating point computations that it can perform per cycle. Multiply these two numbers together, and you get the rate at which floating point computations can be performed, measured in GFLOPS/sec (billions of floating point operations per second). The below graph reports performance obtained on a laptop of ours. The details of the processor are not important for this descussion, since the performance is typical.



Along the x-axis, the matrix sizes $m = n = k$ are reported. Along the y-axis performance is reported in GFLOPS/sec. The important thing is that the top of the graph represents the peak of the processor, so that it is easy to judge what percent of peak is attained.

The blue line represents a basic implementation with a triple-nested loop. When the matrices are small, the data fits in the L2 cache, and performance is (somewhat) better. As the problem sizes increase, memory becomes more and more a bottleneck. Pathetic performance is achieved. The red line is a careful implementation that also blocks for better cache reuse. Obviously, considerable improvement is achieved.

Try It Yourself!



☛ YouTube

If you know how to program in C and have access to a computer that runs the Linux operating system, you may want to try the exercise on the following wiki page:

`https://github.com/flame/how-to-optimize-gemm/wiki`

Others may still learn something by having a look without trying it themselves.

No, we do not have time to help you with this exercise... You can ask each other questions online, but we cannot help you with this... We are just too busy with the MOOC right now...

Further Reading

- Kazushige Goto is famous for his implementation of matrix-matrix multiplication. The following New York Times article on his work may amuse you:

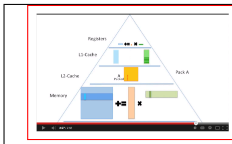
  Writing the Fastest Code, by Hand, for Fun: A Human Computer Keeps ..

- An article that describes his approach to matrix-matrix multiplication is

  Kazushige Goto, Robert A. van de Geijn.
  Anatomy of high-performance matrix multiplication.
  ACM Transactions on Mathematical Software (TOMS), 2008.

It can be downloaded for free by first going to the FLAME publication webpage and clicking on Journal Publication #11. We believe you will be happy to find that you can understand at least the high level issues in that paper.

The following animation of how the memory hierarchy is utilized in Goto's approach may help clarify the above paper:



☛ YouTube

- A more recent paper that takes the insights further is

  Field G. Van Zee, Robert A. van de Geijn.
  BLIS: A Framework for Rapid Instantiation of BLAS Functionality.
  ACM Transactions on Mathematical Software.
  (to appear)

  It is also available from the FLAME publication webpage by clicking on Journal Publication #33.

- A paper that then extends these techniques to what are considered "many-core" architectures is

  Tyler M. Smith, Robert van de Geijn, Mikhail Smelyanskiy, Jeff R. Hammond, and Field G. Van Zee.
  Anatomy of High-Performance Many-Threaded Matrix Multiplication.
  International Parallel and Distributed Processing Symposium 2014. (to appear)

  It is also available from the FLAME publication webpage by clicking on Conference Publication #35. Around 90% of peak on 60 cores running 240 threads... At the risk of being accused of bragging, this is quite exceptional.

Notice that two of these papers have not even been published in print yet. You have arrived at the frontier of National Science Foundation (NSF) sponsored research, after only five weeks.

5.5 Wrap Up

5.5.1 Homework

For all of the below homeworks, only consider matrices that have real valued elements.

Homework 5.5.1.1 Let A and B be matrices and AB be well-defined. $(AB)^2 = A^2B^2$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.2 Let A be symmetric. A^2 is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.3 Let $A, B \in \mathbb{R}^{n\times n}$ both be symmetric. AB is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.4 Let $A, B \in \mathbb{R}^{n\times n}$ both be symmetric. $A^2 - B^2$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.5 Let $A, B \in \mathbb{R}^{n\times n}$ both be symmetric. $(A+B)(A-B)$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.6 Let $A, B \in \mathbb{R}^{n\times n}$ both be symmetric. ABA is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.7 Let $A, B \in \mathbb{R}^{n \times n}$ both be symmetric. $ABAB$ is symmetric.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.8 Let A be symmetric. $A^T A = AA^T$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 5.5.1.9 If $A = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ then $A^T A = AA^T$.

True/False

☛ SEE ANSWER

Homework 5.5.1.10 Propose an algorithm for computing $C := UR$ where C, U, and R are all upper triangular matrices by completing the below algorithm.

Algorithm: $[C] := \text{TRTRMM\_UU\_UNB\_VAR1}(U,R,C)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right)$, $C \rightarrow \left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right)$

where U_{TL} is 0×0, R_{TL} is 0×0, C_{TL} is 0×0

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$$

where υ_{11} is 1×1, ρ_{11} is 1×1, γ_{11} is 1×1

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$$

endwhile

Hint: consider Homework 5.2.4.10. Then implement and test it.

☛ SEE ANSWER

Challenge 5.5.1.11 Propose many algorithms for computing $C := UR$ where C, U, and R are all upper triangular matrices. Hint: Think about how we created matrix-vector multiplication algorithms for the case where A was triangular. How can you similarly take the three different algorithms discussed in Units 5.3.2-4 and transform them into algorithms that take advantage of the triangular shape of the matrices?

Challenge 5.5.1.12 Propose many algorithms for computing $C := UR$ where C, U, and R are all upper triangular matrices. This time, derive all algorithm systematically by following the methodology in

The Science of Programming Matrix Computations.

(You will want to read Chapters 2-5.)
(You may want to use the blank "worksheet" on the next page.)

<table>
<tr><th>Step</th><th>Annotated Algorithm: $[C] := \text{TRTRMM\_UU\_UNB}\ (U,R,C)$</th></tr>
<tr><td>1a</td><td>$\left\{C = \widehat{C}\right\}$</td></tr>
<tr><td>4</td><td>Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right), R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right), C \rightarrow \left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right)$
where U_{TL} is 0×0, R_{TL} is 0×0, C_{TL} is 0×0</td></tr>
<tr><td>2</td><td>$\left\{\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) = \qquad \right\}$</td></tr>
<tr><td>3</td><td>while $m(U_{TL}) < m(U)$ do</td></tr>
<tr><td>2,3</td><td>$\left\{\left(\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) = \qquad \right) \wedge (m(U_{TL}) < m(U))\right\}$</td></tr>
<tr><td>5a</td><td>Repartition
$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right), \left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$
where υ_{11} is 1×1, ρ_{11} is 1×1, γ_{11} is 1×1</td></tr>
<tr><td>6</td><td>$\left\{\left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right) = \qquad \right\}$</td></tr>
<tr><td>8</td><td></td></tr>
<tr><td>5b</td><td>Continue with
$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right), \left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$</td></tr>
<tr><td>7</td><td>$\left\{\left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right) = \qquad \right\}$</td></tr>
<tr><td>2</td><td>$\left\{\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) = \qquad \right\}$</td></tr>
<tr><td></td><td>endwhile</td></tr>
<tr><td>2,3</td><td>$\left\{\left(\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) = \qquad \right) \wedge \neg(m(U_{TL}) < m(U))\right\}$</td></tr>
<tr><td>1b</td><td>$\{C = UR\}$</td></tr>
</table>

5.5.2 Summary

Theorem 5.7 *Let $C \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{m \times k}$, and $B \in \mathbb{R}^{k \times n}$. Let*

- $m = m_0 + m_1 + \cdots m_{M-1}$, $m_i \geq 0$ *for* $i = 0, \ldots, M-1$;
- $n = n_0 + n_1 + \cdots n_{N-1}$, $n_j \geq 0$ *for* $j = 0, \ldots, N-1$; *and*
- $k = k_0 + k_1 + \cdots k_{K-1}$, $k_p \geq 0$ *for* $p = 0, \ldots, K-1$.

Partition

$$C = \left(\begin{array}{c|c|c|c} C_{0,0} & C_{0,1} & \cdots & C_{0,N-1} \\ \hline C_{1,0} & C_{1,1} & \cdots & C_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline C_{M-1,0} & C_{M-1,1} & \cdots & C_{M-1,N-1} \end{array}\right), A = \left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,K-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,K-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,K-1} \end{array}\right),$$

$$\text{and } B = \left(\begin{array}{c|c|c|c} B_{0,0} & B_{0,1} & \cdots & B_{0,N-1} \\ \hline B_{1,0} & B_{1,1} & \cdots & B_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline B_{K-1,0} & B_{K-1,1} & \cdots & B_{K-1,N-1} \end{array}\right),$$

with $C_{i,j} \in \mathbb{R}^{m_i \times n_j}$, $A_{i,p} \in \mathbb{R}^{m_i \times k_p}$, and $B_{p,j} \in \mathbb{R}^{k_p \times n_j}$. Then $C_{i,j} = \sum_{p=0}^{K-1} A_{i,p} B_{p,j}$.

If one partitions matrices C, A, and B into blocks, **and** one makes sure the dimensions match up, **then** blocked matrix-matrix multiplication proceeds exactly as does a regular matrix-matrix multiplication **except** that individual multiplications of scalars commute while (in general) individual multiplications with matrix blocks (submatrices) do not.

Properties of matrix-matrix multiplication

- Matrix-matrix multiplication is *not* commutative: In general, $AB \neq BA$.
- Matrix-matrix multiplication is associative: $(AB)C = A(BC)$. Hence, we can just write ABC.
- Special case: $e_i^T(Ae_j) = (e_i^T A)e_j = e_i^T A e_j = \alpha_{i,j}$ (the i, j element of A).
- Matrix-matrix multiplication is distributative: $A(B+C) = AB + AC$ and $(A+B)C = AC + BC$.

Transposing the product of two matrices

$$(AB)^T = B^T A^T$$

Product with identity matrix

In the following, assume the matrices are "of appropriate size."

$$IA = AI = A$$

Product with a diagonal matrix

$$\left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array}\right) \begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix} = \left(\begin{array}{c|c|c|c} \delta_0 a_0 & \delta_1 a_1 & \cdots & \delta_1 a_{n-1} \end{array}\right)$$

$$\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{m-1} \end{pmatrix} \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array}\right) = \left(\begin{array}{c} \delta_0 \widetilde{a}_0^T \\ \hline \delta_1 \widetilde{a}_1^T \\ \hline \vdots \\ \hline \delta_{m-1} \widetilde{a}_{m-1}^T \end{array}\right)$$

Product of triangular matrices

In the following, assume the matrices are "of appropriate size."

- The product of two lower triangular matrices is lower triangular.
- The product of two upper triangular matrices is upper triangular.

Matrix-matrix multiplication involving symmetric matrices

In the following, assume the matrices are "of appropriate size."

- $A^T A$ is symmetric.
- AA^T is symmetric.
- If A is symmetric then $A + \beta xx^T$ is symmetric.

Loops for computing $C := AB$

$$
\begin{aligned}
C &= \left(\begin{array}{c|c|c|c} \gamma_{0,0} & \gamma_{0,1} & \cdots & \gamma_{0,n-1} \\ \hline \gamma_{1,0} & \gamma_{1,1} & \cdots & \gamma_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \gamma_{m-1,0} & \gamma_{m-1,1} & \cdots & \gamma_{m-1,n-1} \end{array}\right) = \left(\begin{array}{c} \tilde a_0^T \\ \hline \tilde a_1^T \\ \hline \vdots \\ \hline \tilde a_{m-1}^T \end{array}\right) \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right) \\
&= \left(\begin{array}{c|c|c|c} \tilde a_0^T b_0 & \tilde a_0^T b_1 & \cdots & \tilde a_0^T b_{n-1} \\ \hline \tilde a_1^T b_0 & \tilde a_1^T b_1 & \cdots & \tilde a_1^T b_{n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \tilde a_{m-1}^T b_0 & \tilde a_{m-1}^T b_1 & \cdots & \tilde a_{m-1}^T b_{n-1} \end{array}\right).
\end{aligned}
$$

Algorithms for computing $C := AB + C$ via dot products.

$$
\begin{array}{l}
\textbf{for } j = 0, \ldots, n-1 \\
\quad \textbf{for } i = 0, \ldots, m-1 \\
\quad\quad \left.\begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \textbf{endfor} \end{array}\right\} \gamma_{i,j} := \tilde a_i^T b_j + \gamma_{i,j} \\
\quad \textbf{endfor} \\
\textbf{endfor}
\end{array}
\quad \text{or} \quad
\begin{array}{l}
\textbf{for } i = 0, \ldots, m-1 \\
\quad \textbf{for } j = 0, \ldots, n-1 \\
\quad\quad \left.\begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \textbf{endfor} \end{array}\right\} \gamma_{i,j} := \tilde a_i^T b_j + \gamma_{i,j} \\
\quad \textbf{endfor} \\
\textbf{endfor}
\end{array}
$$

Computing $C := AB$ by columns

$$
\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array}\right) = C = AB = A \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right) = \left(\begin{array}{c|c|c|c} Ab_0 & Ab_1 & \cdots & Ab_{n-1} \end{array}\right).
$$

Algorithms for computing $C := AB + C$:

$$
\begin{array}{l}
\textbf{for } j = 0, \ldots, n-1 \\
\quad \left.\begin{array}{l} \textbf{for } i = 0, \ldots, m-1 \\ \quad \textbf{for } p = 0, \ldots, k-1 \\ \quad\quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}\right\} c_j := Ab_j + c_j \\
\textbf{endfor}
\end{array}
\quad \text{or} \quad
\begin{array}{l}
\textbf{for } j = 0, \ldots, n-1 \\
\quad \left.\begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \textbf{for } i = 0, \ldots, m-1 \\ \quad\quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}\right\} c_j := Ab_j + c_j \\
\textbf{endfor}
\end{array}
$$

Algorithm: $C := \text{GEMM\_UNB\_VAR1}(A,B,C)$

Partition $B \rightarrow \left(B_L \,\middle|\, B_R \right)$, $C \rightarrow \left(C_L \,\middle|\, C_R \right)$
  where B_L has 0 columns, C_L has 0 columns

while $n(B_L) < n(B)$ **do**

  Repartition

$$\left(B_L \,\middle|\, B_R \right) \rightarrow \left(B_0 \,\middle|\, b_1 \,\middle|\, B_2 \right), \left(C_L \,\middle|\, C_R \right) \rightarrow \left(C_0 \,\middle|\, c_1 \,\middle|\, C_2 \right)$$

  where b_1 has 1 column, c_1 has 1 column

$c_1 := Ab_1 + c_1$

  Continue with

$$\left(B_L \,\middle|\, B_R \right) \leftarrow \left(B_0 \,\middle|\, b_1 \,\middle|\, B_2 \right), \left(C_L \,\middle|\, C_R \right) \leftarrow \left(C_0 \,\middle|\, c_1 \,\middle|\, C_2 \right)$$

endwhile

Computing $C := AB$ by rows

$$\left(\begin{array}{c} \tilde{c}_0^T \\ \hline \tilde{c}_1^T \\ \hline \vdots \\ \hline \tilde{c}_{m-1}^T \end{array} \right) = C = AB = \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right) B = \left(\begin{array}{c} \tilde{a}_0^T B \\ \hline \tilde{a}_1^T B \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T B \end{array} \right).$$

Algorithms for computing $C := AB + C$ by rows:

$$\begin{array}{l} \textbf{for } i = 0, \ldots, m-1 \\ \quad \left. \begin{array}{l} \textbf{for } j = 0, \ldots, n-1 \\ \quad \textbf{for } p = 0, \ldots, k-1 \\ \quad\quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array} \right\} \tilde{c}_i^T := \tilde{a}_i^T B + \tilde{c}_i^T \\ \textbf{endfor} \end{array} \quad \text{or} \quad \begin{array}{l} \textbf{for } i = 0, \ldots, m-1 \\ \quad \left. \begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \textbf{for } j = 0, \ldots, n-1 \\ \text{p} \quad\quad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array} \right\} \tilde{c}_i^T := \tilde{a}_i^T B + \tilde{c}_i^T \\ \textbf{endfor} \end{array}$$

Algorithm: $C := \text{GEMM\_UNB\_VAR2}(A, B, C)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $C \rightarrow \left(\begin{array}{c} C_T \\ \hline C_B \end{array}\right)$

where A_T has 0 rows, C_T has 0 rows

while $m(A_T) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} C_T \\ \hline C_B \end{array}\right) \rightarrow \left(\begin{array}{c} C_0 \\ \hline c_1^T \\ \hline C_2 \end{array}\right)$$

where a_1 has 1 row, c_1 has 1 row

$c_1^T := a_1^T B + c_1^T$

Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} C_T \\ \hline C_B \end{array}\right) \leftarrow \left(\begin{array}{c} C_0 \\ \hline c_1^T \\ \hline C_2 \end{array}\right)$$

endwhile

Computing $C := AB$ via rank-1 updates

$$C = AB = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{k-1} \end{array}\right) \left(\begin{array}{c} \tilde{b}_0^T \\ \hline \tilde{b}_1^T \\ \hline \vdots \\ \hline \tilde{b}_{k-1}^T \end{array}\right) = a_0 \tilde{b}_0^T + a_1 \tilde{b}_1^T + \cdots + a_{k-1} \tilde{b}_{k-1}^T.$$

Algorithm for computing $C := AB + C$ via rank-1 updates:

$$\begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \left.\begin{array}{l} \textbf{for } j = 0, \ldots, n-1 \\ \quad \textbf{for } i = 0, \ldots, m-1 \\ \qquad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}\right\} C := a_p \tilde{b}_p^T + C \\ \textbf{endfor} \end{array} \quad \text{or} \quad \begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \left.\begin{array}{l} \textbf{for } i = 0, \ldots, m-1 \\ \quad \textbf{for } j = 0, \ldots, n-1 \\ \qquad \gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j} \\ \quad \textbf{endfor} \\ \textbf{endfor} \end{array}\right\} C := a_p \tilde{b}_p^T + C \\ \textbf{endfor} \end{array}$$

Algorithm: $C := \text{GEMM\_UNB\_VAR3}(A, B, C)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right), B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right)$

where A_L has 0 columns, B_T has 0 rows

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array} \right)$$

where a_1 has 1 column, b_1 has 1 row

$C := a_1 b_1^T + C$

Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array} \right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array} \right)$$

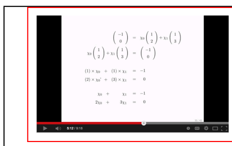
endwhile

Week 6

Gaussian Elimination

6.1 Opening Remarks

6.1.1 Solving Linear Systems



☛ YouTube

6.1.2 Outline

6.1.3 What You Will Learn

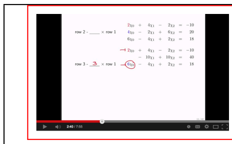
Upon completion of this unit, you should be able to

- Apply Gaussian elimination to reduce a system of linear equations into an upper triangular system of equations.
- Apply back(ward) substitution to solve an upper triangular system in the form $Ux = b$.
- Apply forward substitution to solve a lower triangular system in the form $Lz = b$.
- Represent a system of equations using an appended matrix.
- Reduce a matrix to an upper triangular matrix with Gauss transforms and then apply the Gauss transforms to a right-hand side.
- Solve the system of equations in the form $Ax = b$ using LU factorization.
- Relate LU factorization and Gaussian elimination.
- Relate solving with a unit lower triangular matrix and forward substitution.
- Relate solving with an upper triangular matrix and back substitution.
- Create code for various algorithms for Gaussian elimination, forward substitution, and back substitution.
- Determine the cost functions for LU factorization and algorithms for solving with triangular matrices.

Track your progress in Appendix B.

6.2 Gaussian Elimination

6.2.1 Reducing a System of Linear Equations to an Upper Triangular System



☛ YouTube

A system of linear equations

Consider the system of linear equations

$$\begin{array}{rcrcrcr} 2x & + & 4y & - & 2z & = & -10 \\ 4x & - & 2y & + & 6z & = & 20 \\ 6x & - & 4y & + & 2z & = & 18. \end{array}$$

Notice that x, y, and z are just variables for which we can pick any symbol or letter we want. To be consistent with the notation we introduced previously for naming components of vectors, we identify them instead with χ_0, χ_1, and and χ_2, respectively:

$$\begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ 4\chi_0 & - & 2\chi_1 & + & 6\chi_2 & = & 20 \\ 6\chi_0 & - & 4\chi_1 & + & 2\chi_2 & = & 18. \end{array}$$

Gaussian elimination (transform linear system of equations to an upper triangular system)

Solving the above linear system relies on the fact that its solution does not change if

1. Equations are reordered (not used until next week);
2. An equation in the system is modified by subtracting a multiple of another equation in the system from it; and/or
3. Both sides of an equation in the system are scaled by a nonzero number.

These are the tools that we will employ.

The following steps are knows as (Gaussian) elimination. They transform a system of linear equations to an equivalent upper triangular system of linear equations:

- Subtract $\lambda_{1,0} = (4/2) = 2$ times the first equation from the second equation:

$$\begin{array}{c|c} \text{Before} & \text{After} \\ \begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ 4\chi_0 & - & 2\chi_1 & + & 6\chi_2 & = & 20 \\ 6\chi_0 & - & 4\chi_1 & + & 2\chi_2 & = & 18 \end{array} & \begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ & - & 10\chi_1 & + & 10\chi_2 & = & 40 \\ 6\chi_0 & - & 4\chi_1 & + & 2\chi_2 & = & 18 \end{array} \end{array}$$

- Subtract $\lambda_{2,0} = (6/2) = 3$ times the first equation from the third equation:

$$\begin{array}{c|c} \text{Before} & \text{After} \\ \begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ & - & 10\chi_1 & + & 10\chi_2 & = & 40 \\ 6\chi_0 & - & 4\chi_1 & + & 2\chi_2 & = & 18 \end{array} & \begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ & - & 10\chi_1 & + & 10\chi_2 & = & 40 \\ & - & 16\chi_1 & + & 8\chi_2 & = & 48 \end{array} \end{array}$$

- Subtract $\lambda_{2,1} = ((-16)/(-10)) = 1.6$ times the second equation from the third equation:

Before	After
$\begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ & - & 10\chi_1 & + & 10\chi_2 & = & 40 \\ & - & 16\chi_1 & + & 8\chi_2 & = & 48 \end{array}$	$\begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ & - & 10\chi_1 & + & 10\chi_2 & = & 40 \\ & & & - & 8\chi_2 & = & -16 \end{array}$

This now leaves us with an upper triangular system of linear equations.

> In the above Gaussian elimination procedure, $\lambda_{1,0}$, $\lambda_{2,0}$, and $\lambda_{2,1}$ are called the *multipliers*. Notice that their subscripts indicate the coefficient in the linear system that is being eliminated.

Back substitution (solve the upper triangular system)

The equivalent upper triangular system of equations is now solved via *back substitution*:

- Consider the last equation,
$$-8\chi_2 = -16.$$
Scaling both sides by by $1/(-8)$ we find that
$$\chi_2 = -16/(-8) = 2.$$

- Next, consider the second equation,
$$-10\chi_1 + 10\chi_2 = 40.$$
We know that $\chi_2 = 2$, which we plug into this equation to yield
$$-10\chi_1 + 10(2) = 40.$$
Rearranging this we find that
$$\chi_1 = (40 - 10(2))/(-10) = -2.$$

- Finally, consider the first equation,
$$2\chi_0 + 4\chi_1 - 2\chi_2 = -10$$
We know that $\chi_2 = 2$ and $\chi_1 = -2$, which we plug into this equation to yield
$$2\chi_0 + 4(-2) - 2(2) = -10.$$
Rearranging this we find that
$$\chi_0 = (-10 - (4(-2) - (2)(2)))/2 = 1.$$

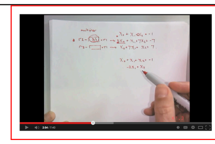
Thus, the solution is the vector
$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ -2 \\ 2 \end{pmatrix}.$$

Check your answer (ALWAYS!)

Check the answer (by plugging $\chi_0 = 1$, $\chi_1 = -2$, and $\chi_2 = 2$ into the original system)

$$\begin{array}{rcrcrcrl} 2(1) & + & 4(-2) & - & 2(2) & = & -10 & \checkmark \\ 4(1) & - & 2(-2) & + & 6(2) & = & 20 & \checkmark \\ 6(1) & - & 4(-2) & + & 2(2) & = & 18 & \checkmark \end{array}$$

Homework 6.2.1.1



☛ YouTube

Practice reducing a system of linear equations to an upper triangular system of linear equations by visiting the Practice with Gaussian Elimination webpage we created for you. For now, only work with the top part of that webpage.

☛ SEE ANSWER

Homework 6.2.1.2 Compute the solution of the linear system of equations given by

$$\begin{array}{rcrcrcr} -2\chi_0 & + & \chi_1 & + & 2\chi_2 & = & 0 \\ 4\chi_0 & - & \chi_1 & - & 5\chi_2 & = & 4 \\ 2\chi_0 & - & 3\chi_1 & - & \chi_2 & = & -6 \end{array}$$

- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \square \\ \square \\ \square \end{pmatrix}$

☛ SEE ANSWER

Homework 6.2.1.3 Compute the coefficients γ_0, γ_1, and γ_2 so that

$$\sum_{i=0}^{n-1} i = \gamma_0 + \gamma_1 n + \gamma_2 n^2$$

(by setting up a system of linear equations).

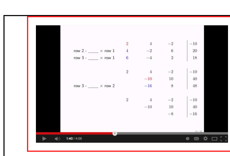
☛ SEE ANSWER

Homework 6.2.1.4 Compute γ_0, γ_1, γ_2, and γ_3 so that

$$\sum_{i=0}^{n-1} i^2 = \gamma_0 + \gamma_1 n + \gamma_2 n^2 + \gamma_3 n^3.$$

☛ SEE ANSWER

6.2.2 Appended Matrices



☛ YouTube

Representing the system of equations with an appended matrix

Now, in the above example, it becomes very cumbersome to always write the entire equation. The information is encoded in the coefficients in front of the χ_i variables, and the values to the right of the equal signs. Thus, we could just let

$$\left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ 4 & -2 & 6 & 20 \\ 6 & -4 & 2 & 18 \end{array}\right) \quad \text{represent} \quad \begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ 4\chi_0 & - & 2\chi_1 & + & 6\chi_2 & = & 20 \\ 6\chi_0 & - & 4\chi_1 & + & 2\chi_2 & = & 18. \end{array}$$

Then Gaussian elimination can simply operate on this array of numbers as illustrated next.

Gaussian elimination (transform to upper triangular system of equations)

- Subtract $\lambda_{1,0} = (4/2) = 2$ times the first row from the second row:

$$\begin{array}{c|c} \text{Before} & \text{After} \\ \left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ 4 & -2 & 6 & 20 \\ 6 & -4 & 2 & 18 \end{array}\right) & \left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ & -10 & 10 & 40 \\ 6 & -4 & 2 & 18 \end{array}\right). \end{array}$$

- Subtract $\lambda_{2,0} = (6/2) = 3$ times the first row from the third row:

$$\begin{array}{c|c} \text{Before} & \text{After} \\ \left(\begin{array}{rrr|r} 2 & 4 & 2 & -10 \\ & -10 & 10 & 40 \\ 6 & -4 & 2 & 18 \end{array}\right) & \left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ & -10 & 10 & 40 \\ & -16 & 8 & 48 \end{array}\right). \end{array}$$

- Subtract $\lambda_{2,1} = ((-16)/(-10)) = 1.6$ times the second row from the third row:

$$\begin{array}{c|c} \text{Before} & \text{After} \\ \left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ & -10 & 10 & 40 \\ & -16 & 8 & 48 \end{array}\right) & \left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ & -10 & 10 & 40 \\ & & -8 & -16 \end{array}\right). \end{array}$$

This now leaves us with an upper triangular system of linear equations.

Back substitution (solve the upper triangular system)

The equivalent upper triangular system of equations is now solved via *back substitution*:

- The final result above represents

$$\begin{pmatrix} 2 & 4 & -2 \\ & -10 & 10 \\ & & -8 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -10 \\ 40 \\ -16 \end{pmatrix}$$

  or, equivalently,

$$\begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & - & 2\chi_2 & = & -10 \\ & - & 10\chi_1 & + & 10\chi_2 & = & 40 \\ & & & - & 8\chi_2 & = & -16 \end{array}$$

- Consider the last equation,
$$8\chi_2 = -16.$$
Scaling both sides by by $1/(-8)$ we find that
$$\chi_2 = -16/(-8) = 2.$$
- Next, consider the second equation,
$$-10\chi_1 + 10\chi_2 = 40.$$
We know that $\chi_2 = 2$, which we plug into this equation to yield
$$-10\chi_1 + 10(2) = 40.$$
Rearranging this we find that
$$\chi_1 = (40 - 10(2))/(-10) = -2.$$
- Finally, consider the first equation,
$$2\chi_0 + 4\chi_1 - 2\chi_2 = -10$$
We know that $\chi_2 = 2$ and $\chi_1 = -2$, which we plug into this equation to yield
$$2\chi_0 + 4(-2) - 2(2) = -10.$$
Rearranging this we find that
$$\chi_0 = (-10 - (4(-2) + (-2)(-2)))/2 = 1.$$

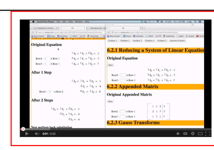
Thus, the solution is the vector
$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ -2 \\ 2 \end{pmatrix}.$$

Check your answer (ALWAYS!)

Check the answer (by plugging $\chi_0 = 1$, $\chi_1 = -2$, and $\chi_2 = 2$ into the original system)
$$\begin{array}{rcrcrcrl} 2(1) & + & 4(-2) & - & 2(2) & = & -10 & \checkmark \\ 4(1) & - & 2(-2) & + & 6(2) & = & 20 & \checkmark \\ 6(1) & - & 4(-2) & + & 2(2) & = & 18 & \checkmark \end{array}$$

Alternatively, you can check that
$$\begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ -2 \\ 2 \end{pmatrix} = \begin{pmatrix} -10 \\ 20 \\ 18 \end{pmatrix} \quad \checkmark$$

Homework 6.2.2.1



☛ YouTube

Practice reducing a system of linear equations to an upper triangular system of linear equations by visiting the Practice with Gaussian Elimination webpage we created for you. For now, only work with the top two parts of that webpage.

☛ SEE ANSWER

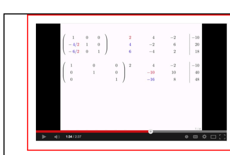
Homework 6.2.2.2 Compute the solution of the linear system of equations expressed as an appended matrix given by

$$\left(\begin{array}{rrr|r} -1 & 2 & -3 & 2 \\ -2 & 2 & -8 & 10 \\ 2 & -6 & 6 & -2 \end{array}\right)$$

- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \Box \\ \Box \\ \Box \end{pmatrix}$

☛ SEE ANSWER

6.2.3 Gauss Transforms



☛ YouTube

Homework 6.2.3.1
Compute ONLY the values in the boxes. A $\star$ means a value that we don't care about.

- $\begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 345 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \star & \star & \star \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ \Box & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 2 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ 0 & \Box & \Box \\ \Box & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ \Box & 1 & 0 \\ \Box & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 2 & -2 & 6 \\ -4 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ 0 & \Box & \Box \\ 0 & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \Box & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & -16 & 8 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & 0 & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & \Box \\ 0 & 1 & \Box \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -8 \\ 1 & 1 & -4 \\ -1 & -2 & 4 \end{pmatrix} = \begin{pmatrix} \Box & \Box & 0 \\ \Box & \Box & 0 \\ \Box & \Box & \Box \end{pmatrix}.$

☛ SEE ANSWER

Theorem 6.1 *Let $\widehat{L}_j$ be a matrix that equals the identity, except that for $i > j$the (i, j) elements (the ones below the diagonal in the jth column) have been replaced with $-\lambda_{i,,j}$:*

$$\widehat{L}_j = \begin{pmatrix} I_j & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+1,j} & 1 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+2,j} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -\lambda_{m-1,j} & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Then $\widehat{L}_j A$ equals the matrix A except that for $i > j$ the ith row is modified by subtracting $\lambda_{i,j}$ times the jth row from it. Such a *matrix $\widehat{L}_j$ is called a Gauss transform.*

Proof: Let

$$\widehat{L}_j = \begin{pmatrix} I_j & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+1,j} & 1 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+2,j} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -\lambda_{m-1,j} & 0 & 0 & \cdots & 1 \end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix} A_{0:j-1,:} \\ \check{a}_j^T \\ \check{a}_{j+1}^T \\ \check{a}_{j+2}^T \\ \vdots \\ \check{a}_{m-1}^T \end{pmatrix},$$

where I_k equals a $k \times k$ identity matrix, $A_{s:t,:}$ equals the matrix that consists of rows s through t from matrix A, and $\check{a}_k^T$ equals the kth row of A. Then

$$\begin{aligned} \widehat{L}_j A &= \begin{pmatrix} I_j & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+1,j} & 1 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+2,j} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -\lambda_{m-1,j} & 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} A_{0:j-1,:} \\ \check{a}_j^T \\ \check{a}_{j+1}^T \\ \check{a}_{j+2}^T \\ \vdots \\ \check{a}_{m-1}^T \end{pmatrix} \\ &= \begin{pmatrix} A_{0:j-1,:} \\ \check{a}_j^T \\ -\lambda_{j+1,j}\check{a}_j^T + \check{a}_{j+1}^T \\ -\lambda_{j+2,j}\check{a}_j^T + \check{a}_{j+2}^T \\ \vdots \\ -\lambda_{m-1,j}\check{a}_j^T + \check{a}_{m-1}^T \end{pmatrix} = \begin{pmatrix} A_{0:j-1,:} \\ \check{a}_j^T \\ \check{a}_{j+1}^T - \lambda_{j+1,j}\check{a}_j^T \\ \check{a}_{j+2}^T - \lambda_{j+2,j}\check{a}_j^T \\ \vdots \\ \check{a}_{m-1}^T - \lambda_{m-1,j}\check{a}_j^T \end{pmatrix}. \end{aligned}$$

Gaussian elimination (transform to upper triangular system of equations)

- Subtract $\lambda_{1,0} = (4/2) = 2$ times the first row from the second row *and* subtract $\lambda_{2,0} = (6/2) = 3$ times the first row from the third row:

Before

$$\begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ 4 & -2 & 6 & 20 \\ 6 & -4 & 2 & 18 \end{array}\right)$$

After

$$\left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ 0 & -10 & 10 & 40 \\ 0 & -16 & 8 & 48 \end{array}\right)$$

- Subtract $\lambda_{2,1} = ((-16)/(-10)) = 1.6$ times the second row from the third row:

Before

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix} \left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ 0 & -10 & 10 & 40 \\ 0 & -16 & 8 & 48 \end{array}\right)$$

After

$$\left(\begin{array}{rrr|r} 2 & 4 & -2 & -10 \\ 0 & -10 & 10 & 40 \\ 0 & 0 & -8 & -16 \end{array}\right)$$

This now leaves us with an upper triangular appended matrix.

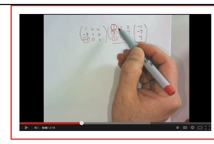
Back substitution (solve the upper triangular system)

As before.

Check your answer (ALWAYS!)

As before.

Homework 6.2.3.2

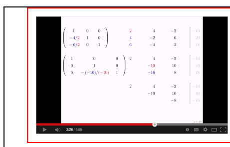


☛ YouTube

Practice reducing an appended sytem to an upper triangular form with Gauss transforms by visiting the Practice with Gaussian Elimination webpage we created for you. For now, only work with the top three parts of that webpage.

☛ SEE ANSWER

6.2.4 Computing Separately with the Matrix and Right-Hand Side (Forward Substitution)



☛ YouTube

Transform to matrix to upper triangular matrix

- Subtract $\lambda_{1,0} = (4/2) = 2$ times the first row from the second row *and* subtract $\lambda_{2,0} = (6/2) = 3$ times the first row from the third row:

Before	After
$\left(\begin{array}{rrr} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{array}\right)\left(\begin{array}{rrr} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{array}\right)$	$\left(\begin{array}{rrr} 2 & 4 & -2 \\ 2 & -10 & 10 \\ 3 & -16 & 8 \end{array}\right)$

  Notice that we are storing the multipliers over the zeroes that are introduced.

- Subtract $\lambda_{2,1} = ((-16)/(-10)) = 1.6$ times the second row from the third row:

Before	After
$\left(\begin{array}{rrr} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{array}\right)\left(\begin{array}{rrr} 2 & 4 & -2 \\ 2 & -10 & 10 \\ 3 & -16 & 8 \end{array}\right)$	$\left(\begin{array}{rrr} 2 & 4 & -2 \\ 2 & -10 & 10 \\ 3 & 1.6 & -8 \end{array}\right)$

  (The transformation does not affect the $(2,0)$ element that equals 3 because we are merely storing a previous multiplier there.) Again, notice that we are storing the multiplier over the zeroes that are introduced.

This now leaves us with an upper triangular matrix *and* the multipliers used to transform the matrix to the upper triangular matrix.

Forward substitution (applying the transforms to the right-hand side)

We now take the transforms (multipliers) that were computed during Gaussian Elimination (and stored over the zeroes) and apply them to the right-hand side vector.

- Subtract $\lambda_{1,0} = 2$ times the first component from the second component *and* subtract $\lambda_{2,0} = 3$ times the first component from the third component:

$$\begin{array}{c|c} \text{Before} & \text{After} \\ \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \begin{pmatrix} -10 \\ 20 \\ 18 \end{pmatrix} & \begin{pmatrix} -10 \\ 40 \\ 48 \end{pmatrix} \end{array}$$

- Subtract $\lambda_{2,1} = 1.6$ times the second component from the third component:

$$\begin{array}{c|c} \text{Before} & \text{After} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix} \begin{pmatrix} -10 \\ 40 \\ 48 \end{pmatrix} & \begin{pmatrix} -10 \\ 40 \\ -16 \end{pmatrix} \end{array}$$

The important thing to realize is that this updates the right-hand side exactly as the appended column was updated in the last unit. This process is often referred to as *forward substitution*.

Back substitution (solve the upper triangular system)

As before.

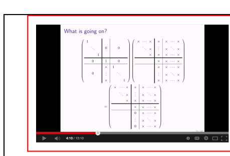
Check your answer (ALWAYS!)

As before.

Homework 6.2.4.1 No video this time! We trust that you have probably caught on to how to use the webpage. Practice reducing a matrix to an upper triangular matrix with Gauss transforms and then applying the Gauss transforms to a right-hand side by visiting the Practice with Gaussian Elimination webpage we created for you. Now you can work with all parts of the webpage. Be sure to compare and contrast!

☛ SEE ANSWER

6.2.5 Towards an Algorithm



☛ YouTube

Gaussian elimination (transform to upper triangular system of equations)

- As is shown below, compute $\begin{pmatrix} \lambda_{1,0} \\ \lambda_{2,0} \end{pmatrix} = \begin{pmatrix} 4 \\ 6 \end{pmatrix} / 2 = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$ and apply the Gauss transform to the matrix:

Algorithm: $A := \text{GAUSSIAN\_ELIMINATION}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

  where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

$$a_{21} := a_{21}/\alpha_{11} \qquad (= l_{21})$$
$$A_{22} := A_{22} - a_{21}a_{12}^T \qquad (= A_{22} - l_{21}a_{12}^T)$$

  Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 6.1: Algorithm that transforms a matrix A into an upper triangular matrix U, overwriting the uppertriangular part of A with that U. The elements of A below the diagonal are overwritten with the multipliers.

Before	After
$\left(\begin{array}{\|c\|cc} 1 & 0 & 0 \\ \hline -2 & 1 & 0 \\ -3 & 0 & 1 \end{array}\right)\left(\begin{array}{\|c\|cc} \color{red}{2} & 4 & -2 \\ \hline \color{blue}{4} & -2 & 6 \\ \color{blue}{6} & -4 & 2 \end{array}\right)$	$\left(\begin{array}{\|c\|cc} 2 & 4 & -2 \\ \hline {\scriptstyle 2} & -10 & 10 \\ {\scriptstyle 3} & -16 & 8 \end{array}\right)$

- As is shown below, compute $\left(\begin{array}{c} \lambda_{2,1} \end{array}\right) = \left(\begin{array}{c} -16 \end{array}\right)/(-10) = \left(\begin{array}{c} 1.6 \end{array}\right)$ and apply the Gauss transform to the matrix:

Before	After
$\left(\begin{array}{c\|c\|c} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1.6 & 1 \end{array}\right)\left(\begin{array}{c\|c\|c} 2 & 4 & -2 \\ \hline {\scriptstyle 2} & \color{red}{-10} & 10 \\ {\scriptstyle 3} & \color{blue}{-16} & 8 \end{array}\right)$	$\left(\begin{array}{ccc} 2 & 4 & -2 \\ {\scriptstyle 2} & -10 & 10 \\ {\scriptstyle 3} & {\scriptstyle 1.6} & -8 \end{array}\right)$

(The transformation does not affect the $(2,0)$ element that equals ${\scriptstyle 3}$ because we are merely storing a previous multiplier there.)

This now leaves us with an upper triangular matrix *and* the multipliers used to transform the matrix to the upper triangular matrix.

The insights in this section are summarized in the algorithm in Figure 6.1, in which the original matrix A is overwritten with the upper triangular matrix that results from Gaussian elimination and the strictly lower triangular elements are overwritten by the multipliers.

Algorithm: $b := \text{FORWARD\_SUBSTITUTION}(A, b)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

  where A_{TL} is 0×0, b_T has 0 rows

while $m(A_{TL}) < m(A)$ **do**

 Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$$b_2 := b_2 - \beta_1 a_{21} \qquad (= b_2 - \beta_1 l_{21})$$

 Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 6.2: Algorithm that applies the multipliers (stored in the elements of A below the diagonal) to a right-hand side vector b.

Forward substitution (applying the transforms to the right-hand side)

We now take the transforms (multipliers) that were computed during Gaussian Elimination (and stored over the zeroes) and apply them to the right-hand side vector.

- Subtract $\lambda_{1,0} = 2$ times the first component from the second component *and* subtract $\lambda_{2,0} = 3$ times the first component from the third component:

Before	After
$\left(\begin{array}{r\|rr} 1 & 0 & 0 \\ \hline -2 & 1 & 0 \\ -3 & 0 & 1 \end{array}\right) \left(\begin{array}{r} -10 \\ \hline 20 \\ 18 \end{array}\right)$	$\left(\begin{array}{r} -10 \\ \hline 40 \\ 48 \end{array}\right)$

- Subtract $\lambda_{2,1} = 1.6$ times the second component from the third component:

Before	After
$\left(\begin{array}{r\|r\|r} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1.6 & 1 \end{array}\right) \left(\begin{array}{r} -10 \\ \hline 40 \\ \hline 48 \end{array}\right)$	$\left(\begin{array}{r} -10 \\ \hline 40 \\ \hline -16 \end{array}\right)$

The important thing to realize is that this updates the right-hand side exactly as the appended column was updated in the last unit. This process is often referred to as *forward substitution*.

The above observations motivate the algorithm for forward substitution in Figure 6.2.

Back substitution (solve the upper triangular system)

As before.

Check your answer (ALWAYS!)

As before.

Homework 6.2.5.1 Implement the algorithms in Figures 6.1 and 6.2

- `[ A_out ] = GaussianElimination( A )`
- `[ b_out ] = ForwardSubstitution( A, b )`

You can check that they compute the right answers with the following script:

- `test_GausianElimination.m`

This script exercises the functions by factoring the matrix

```
A = [
    2    0    1    2
   -2   -1    1   -1
    4   -1    5    4
   -4    1   -3   -8
]
```

by calling

```
LU = GaussianElimination( A )
```

Next, solve $Ax = b$ where

```
b = [
    2
    2
   11
   -3
]
```

by first apply forward substitution to b, using the output matrix `LU`:

```
bhat = ForwardSubstitution( LU, b )
```

extracting the upper triangular matrix U from `LU`:

```
U = triu( LU )
```

and then solving $Ux = \widehat{b}$ (which is equivalent to backward substitution) with the MATLAB intrinsic function:

```
x = U \ bhat
```

Finally, check that you got the right answer:

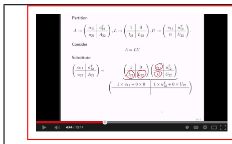
```
b - A * x
```

(the result should be a zero vector with four elements).

☛ SEE ANSWER

6.3 Solving $Ax = b$ via LU Factorization

6.3.1 LU factorization (Gaussian elimination)



☛ YouTube

In this unit, we will use the insights into how blocked matrix-matrix and matrix-vector multiplication works to derive and state algorithms for solving linear systems in a more concise way that translates more directly into algorithms.

The idea is that, under circumstances to be discussed later, a matrix $A \in \mathbb{R}^{n\times n}$ can be factored into the product of two matrices $L, U \in \mathbb{R}^{n\times n}$:

$$A = LU,$$

where L is unit lower triangular and U is upper triangular.

Assume $A \in \mathbb{R}^{n\times n}$ is given and that L and U are to be computed such that $A = LU$, where $L \in \mathbb{R}^{n\times n}$ is unit lower triangular and $U \in \mathbb{R}^{n\times n}$ is upper triangular. We derive an algorithm for computing this operation by partitioning

$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right), \quad L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), \quad \text{and} \quad U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right).$$

Now, $A = LU$ implies (using what we learned about multiplying matrices that have been partitioned into submatrices)

$$\begin{aligned}
\overbrace{\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right)}^{A} &= \overbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)}^{L} \overbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)}^{U} \\
&= \overbrace{\left(\begin{array}{c|c} 1 \times \upsilon_{11} + 0 \times 0 & 1 \times u_{12}^T + 0 \times U_{22} \\ \hline l_{21}\upsilon_{11} + L_{22} \times 0 & l_{21}u_{12}^T + L_{22}U_{22} \end{array}\right)}^{LU}. \\
&= \overbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline l_{21}\upsilon_{11} & l_{21}u_{12}^T + L_{22}U_{22} \end{array}\right)}^{LU}.
\end{aligned}$$

For two matrices to be equal, their elements must be equal, and therefore, if they are partitioned conformally, their submatrices must be equal:

$$\begin{array}{c|c} \alpha_{11} = \upsilon_{11} & a_{12}^T = u_{12}^T \\ \hline a_{21} = l_{21}\upsilon_{11} & A_{22} = l_{21}u_{12}^T + L_{22}U_{22} \end{array}$$

or, rearranging,

$$\begin{array}{c|c} \upsilon_{11} = \alpha_{11} & u_{12}^T = a_{12}^T \\ \hline l_{21} = a_{21}/\upsilon_{11} & L_{22}U_{22} = A_{22} - l_{21}u_{12}^T \end{array}.$$

This suggests the following steps for **overwriting** a matrix A with its LU factorization:

- Partition

$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right).$$

- Update $a_{21} = a_{21}/\alpha_{11} (= l_{21})$. (Scale a_{21} by $1/\alpha_{11}$!)

Algorithm: $A := \text{LU\_UNB\_VAR5}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

$a_{21} := a_{21}/\alpha_{11} \qquad (= l_{21})$

$A_{22} := A_{22} - a_{21}a_{12}^T \qquad (= A_{22} - l_{21}a_{12}^T)$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 6.3: LU factorization algorithm.

- Update $A_{22} = A_{22} - a_{21}a_{12}^T (= A_{22} - l_{21}u_{12}^T)$ (Rank-1 update!).
- Overwrite A_{22} with L_{22} and U_{22} by repeating with $A = A_{22}$.

This will leave U in the upper triangular part of A and the strictly lower triangular part of L in the strictly lower triangular part of A. The diagonal elements of L need not be stored, since they are known to equal one.

The above can be summarized in Figure 6.14. If one compares this to the algorithm GAUSSIAN_ELIMINATION we arrived at in Unit 6.2.5, you find they are identical! LU factorization **is** Gaussian elimination.

We illustrate in Fgure 6.4 how LU factorization with a 3×3 matrix proceeds. Now, compare this to Gaussian elimination with an augmented system, in Figure 6.5. It should strike you that exactly the same computations are performed with the coefficient matrix to the left of the vertical bar.

Step	$\left(\begin{array}{c\|c\|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$	a_{21}/α_{11}	$A_{22} - a_{21}a_{12}^T$
1-2	$\left(\begin{array}{c\|cc} -2 & -1 & 1 \\ \hline 2 & -2 & -3 \\ -4 & 4 & 7 \end{array}\right)$	$\begin{pmatrix} 2 \\ -4 \end{pmatrix} / (-2) = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$	$\begin{pmatrix} -2 & -3 \\ 4 & 7 \end{pmatrix} - \begin{pmatrix} -1 \\ 2 \end{pmatrix} \begin{pmatrix} -1 & 1 \end{pmatrix} = \begin{pmatrix} -3 & -2 \\ 6 & 5 \end{pmatrix}$
3	$\left(\begin{array}{c\|c\|c} -2 & -1 & 1 \\ \hline -1 & -3 & -2 \\ \hline 2 & 6 & 5 \end{array}\right)$	$\begin{pmatrix} 6 \end{pmatrix} / (-3) = \begin{pmatrix} -2 \end{pmatrix}$	$\begin{pmatrix} 5 \end{pmatrix} - \begin{pmatrix} -2 \end{pmatrix} \begin{pmatrix} -2 \end{pmatrix} = \begin{pmatrix} 1 \end{pmatrix}$
	$\left(\begin{array}{cc\|c} -2 & -1 & 1 \\ -1 & -3 & -2 \\ \hline 2 & -2 & 1 \end{array}\right)$		

Figure 6.4: LU factorization with a 3×3 matrix

Step	Current system	Multiplier	Operation
1	$\left(\begin{array}{rrr\|r} -2 & -1 & 1 & 6 \\ 2 & -2 & -3 & 3 \\ -4 & 4 & 7 & -3 \end{array}\right)$	$\frac{2}{-2} = -1$	$\begin{array}{rrrrr} & 2 & -2 & -3 & 3 \\ -1\times(& -2 & -1 & 1 & 6) \\ \hline & 0 & -3 & -2 & 9 \end{array}$
2	$\left(\begin{array}{rrr\|r} -2 & -1 & 1 & 6 \\ 0 & -3 & -2 & 9 \\ -4 & 4 & 7 & -3 \end{array}\right)$	$\frac{-4}{-2} = 2$	$\begin{array}{rrrrr} & -4 & 4 & 7 & -3 \\ -(2)\times(& -2 & -1 & 1 & 6) \\ \hline & 0 & 6 & 5 & -15 \end{array}$
3	$\left(\begin{array}{rrr\|r} -2 & -1 & 1 & 6 \\ 0 & -3 & -2 & 9 \\ 0 & 6 & 5 & -15 \end{array}\right)$	$\frac{6}{-3} = -2$	$\begin{array}{rrrrr} & 0 & 6 & 5 & -15 \\ -(-2)\times(& 0 & -3 & -2 & 9) \\ \hline & 0 & 0 & 1 & 3 \end{array}$
4	$\left(\begin{array}{rrr\|r} -2 & -1 & 1 & 6 \\ 0 & -3 & -2 & 9 \\ 0 & 0 & 1 & 3 \end{array}\right)$		

Figure 6.5: Gaussian elimination with an augmented system.

Homework 6.3.1.1 Implement the algorithm in Figures 6.4.

- `[ A_out ] = LU_unb_var5( A )`

You can check that they compute the right answers with the following script:

- `test_LU_unb_var5.m`

This script exercises the functions by factoring the matrix

```
A = [
     2     0     1     2
    -2    -1     1    -1
     4    -1     5     4
    -4     1    -3    -8
]
```

by calling

```
LU = LU_unb_var5( A )
```

Next, it extracts the unit lower triangular matrix and upper triangular matrix:

```
L = tril( LU, -1 ) + eye( size( A ) )

U = triu( LU )
```

and checks if the correct factors were computed:

```
A - L * U
```

which should yield a 4×4 zero matrix.

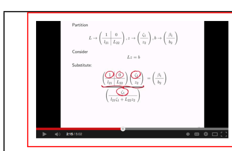
☛ SEE ANSWER

Homework 6.3.1.2 Compute the LU factorization of

$$\begin{pmatrix} 1 & -2 & 2 \\ 5 & -15 & 8 \\ -2 & -11 & -11 \end{pmatrix}.$$

☛ SEE ANSWER

6.3.2 Solving $Lz = b$ (Forward substitution)



☛ YouTube

Next, we show how forward substitution is the same as solving the linear system $Lz = b$ where b is the right-hand side and L is the matrix that resulted from the LU factorization (and is thus unit lower triangular, with the multipliers from Gaussian Elimination stored below the diagonal).

Given a unit lower triangular matrix $L \in \mathbb{R}^{n \times n}$ and vectors $z, b \in \mathbb{R}^n$, consider the equation $Lz = b$ where L and b are known and z is to be computed. Partition

$$L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), \quad z \rightarrow \left(\begin{array}{c} \zeta_1 \\ \hline z_2 \end{array}\right), \quad \text{and} \quad b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right).$$

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR1}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

where λ_{11} is 1×1 , β_1 has 1 row

$b_2 := b_2 - \beta_1 l_{21}$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 6.6: Algorithm for solving $Lx = b$, overwriting b with the result vector x. Here L is a lower triangular matrix.

(Recall: the horizontal line here partitions the result. It is *not* a division.) Now, $Lz = b$ implies that

$$\overbrace{\left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)}^{b} = \overbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)}^{L} \overbrace{\left(\begin{array}{c} \zeta_1 \\ \hline z_2 \end{array}\right)}^{z}$$

$$= \overbrace{\left(\begin{array}{c} 1 \times \zeta_1 + 0 \times z_2 \\ \hline l_{21}\zeta_1 + L_{22}z_2 \end{array}\right)}^{Lz} = \overbrace{\left(\begin{array}{c} \zeta_1 \\ \hline l_{21}\zeta_1 + L_{22}z_2 \end{array}\right)}^{Lz}$$

so that

$$\begin{array}{c} \beta_1 = \zeta_1 \\ \hline b_2 = l_{21}\zeta_1 + L_{22}z_2 \end{array} \quad \text{or, equivalently,} \quad \begin{array}{c} \zeta_1 = \beta_1 \\ \hline L_{22}z_2 = b_2 - \zeta_1 l_{21} \end{array}.$$

This suggests the following steps for **overwriting** the vector b with the solution vector z:

- Partition

$$L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right) \quad \text{and} \quad b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

- Update $b_2 = b_2 - \beta_1 l_{21}$ (this is an AXPY operation!).

- Continue with $L = L_{22}$ and $b = b_2$.

This motivates the algorithm in Figure 6.15. If you compare this algorithm to FORWARD_SUBSTITUTION in Unit 6.2.5, you find them to be the same algorithm, except that matrix A has now become matrix L! So, solving $Lz = b$, overwriting b with z, **is** forward substitution when L is the unit lower triangular matrix that results from LU factorization.

We illustrate solving $Lz = b$ in Figure 6.8. Compare this to forward substitution with multipliers stored below the diagonal after Gaussian elimination, in Figure **??**.

Homework 6.3.2.1 Implement the algorithm in Figure 6.15.

- `[ b_out ] = Ltrsv_unb_var1( L, b )`

You can check that they compute the right answers with the following script:

- `test_Ltrsv_unb_var1.m`

This script exercises the function by setting the matrix

```
L = [
     1     0     0     0
    -1     1     0     0
     2     1     1     0
    -2    -1     1     1
]
```

and solving $Lx = b$ with the right-hand size vector

```
b = [
     2
     2
    11
    -3
]
```

by calling

```
x = Ltrsv_unb_var1( L, b )
```

Finally, it checks if `x` is indeed the answer by checking if

```
b - L * x
```

equals the zero vector.

```
x = U \ z
```

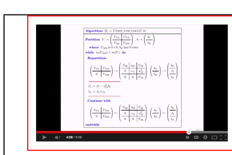
We can the check if this solves $Ax = b$ by computing

```
b - A * x
```

which should yield a zero vector.

☛ SEE ANSWER

6.3.3 Solving $Ux = b$ (Back substitution)



☛ YouTube

Next, let us consider how to solve a linear system $Ux = b$. We will conclude that the algorithm we come up with is the same as backward substitution.

Step	$\left(\begin{array}{c\|c\|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right)$	$\left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$	$b_2 - l_{21}\beta_1$
1-2	$\left(\begin{array}{\|c\|cc} 1 & 0 & 0 \\ \hline -1 & 1 & 0 \\ 2 & -2 & 1 \end{array}\right)$	$\left(\begin{array}{c} 6 \\ \hline 3 \\ -3 \end{array}\right)$	$\begin{pmatrix} 3 \\ -3 \end{pmatrix} - \begin{pmatrix} -1 \\ 2 \end{pmatrix}(6) = \begin{pmatrix} 9 \\ -15 \end{pmatrix}$
3	$\left(\begin{array}{c\|c\|c} 1 & 0 & 0 \\ \hline -1 & 1 & 0 \\ \hline 2 & -2 & 1 \end{array}\right)$	$\left(\begin{array}{c} 6 \\ \hline 9 \\ \hline -15 \end{array}\right)$	$\begin{pmatrix} -15 \end{pmatrix} - \begin{pmatrix} -2 \end{pmatrix}(9) = (3)$
	$\left(\begin{array}{cc\|c} 1 & 0 & 0 \\ -1 & 1 & 0 \\ \hline 2 & -2 & 1 \end{array}\right)$	$\left(\begin{array}{c} 6 \\ 9 \\ \hline 3 \end{array}\right)$	

Figure 6.7: Solving $Lz = b$ where L is a unit lower triangular matrix. Vector z overwrites vector b.

Step	Stored multipliers and right-hand side	Operation
1	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 3 \\ 2 & -2 & - & -3 \end{array}\right)$	$\begin{array}{r} 3 \\ -(-1)\times(\;6) \\ \hline 9 \end{array}$
2	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 9 \\ 2 & -2 & - & -3 \end{array}\right)$	$\begin{array}{r} -3 \\ -(2)\times(\;6) \\ \hline -15 \end{array}$
3	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 9 \\ 2 & -2 & - & -15 \end{array}\right)$	$\begin{array}{r} -15 \\ -(-2)\times(\;9) \\ \hline 3 \end{array}$
4	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 9 \\ 2 & -2 & - & 3 \end{array}\right)$	

Figure 6.8: Forward substitutions with multipliers stored below the diagonal (e.g., as output from Gaussian Elimination).

Given upper triangular matrix $U \in \mathbb{R}^{n\times n}$ and vectors $x, b \in \mathbb{R}^n$, consider the equation $Ux = b$ where U and b are known and x is to be computed. Partition

$$U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right), \quad x \rightarrow \left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right) \quad \text{and} \quad b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right).$$

Now, $Ux = b$ implies

$$\begin{aligned} \overbrace{\left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)}^{b} &= \overbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)}^{U} \overbrace{\left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right)}^{x} \\ &= \overbrace{\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 \\ \hline 0 \times \chi_1 + U_{22}x_2 \end{array}\right)}^{Ux} = \overbrace{\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 \\ \hline U_{22}x_2 \end{array}\right)}^{Ux} \end{aligned}$$

so that

$$\begin{array}{c} \beta_1 = \upsilon_{11}\chi_1 + u_{12}^T x_2 \\ \hline b_2 = U_{22}x_2 \end{array} \quad \text{or, equivalently,} \quad \begin{array}{c} \chi_1 = (\beta_1 - u_{12}^T x_2)/\upsilon_{11} \\ \hline U_{22}x_2 = b_2 \end{array}.$$

This suggests the following steps for overwriting the vector b with the solution vector x:

- Partition
$$U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right), \quad \text{and} \quad b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$
- Solve $U_{22}x_2 = b_2$ for x_2, overwriting b_2 with the result.
- Update $\beta_1 = (\beta_1 - u_{12}^T b_2)/\upsilon_{11} (= (\beta_1 - u_{12}^T x_2)/\upsilon_{11})$.
  (This requires a dot product followed by a scaling of the result by $1/\upsilon_{11}$.)

This suggests the following algorithm: Notice that the algorithm does not have "Solve $U_{22}x_2 = b_2$" as an update. The reason is that the algorithm marches through the matrix from the bottom-right to the top-left and through the vector from bottom to top. Thus, for a given iteration of the while loop, all elements in x_2 have already been computed and have overwritten b_2. Thus, the "Solve $U_{22}x_2 = b_2$" has already been accomplished by prior iterations of the loop. As a result, in this iteration, only β_1 needs to be updated with χ_1 via the indicated computations.

Homework 6.3.3.1 Side-by-side, solve the upper triangular linear system

$$\begin{aligned} -2\chi_0 - \chi_1 + \chi_2 &= 6 \\ -3\chi_1 - 2\chi_2 &= 9 \\ \chi_2 &= 3 \end{aligned}$$

via back substitution and by executing the above algorithm with

$$U = \begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 6 \\ 9 \\ 3 \end{pmatrix}.$$

Compare and contrast!

☛ SEE ANSWER

Algorithm: $[b] := \text{UTRSV\_UNB\_VAR1}(U, b)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

  where U_{BR} is 0×0, b_B has 0 rows

while $m(U_{BR}) < m(U)$ **do**

  Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$\beta_1 := \beta_1 - u_{12}^T b_2$

$\beta_1 := \beta_1 / \upsilon_{11}$

  Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 6.9: Algorithm for solving $Ux = b$ where U is an uppertriangular matrix. Vector b is overwritten with the result vector x.

Homework 6.3.3.2 Implement the algorithm in Figure 6.16.

- `[ b_out ] = Utrsv_unb_var1( U, b )`

You can check that it computes the right answer with the following script:

- `test_Utrsv_unb_var1.m`

This script exercises the function by starting with matrix

```
U = [
      2     0     1     2
      0    -1     2     1
      0     0     1    -1
      0     0     0    -2
]
```

Next, it solves $Ux = b$ with the right-hand size vector

```
b = [
      2
      4
      3
      2
]
```

by calling

```
x = Utrsv_unb_var1( U, b )
```

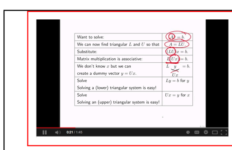
Finally, it checks if `x` indeed solves $Ux = b$ by computing

```
b - U * x
```

which should yield a zero vector of size four.

☛ SEE ANSWER

6.3.4 Putting it all together to solve $Ax = b$



☛ YouTube

Now, the week started with the observation that we would like to solve linear systems. These could then be written more concisely as $Ax = b$, where $n \times n$ matrix A and vector b of size n are given, and we would like to solve for x, which is the vectors of unknowns. We now have algorithms for

- Factoring A into the product LU where L is unit lower triangular;
- Solving $Lz = b$; and
- Solving $Ux = b$.

We now discuss how these algorithms (and functions that implement them) can be used to solve $Ax = b$.

Start with

$$Ax = b.$$

If we have L and U so that $A = LU$, then we can replace A with LU:

$$\underbrace{(LU)}_{A} x = b.$$

Now, we can treat matrices and vectors alike as matrices, and invoke the fact that matrix-matrix multiplication is associative to place some convenient parentheses:

$$L(Ux) = b.$$

We can then recognize that Ux is a vector, which we can call z:

$$L \underbrace{(Ux)}_{z} = b$$

so that

$$Lz = b \quad \text{and} \quad Ux = z.$$

Thus, the following steps will solve $Ax = b$:

- Factor A into L and U so that $A = LU$ (LU factorization).

- Solve $Lz = b$ for z (forward substitution).

- Solve $Ux = z$ for x (back substitution).

This works if A has the right properties for the LU factorization to exist, which is what we will discuss next week...

Homework 6.3.4.1 Implement the function

- `[ A_out, b_out ] = Solve( A, b )`

that

- Computes the LU factorization of matrix A, $A = LU$, overwriting the upper triangular part of A with U and the strictly lower triangular part of A with the strictly lower triangular part of L. The result is then returned in variable `A_out`.
- Uses the factored matrix to solve $Ax = b$.

Use the routines you wrote in the previous subsections (6.3.1-6.3.3).
You can check that it computes the right answer with the following script:

- `test_Solve.m`

This script exercises the function by starting with matrix

```
A = [
     2    0    1    2
    -2   -1    1   -1
     4   -1    5    4
    -4    1   -3   -8
]
```

Next, it solves $Ax = b$ with

```
b = [
     2
     2
    11
    -3
]
```

by calling

```
x = Solve( A, b )
```

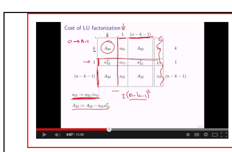
Finally, it checks if x indeed solves $Ax = b$ by computing

```
b - A * x
```

which should yield a zero vector of size four.

☛ SEE ANSWER

6.3.5 Cost



☛ YouTube

LU factorization

Let's look at how many floating point computations are needed to compute the LU factorization of an $n \times n$ matrix A. Let's focus on the algorithm:

Assume that during the kth iteration A_{TL} is $k \times k$. Then

- A_{00} is a $k \times k$ matrix.

Algorithm: $A := \text{LU\_UNB\_VAR5}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

$a_{21} := a_{21}/\alpha_{11} \qquad (= l_{21})$

$A_{22} := A_{22} - a_{21}a_{12}^T \qquad (= A_{22} - l_{21}a_{12}^T)$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 6.10: LU factorization algorithm.

- a_{21} is a column vector of size $n-k-1$.
- a_{12}^T is a row vector of size $n-k-1$.
- A_{22} is a $(n-k-1) \times (n-k-1)$ matrix.

Now,

- a_{21}/α_{11} is typically implemented as $(1/\alpha_{11}) \times a_{21}$ so that only one division is performed (divisions are EXPENSIVE) and $(n-k-1)$ multiplications are performed.
- $A_{22} := A_{22} - a_{21}a_{12}^T$ is a rank-1 update. In a rank-1 update, for each element in the matrix one multiply and one add (well, subtract in this case) is performed, for a total of $2(n-k-1)^2$ floating point operations.

Now, we need to sum this over all iterations $k = 0, \ldots, (n-1)$:

$$\sum_{k=0}^{n-1} \left((n-k-1) + 2(n-k-1)^2\right) \text{ floating point operations.}$$

Here we ignore the divisions. Clearly, there will only be n of those (one per iteration of the algorithm).

Let us compute how many flops this equals.

$\sum_{k=0}^{n-1}\left((n-k-1)+2(n-k-1)^2\right)$
$\quad = \quad <$ Change of variable: $p=n-k-1$ so that $p=0$ when $k=n-1$ and $p=n-1$ when $k=0$ $>$
$\sum_{p=n-1}^{0}\left(p+2p^2\right)$
$\quad = \quad <$ Sum in reverse order $>$
$\sum_{p=0}^{n-1}\left(p+2p^2\right)$
$\quad = \quad <$ Split into two sums $>$

$\sum_{p=0}^{n-1} p+\sum_{p=0}^{n-1}(2p^2)$
$\quad = \quad <$ Results from Week 2! $>$
$\frac{(n-1)n}{2}+2\frac{(n-1)n(2n-1)}{6}$
$\quad = \quad <$ Algebra $>$
$\frac{3(n-1)n}{6}+2\frac{(n-1)n(2n-1)}{6}$
$\quad = \quad <$ Algebra $>$
$\frac{(n-1)n(4n+1)}{6}$

Now, when n is large $n-1$ and $4n+1$ equal, approximately, n and $4n$, respectively, so that the cost of LU factorization equals, approximately,

$$\frac{2}{3}n^3 \text{ flops.}$$

Forward substitution

Next, let us look at how many flops are needed to solve $Lx=b$. Again, focus on the algorithm: Assume that during the kth iteration L_{TL} is $k\times k$. Then

- L_{00} is a $k\times k$ matrix.
- l_{21} is a column vector of size $n-k-1$.
- b_2 is a column vector of size $n-k-1$.

Now,

- The `axpy` operation $b_2 := b_2 - \beta_1 l_{21}$ requires $2(n-k-1)$ flops since the vectors are of size $n-k-1$.

We need to sum this over all iterations $k=0,\ldots,(n-1)$:

$$\sum_{k=0}^{n-1}(n-k-1) \text{ flops.}$$

Let us compute how many flops this equals:

$\sum_{k=0}^{n-1} 2(n-k-1)$
$\quad = \quad <$ Factor out 2 $>$
$2\sum_{k=0}^{n-1}(n-k-1)$
$\quad = \quad <$ Change of variable: $p=n-k-1$ so that $p=0$ when $k=n-1$ and $p=n-1$ when $k=0$ $>$
$2\sum_{p=n-1}^{0} 2p$
$\quad = \quad <$ Sum in reverse order $>$

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR1}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

  where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

  Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

  where λ_{11} is 1×1 , β_1 has 1 row

$b_2 := b_2 - \beta_1 l_{21}$

  Continue with

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 6.11: Algorithm for solving $Lx = b$, overwriting b with the result vector x. Here L is a lower triangular matrix.

$$\begin{array}{l} 2\sum_{p=0}^{n-1} p \\ \quad = \quad < \text{Results from Week 2!} > \\ 2\frac{(n-1)n}{2} \\ \quad = \quad < \text{Algebra} > \\ (n-1)n. \end{array}$$

Now, when n is large $n-1$ equals, approximately, n so that the cost for the forward substitution equals, approximately,

$$n^2 \text{ flops.}$$

Back substitution

Finally, let us look at how many flops are needed to solve $Ux = b$. Focus on the algorithm:

Algorithm: $[b] := \text{UTRSV\_UNB\_VAR1}(U, b)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

  where U_{BR} is 0×0, b_B has 0 rows

while $m(U_{BR}) < m(U)$ **do**

  Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$\beta_1 := \beta_1 - u_{12}^T b_2$

$\beta_1 := \beta_1 / \upsilon_{11}$

  Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 6.12: Algorithm for solving $Ux = b$ where U is an uppertriangular matrix. Vector b is overwritten with the result vector x.

Homework 6.3.5.1 Assume that during the kth iteration U_{BR} is $k \times k$. (Notice we are purposely saying that U_{BR} is $k \times k$ because this algorithm moves in the opposite direction!)
Then answer the following questions:

- U_{22} is a ?×? matrix.
- u_{12}^T is a column/row vector of size ????.
- b_2 is a column vector of size ???.

Now,

- The `axpy`/`dot` operation $\beta_1 := \beta_1 - u_{12}^T b_2$ requires ??? flops since the vectors are of size ????.

We need to sum this over all iterations $k = 0, \ldots, (n-1)$ (You may ignore the divisions):

$$????? \text{ flops.}$$

Compute how many floating point operations this equal. Then, approximate the result.

☛ SEE ANSWER

Total cost

The total cost of first factoring A and then performing forward and back substitution is, approximately,

$$\frac{2}{3}n^3 + n^2 + n^2 = \frac{2}{3}n^3 + 2n^2 \text{ flops.}$$

When n is large n^2 is very small relative to n^3 and hence the total cost is typically given as

$$\frac{2}{3}n^3 \text{ flops.}$$

Notice that this explains why we prefer to do the LU factorization separate from the forward and back substitutions. If we solve $Ax = b$ via these three steps, and afterwards a new right-hand side b comes along with which we wish to solve, then we need not refactor A since we already have L and U (overwritten in A). But it is the factorization of A where most of the expense is, so solving with this new right-hand side is almost free.

6.4 Enrichment

6.4.1 Blocked LU Factorization

What you saw in Week 5, Units 5.4.1 and 5.4.2, was that by carefully implementing matrix-matrix multiplication, the performance of this operation could be improved from a few percent of the peak of a processor to better than 90%. This came at a price: clearly the implementation was not nearly as “clean” and easy to understand as the routines that you wrote so far in this course.

Imagine implementing all the operations you have encountered so far in this way. When a new architecture comes along, you will have to reimplement to optimize for that architecture. While this guarantees job security for those with the skill and patience to do this, it quickly becomes a distraction from more important work.

So, how to get around this problem? Widely used linear algebra libraries like LAPACK (written in Fortran)

> E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, Jack J. Dongarra, J. Du Croz, S. Hammarling, A. Greenbaum, A. McKenney, and D. Sorensen.
> *LAPACK Users' guide (third ed.).*
> SIAM, 1999.

and the `libflame` library (developed as part of our FLAME project and written in C)

F. G. Van Zee, E. Chan, R. A. van de Geijn, E. S. Quintana-Orti, G. Quintana-Orti.
The libflame Library for Dense Matrix Computations.
IEEE Computing in Science and Engineering, Vol. 11, No 6, 2009.

F. G. Van Zee.
libflame: The Complete Reference.
www.lulu.com , 2009

implement many linear algebra operations so that most computation is performed by a call to a matrix-matrix multiplication routine. The `libflame` library is coded with an API that is very similar to the FLAME@lab API that you have been using for your routines.

More generally, in the scientific computing community there is a set of operations with a standardized interface known as the Basic Linear Algebra Subprograms (BLAS) in terms of which applications are written.

C. L. Lawson, R. J. Hanson, D. R. Kincaid, F. T. Krogh.
Basic Linear Algebra Subprograms for Fortran Usage.
ACM Transactions on Mathematical Software, 1979.

J. J. Dongarra, J. Du Croz, S. Hammarling, R. J. Hanson.
An Extended Set of FORTRAN Basic Linear Algebra Subprograms.
ACM Transactions on Mathematical Software, 1988.

J. J. Dongarra, J. Du Croz, S. Hammarling, I. Duff.
A Set of Level 3 Basic Linear Algebra Subprograms.
ACM Transactions on Mathematical Software, 1990.

F. G. Van Zee, R. A. van de Geijn.
BLIS: A Framework for Rapid Instantiation of BLAS Functionality.
ACM Transactions on Mathematical Software, to appear.

It is then expected that *someone* optimizes these routines. When they are highly optimized, any applications and libraries written in terms of these routines also achieve high performance.

In this enrichment, we show how to cast LU factorization so that most computation is performed by a matrix-matrix multiplication. Algorithms that do this are called *blocked* algorithms.

Blocked LU factorization

It is difficult to describe how to attain a blocked LU factorization algorithm by starting with Gaussian elimination as we did in Section 6.2, but it is easy to do so by starting with $A = LU$ and following the techniques exposed in Unit 6.3.1.

We start again by assuming that matrix $A \in \mathbb{R}^{n\times n}$ can be factored into the product of two matrices $L,U \in \mathbb{R}^{n\times n}$, $A = LU$, where L is unit lower triangular and U is upper triangular. Matrix $A \in \mathbb{R}^{n\times n}$ is given and that L and U are to be computed such that $A = LU$, where $L \in \mathbb{R}^{n\times n}$ is unit lower triangular and $U \in \mathbb{R}^{n\times n}$ is upper triangular.

We derive a blocked algorithm for computing this operation by partitioning

$$A \rightarrow \left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right), \quad L \rightarrow \left(\begin{array}{c|c} L_{11} & 0 \\ \hline L_{21} & L_{22} \end{array}\right), \quad \text{and} \quad U \rightarrow \left(\begin{array}{c|c} U_{11} & U_{12} \\ \hline 0 & U_{22} \end{array}\right),$$

where $A_{11}, L_{11}, U_{11} \in \mathbb{R}^{b\times b}$. The integer b is the block size for the algorithm. In Unit 6.3.1, $b = 1$ so that $A_{11} = \alpha_{11}$, $L_{11} = 1$, and so forth. Here, we typically choose $b > 1$ so that L_{11} is a unit lower triangular matrix and U_{11} is an upper triangular matrix. How to choose b is closely related to how to optimize matrix-matrix multiplication (Units 5.4.1 and 5.4.2).

Now, $A = LU$ implies (using what we learned about multiplying matrices that have been partitioned into submatrices)

$$\overbrace{\left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right)}^{A} = \overbrace{\left(\begin{array}{c|c} L_{11} & 0 \\ \hline L_{21} & L_{22} \end{array}\right)}^{L} \overbrace{\left(\begin{array}{c|c} U_{11} & U_{12} \\ \hline 0 & U_{22} \end{array}\right)}^{U}$$

$$= \overbrace{\left(\begin{array}{c|c} L_{11} \times U_{11} + 0 \times 0 & L_{11} \times U_{12} + 0 \times U_{22} \\ \hline L_{21} \times U_{11} + L_{22} \times 0 & L_{21} \times U_{12} + L_{22} \times U_{22} \end{array}\right)}^{LU}.$$

$$= \overbrace{\left(\begin{array}{c|c} L_{11}U_{11} & L_{11}U_{12} \\ \hline L_{21}U_{11} & L_{21}U_{12} + L_{22}U_{22} \end{array}\right)}^{LU}.$$

For two matrices to be equal, their elements must be equal and therefore, if they are partitioned conformally, their submatrices must be equal:

$$\begin{array}{c|c} A_{11} = L_{11}U_{11} & A_{12} = L_{11}U_{12} \\ \hline A_{21} = L_{21}U_{11} & A_{22} = L_{21}U_{12} + L_{22}U_{22} \end{array}$$

or, rearranging,

$$\begin{array}{c|c} A_{11} = L_{11}U_{11} & A_{12} = L_{11}U_{12} \\ \hline A_{21} = L_{21}U_{11} & A_{22} - L_{21}U_{12} = L_{22}U_{22} \end{array}$$

This suggests the following steps for **overwriting** a matrix A with its LU factorization:

- Partition

$$A \rightarrow \left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right).$$

- Compute the LU factorization of A_{11}: $A_{11} \rightarrow L_{11}U_{11}$. Overwrite A_{11} with this factorization.
  Note: one can use an unblocked algorithm for this.

- Now that we know L_{11}, we can solve $L_{11}U_{12} = A_{12}$, where L_{11} and A_{12} are given. U_{12} overwrites A_{12}.
  This is known as an triangular solve with multiple right-hand sides. More on this later in this unit.

- Now that we know U_{11} (still from the first step), we can solve $L_{21}U_{11} = A_{21}$, where U_{11} and A_{21} are given. L_{21} overwrites A_{21}.
  This is also known as a triangular solve with multiple right-hand sides. More on this also later in this unit.

- Update $A_{22} = A_{22} - A_{21}A_{12} (= A_{22} - L_{21}U_{12})$.
  This is a matrix-matrix multiplication and is where, if b is small relative to n, most of the computation is performed.

- Overwrite A_{22} with L_{22} and U_{22} by repeating the above steps with $A = A_{22}$.

This will leave U in the upper triangular part of A and the strictly lower triangular part of L in the strictly lower triangular part of A. The diagonal elements of L need not be stored, since they are known to equal one.

The above is summarized in Figure 6.13. In that figure, the derivation of the unblocked algorithm from Unit 6.3.1 is given on the left and the above derivation of the blocked algorithm is given on the right, for easy comparing and contrasting. The resulting algorithms, in FLAME notation, are given as well. It is important to note that the algorithm now progresses b rows and b columns at a time, since A_{11} is a $b \times b$ block.

Triangular solve with multiple right-hand sides

In the above algorithm, we needed to perform two subproblems:

- $L_{11}U_{12} = A_{12}$ where unit lower triangular matrix L_{11} and (general) matrix A_{12} are known and (general) matrix U_{12} is to be computed; and

- $L_{21}U_{11} = A_{21}$ where upper triangular matrix U_{11} and (general) matrix A_{21} are known and (general) matrix L_{21} is to be computed.

<table>
<tr><th>Unblocked algorithm</th><th>Blocked algorithm</th></tr>
<tr><td>$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right), L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)$</td><td>$A \rightarrow \left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right), L \rightarrow \left(\begin{array}{c|c} L_{11} & 0 \\ \hline L_{21} & L_{22} \end{array}\right), U \rightarrow \left(\begin{array}{c|c} U_{11} & U_{12} \\ \hline 0 & U_{22} \end{array}\right)$</td></tr>
<tr><td>$\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \underbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)}_{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline l_{21}\upsilon_{11} & l_{21}u_{12}^T + L_{22}U_{22} \end{array}\right)}$</td><td>$\left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right) = \underbrace{\left(\begin{array}{c|c} L_{11} & 0 \\ \hline L_{21} & L_{22} \end{array}\right)\left(\begin{array}{c|c} U_{11} & U_{12} \\ \hline 0 & U_{22} \end{array}\right)}_{\left(\begin{array}{c|c} L_{11}U_{11} & L_{11}U_{12} \\ \hline L_{21}U_{11} & L_{21}U_{12}^T + L_{22}U_{22} \end{array}\right)}$</td></tr>
<tr><td>$\begin{array}{c|c} \alpha_{11} = \upsilon_{11} & a_{12}^T = u_{12}^T \\ \hline a_{21} = l_{21}\upsilon_{11} & A_{22} = l_{21}u_{12}^T + L_{22}U_{22} \end{array}$</td><td>$\begin{array}{c|c} A_{11} = L_{11}U_{11} & A_{12} = L_{11}U_{12} \\ \hline A_{21} = L_{21}U_{11} & A_{22} = L_{21}U_{12} + L_{22}U_{22} \end{array}$</td></tr>
<tr><td>$\alpha_{11} := \alpha_{11}$
$a_{12}^T := a_{12}^T$
$a_{21} := a_{21}/\alpha_{11}$
$A_{22} := A_{22} - a_{21}a_{12}^T$</td><td>$A_{11} \rightarrow L_{11}U_{11}$ (overwriting A_{11} with L_{11} and U_{11})
Solve $L_{11}U_{12} := A_{12}$ (overwriting A_{12} with U_{12})
Solve $L_{21}U_{11} := A_{21}$ (overwriting A_{21} with L_{21})
$A_{22} := A_{22} - A_{21}A_{12}$</td></tr>
<tr><td>Algorithm: $[A] := \text{LU\_UNB\_VAR5}(A)$
Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
  where A_{TL} is 0×0
while $m(A_{TL}) < m(A)$ do
  Repartition
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$

$a_{21} := a_{21}/\alpha_{11}$
$A_{22} := A_{22} - a_{21}a_{12}^T$

  Continue with
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$
endwhile</td><td>Algorithm: $[A] := \text{LU\_BLK\_VAR5}(A)$
Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
  where A_{TL} is 0×0
while $m(A_{TL}) < m(A)$ do
  Repartition
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right)$

Factor $A_{11} \rightarrow L_{11}U_{11}$ (Overwrite A_{11})
Solve $L_{11}U_{12} = A_{12}$ (Overwrite A_{12})
Solve $L_{21}U_{11} = A_{21}$ (Overwrite A_{21})
$A_{22} := A_{22} - A_{21}A_{12}$

  Continue with
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right)$
endwhile</td></tr>
</table>

Figure 6.13: Side-by-side derivation of the unblocked and blocked algorithms.

These operations are known as special cases of the “triangular solve with multiple right-hand side” operation.

Let’s simplify the discussion to solving $LX = B$ where L is unit lower triangular and X and B are general matrices. Here L and B are known and X is to be computed. We slice and dice B and X into columns to observe that

$$L \left(\begin{array}{c|c|c} x_0 & x_1 & \cdots \end{array} \right) = \left(\begin{array}{c|c|c} b_0 & b_1 & \cdots \end{array} \right)$$

and hence

$$\left(\begin{array}{c|c|c} Lx_0 & Lx_1 & \cdots \end{array} \right) = \left(\begin{array}{c|c|c} b_0 & b_1 & \cdots \end{array} \right).$$

We therefore conclude that $Lx_j = b_j$ for all pairs of columns x_j and b_j. But that means that to compute x_j from L and b_j we need to solve with a unit lower triangular matrix L. Thus the name “triangular solve with multiple right-hand sides”. The multiple right-hand sides are the columns b_j.

Now let us consider solving $XU = B$, where U is upper triangular. If we transpose both sides, we get that $(XU)^T = B^T$ or $U^T X^T = B^T$. If we partition X and B by columns so that

$$X = \left(\begin{array}{c} \widetilde{x}_0^T \\ \hline \widetilde{x}_1^T \\ \hline \vdots \end{array} \right) \quad \text{and} \quad B = \left(\begin{array}{c} \widetilde{b}_0^T \\ \hline \widetilde{b}_1^T \\ \hline \vdots \end{array} \right),$$

then

$$U^T \left(\begin{array}{c|c|c} \widetilde{x}_0 & \widetilde{x}_1 & \cdots \end{array} \right) = \left(\begin{array}{c|c|c} U^T\widetilde{x}_0 & U^T\widetilde{x}_1 & \cdots \end{array} \right) = \left(\begin{array}{c|c|c} \widetilde{b}_0 & \widetilde{b}_1 & \cdots \end{array} \right).$$

We notice that this, again, is a matter of solving multiple right-hand sides (now rows of B that have been transposed) with a lower triangular matrix (U^T). In practice, none of the matrices are transposed.

Cost analysis

Let us analyze where computation is spent in just the first step of a blocked LU factorization. We will assume that A is $n \times n$ and that a block size of b is used:

- Partition
$$A \rightarrow \left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array} \right).$$
This carries no substantial cost, since it just partitions the matrix.

- Compute the LU factorization of A_{11}: $A_{11} \rightarrow L_{11}U_{11}$. Overwrite A_{11} with this factorization.
One can use an unblocked algorithm for this and we saw that the cost of that algorithm, for a $b \times b$ matrix, is approximately $\frac{2}{3}b^3$.

- Now that we know L_{11}, we can solve $L_{11}U_{12} = A_{12}$, where L_{11} and A_{12} are given. U_{12} overwrites A_{12}.
This is a triangular solve with multiple right-hand sides with a matrix A_{12} that is $b \times (n-b)$. Now, each triangular solve with each column of A_{12} costs, approximately, b^2 flops for a total of $b^2(n-b)$ flops.

- Now that we know U_{11}, we can solve $L_{21}U_{11} = A_{21}$, where U_{11} and A_{21} are given. L_{21} overwrites A_{21}.
This is a triangular solve with multiple right-hand sides with a matrix A_{21} that is $(n-b) \times b$. Now, each triangular solve with each row of A_{21} costs, approximately, b^2 flops for a total of $b^2(n-b)$ flops.

- Update $A_{22} = A_{22} - A_{21}A_{12} (= A_{22} - L_{21}U_{12})$.
This is a matrix-matrix multiplication that multiplies $(n-b) \times b$ matrix A_{21} times $b \times (n-b)$ matrix A_{12} to update A_{22}. This requires $b(n-b)^2$ flops.

- Overwrite A_{22} with L_{22} and U_{22} by repeating with $A = A_{22}$, in future iterations. We don’t count that here, since we said we were only going to analyze the first iteration of the blocked LU factorization.

Now, if n is much larger than b, $\frac{2}{3}b^3$ is small compared to $b^2(n-b)$ which is itself small relative to $2b(n-b)^2$. Thus, if n is much larger than b, most computational time is spent in the matrix-matrix multiplication $A_{22} := A_{22} - A_{21}A_{12}$. Since we saw in the enrichment of Week 5 that such a matrix-matrix multiplication can achieve extremely high performance, the blocked LU factorization can achieve extremely high performance (if n is large).

It is important to note that the blocked LU factorization algorithm executes exactly the same number of floating point operations as does the unblocked algorithm. It just does so in a different order so that matrix-matrix multiplication can be utilized.

More

A large number of algorithms, both unblocked and blocked, that are expressed with our FLAME notation can be found in the following technical report:

> P. Bientinesi and R. van de Geijn.
> Representing Dense Linear Algebra Algorithms: A Farewell to Indices.
> FLAME Working Note #17. The University of Texas at Austin, Department of Computer Sciences. Technical Report TR-2006-10, 2006.

It is available from the FLAME Publications webpage.

6.4.2 How Ordinary Elimination Became Gaussian Elimination

Read

> Joseph F. Grcar.
> How Ordinary Elimination Became Gaussian Elimination.

Cite as

> Joseph F. Grcar.
> How ordinary elimination became Gaussian elimination.
> Historia Math, 2011.

6.4.3 Formal Derivation of LU factorization

Robert used to teach an introductory graduate level course on Numerical Linear Algebra. One of the lectures, about half way into the course, discusses how to derive LU factorization algorithms in a goal-oriented fashion that would have (perhaps) made Dijkstra proud. You may want to revisit Unit 2.5.1 (in particular, the paper ”The Science of Deriving Dense Linear Algebra Algorithms”) and then watch the below video. The discussion on derivation starts around minute 9 or 10.

(Robert simply set up a camera in his classroom, so the quality is pretty low. This also convinced us that making people watch hour long videos is probably hard on people’s attention span...)

For more info on this class (and the notes), visit www.ulaff.net (bottom of the page).

6.5 Wrap Up

6.5.1 Homework

There is no additional graded homework. However, we have an additional version of the ”Gaussian Elimination” webpage:

- Practice with four equations in four unknowns.

Now, we always joke that in a standard course on matrix computations the class is asked to solve systems with three equations with pencil and paper. What defines an honor version of the course is that the class is asked to solve systems with four equations with pencil and paper...

Of course, there is little insight gained from the considerable extra work. However, here we have webpages that automate most of the rote work, and hence it IS worthwhile to at least observe how the methodology extends to larger systems. DO NOT DO THE WORK BY HAND. Let the webpage do the work and focus on the insights that you can gain from this.

6.5.2 Summary

Linear systems of equations

A linear system of equations with m equations in n unknowns is given by

$$\begin{array}{ccccccccc}
\alpha_{0,0}\chi_0 & + & \alpha_{0,1}\chi_1 & + & \cdots & + & \alpha_{0,n-1}\chi_{n-1} & = & \beta_0 \\
\alpha_{1,0}\chi_0 & + & \alpha_{1,1}\chi_1 & + & \cdots & + & \alpha_{1,n-1}\chi_{n-1} & = & \beta_1 \\
\vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\
\alpha_{m-1,0}\chi_0 & + & \alpha_{m-1,1}\chi_1 & + & \cdots & + & \alpha_{m-1,n-1}\chi_{n-1} & = & \beta_{m-1}
\end{array}$$

Variables $\chi_0, \chi_1, \ldots, \chi_{n-1}$ are the unknowns.

This Week, we only considered the case where $m = n$:

$$\begin{array}{ccccccccc}
\alpha_{0,0}\chi_0 & + & \alpha_{0,1}\chi_1 & + & \cdots & + & \alpha_{0,n-1}\chi_{n-1} & = & \beta_0 \\
\alpha_{1,0}\chi_0 & + & \alpha_{1,1}\chi_1 & + & \cdots & + & \alpha_{1,n-1}\chi_{n-1} & = & \beta_1 \\
\vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\
\alpha_{n-1,0}\chi_0 & + & \alpha_{n-1,1}\chi_1 & + & \cdots & + & \alpha_{n-1,n-1}\chi_{n-1} & = & \beta_{n-1}
\end{array}$$

Here the $\alpha_{i,j}$s are the coefficients in the linear system. The β_is are the right-hand side values.

Basic tools

Solving the above linear system relies on the fact that its solution does not change if

1. Equations are reordered (not used until next week);
2. An equation in the system is modified by subtracting a multiple of another equation in the system from it; and/or
3. Both sides of an equation in the system are scaled by a nonzero.

Gaussian elimination is a method for solving systems of linear equations that employs these three basic rules in an effort to reduce the system to an upper triangular system, which is easier to solve.

Appended matrices

The above system of n linear equations in n unknowns is more concisely represented as an appended matrix:

$$\left(\begin{array}{cccc|c}
\alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} & \beta_0 \\
\alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} & \beta_1 \\
\vdots & \vdots & & \vdots & \vdots \\
\alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-1} & \beta_{n-1}
\end{array}\right)$$

This representation allows one to just work with the coefficients and right-hand side values of the system.

Matrix-vector notation

The linear system can also be represented as $Ax = b$ where

$$A = \left(\begin{array}{cccc}
\alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\
\alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\
\vdots & \vdots & & \vdots \\
\alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-1}
\end{array}\right), \quad x = \left(\begin{array}{c} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{array}\right), \quad \text{and} \quad \left(\begin{array}{c} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{n-1} \end{array}\right).$$

Here, A is the matrix of coefficients from the linear system, x is the solution vector, and b is the right-hand side vector.

Gauss transforms

A Gauss transform is a matrix of the form

$$L_j = \begin{pmatrix} I_j & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+1,j} & 1 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+2,j} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -\lambda_{n-1,j} & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

When applied to a matrix (or a vector, which we think of as a special case of a matrix), it subtracts $\lambda_{i,j}$ times the jth row from the ith row, for $i = j+1,\ldots,n-1$. Gauss transforms can be used to express the operations that are inherently performed as part of Gaussian elimination with an appended system of equations.

The action of a Gauss transform on a matrix, $A := L_jA$ can be described as follows:

$$\begin{pmatrix} I_j & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+1,j} & 1 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+2,j} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -\lambda_{n-1,j} & 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} A_0 \\ \widetilde{a}_j^T \\ \widetilde{a}_{j+1}^T \\ \vdots \\ \widetilde{a}_{n-1}^T \end{pmatrix} = \begin{pmatrix} A_0 \\ \widetilde{a}_j^T \\ \widetilde{a}_{j+1}^T - \lambda_{j+1,j}\widetilde{a}_j^T \\ \vdots \\ \widetilde{a}_{n-1}^T - \lambda_{n-1,j}\widetilde{a}_j^T \end{pmatrix}.$$

An important observation that was NOT made clear enough this week is that the rows of A are updates with an AXPY! A multiple of the jth row is subtracted from the ith row.

A more concise way to describe a Gauss transforms is

$$\widetilde{L} = \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right).$$

Now, applying to a matrix A, $\widetilde{L}A$ yields

$$\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right) \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right) = \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 - l_{21}a_1^T \end{array}\right).$$

In other words, A_2 is updated with a rank-1 update. **An important observation that was NOT made clear enough this week is that a rank-1 update can be used to simultaneously subtract multiples of a row of A from other rows of A.**

Forward substitution

Forward substitution applies the same transformations that were applied to the matrix to a right-hand side vector.

Back(ward) substitution

Backward substitution solves the upper triangular system

$$\begin{array}{ccccccccc} \alpha_{0,0}\chi_0 & + & \alpha_{0,1}\chi_1 & + & \cdots & + & \alpha_{0,n-1}\chi_{n-1} & = & \beta_0 \\ & & \alpha_{1,1}\chi_1 & + & \cdots & + & \alpha_{1,n-1}\chi_{n-1} & = & \beta_1 \\ & & & \ddots & & & \vdots & \vdots & \vdots \\ & & & & & & \alpha_{n-1,n-1}\chi_{n-1} & = & \beta_{n-1} \end{array}$$

Algorithm: $A := \text{LU\_UNB\_VAR5}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

$a_{21} := a_{21}/\alpha_{11} \qquad (= l_{21})$

$A_{22} := A_{22} - a_{21}a_{12}^T \qquad (= A_{22} - l_{21}a_{12}^T)$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 6.14: LU factorization algorithm.

This algorithm overwrites b with the solution x.

LU factorization

The LU factorization factorization of a square matrix A is given by $A = LU$, where L is a unit lower triangular matrix and U is an upper triangular matrix. An algorithm for computing the LU factorization is given by

This algorithm overwrites A with the matrices L and U. Since L is unit lower triangular, its diagonal needs not be stored.

The operations that compute an LU factorization are the same as the operations that are performed when reducing a system of linear equations to an upper triangular system of equations.

Solving $Lz = b$

Forward substitution is equivalent to solving a unit lower triangular system $Lz = b$. An algorithm for this is given by This algorithm overwrites b with the solution z.

Solving $Ux = b$

Back(ward) substitution is equivalent to solving an upper triangular system $Ux = b$. An algorithm for this is given by This algorithm overwrites b with the solution x.

Solving $Ax = b$

If LU factorization completes with an upper triangular matrix U that does not have zeroes on its diagonal, then the following three steps can be used to solve $Ax = b$:

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR1}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

where λ_{11} is 1×1 , β_1 has 1 row

$b_2 := b_2 - \beta_1 l_{21}$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 6.15: Algorithm for solving $Lx = b$, overwriting b with the result vector x. Here L is a lower triangular matrix.

- Factor $A = LU$.
- Solve $Lz = b$.
- Solve $Ux = z$.

Cost

- Factoring $A = LU$ requires, approximately, $\frac{2}{3}n^3$ floating point operations.
- Solve $Lz = b$ requires, approximately, n^2 floating point operations.
- Solve $Ux = z$ requires, approximately, n^2 floating point operations.

Algorithm: $[b] := \text{UTRSV\_UNB\_VAR1}(U, b)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

 where U_{BR} is 0×0, b_B has 0 rows

while $m(U_{BR}) < m(U)$ **do**

 Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$\beta_1 := \beta_1 - u_{12}^T b_2$

$\beta_1 := \beta_1 / \upsilon_{11}$

 Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

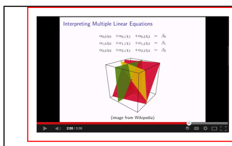
Figure 6.16: Algorithm for solving $Ux = b$ where U is an uppertriangular matrix. Vector b is overwritten with the result vector x.

Week 7

More Gaussian Elimination and Matrix Inversion

7.1 Opening Remarks

7.1.1 Introduction



☛ YouTube

7.1.2 Outline

.13

7.1.3 What You Will Learn

Upon completion of this unit, you should be able to

- Determine, recognize, and apply permutation matrices.
- Apply permutation matrices to vectors and matrices.
- Identify and interpret permutation matrices and fluently compute the multiplication of a matrix on the left and right by a permutation matrix.
- Reason, make conjectures, and develop arguments about properties of permutation matrices.
- Recognize when Gaussian elimination breaks down and apply row exchanges to solve the problem when appropriate.
- Recognize when LU factorization fails and apply row pivoting to solve the problem when appropriate.
- Recognize that when executing Gaussian elimination (LU factorization) with $Ax = b$ where A is a square matrix, one of three things can happen:
  1. The process completes with no zeroes on the diagonal of the resulting matrix U. Then $A = LU$ and $Ax = b$ has a unique solution, which can be found by solving $Lz = b$ followed by $Ux = z$.
  2. The process requires row exchanges, completing with no zeroes on the diagonal of the resulting matrix U. Then $PA = LU$ and $Ax = b$ has a unique solution, which can be found by solving $Lz = Pb$ followed by $Ux = z$.
  3. The process requires row exchanges, but at some point no row can be found that puts a nonzero on the diagonal, at which point the process fails (unless the zero appears as the last element on the diagonal, in which case it completes, but leaves a zero on the diagonal of the upper triangular matrix). In Week 8 we will see that this means $Ax = b$ does not have a unique solution.
- Reason, make conjectures, and develop arguments about properties of inverses.
- Find the inverse of a simple matrix by understanding how the corresponding linear transformation is related to the matrix-vector multiplication with the matrix.
- Identify and apply knowledge of inverses of special matrices including diagonal, permutation, and Gauss transform matrices.
- Determine whether a given matrix is an inverse of another given matrix.
- Recognize that a 2×2 matrix $A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}$ has an inverse if and only if its determinant is not zero: $\det(A) = \alpha_{0,0}\alpha_{1,1} - \alpha_{0,1}\alpha_{1,0} \neq 0$.
- Compute the inverse of a 2×2 matrix A if that inverse exists.

Track your progress in Appendix B.

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR1}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

Repartition

- $$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$
  where λ_{11} is 1×1 , β_1 has 1 row

$b_2 := b_2 - \beta_1 l_{21}$

Continue with

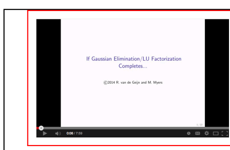
$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 7.1: Algorithm for solving $Lz = b$ when L is a unit lower triangular matrix. The right-hand side vector b is overwritten with the solution vector z.

7.2 When Gaussian Elimination Breaks Down

7.2.1 When Gaussian Elimination Works



☛ YouTube

We know that *if* Gaussian elimination completes (the LU factorization of a given matrix can be computed) *and* the upper triangular factor U has no zeroes on the diagonal, then $Ax = b$ can be solved for all right-hand side vectors b.

Why?

- If Gaussian elimination completes (the LU factorization can be computed), then $A = LU$ for some unit lower triangular matrix L and upper triangular matrix U. We know this because of the equivalence of Gaussian elimination and LU factorization.

  If you look at the algorithm for forward substitition (solving $Lz = b$) in Figure 7.1 you notice that the only computations that are encountered are multiplies and adds. Thus, the algorithm will complete.

  Similarly, the backward substitution algorithm (for solving $Ux = z$) in Figure 7.2 can only break down if the division causes an error. And that can only happen if U has a zero on its diagonal.

So, under the mentioned circumstances, we can compute *a* solution to $Ax = b$ via Gaussian elimination, forward substitution, and back substitution. Last week we saw how to compute this solution.

Algorithm: $[b] := \text{UTRSV\_UNB\_VAR1}(U, b)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

  where U_{BR} is 0×0, b_B has 0 rows

while $m(U_{BR}) < m(U)$ **do**

  Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$\beta_1 := \beta_1 - u_{12}^T b_2$

$\beta_1 := \beta_1 / \upsilon_{11}$

  Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 7.2: Algorithm for solving $Ux = b$ when U is an upper triangular matrix. The right-hand side vector b is overwritten with the solution vector x.

Is this the only solution?

We first give an intuitive explanation, and then we move on and walk you through a rigorous proof.

The reason is as follows: Assume that $Ax = b$ has two solutions: u and v. Then

- $Au = b$ and $Av = b$.
- This then means that vector $w = u - v$ satisfies
$$Aw = A(u - v) = Au - Av = b - b = 0.$$
- Since Gaussian elimination completed we know that
$$(LU)w = 0,$$
or, equivalently,
$$Lz = 0 \quad \text{and} \quad Uw = z.$$
- It is not hard to see that if $Lz = 0$ then $z = 0$:
$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \lambda_{1,0} & 1 & 0 & \cdots & 0 \\ \lambda_{2,0} & \lambda_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ \lambda_{n-1,0} & \lambda_{n-1,1} & \lambda_{n-1,2} & \cdots & 1 \end{pmatrix} \begin{pmatrix} \zeta_0 \\ \zeta_1 \\ \zeta_2 \\ \vdots \\ \zeta_{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$
means $\zeta_0 = 0$. But then $\lambda_{1,0}\zeta_0 + \zeta_1 = 0$ means $\zeta_1 = 0$. In turn $\lambda_{2,0}\zeta_0 + \lambda_{2,1}\zeta_1 + \zeta_2 = 0$ means $\zeta_2 = 0$. And so forth.

- Thus, $z = 0$ and hence $Uw = 0$.
- It is not hard to see that if $Uw = 0$ then $w = 0$:

$$\begin{pmatrix} \upsilon_{0,0} & \cdots & \upsilon_{0,n-3} & \upsilon_{0,n-2} & \upsilon_{0,n-1} \\ \vdots & \ddots & \vdots & \vdots & \\ 0 & \cdots & \upsilon_{n-3,n-3} & \upsilon_{n-3,n-2} & \upsilon_{n-3,n-1} \\ 0 & \cdots & 0 & \upsilon_{n-2,n-2} & \upsilon_{n-2,n-1} \\ 0 & \cdots & 0 & 0 & \upsilon_{n-11,n-1} \end{pmatrix} \begin{pmatrix} \omega_0 \\ \vdots \\ \omega_{n-3} \\ \omega_{n-2} \\ \omega_{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

means $\upsilon_{n-1,n-1}\omega_{n-1} = 0$ and hence $\omega_{n-1} = 0$ (since $\upsilon_{n-1,n-1} \neq 0$). But then $\upsilon_{n-2,n-2}\omega_{n-2} + \upsilon_{n-2,n-1}\omega_{n-1} = 0$ means $\omega_{n-2} = 0$. And so forth.

We conclude that

> If Gaussian elimination completes with an upper triangular system that has no zero diagonal coefficients (LU factorization computes with L and U where U has no diagonal zero elements), then for all right-hand side vectors, b, the linear system $Ax = b$ has a unique solution x.

A rigorous proof

> Let $A \in \mathbb{R}^{n\times n}$. *If* Gaussian elimination completes and the resulting upper triangular system has no zero coefficients on the diagonal (U has no zeroes on its diagonal), *then* there is a unique solution x to $Ax = b$ for all $b \in \mathbb{R}$.
>
> Always/Sometimes/Never

We don't yet state this as a homework problem, because to get to that point we are going to make a number of observations that lead you to the answer.

Homework 7.2.1.1 Let $L \in \mathbb{R}^{1\times 1}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.2 Give the solution of $\begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$.

☛ SEE ANSWER

Homework 7.2.1.3 Give the solution of $\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$.

(Hint: look carefully at the last problem, and you will be able to save yourself some work.)

☛ SEE ANSWER

Homework 7.2.1.4 Let $L \in \mathbb{R}^{2\times 2}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.5 Let $L \in \mathbb{R}^{3\times 3}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR2}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

where λ_{11} is 1×1 , β_1 has 1 row

Continue with

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 7.3: Blank algorithm for solving $Lx = b$, overwriting b with the result vector x for use in Homework 7.2.1.7. Here L is a lower triangular matrix.

Homework 7.2.1.6 Let $L \in \mathbb{R}^{n \times n}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.7 The proof for the last exercise suggests an alternative algorithm (Variant 2) for solving $Lx = b$ when L is unit lower triangular. Use Figure 7.3 to state this alternative algorithm and then implement it, yielding

- `[ b_out ] = Ltrsv_unb_var2( L, b )`

You can check that they compute the right answers with the script in

- `test_Ltrsv_unb_var2.m`

☛ SEE ANSWER

33

Homework 7.2.1.8 Let $L \in \mathbb{R}^{n \times n}$ be a unit lower triangular matrix. $Lx = 0$, where 0 is the zero vector of size n, has the unique solution $x = 0$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.9 Let $U \in \mathbb{R}^{1\times 1}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.10 Give the solution of $\begin{pmatrix} -1 & 1 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$.

☛ SEE ANSWER

Homework 7.2.1.11 Give the solution of $\begin{pmatrix} -2 & 1 & -2 \\ 0 & -1 & 1 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix}$.

☛ SEE ANSWER

Homework 7.2.1.12 Let $U \in \mathbb{R}^{2\times 2}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.13 Let $U \in \mathbb{R}^{3\times 3}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.14 Let $U \in \mathbb{R}^{n\times n}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

☛ SEE ANSWER

The proof for the last exercise closely mirrors how we derived Variant 1 for solving $Ux = b$ last week.

Homework 7.2.1.15 Let $U \in \mathbb{R}^{n\times n}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = 0$, where 0 is the zero vector of size n, has the unique solution $x = 0$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.1.16 Let $A \in \mathbb{R}^{n\times n}$. *If* Gaussian elimination completes and the resulting upper triangular system has no zero coefficients on the diagonal (U has no zeroes on its diagonal), *then* there is a unique solution x to $Ax = b$ for all $b \in \mathbb{R}$.

Always/Sometimes/Never

☛ SEE ANSWER

7.2.2 The Problem

☛ **YouTube**

The question becomes "Does Gaussian elimination always solve a linear system of n equations and n unknowns?" Or, equivalently, can an LU factorization always be computed for an $n \times n$ matrix? In this unit we show that there are linear systems where $Ax = b$ has a unique solution but Gaussian elimination (LU factorization) breaks down. In this and the next sections we will discuss what modifications must be made to Gaussian elimination and LU factorization so that *if* $Ax = b$ has a unique solution, *then* these modified algorithms complete and can be used to solve $Ax = b$.

A simple example where Gaussian elimination and LU factorization break down involves the matrix $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. In the first step, the multiplier equals $1/0$, which will cause a "division by zero" error.

Now, $Ax = b$ is given by the set of linear equations

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \beta_1 \\ \beta_0 \end{pmatrix}$$

so that $Ax = b$ is equivalent to

$$\begin{pmatrix} \chi_1 \\ \chi_0 \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}$$

and the solution to $Ax = b$ is given by the vector $x = \begin{pmatrix} \beta_1 \\ \beta_0 \end{pmatrix}$.

Homework 7.2.2.1 Solve the following linear system, via the steps in Gaussian elimination that you have learned so far.

$$\begin{aligned} 2\chi_0 + \quad 4\chi_1 + (-2)\chi_2 &= -10 \\ 4\chi_0 + \quad 8\chi_1 + \quad 6\chi_2 &= 20 \\ 6\chi_0 + (-4)\chi_1 + \quad 2\chi_2 &= 18 \end{aligned}$$

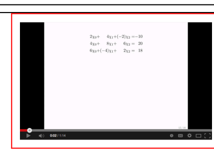
Mark all that are correct:

(a) The process breaks down.

(b) There is no solution.

(c) $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 4 \end{pmatrix}$

☛ SEE ANSWER



☛ YouTube

Now you try an example:

Homework 7.2.2.2 Perform Gaussian elimination with

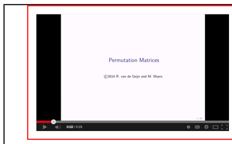
$$\begin{aligned} 0\chi_0 + \quad 4\chi_1 + (-2)\chi_2 &= -10 \\ 4\chi_0 + \quad 8\chi_1 + \quad 6\chi_2 &= 20 \\ 6\chi_0 + (-4)\chi_1 + \quad 2\chi_2 &= 18 \end{aligned}$$

☛ SEE ANSWER

We now understand how to modify Gaussian elimination so that it completes when a zero is encountered on the diagonal and a nonzero appears somewhere below it.

The above examples suggest that the LU factorization algorithm needs to be modified to allow for row exchanges. But to do so, we need to develop some machinery.

7.2.3 Permutations

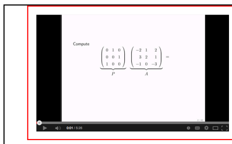


☛ YouTube

Homework 7.2.3.1 Compute

$$\underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}}_{P} \underbrace{\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}}_{A} =$$

☛ SEE ANSWER



☛ YouTube

Examining the matrix P in the above exercise, we see that each row of P equals a unit basis vector. This leads us to the following definitions that we will use to help express permutations:

Definition 7.1 *A vector with integer components*

$$p = \begin{pmatrix} k_0 \\ k_1 \\ \vdots \\ k_{n-1} \end{pmatrix}$$

is said to be a permutation vector if

- $k_j \in \{0, \ldots, n-1\}$ *, for* $0 \leq j < n$*; and*
- $k_i = k_j$ *implies* $i = j$*.*

In other words, p *is a rearrangement of the numbers* $0, \ldots, n-1$ *(without repetition).*

We will often write $(k_0, k_1, \ldots, k_{n-1})^T$ to indicate the column vector, for space considerations.

Definition 7.2 *Let* $p = (k_0, \ldots, k_{n-1})^T$ *be a permutation vector. Then*

$$P = P(p) = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}$$

is said to be a permutation matrix.

In other words, P is the identity matrix with its rows rearranged as indicated by the permutation vector $(k_0, k_1, \ldots, k_{n-1})$. We will frequently indicate this permutation matrix as $P(p)$ to indicate that the permutation matrix corresponds to the permutation vector p.

Homework 7.2.3.2 For each of the following, give the permutation matrix $P(p)$:

- If $p = \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \end{pmatrix}$ then $P(p) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$,
- If $p = \begin{pmatrix} 3 \\ 2 \\ 1 \\ 0 \end{pmatrix}$ then $P(p) =$
- If $p = \begin{pmatrix} 1 \\ 0 \\ 2 \\ 3 \end{pmatrix}$ then $P(p) =$
- If $p = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 0 \end{pmatrix}$ then $P(p) =$

☛ SEE ANSWER

Homework 7.2.3.3 Let $p = (2,0,1)^T$. Compute

- $P(p) \begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} =$
- $P(p) \begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} =$

☛ SEE ANSWER

Homework 7.2.3.4 Let $p = (2,0,1)^T$ and $P = P(p)$. Compute

$$\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} P^T =$$

☛ SEE ANSWER

Homework 7.2.3.5 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation vector. Consider

$$x = \left(\begin{array}{c} \chi_0 \\ \hline \chi_1 \\ \hline \vdots \\ \hline \chi_{n-1} \end{array}\right).$$

Applying permuation matrix $P = P(p)$ to x yields

$$Px = \left(\begin{array}{c} \chi_{k_0} \\ \hline \chi_{k_1} \\ \hline \vdots \\ \hline \chi_{k_{n-1}} \end{array}\right).$$

Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.3.6 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation. Consider

$$A = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{n-1}^T \end{array}\right).$$

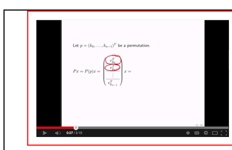
Applying $P = P(p)$ to A yields

$$PA = \left(\begin{array}{c} \widetilde{a}_{k_0}^T \\ \hline \widetilde{a}_{k_1}^T \\ \hline \vdots \\ \hline \widetilde{a}_{k_{n-1}}^T \end{array}\right).$$

Always/Sometimes/Never

☛ SEE ANSWER

In other words, Px and PA rearrange the elements of x and the rows of A in the order indicated by permutation vector p.



☛ YouTube

Homework 7.2.3.7 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation, $P = P(p)$, and $A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array}\right)$.
$AP^T = \left(\begin{array}{c|c|c|c} a_{k_0} & a_{k_1} & \cdots & a_{k_{n-1}} \end{array}\right)$.

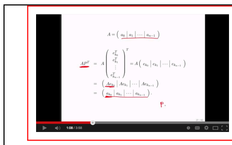
Aways/Sometimes/Never

☛ SEE ANSWER

Homework 7.2.3.8 If P is a permutation matrix, then so is P^T.

True/False

☛ SEE ANSWER



☛ YouTube

Definition 7.3 *Let us call the special permutation matrix of the form*

$$\widetilde{P}(\pi) = \left(\begin{array}{c} \boxed{e_\pi^T} \\ e_1^T \\ \vdots \\ e_{\pi-1}^T \\ \boxed{e_0^T} \\ e_{\pi+1}^T \\ \vdots \\ e_{n-1}^T \end{array}\right) = \left(\begin{array}{c|cccc|c|ccc} 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \hline 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ \hline 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{array}\right)$$

a pivot matrix.

$$\widetilde{P}(\pi) = (\widetilde{P}(\pi))^T.$$

Homework 7.2.3.9 Compute

$$\widetilde{P}(1)\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} = \quad \text{and} \quad \widetilde{P}(1)\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} = .$$

☛ SEE ANSWER

Homework 7.2.3.10 Compute

$$\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}\widetilde{P}(1) = .$$

☛ SEE ANSWER

Homework 7.2.3.11 When $\widetilde{P}(\pi)$ (of appropriate size) multiplies a matrix from the left, it swaps row 0 and row π, leaving all other rows unchanged.

Always/Sometimes/Never

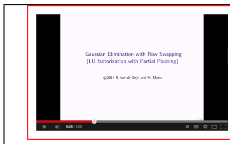
☛ SEE ANSWER

Homework 7.2.3.12 When $\widetilde{P}(\pi)$ (of appropriate size) multiplies a matrix from the right, it swaps column 0 and column π, leaving all other columns unchanged.

Always/Sometimes/Never

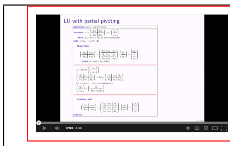
☛ SEE ANSWER

7.2.4 Gaussian Elimination with Row Swapping (LU Factorization with Partial Pivoting)



☛ YouTube

Gaussian elimination with row pivoting



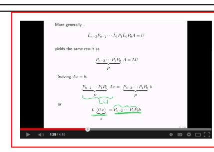
☛ YouTube

We start our discussion with the example in Figure 7.4.

Homework 7.2.4.1 Compute

- $\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \end{array}\right)\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right)\left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}\right) =$
- $\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 3 & 1 & 0 \\ 2 & 0 & 1 \end{array}\right)\left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 0 & -16 & 8 \\ 0 & 0 & 10 \end{array}\right) =$
- What do you notice?

☛ SEE ANSWER



☛ YouTube

What the last homework is trying to demonstrate is that, for given matrix A,

- Let $L = \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 3 & 1 & 0 \\ 2 & 0 & 1 \end{array}\right)$ be the matrix in which the multipliers have been collected (the unit lower triangular matrix that has overwritten the strictly lower triangular part of the matrix).
- Let $U = \left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 0 & -16 & 8 \\ 0 & 0 & 10 \end{array}\right)$ be the upper triangular matrix that overwrites the matrix.
- Let P be the net result of multiplying all the permutation matrices together, from last to first as one goes from lef t to right:

$$P = \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \end{array}\right)\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \end{array}\right)$$

Then

$$PA = LU.$$

Example 7.4
(You may want to print the blank worksheet at the end of this week so you can follow along.)
In this example, we incorporate the insights from the last two units (Gaussian elimination with row interchanges and permutation matrices) into the explanation of Gaussian elimination that uses Gauss transforms:

i	L_i	$\tilde{P}$	A	p
0		$\begin{array}{\|c\|cc} \hline 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}$	$\begin{array}{\|c\|cc} \hline 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}$	$\begin{array}{c} \hline 0 \\ \hline \cdot \\ \cdot \end{array}$
	$\begin{array}{\|c\|cc} \hline 1 & 0 & 0 \\ \hline -2 & 1 & 0 \\ -3 & 0 & 1 \end{array}$		$\begin{array}{\|c\|cc} \hline 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}$	$\begin{array}{c} \hline 0 \\ \hline \cdot \\ \cdot \end{array}$
1		$\begin{array}{c\|c\|c} & & \\ \hline & 0 & 1 \\ \hline & 1 & 0 \end{array}$	$\begin{array}{c\|c\|c} 2 & 4 & -2 \\ \hline 2 & 0 & 10 \\ \hline 3 & -16 & 8 \end{array}$	$\begin{array}{c} 0 \\ \hline 1 \\ \hline \cdot \end{array}$
	$\begin{array}{c\|c\|c} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -0 & 1 \end{array}$		$\begin{array}{c\|c\|c} 2 & 4 & -2 \\ \hline 3 & -16 & 8 \\ \hline 2 & 0 & 10 \end{array}$	$\begin{array}{c} 0 \\ \hline 1 \\ \hline \cdot \end{array}$
2			$\begin{array}{cc\|c\|} 2 & 4 & -2 \\ 3 & -16 & 8 \\ \hline 2 & 0 & 10 \\ \hline \end{array}$	$\begin{array}{c} 0 \\ 1 \\ \hline 0 \\ \hline \end{array}$

Figure 7.4: Example of a linear system that requires row swapping to be added to Gaussian elimination.

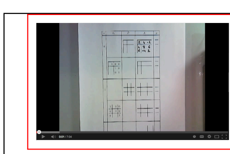
In other words, Gaussian elimination with row interchanges computes the LU factorization of a permuted matrix. Of course, one does not generally know ahead of time (*a priori*) what that permutation must be, because one doesn't know when a zero will appear on the diagonal. The key is to notice that when we pivot, we also interchange the multipliers that have overwritten the zeroes that were introduced.

Homework 7.2.4.2
(You may want to print the blank worksheet at the end of this week so you can follow along.)
Perform Gaussian elimination with row swapping (row pivoting):

i	L_i	$\tilde{P}$	A	p
0			$\begin{array}{c\|cc} \hline 0 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}$	
1				
2				

☛ SEE ANSWER

The example and exercise motivate the modification to the LU factorization algorithm in Figure 7.5. In that algorithm, PIVOT(x) returns the index of the first nonzero component of x. This means that the algorithm only works if it is always the case that $\alpha_{11} \neq 0$ or vector a_{21} contains a nonzero component.



☛ YouTube

Algorithm: $[A, p] := \text{LU\_PIV}(A, p)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $p \rightarrow \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right)$
  where A_{TL} is 0×0 and p_T has 0 components
while $m(A_{TL}) < m(A)$ **do**
  Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right) \rightarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ \hline p_2 \end{array}\right)$$

$$\pi_1 = \text{PIVOT}\left(\left(\begin{array}{c} \alpha_{11} \\ \hline a_{21} \end{array}\right)\right)$$

$$\left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) := P(\pi_1)\left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

$a_{21} := a_{21}/\alpha_{11}$ (a_{21} now contains l_{21})

$$\left(\begin{array}{c} a_{12}^T \\ \hline A_{22} \end{array}\right) = \left(\begin{array}{c} a_{12}^T \\ \hline A_{22} - a_{21}a_{12}^T \end{array}\right)$$

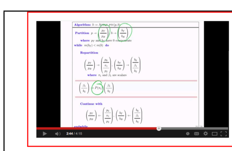
  Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right) \leftarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ \hline p_2 \end{array}\right)$$

endwhile

Figure 7.5: LU factorization algorithm that incorporates row (partial) pivoting.

Solving the linear system



☛ YouTube

Here is the cool part: We have argued that Gaussian elimination with row exchanges (LU factorization with row pivoting) computes the equivalent of a pivot matrix P and factors L and U (unit lower triangular and upper triangular, respectively) so that $PA = LU$. If we want to solve the system $Ax = b$, then

$$Ax = b$$

is equivalent to

$$PAx = Pb.$$

Now, $PA = LU$ so that

$$\underbrace{(LU)}_{PA} x = Pb.$$

- **Algorithm:** $b := \text{APPLY\_PIV}(p, b)$

  Partition $p \rightarrow \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

  where p_T and b_T have 0 components

  while $m(b_T) < m(b)$ **do**

  Repartition

  $$\left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right) \rightarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ p_2 \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ b_2 \end{array}\right)$$

  $$\left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right) := P(\pi_1) \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

  Continue with

  $$\left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right) \leftarrow \left(\begin{array}{c} p_0 \\ \pi_1 \\ \hline p_2 \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \beta_1 \\ \hline b_2 \end{array}\right)$$

  endwhile

Figure 7.6: Algorithm for applying the same exchanges rows that happened during the LU factorization with row pivoting to the components of the right-hand side.

So, solving $Ax = b$ is equivalent to solving

$$L \underbrace{(Ux)}_{z} = Pb.$$

This leaves us with the following steps:

Update $b := Pb$ by applying the pivot matrices that were encountered during Gaussian elimination with row exchanges to vector b, *in the same order*. A routine that, given the vector with pivot information p, does this is given in Figure 7.6.

- Solve $Lz = b$ with this updated vector b, overwriting b with z. For this we had the routine `Ltrsv_unit`.
- Solve $Ux = b$, overwriting b with x. For this we had the routine `Utrsv_nonunit`.

Uniqueness of solution

> If Gaussian elimination with row exchanges (LU factorization with pivoting) completes with an upper triangular system that has no zero diagonal coefficients, then for all right-hand side vectors, b, the linear system $Ax = b$ has a unique solution, x.

7.2.5 When Gaussian Elimination Fails Altogether

Now, we can see that when executing Gaussian elimination (LU factorization) with $Ax = b$ where A is a square matrix, one of three things can happen:

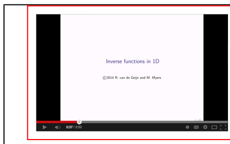
- The process completes with no zeroes on the diagonal of the resulting matrix U. Then $A = LU$ and $Ax = b$ has a unique solution, which can be found by solving $Lz = b$ followed by $Ux = z$.

- The process requires row exchanges, completing with no zeroes on the diagonal of the resulting matrix U. Then $PA = LU$ and $Ax = b$ has a unique solution, which can be found by solving $Lz = Pb$ followed by $Ux = z$.
- The process requires row exchanges, but at some point no row can be found that puts a nonzero on the diagonal, at which point the process fails (unless the zero appears as the last element on the diagonal, in which case it completes, but leaves a zero on the diagonal).

This last case will be studied in great detail in future weeks. For now, we simply state that in this case $Ax = b$ either has *no* solutions, or it has an *infinite* number of solutions.

7.3 The Inverse Matrix

7.3.1 Inverse Functions in 1D



☛ YouTube

In high school, you should have been exposed to the idea of an inverse of a function of one variable. If

- $f : \mathbb{R} \to \mathbb{R}$ maps a real to a real; and
- it is a *bijection* (both one-to-one and onto)

then

- $f(x) = y$ has a unique solution for all $y \in \mathbb{R}$.
- The function that maps y to x so that $g(y) = x$ is called the inverse of f.
- It is denoted by $f^{-1} : \mathbb{R} \to \mathbb{R}$.
- Importantly, $f(f^{-1}(x)) = x$ and $f^{-1}(f(x)) = x$.

In the next units we will examine how this extends to vector functions and linear transformations.

7.3.2 Back to Linear Transformations

☛ YouTube

Theorem 7.5 *Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a vector function. Then f is one-to-one and onto (a bijection) implies that $m = n$.*

The proof of this hinges on the dimensionality of $\mathbb{R}^m$ and $\mathbb{R}^n$. We won't give it here.

Corollary 7.6 *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a vector function that is a bijection. Then there exists a function $f^{-1} : \mathbb{R}^n \to \mathbb{R}^n$, which we will call its inverse, such that $f(f^{-1}(x)) = f^{-1}(f(x)) = x$.*

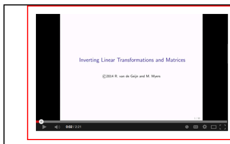
This is an immediate consequence of the fact that for every y there is a unique x such that $f(x) = y$ and $f^{-1}(y)$ can then be defined to equal that x.

Homework 7.3.2.1 Let $L : \mathbb{R}^n \to \mathbb{R}^n$ be a linear transformation that is a bijection and let L^{-1} denote its inverse.

L^{-1} is a linear transformation.

Always/Sometimes/Never

☛ SEE ANSWER



☛ YouTube

What we conclude is that if $A \in \mathbb{R}^{n \times n}$ is the matrix that represents a linear transformation that is a bijection L, then there is a matrix, which we will denote by A^{-1}, that represents L^{-1}, the inverse of L. Since for all $x \in \mathbb{R}^n$ it is the case that $L(L^{-1}(x)) = L^{-1}(L(x)) = x$, we know that $AA^{-1} = A^{-1}A = I$, the identity matrix.

Theorem 7.7 *Let $L : \mathbb{R}^n \to \mathbb{R}^n$ be a linear transformation, and let A be the matrix that represents L. If there exists a matrix B such that $AB = BA = I$, then L has an inverse, L^{-1}, and B equals the matrix that represents that linear transformation.*

Actually, it suffices to require there to be a matrix B such that $AB = I$ *or* $BA = I$. But we don't quite have the knowledge at this point to be able to prove it from that weaker assumption.

Proof: We need to show that L is a bijection. Clearly, for every $x \in \mathbb{R}^n$ there is a $y \in \mathbb{R}^n$ such that $y = L(x)$. The question is whether, given any $y \in \mathbb{R}^n$, there is a vector $x \in \mathbb{R}^n$ such that $L(x) = y$. But

$$L(By) = A(By) = (AB)y = Iy = y.$$

So, $x = By$ has the property that $L(x) = y$.

But is this vector x unique? If $Ax_0 = y$ and $Ax_1 = y$ then $A(x_0 - x_1) = 0$. Since $BA = I$ we find that $BA(x_0 - x_1) = x_0 - x_1$ and hence $x_0 - x_1 = 0$, meaning that $x_0 = x_1$.

Let $L : \mathbb{R}^n \to \mathbb{R}^n$ and let A be the matrix that represents L. Then L has an inverse if and only if there exists a matrix B such that $AB = BA = I$. We will call matrix B the inverse of A, denote it by A^{-1} and note that if $AA^{-1} = I$ then $A^{-1}A = I$.

Definition 7.8 *A matrix A is said to be invertible if the inverse, A^{-1}, exists. An equivalent term for invertible is nonsingular.*

We are going to collect a string of conditions that are equivalent to the statement "A is invertible". Here is the start of that collection.

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.

We will add to this collection as the course proceeds.

Homework 7.3.2.2 Let A, B, and C all be $n \times n$ matrices. If $AB = I$ and $CA = I$ then $B = C$.

True/False

☛ SEE ANSWER

7.3.3 Simple Examples

☛ YouTube

General principles

Given a matrix A for which you want to find the inverse, the first thing you have to check is that A is square. Next, you want to ask yourself the question: "What is the matrix that undoes Ax?" Once you guess what that matrix is, say matrix B, you prove it to yourself by checking that $BA = I$ or $AB = I$.

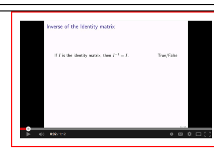
If that doesn't lead to an answer or if that matrix is too complicated to guess at an inverse, you should use a more systematic approach which we will teach you in the next unit. We will then teach you a fool-proof method next week.

Inverse of the Identity matrix

Homework 7.3.3.1 If I is the identity matrix, then $I^{-1} = I$.

True/False

☛ SEE ANSWER



☛ YouTube

Inverse of a diagonal matrix

Homework 7.3.3.2 Find

$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix}^{-1} =$$

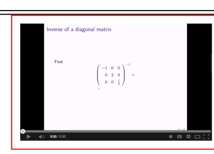
☛ SEE ANSWER

Homework 7.3.3.3 Assume $\delta_j \neq 0$ for $0 \leq j < n$.

$$\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{\delta_0} & 0 & \cdots & 0 \\ 0 & \frac{1}{\delta_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\delta_{n-1}} \end{pmatrix}.$$

Always/Sometimes/Never

☛ SEE ANSWER



☛ YouTube

Inverse of a Gauss transform

Homework 7.3.3.4 Find

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix}^{-1} =$$

Important: read the answer!

☛ SEE ANSWER

Homework 7.3.3.5

$$\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right)^{-1} = \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right).$$

True/False

☛ SEE ANSWER

The observation about how to compute the inverse of a Gauss transform explains the link between Gaussian elimination with Gauss transforms and LU factorization.

Let's review the example from Section 6.2.4:

Before	After
• $\begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix}$	$\begin{pmatrix} 2 & 4 & -2 \\ & -10 & 10 \\ & -16 & 8 \end{pmatrix}.$

Before	After
• $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ & -10 & 10 \\ & -16 & 8 \end{pmatrix}$	$\begin{pmatrix} 2 & 4 & -2 \\ & -10 & 10 \\ & & -8 \end{pmatrix}.$

Now, we can summarize the above by

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & 0 & -8 \end{pmatrix}.$$

Now

$$\begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & 0 & -8 \end{pmatrix}.$$

so that

$$\begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & 0 & -8 \end{pmatrix}.$$

But, given our observations about the inversion of Gauss transforms, this translates to

$$\begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1.6 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & 0 & -8 \end{pmatrix}.$$

But, miraculously,

$$\begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1.6 & 1 \end{pmatrix}}_{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 1.6 & 1 \end{pmatrix}} \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & 0 & -8 \end{pmatrix}.$$

But this gives us the LU factorization of the original matrix:

$$\begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 1.6 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & 0 & -8 \end{pmatrix}.$$

Now, the LU factorization (overwriting the strictly lower triangular part of the matrix with the multipliers) yielded

$$\begin{pmatrix} 2 & 4 & -2 \\ {\scriptstyle 2} & -10 & 10 \\ {\scriptstyle 3} & {\scriptstyle 1.6} & -8 \end{pmatrix}.$$

NOT a coincidence!

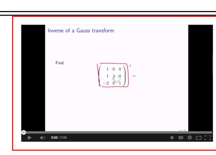
The following exercise explains further:

Homework 7.3.3.6 Assume the matrices below are partitioned conformally so that the multiplications and comparison are legal.

$$\left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & 0 & I \end{array}\right) \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) = \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER



☛ YouTube

Inverse of a permutation

Homework 7.3.3.7 Find

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} =$$

☛ SEE ANSWER

Homework 7.3.3.8 Find

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{-1} =$$

☛ SEE ANSWER

Homework 7.3.3.9 Let P be a permutation matrix. Then $P^{-1} = P$.

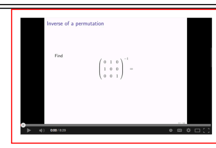
Always/Sometimes/Never

☛ SEE ANSWER

Homework 7.3.3.10 Let P be a permutation matrix. Then $P^{-1} = P^T$.

Always/Sometimes/Never

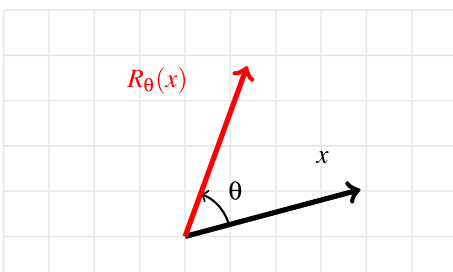
☛ SEE ANSWER



☛ YouTube

Inverting a 2D rotation

Homework 7.3.3.11 Recall from Week 2 how $R_\theta(x)$ rotates a vector x through angle θ:



R_θ is represented by the matrix

$$R = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}.$$

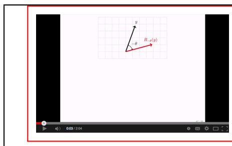
What transformation will "undo" this rotation through angle θ? (Mark all correct answers)

(a) $R_{-\theta}(x)$

(b) Ax, where $A = \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{pmatrix}$

(c) Ax, where $A = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}$

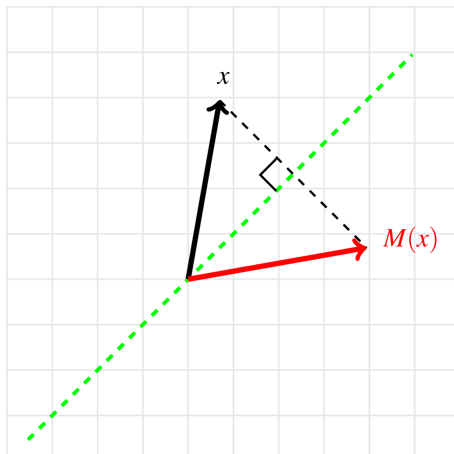
☛ SEE ANSWER



☛ YouTube

Inverting a 2D reflection

Homework 7.3.3.12 Consider a reflection with respect to the 45 degree line:



If A represents the linear transformation M, then

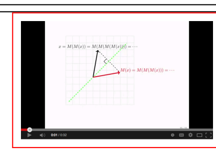
(a) $A^{-1} = -A$

(b) $A^{-1} = A$

(c) $A^{-1} = I$

(d) All of the above.

☛ SEE ANSWER



☛ YouTube

7.3.4 More Advanced (but Still Simple) Examples

☛ YouTube

More general principles

Notice that $AA^{-1} = I$. Let's label A^{-1} with the letter B instead. Then $AB = I$. Now, partition both B and I by columns. Then

$$A \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} e_0 & e_1 & \cdots & e_{n-1} \end{array} \right)$$

and hence $Ab_j = e_j$. So.... the jth column of the inverse equals the solution to $Ax = e_j$ where A and e_j are input, and x is output.

We can now add to our string of equivalent conditions:

The following statements are equivalent statements about $A \in \mathbb{R}^{n\times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.

Inverse of a triangular matrix

Homework 7.3.4.1 Compute $\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix}^{-1} =$

☛ SEE ANSWER

☛ YouTube

Homework 7.3.4.2 Find

$$\begin{pmatrix} 1 & -2 \\ 0 & 2 \end{pmatrix}^{-1} =$$

☛ SEE ANSWER

Homework 7.3.4.3 Let $\alpha_{0,0} \neq 0$ and $\alpha_{1,1} \neq 0$. Then

$$\begin{pmatrix} \alpha_{0,0} & 0 \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{\alpha_{0,0}} & 0 \\ -\frac{\alpha_{1,0}}{\alpha_{0,0}\alpha_{1,1}} & \frac{1}{\alpha_{1,1}} \end{pmatrix}$$

True/False

☛ SEE ANSWER

Homework 7.3.4.4 Partition lower triangular matrix L as

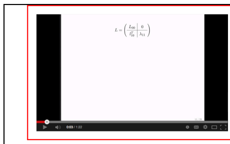
$$L = \left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)$$

Assume that L has no zeroes on its diagonal. Then

$$L^{-1} = \left(\begin{array}{c|c} L_{00}^{-1} & 0 \\ \hline -\frac{1}{\lambda_{11}} l_{10}^T L_{00}^{-1} & \frac{1}{\lambda_{11}} \end{array}\right)$$

True/False

☛ SEE ANSWER



☛ YouTube

Homework 7.3.4.5 The inverse of a lower triangular matrix with no zeroes on its diagonal is a lower triangular matrix.

True/False

☛ SEE ANSWER

Challenge 7.3.4.6 The answer to the last exercise suggests an algorithm for inverting a lower triangular matrix. See if you can implement it!

Inverting a 2×2 matrix

Homework 7.3.4.7 Find

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}^{-1} =$$

☛ SEE ANSWER

Homework 7.3.4.8 If $\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1} \neq 0$ then

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}^{-1} = \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix}$$

(Just check by multiplying... Deriving the formula is time consuming.)

True/False

☛ SEE ANSWER

Homework 7.3.4.9 The 2×2 matrix $A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}$ has an inverse if and only if $\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1} \neq 0$.

True/False

☛ SEE ANSWER

☛ YouTube

The expression $\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1} \neq 0$ is known as the *determinant* of

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}.$$

This 2×2 matrix has an inverse if and only if its determinant is nonzero. We will see how the determinant is useful again late in the course, when we discuss how to compute eigenvalues of small matrices. The determinant of a $n \times n$ matrix can be defined and is similarly a condition for checking whether the matrix is invertible. For this reason, we add it to our list of equivalent conditions:

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.

7.3.5 Properties

Inverse of product

Homework 7.3.5.1 Let $\alpha \neq 0$ and B have an inverse. Then

$$(\alpha B)^{-1} = \frac{1}{\alpha} B^{-1}.$$

True/False

☛ SEE ANSWER

Homework 7.3.5.2 Which of the following is true regardless of matrices A and B (as long as they have an inverse and are of the same size)?

(a) $(AB)^{-1} = A^{-1}B^{-1}$

(b) $(AB)^{-1} = B^{-1}A^{-1}$

(c) $(AB)^{-1} = B^{-1}A$

(d) $(AB)^{-1} = B^{-1}$

☛ SEE ANSWER

Homework 7.3.5.3 Let square matrices $A, B, C \in \mathbb{R}^{n\times n}$ have inverses A^{-1}, B^{-1}, and C^{-1}, respectively. Then $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$.

Always/Sometimes/Never
☛ SEE ANSWER

Inverse of transpose

Homework 7.3.5.4 Let square matrix A have inverse A^{-1}. Then $(A^T)^{-1} = (A^{-1})^T$.

Always/Sometimes/Never
☛ SEE ANSWER

Inverse of inverse

Homework 7.3.5.5

$$(A^{-1})^{-1} = A$$

Always/Sometimes/Never
☛ SEE ANSWER

7.4 Enrichment

7.4.1 Library Routines for LU with Partial Pivoting

Various linear algebra software libraries incorporate LU factorization with partial pivoting.

LINPACK

The first such library was LINPACK:

> J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart.
> LINPACK Users' Guide.
> SIAM, 1979.

A link to the implementation of the routine `DGEFA` can be found at

http://www.netlib.org/linpack/dgefa.f.

You will notice that it is written in Fortran and uses what are now called Level-1 BLAS routines. LINPACK preceded the introduction of computer architectures with cache memories, and therefore no blocked algorithm is included in that library.

LAPACK

LINPACK was replaced by the currently most widely used library, LAPACK:

> E. Anderson, Z. Bai, J. Demmel, J. J. Dongarra, J. Ducroz, A. Greenbaum, S. Hammarling, A. E. McKenney, S. Ostroucho, and D. Sorensen.
> LAPACK Users' Guide.
> SIAM 1992.

> E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. J. Dongarra, J. Ducroz, A. Greenbaum, S. Hammarling, A. E. McKenney, S. Ostroucho, and D. Sorensen.
> LAPACK Users' Guide (3rd Edition).
> SIAM 1999.

Implementations in this library include

- DGETF2 (unblocked LU factorization with partial pivoting).
- DGETRF (blocked LU factorization with partial pivoting).

It, too, is written in Fortran. The unblocked implementation makes calls to Level-1 (vector-vector) and Level-2 (matrix-vector) BLAS routines. The blocked implementation makes calls to Level-3 (matrix-matrix) BLAS routines. See if you can recognize some of the names of routines.

ScaLAPACK

ScaLAPACK is version of LAPACK that was (re)written for large distributed memory architectures. The design decision was to make the routines in ScaLAPACK reflect as closely as possible the corresponding routines in LAPACK.

> L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, R. C. Whaley.
> ScaLAPACK Users' Guilde.
> SIAM, 1997.

Implementations in this library include

- PDGETRF (blocked LU factorization with partial pivoting).

ScaLAPACK is wirtten in a mixture of Fortran and C. The unblocked implementation makes calls to Level-1 (vector-vector) and Level-2 (matrix-vector) BLAS routines. The blocked implementation makes calls to Level-3 (matrix-matrix) BLAS routines. See if you can recognize some of the names of routines.

`libflame`

We have already mentioned libflame. It targets sequential and multithreaded architectures.

> F. G. Van Zee, E. Chan, R. A. van de Geijn, E. S. Quintana-Orti, G. Quintana-Orti.
> The libflame Library for Dense Matrix Computations.
> IEEE Computing in Science and Engineering, Vol. 11, No 6, 2009.

> F. G. Van Zee.
> *libflame: The Complete Reference.*
> www.lulu.com , 2009
> (Available from http://www.cs.utexas.edu/ flame/web/FLAMEPublications.html.)

It uses an API so that the code closely resembles the code that you have been writing.

- Various unblocked and blocked implementations.

Elemental

Elemental is a library that targets distributed memory architectures, like ScaLAPACK does.

> Jack Poulson, Bryan Marker, Robert A. van de Geijn, Jeff R. Hammond, Nichols A. Romero. Elemental: A New Framework for Distributed Memory Dense Matrix Computations. ACM Transactions on Mathematical Software (TOMS), 2013.
> (Available from http://www.cs.utexas.edu/ flame/web/FLAMEPublications.html.)

It is coded in C++ in a style that resembles the FLAME APIs.

- Blocked implementation.

7.5 Wrap Up

7.5.1 Homework

(No additional homework this week.)

7.5.2 Summary

Permutations

Definition 7.9 *A vector with integer components*

$$p = \begin{pmatrix} k_0 \\ k_1 \\ \vdots \\ k_{n-1} \end{pmatrix}$$

is said to be a permutation vector if

- $k_j \in \{0, \ldots, n-1\}$ *, for* $0 \leq j < n$*; and*
- $k_i = k_j$ *implies* $i = j$*.*

In other words, p *is a rearrangement of the numbers* $0, \ldots, n-1$ *(without repetition).*

Definition 7.10 *Let* $p = (k_0, \ldots, k_{n-1})^T$ *be a permutation vector. Then*

$$P = P(p) = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}$$

is said to be a permutation matrix.

Theorem 7.11 *Let* $p = (k_0, \ldots, k_{n-1})^T$ *be a permutation vector. Consider*

$$P = P(p) = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}, \quad x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}, \quad \text{and} \quad A = \begin{pmatrix} a_0^T \\ a_1^T \\ \vdots \\ a_{n-1}^T \end{pmatrix}.$$

Then

$$Px = \begin{pmatrix} \chi_{k_0} \\ \chi_{k_1} \\ \vdots \\ \chi_{k_{n-1}} \end{pmatrix}, \quad \text{and} \quad PA = \begin{pmatrix} a_{k_0}^T \\ a_{k_1}^T \\ \vdots \\ a_{k_{n-1}}^T \end{pmatrix}.$$

Theorem 7.12 *Let* $p = (k_0, \ldots, k_{n-1})^T$ *be a permutation vector. Consider*

$$P = P(p) = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \end{pmatrix}.$$

Then

$$AP^T = \begin{pmatrix} a_{k_0} & a_{k_1} & \cdots & a_{k_{n-1}} \end{pmatrix}.$$

Theorem 7.13 *If* P *is a permutation matrix, so is* P^T*.*

Definition 7.14 *Let us call the special permutation matrix of the form*

$$
\widetilde{P}(\pi) = \left(\begin{array}{c} \boxed{e_\pi^T} \\ e_1^T \\ \vdots \\ e_{\pi-1}^T \\ \boxed{e_0^T} \\ e_{\pi+1}^T \\ \vdots \\ e_{n-1}^T \end{array} \right) = \left(\begin{array}{|c|cccc|c|ccc} \hline 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \hline 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ \hline 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \\ \hline \end{array} \right)
$$

a pivot matrix.

Theorem 7.15 *When $\widetilde{P}(\pi)$ (of appropriate size) multiplies a matrix from the left, it swaps row 0 and row π, leaving all other rows unchanged.*

When $\widetilde{P}(\pi)$ (of appropriate size) multiplies a matrix from the right, it swaps column 0 and column π, leaving all other columns unchanged.

LU with row pivoting

Algorithm: $[A, p] := \text{LU\_PIV}(A, p)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array} \right)$, $p \rightarrow \left(\begin{array}{c} p_T \\ \hline p_B \end{array} \right)$

where A_{TL} is 0×0 and p_T has 0 components

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$
\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array} \right), \left(\begin{array}{c} p_T \\ \hline p_B \end{array} \right) \rightarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ \hline p_2 \end{array} \right)
$$

$\pi_1 = \text{PIVOT}\left(\left(\begin{array}{c} \alpha_{11} \\ \hline a_{21} \end{array} \right) \right)$

$\left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array} \right) := P(\pi_1) \left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array} \right)$

$a_{21} := a_{21} / \alpha_{11}$ (a_{21} now contains l_{21})

$\left(\begin{array}{c} a_{12}^T \\ \hline A_{22} \end{array} \right) = \left(\begin{array}{c} a_{12}^T \\ \hline A_{22} - a_{21} a_{12}^T \end{array} \right)$

Continue with

$$
\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array} \right), \left(\begin{array}{c} p_T \\ \hline p_B \end{array} \right) \leftarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ \hline p_2 \end{array} \right)
$$

endwhile

Algorithm: $b := \text{APPLY\_PIV}(p, b)$

Partition $p \rightarrow \left(\begin{array}{c} p_T \\ \hline p_B \end{array} \right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array} \right)$

where p_T and b_T have 0 components

while $m(b_T) < m(b)$ **do**

Repartition

$$
\left(\begin{array}{c} p_T \\ \hline p_B \end{array} \right) \rightarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ \hline p_2 \end{array} \right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array} \right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array} \right)
$$

$\left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array} \right) := P(\pi_1) \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array} \right)$

Continue with

$$
\left(\begin{array}{c} p_T \\ \hline p_B \end{array} \right) \leftarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ \hline p_2 \end{array} \right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array} \right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array} \right)
$$

endwhile

- LU factorization with row pivoting, starting with a square nonsingular matrix A, computes the LU factorization of a permuted matrix A: $PA = LU$ (via the above algorithm LU_PIV).
- $Ax = b$ then can be solved via the following steps:
  - Update $b := Pb$ (via the above algorithm APPLY_PIV).

- Solve $Lz = b$, overwriting b with z (via the algorithm from 6.3.2).
- Solve $Ux = b$, overwriting b with x (via the algorithm from 6.3.3).

Theorem 7.16 *Let $L : \mathbb{R}^n \to \mathbb{R}^n$ be a linear transformation that is a bijection. Then the inverse function $L^{-1} : \mathbb{R}^n \to \mathbb{R}^n$ exists and is a linear transformation.*

Theorem 7.17 *If A has an inverse, A^{-1}, then A^{-1} is unique.*

Inverses of special matrices

Type	A	A^{-1}
Identity matrix	$I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$	$I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$
Diagonal matrix	$D = \begin{pmatrix} \delta_{0,0} & 0 & \cdots & 0 \\ 0 & \delta_{1,1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1,n-1} \end{pmatrix}$	$D^{-1} = \begin{pmatrix} \delta_{0,0}^{-1} & 0 & \cdots & 0 \\ 0 & \delta_{1,1}^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1,n-1}^{-1} \end{pmatrix}$
Gauss transform	$\widetilde{L} = \left(\begin{array}{c\|c\|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right)$	$\widetilde{L}^{-1} = \left(\begin{array}{c\|c\|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right).$
Permutation matrix	P	P^T
2D Rotation	$R = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$	$R^{-1} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} = R^T$
2D Reflection	A	A
Lower triangular matrix	$L = \left(\begin{array}{c\|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)$	$L^{-1} = \left(\begin{array}{c\|c} L_{00}^{-1} & 0 \\ \hline -\frac{1}{\lambda_{11}} l_{10}^T L_{00}^{-1} & \frac{1}{\lambda_{11}} \end{array}\right)$
Upper triangular matrix	$U = \left(\begin{array}{c\|c} U_{00} & u_{01} \\ \hline 0 & \upsilon_{11} \end{array}\right)$	$U^{-1} = \left(\begin{array}{c\|c} U_{00}^{-1} & -U_{00}^{-1} u_{01}/\upsilon_{11} \\ \hline 0 & \frac{1}{\upsilon_{11}} \end{array}\right)$
General 2×2 matrix	$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}$	$\frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix}$

The following matrices have inverses:

- Triangular matrices that have no zeroes on their diagonal.
- Diagonal matrices that have no zeroes on their diagonal.
  (Notice: this is a special class of triangular matrices!).
- Gauss transforms.

  (In Week 8 we will generalize the notion of a Gauss transform to matrices of the form $\left(\begin{array}{c|c|c} I & u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & 0 \end{array}\right)$.)
- Permutation matrices.

- 2D Rotations.
- 2D Reflections.

General principle

If $A, B \in \mathbb{R}^{n \times n}$ and $AB = I$, then $Ab_j = e_j$, where b_j is the jth column of B and e_j is the jth unit basis vector.

Properties of the inverse

Assume A, B, and C are square matrices that are nonsingular. Then

- $(\alpha B)^{-1} = \frac{1}{\alpha} B^{-1}$.
- $(AB)^{-1} = B^{-1}A^{-1}$.
- $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$.
- $(A^T)^{-1} = (A^{-1})^T$.
- $(A^{-1})^{-1} = A$.

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.

Blank worksheet for pivoting exercises

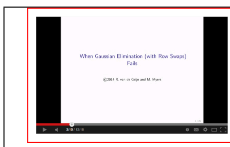
i	L_i	$\tilde{P}$	A	p
0				$\cdot$ $\cdot$
	$\begin{array}{c\|cc} 1 & 0 & 0 \\ \hline & 1 & 0 \\ & 0 & 1 \end{array}$			$\cdot$ $\cdot$
1				$\cdot$
	$\begin{array}{c\|c\|c} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & & 1 \end{array}$			$\cdot$
2				

Week 8

More on Matrix Inversion

8.1 Opening Remarks

8.1.1 When LU Factorization with Row Pivoting Fails



☛ YouTube

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.

Homework 8.1.1.1 Assume that $A, B, C \in \mathbb{R}^{n \times n}$, let $BA = C$, and B be nonsingular.

A is nonsingular if and only if C is nonsingular.

True/False

☛ SEE ANSWER

The reason the above result is important is that we have seen that LU factorization computes a sequence of pivot matrices and Gauss transforms in an effort to transform the matrix into an upper triangular matrix. We know that the permutation matrices and Gauss transforms are all nonsingular since we saw last week that inverses could be constructed. If we now look at under what circumstance LU factorization with row pivoting breaks down, we will see that with the help of the above result we can conclude that the matrix is singular (does not have an inverse).

Let us assume that a number of pivot matrices and Gauss transforms have been successfully computed by LU factorization with partial pivoting:

$$\widetilde{L}_{k-1}P_{k-1}\cdots\widetilde{L}_0 P_0 \widehat{A} = \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & a_{21} & A_{22} \end{array}\right)$$

where $\widehat{A}$ equals the original matrix with which the LU factorization with row pivoting started and the values on the right of $=$ indicate what is currently in matrix A, which has been overwritten. The following picture captures when LU factorization breaks down, for $k = 2$:

$$\overbrace{\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -\times & 1 & 0 & 0 \\ 0 & -\times & 0 & 1 & 0 \\ 0 & -\times & 0 & 0 & 1 \end{pmatrix}}^{\widetilde{L}_1} P_1 \overbrace{\begin{pmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \end{pmatrix}}^{\widetilde{L}_0 P_0 A} = \overbrace{\left(\begin{array}{cc|c|cc} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ \hline 0 & 0 & 0 & \times & \times \\ \hline 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times & \times \end{array}\right)}^{\left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & a_{21} & A_{22} \end{array}\right)}.$$

Here the $\times$s are "representative" elements in the matrix. In other words, if in the current step $\alpha_{11} = 0$ and $a_{21} = 0$ (the zero vector), then no row can be found with which to pivot so that $\alpha_{11} \neq 0$, and the algorithm fails.

Now, repeated application of the insight in the homework tells us that matrix A is nonsingular if and only if the matrix to the right is nonsingular. We recall our list of equivalent conditions:

The following statements are equivalent statements about $A \in \mathbb{R}^{n\times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.

It is the condition "$Ax = 0$ implies that $x = 0$" that we will use. We show that if LU factorization with partial pivoting breaks down, then there is a vector $x \neq 0$ such that $Ax = 0$ for the current (updated) matrix A:

$$\left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & 0 & a_{12}^T \\ \hline 0 & 0 & A_{22} \end{array}\right) \overbrace{\left(\begin{array}{c} -U_{00}^{-1}u_{01} \\ \hline 1 \\ \hline 0 \end{array}\right)}^{x} = \left(\begin{array}{c} -U_{00}U_{00}^{-1}u_{01} + u_{01} \\ \hline 0 \\ \hline 0 \end{array}\right) = \left(\begin{array}{c} 0 \\ \hline 0 \\ \hline 0 \end{array}\right)$$

We conclude that if LU factorization with partial pivoting breaks down, then the original matrix A is not nonsingular. (In other words, it is singular.)

This allows us to add another condition to the list of equivalent conditions:

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.
- LU with partial pivoting does not break down.

8.1.2 Outline

8.1.3 What You Will Learn

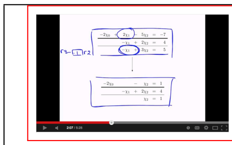
Upon completion of this unit, you should be able to

- Determine with Gaussian elimination (LU factorization) when a system of linear equations with n equations in n unknowns does not have a unique solution.
- Understand and apply Gauss Jordan elimination to solve linear systems with one or more right-hand sides and to find the inverse of a matrix.
- Identify properties that indicate a linear transformation has an inverse.
- Identify properties that indicate a matrix has an inverse.
- Create an algorithm to implement Gauss-Jordan elimination and determine the cost function.
- Recognize and understand that inverting a matrix is not the method of choice for solving a linear system.
- Identify specialized factorizations of matrices with special structure and/or properties and create algorithms that take advantage of this (enrichment).

Track your progress in Appendix B.

8.2 Gauss-Jordan Elimination

8.2.1 Solving $Ax = b$ via Gauss-Jordan Elimination



☛ YouTube

In this unit, we discuss a variant of Gaussian elimination that is often referred to as Gauss-Jordan elimination.

Homework 8.2.1.1 Perform the following steps

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ \hline 2\chi_0 & - & 3\chi_1 & + & 7\chi_2 & = & 11 \\ -4\chi_0 & + & 3\chi_1 & - & 7\chi_2 & = & -9 \end{array} \longrightarrow \begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ \hline & & -\chi_1 & + & 2\chi_2 & = & 4 \\ & & -\chi_1 & + & 3\chi_2 & = & 5 \end{array}$$

one must subtract $\lambda_{1,0} = \square$ times the first row from the second row and subtract $\lambda_{2,0} = \square$ times the first row from the third row.

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ \hline & & -\chi_1 & + & 2\chi_2 & = & 4 \\ \hline & & -\chi_1 & + & 3\chi_2 & = & 5 \end{array} \longrightarrow \begin{array}{rcrcrcr} -2\chi_0 & & & - & \chi_2 & = & 1 \\ \hline & & -\chi_1 & + & 2\chi_2 & = & 4 \\ \hline & & & & \chi_2 & = & 1 \end{array}$$

one must subtract $\upsilon_{0,1} = \square$ times the second row from the first row and subtract $\lambda_{2,1} = \square$ times the second row from the third row.

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} -2\chi_0 & & & - & \chi_2 & = & 1 \\ & & -\chi_1 & + & 2\chi_2 & = & 4 \\ \hline & & & & \chi_2 & = & 1 \\ \hline \end{array} \longrightarrow \begin{array}{rcrcrcr} -2\chi_0 & & & & & = & 2 \\ & & -\chi_1 & & & = & 2 \\ \hline & & & & \chi_2 & = & 1 \\ \hline \end{array}$$

one must subtract $\upsilon_{0,2} = \square$ times the third row from the first row and subtract $\upsilon_{1,2} = \square$ times the third row from the first row.

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} -2\chi_0 & & & & & = & 2 \\ & & -\chi_1 & & & = & 2 \\ & & & & \chi_2 & = & 1 \end{array} \longrightarrow \begin{array}{rcrcrcr} \chi_0 & & & & & = & -1 \\ & & \chi_1 & & & = & -2 \\ & & & & \chi_2 & = & 1 \end{array}$$

one must multiply the first row by $\delta_{0,0} = \square$, the second row by $\delta_{1,1} = \square$, and the third row by $\delta_{2,2} = \square$.

- Use the above exercises to compute the vector x that solves

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ 2\chi_0 & - & 3\chi_1 & + & 7\chi_2 & = & 11 \\ -4\chi_0 & + & 3\chi_1 & - & 7\chi_2 & = & -9 \end{array}$$

☛ SEE ANSWER

Be sure to compare and contrast the above order of eliminating elements in the matrix to what you do with Gaussian elimination.

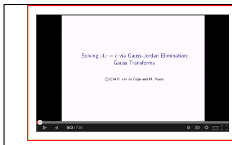
Homework 8.2.1.2 Perform the process illustrated in the last exercise to solve the systems of linear equations

- $$\begin{pmatrix} 3 & 2 & 10 \\ -3 & -3 & -14 \\ 3 & 1 & 3 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -7 \\ 9 \\ -5 \end{pmatrix}$$

- $$\begin{pmatrix} 2 & -3 & 4 \\ 2 & -2 & 3 \\ 6 & -7 & 9 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -8 \\ -5 \\ -17 \end{pmatrix}$$

☛ SEE ANSWER

8.2.2 Solving $Ax = b$ via Gauss-Jordan Elimination: Gauss Transforms



☛ YouTube

We again discuss Gauss-Jordan elimination, but now with an appended system.

Homework 8.2.2.1 Evaluate

- $$\left(\begin{array}{r|rr} 1 & 0 & 0 \\ \hline 1 & 1 & 0 \\ -2 & 0 & 1 \end{array}\right)\left(\begin{array}{r|rr||r} -2 & 2 & -5 & -7 \\ \hline 2 & -3 & 7 & 11 \\ -4 & 3 & -7 & -9 \end{array}\right) =$$

- $$\left(\begin{array}{r|r|r} 1 & 2 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1 & 1 \end{array}\right)\left(\begin{array}{r|r|r||r} -2 & 2 & -5 & -7 \\ \hline 0 & -1 & 2 & 4 \\ \hline 0 & -1 & 3 & 5 \end{array}\right) =$$

- $$\left(\begin{array}{rr|r} 1 & 0 & 1 \\ 0 & 1 & -2 \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{rr|r||r} -2 & 0 & -1 & 1 \\ 0 & -1 & 2 & 4 \\ \hline 0 & 0 & 1 & 1 \end{array}\right) =$$

- $$\left(\begin{array}{rrr} -\frac{1}{2} & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{array}\right)\left(\begin{array}{rrr||r} -2 & 0 & 0 & 2 \\ 0 & -1 & 0 & 2 \\ 0 & 0 & 1 & 1 \end{array}\right) =$$

- Use the above exercises to compute $x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$ that solves

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ 2\chi_0 & - & 3\chi_1 & + & 7\chi_2 & = & 11 \\ -4\chi_0 & + & 3\chi_1 & - & 7\chi_2 & = & -9 \end{array}$$

☛ SEE ANSWER

Homework 8.2.2.2 This exercise shows you how to use MATLAB to do the heavy lifting for Homework 8.2.2.1. Again solve

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ 2\chi_0 & - & 3\chi_1 & + & 7\chi_2 & = & 11 \\ -4\chi_0 & + & 3\chi_1 & - & 7\chi_2 & = & -9 \end{array}$$

via Gauss-Jordan elimination. This time we set this up as an appended matrix:

$$\left(\begin{array}{rrr|r} -2 & 2 & -5 & -7 \\ 2 & -3 & 7 & 11 \\ -4 & 3 & -7 & -9 \end{array} \right).$$

We can enter this into MATLAB as

```
A = [
-2  2 -5  ??
 2 -3  7  ??
-4  3 -7  ??
]
```

(You enter `??`.) Create the Gauss transform, G_0, that zeroes the entries in the first column below the diagonal:

```
G0 = [
 1  0  0
??  1  0
??  0  1
]
```

(You fill in the `??`). Now apply the Gauss transform to the appended system:

```
A0 = G0 * A
```

Similarly create G_1,

```
G1 = [
 1 ??  0
 0  1  0
 0 ??  1
]
```

A_1, G_2, and A_2, where A_2 equals the appended system that has been transformed into a diagonal system. Finally, let D equal to a diagonal matrix so that $A_3 = D * A2$ has the identity for the first three columns.
You can then find the solution to the linear system in the last column.

☛ SEE ANSWER

Homework 8.2.2.3 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array} \right) \left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & b_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21} & A_{22} & b_2 \end{array} \right) = \left(\begin{array}{c|c|c||c} D_{00} & a_{01} - \alpha_{11} u_{01} & A_{02} - u_{01} a_{12}^T & b_0 - \beta_1 u_{01} \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21} - \alpha_{11} l_{21} & A_{22} - l_{21} a_{12}^T & b_2 - \beta_1 l_{21} \end{array} \right)$$

Always/Sometimes/Never

☛ SEE ANSWER

Homework 8.2.2.4 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$; and
- $l_{21} := a_{21}/\alpha_{11}$.

Consider the following expression:

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & b_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21} & A_{22} & b_2 \end{array}\right) = \left(\begin{array}{c|c|c||c} D_{00} & 0 & A_{02} - u_{01}a_{12}^T & b_0 - \beta_1 u_{01} \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & 0 & A_{22} - l_{21}a_{12}^T & b_2 - \beta_1 l_{21} \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER

The above exercises showcase a variant on Gauss transforms that not only take multiples of a current row and add or subtract these from the rows below the current row, but also take multiples of the current row and add or subtract these from the rows above the current row:

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c} A_0 \\ \hline a_1^T \\ \hline A_2 \end{array}\right) = \left(\begin{array}{c} A_0 - u_{01}a_1^T \\ \hline a_1^T \\ \hline A_2 - l_{21}a_1^T \end{array}\right) \begin{array}{l} \longleftarrow \text{ Subtract multiples of } a_1^T \text{ from the rows above } a_1^T \\ \leftarrow \text{ Leave } a_1^T \text{ alone} \\ \longleftarrow \text{ Subtract multiples of } a_1^T \text{ from the rows below } a_1^T \end{array}$$

The discussion in this unit motivates the algorithm GAUSSJORDAN_PART1 in Figure 8.1, which transforms A to a diagonal matrix and updates the right-hand side accordingly, and GAUSSJORDAN_PART2 in Figure 8.2, which transforms the diagonal matrix A to an identity matrix and updates the right-hand side accordingly. The two algorithms together leave A overwritten with the identity and the vector to the right of the double lines with the solution to $Ax = b$.

The reason why we split the process into two parts is that it is easy to create problems for which only integers are encountered during the first part (while matrix A is being transformed into a diagonal). This will make things easier for us when we extend this process so that it computes the inverse of matrix A: fractions only come into play during the second, much simpler, part.

Algorithm: $[A,b] := \text{GAUSSJORDAN\_PART1}(A,b)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where A_{TL} is 0×0, b_T has 0 rows

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$a_{01} := a_{01}/\alpha_{11} \quad (= u_{01})$

$a_{21} := a_{21}/\alpha_{11} \quad (= l_{21})$

$A_{02} := A_{02} - a_{01}a_{12}^T \quad (= A_{02} - u_{01}a_{12}^T)$

$A_{22} := A_{22} - a_{21}a_{12}^T \quad (= A_{22} - l_{21}a_{12}^T)$

$b_0 := b_0 - \beta_1 a_{01} \quad (= b_2 - \beta_1 u_{01})$

$b_2 := b_2 - \beta_1 a_{21} \quad (= b_2 - \beta_1 l_{21})$

$a_{01} := 0$ (zero vector)

$a_{21} := 0$ (zero vector)

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 8.1: Algorithm that transforms matrix A to a diagonal matrix and updates the right-hand side accordingly.

Algorithm: $[A,b] := \textsc{GaussJordan\_Part2}(A,b)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where A_{TL} is 0×0, b_T has 0 rows

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$\beta_1 := \beta_1 / \alpha_{11}$

$\alpha_{11} := 1$

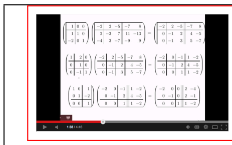
Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 8.2: Algorithm that transforms diagonal matrix A to an identity matrix and updates the right-hand side accordingly.

8.2.3 Solving $Ax = b$ via Gauss-Jordan Elimination: Multiple Right-Hand Sides



☛ YouTube

Homework 8.2.3.1 Evaluate

- $$\left(\begin{array}{|c|cc} 1 & 0 & 0 \\ \hline 1 & 1 & 0 \\ -2 & 0 & 1 \end{array}\right)\left(\begin{array}{|c|cc||cc} -2 & 2 & -5 & -7 & 8 \\ \hline 2 & -3 & 7 & 11 & -13 \\ -4 & 3 & -7 & -9 & 9 \end{array}\right) = \left(\begin{array}{|c|cc||cc} -2 & 2 & -5 & -7 & \square \\ \hline 0 & -1 & 2 & 4 & \square \\ 0 & -1 & 3 & 5 & \square \end{array}\right)$$

- $$\left(\begin{array}{c|c|c} 1 & 2 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1 & 1 \end{array}\right)\left(\begin{array}{c|c|c||cc} -2 & 2 & -5 & -7 & 8 \\ \hline 0 & -1 & 2 & 4 & -5 \\ \hline 0 & -1 & 3 & 5 & -7 \end{array}\right) = \left(\begin{array}{c|c|c||cc} -2 & 0 & -1 & 1 & \square \\ \hline 0 & -1 & 2 & 4 & \square \\ \hline 0 & 0 & 1 & 1 & \square \end{array}\right)$$

- $$\left(\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & 1 & -2 \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{cc|c||cc} -2 & 0 & -1 & 1 & -2 \\ 0 & -1 & 2 & 4 & -5 \\ \hline 0 & 0 & 1 & 1 & -2 \end{array}\right) = \left(\begin{array}{cc|c||cc} -2 & 0 & 0 & 2 & \square \\ 0 & -1 & 0 & 2 & \square \\ \hline 0 & 0 & 1 & 1 & \square \end{array}\right)$$

- $$\left(\begin{array}{ccc} -\frac{1}{2} & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{array}\right)\left(\begin{array}{ccc||cc} -2 & 0 & 0 & 2 & -4 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & 1 & 1 & -2 \end{array}\right) = \left(\begin{array}{ccc||cc} 1 & 0 & 0 & -1 & \square \\ 0 & 1 & 0 & -2 & \square \\ 0 & 0 & 1 & 1 & \square \end{array}\right)$$

- Use the above exercises to compute $x_0 = \begin{pmatrix} \chi_{00} \\ \chi_{10} \\ \chi_{20} \end{pmatrix}$ and $x_1 = \begin{pmatrix} \chi_{01} \\ \chi_{11} \\ \chi_{21} \end{pmatrix}$ that solve

$$\begin{array}{rcrcrcr} -2\chi_{00} & + & 2\chi_{10} & - & 5\chi_{20} & = & -7 \\ 2\chi_{00} & - & 3\chi_{10} & + & 7\chi_{20} & = & 11 \\ -4\chi_{00} & + & 3\chi_{10} & - & 7\chi_{20} & = & -9 \end{array} \quad \text{and} \quad \begin{array}{rcrcrcr} -2\chi_{01} & + & 2\chi_{11} & - & 5\chi_{21} & = & 8 \\ 2\chi_{01} & - & 3\chi_{11} & + & 7\chi_{21} & = & -13 \\ -4\chi_{01} & + & 3\chi_{11} & - & 7\chi_{21} & = & 9 \end{array}$$

☛ SEE ANSWER

Homework 8.2.3.2 This exercise shows you how to use MATLAB to do the heavy lifting for Homework 8.2.3.1. Start with the appended system:

$$\left(\begin{array}{rrr|rr} -2 & 2 & -5 & -7 & 8 \\ 2 & -3 & 7 & 11 & -13 \\ -4 & 3 & -7 & -9 & 9 \end{array}\right)$$

Enter this into MATLAB as

```
A = [
-2  2 -5  ?? ??
 2 -3  7  ?? ??
-4  3 -7  ?? ??
]
```

(You enter `??`.) Create the Gauss transform, G_0, that zeroes the entries in the first column below the diagonal:

```
G0 = [
 1  0  0
??  1  0
??  0  1
]
```

(You fill in the `??`). Now apply the Gauss transform to the appended system:

```
A0 = G0 * A
```

Similarly create G_1,

```
G1 = [
 1 ??  0
 0  1  0
 0 ??  1
]
```

A_1, G_2, and A_2, where A_2 equals the appended system that has been transformed into a diagonal system. Finally, let D equal to a diagonal matrix so that $A_3 = D * A2$ has the identity for the first three columns.
You can then find the solutions to the linear systems in the last column.

☛ SEE ANSWER

Homework 8.2.3.3 Evaluate

- $$\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline \Box & 1 & 0 \\ \Box & 0 & 1 \end{array}\right)\left(\begin{array}{c|cc||cc} 3 & 2 & 10 & -7 & 16 \\ \hline -3 & -3 & -14 & 9 & -25 \\ 3 & 1 & 4 & -5 & 3 \end{array}\right) = \left(\begin{array}{c|cc||cc} 3 & 2 & 10 & \Box & \Box \\ \hline 0 & -1 & -4 & \Box & \Box \\ 0 & -1 & -6 & \Box & \Box \end{array}\right)$$

- $$\left(\begin{array}{c|c|c} 1 & \Box & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & \Box & 1 \end{array}\right)\left(\begin{array}{c|c|c||cc} 3 & 2 & 10 & -7 & 16 \\ \hline 0 & -1 & -4 & 2 & -9 \\ \hline 0 & -1 & -6 & 2 & -13 \end{array}\right) = \left(\begin{array}{c|c|c||cc} 3 & 0 & 2 & \Box & \Box \\ \hline 0 & -1 & -4 & \Box & \Box \\ \hline 0 & 0 & -2 & \Box & \Box \end{array}\right)$$

- $$\left(\begin{array}{cc|c} 1 & 0 & \Box \\ 0 & 1 & \Box \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{cc|c||cc} 3 & 0 & 2 & -3 & -2 \\ 0 & -1 & -4 & 2 & -9 \\ \hline 0 & 0 & -2 & 0 & -4 \end{array}\right) = \left(\begin{array}{cc|c||cc} 3 & 0 & 0 & \Box & \Box \\ 0 & -1 & 0 & \Box & \Box \\ \hline 0 & 0 & -2 & \Box & \Box \end{array}\right)$$

- $$\begin{pmatrix} \Box & 0 & 0 \\ 0 & \Box & 0 \\ 0 & 0 & \Box \end{pmatrix}\left(\begin{array}{ccc||cc} 3 & 0 & 0 & -3 & -6 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & -2 & 0 & -4 \end{array}\right) = \left(\begin{array}{ccc||cc} 1 & 0 & 0 & \Box & \Box \\ 0 & 1 & 0 & \Box & \Box \\ 0 & 0 & 1 & \Box & \Box \end{array}\right)$$

Use the above exercises to compute $x_0 = \begin{pmatrix} \chi_{0,0} \\ \chi_{1,0} \\ \chi_{2,0} \end{pmatrix}$ and $x_1 = \begin{pmatrix} \chi_{0,1} \\ \chi_{1,1} \\ \chi_{2,1} \end{pmatrix}$ that solve

$$\begin{array}{rcrcrcr} 3\chi_{0,0} & + & 2\chi_{1,0} & + & 10\chi_{2,0} & = & -7 \\ -3\chi_{0,0} & - & 3\chi_{1,0} & - & 14\chi_{2,0} & = & 9 \\ 3\chi_{0,0} & + & 1\chi_{1,0} & + & 4\chi_{2,0} & = & -5 \end{array} \quad \text{and} \quad \begin{array}{rcrcrcr} 3\chi_{0,0} & + & 2\chi_{1,0} & + & 10\chi_{2,0} & = & 16 \\ -3\chi_{0,0} & - & 3\chi_{1,0} & - & 14\chi_{2,0} & = & -25 \\ 3\chi_{0,0} & + & 1\chi_{1,0} & + & 4\chi_{2,0} & = & 3 \end{array}$$

(You could use MATLAB to do the heavy lifting, like in the last homework...)

☛ SEE ANSWER

Homework 8.2.3.4 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & B_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_1^T \\ \hline 0 & a_{21} & A_{22} & B_2 \end{array}\right) = \left(\begin{array}{c|c|c||c} D_{00} & a_{01}-\alpha_{11}u_{01} & A_{02}-u_{01}a_{12}^T & B_0-u_{01}b_1^T \\ \hline 0 & \alpha_{11} & a_{12}^T & b_1^T \\ \hline 0 & a_{21}-\alpha_{11}l_{21} & A_{22}-l_{21}a_{12}^T & B_2-l_{21}b_1^T \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER

Algorithm: $[A,B] := \text{GAUSSJORDAN\_MRHS\_PART1}(A,B)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right)$
  where A_{TL} is 0×0, B_T has 0 rows
while $m(A_{TL}) < m(A)$ **do**
  Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array}\right)$$

$$\begin{array}{ll} a_{01} := a_{01}/\alpha_{11} & (= u_{01}) \\ a_{21} := a_{21}/\alpha_{11} & (= l_{21}) \\ A_{02} := A_{02} - a_{01}a_{12}^T & (= A_{02} - u_{01}a_{12}^T) \\ A_{22} := A_{22} - a_{21}a_{12}^T & (= A_{22} - l_{21}a_{12}^T) \\ B_0 := B_0 - a_{01}b_1^T & (= B_0 - u_{01}b_1^T) \\ B_2 := B_2 - a_{21}b_1^T & (= B_2 - l_{21}b_1^T) \\ a_{01} := 0 \quad \text{(zero vector)} & \\ a_{21} := 0 \quad \text{(zero vector)} & \end{array}$$

  Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array}\right)$$

endwhile

Figure 8.3: Algorithm that transforms diagonal matrix A to an identity matrix and updates a matrix B with multiple right-hand sides accordingly.

Homework 8.2.3.5 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$; and
- $l_{21} := a_{21}/\alpha_{11}$.

The following expression holds:

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right) \left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & b_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21} & A_{22} & b_2 \end{array}\right) = \left(\begin{array}{c|c|c||c} D_{00} & 0 & A_{02} - u_{01}a_{12}^T & B_0 - u_{01}b_1^T \\ \hline 0 & \alpha_{11} & a_{12}^T & b_1^T \\ \hline 0 & 0 & A_{22} - l_{21}a_{12}^T & B_2 - l_{21}b_1^T \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER

The above observations justify the two algorithms in Figures 8.3 and 8.4 for "Gauss-Jordan elimination" that work with

Algorithm: $[A,B] := \text{GAUSSJORDAN\_MRHS\_PART2}(A,B)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right)$

where A_{TL} is 0×0, B_T has 0 rows

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array}\right)$$

$b_1^T := (1/\alpha_{11}) b_1^T$

$\alpha_{11} := 1$

Continue with

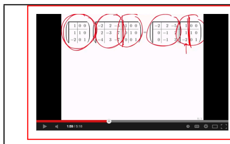
$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ \hline B_2 \end{array}\right)$$

endwhile

Figure 8.4: Algorithm that transforms diagonal matrix A to an identity matrix and updates a matrix B with multiple right-hand sides accordingly.

"multiple right-hand sides" (viewed as the columns of matrix B).

8.2.4 Computing A^{-1} via Gauss-Jordan Elimination



☛ YouTube

Recall the following observation about the inverse of matrix A. If we let X equal the inverse of A, then

$$AX = I$$

or

$$A \left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} e_0 & e_1 & \cdots & e_{n-1} \end{array} \right),$$

so that $Ax_j = e_j$. In other words, the jth column of $X = A^{-1}$ can be computed by solving $Ax = e_j$. Clearly, we can use the routine that performs Gauss-Jordan with the appended system $\left(\begin{array}{c||c} A & B \end{array} \right)$ to compute A^{-1} by feeding it $B = I$!

Homework 8.2.4.1 Evaluate

- $$\left(\begin{array}{|c|cc}\hline 1 & 0 & 0 \\ \hline 1 & 1 & 0 \\ -2 & 0 & 1\end{array}\right)\left(\begin{array}{|c|cc||c|cc}\hline -2 & 2 & -5 & 1 & 0 & 0 \\ \hline 2 & -3 & 7 & 0 & 1 & 0 \\ -4 & 3 & -7 & 0 & 0 & 1\end{array}\right)=\left(\begin{array}{|c|cc||c|cc}\hline -2 & 2 & -5 & \Box & \Box & \Box \\ \hline 0 & -1 & 2 & \Box & \Box & \Box \\ 0 & -1 & 3 & \Box & \Box & \Box\end{array}\right)$$

- $$\left(\begin{array}{c|c|c} 1 & 2 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1 & 1\end{array}\right)\left(\begin{array}{c|c|c||c|c|c} -2 & 2 & -5 & 1 & 0 & 0 \\ \hline 0 & -1 & 2 & 1 & 1 & 0 \\ \hline 0 & -1 & 3 & -2 & 0 & 1\end{array}\right)=\left(\begin{array}{c|c|c||c|c|c} -2 & 0 & -1 & \Box & \Box & \Box \\ \hline 0 & -1 & 2 & \Box & \Box & \Box \\ \hline 0 & 0 & 1 & \Box & \Box & \Box\end{array}\right)$$

- $$\left(\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & 1 & -2 \\ \hline 0 & 0 & 1\end{array}\right)\left(\begin{array}{cc|c||cc|c} -2 & 0 & -1 & 3 & 2 & 0 \\ 0 & -1 & 2 & 1 & 1 & 0 \\ \hline 0 & 0 & 1 & -3 & -1 & 1\end{array}\right)=\left(\begin{array}{cc|c||cc|c} -2 & 0 & 0 & \Box & \Box & \Box \\ 0 & -1 & 0 & \Box & \Box & \Box \\ \hline 0 & 0 & 1 & \Box & \Box & \Box\end{array}\right)$$

- $$\left(\begin{array}{ccc} -\frac{1}{2} & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1\end{array}\right)\left(\begin{array}{ccc||ccc} -2 & 0 & 0 & 0 & 1 & 1 \\ 0 & -1 & 0 & 7 & 3 & -2 \\ 0 & 0 & 1 & -3 & -1 & 1\end{array}\right)=\left(\begin{array}{ccc||ccc} 1 & 0 & 0 & \Box & \Box & \Box \\ 0 & 1 & 0 & \Box & \Box & \Box \\ 0 & 0 & 1 & \Box & \Box & \Box\end{array}\right)$$

- $$\left(\begin{array}{ccc} -2 & 2 & -5 \\ 2 & -3 & 7 \\ -4 & 3 & -7\end{array}\right)\left(\begin{array}{ccc} 0 & -\frac{1}{2} & -\frac{1}{2} \\ -7 & -3 & 2 \\ -3 & -1 & 1\end{array}\right)=$$

☛ SEE ANSWER

Homework 8.2.4.2 In this exercise, you will use MATLAB to compute the inverse of a matrix using the techniques discussed in this unit.

Initialize	`A = [ -2  2 -5`
	`       2 -3  7`
	`      -4  3 -7 ]`

Create an appended matrix by appending the identity	`A_appended = [ A    eye(size(A)) ]`

Create the first Gauss transform to introduce zeros in the first column (fill in the `?`s).	`G0 = [ 1  0  0`
	`       ?  1  0`
	`       ?  0  1 ]`

Apply the Gauss transform to the appended system	`A0 = G0 * A_appended`

Create the second Gauss transform to introduce zeros in the second column	`G1 = [ 1  ?  0`
	`       0  1  0`
	`       0  ?  1 ]`

Apply the Gauss transform to the appended system	`A1 = G1 * A0`

Create the third Gauss transform to introduce zeros in the third column	`G2 = [ 1  0  ?`
	`       0  1  ?`
	`       0  0  1 ]`

Apply the Gauss transform to the appended system	`A2 = G2 * A1`

Create a diagonal matrix to set the diagonal elements to one	`D3 = [ -1/2  0  0`
	`         0  -1  0`
	`         0   0  1 ]`

Apply the diagonal matrix to the appended system	`A3 = D3 * A2`
Extract the (updated) appended columns	`Ainv = A3( :, 4:6 )`
Check that the inverse was computed	`A * Ainv`

The result should be a 3×3 identity matrix.

☛ SEE ANSWER

Homework 8.2.4.3 Compute

- $\begin{pmatrix} 3 & 2 & 9 \\ -3 & -3 & -14 \\ 3 & 1 & 3 \end{pmatrix}^{-1} =$

- $\begin{pmatrix} 2 & -3 & 4 \\ 2 & -2 & 3 \\ 6 & -7 & 9 \end{pmatrix}^{-1} =$

☛ SEE ANSWER

Homework 8.2.4.4 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} D_{00} & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21} & A_{22} & B_{20} & 0 & I \end{array}\right)$$
$$= \left(\begin{array}{c|c|c||c|c|c} D_{00} & a_{01}-\alpha_{11}u_{01} & A_{02}-u_{01}a_{12}^T & B_{00}-u_{01}b_{10}^T & -u_{01} & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21}-\alpha_{11}l_{21} & A_{22}-l_{21}a_{12}^T & B_{20}-l_{21}b_{10}^T & -l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER

Homework 8.2.4.5 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$; and
- $l_{21} := a_{21}/\alpha_{11}$.

Consider the following expression:

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} D_{00} & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21} & A_{22} & B_{20} & 0 & I \end{array}\right)$$
$$= \left(\begin{array}{c|c|c||c|c|c} D_{00} & 0 & A_{02}-u_{01}a_{12}^T & B_{00}-u_{01}b_{10}^T & -u_{01} & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & 0 & A_{22}-l_{21}a_{12}^T & B_{20}-l_{21}b_{10}^T & -l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER

The above observations justify the two algorithms in Figures 8.5 and 8.6 for "Gauss-Jordan elimination" for inverting a matrix.

Algorithm: $[A,B] := \text{GJ\_INVERSE\_PART1}(A,B)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), B \rightarrow \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right)$

where A_{TL} is 0×0, B_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

where α_{11} is 1×1, β_{11} is 1×1

$$\left(\begin{array}{c|c|c} & a_{01} := a_{01}/\alpha_{11} & A_{02} := A_{02} - a_{01}a_{12}^T \\ \hline & & \\ \hline & a_{21} := a_{21}/\alpha_{11} & A_{22} := A_{22} - a_{21}a_{12}^T \end{array}\right) \quad \left(\begin{array}{c|c|c} B_{00} := B_{00} - a_{01}b_{10}^T & b_{01} := -a_{01} & \\ \hline & & \\ \hline B_{20} := B_{20} - a_{21}b_{10}^T & b_{21} := -a_{21} & \end{array}\right)$$

(Note: a_{01} and a_{21} on the left need to be updated first.)

$a_{01} := 0$ (zero vector)

$a_{21} := 0$ (zero vector)

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

endwhile

Figure 8.5: Algorithm that transforms diagonal matrix A to an identity matrix and updates an identity matrix stored in B accordingly.

Algorithm: $[A,B] := \text{GJ\_INVERSE\_PART2}(A,B)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), B \rightarrow \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right)$

  where A_{TL} is 0×0, B_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

  where α_{11} is 1×1, β_{11} is 1×1

$b_{10}^T := b_{10}^T / \alpha_{11}$

$\beta_{11} := \beta_{11} / \alpha_{11}$

$b_{12}^T := b_{12}^T / \alpha_{11}$

$\alpha_{11} := 1$

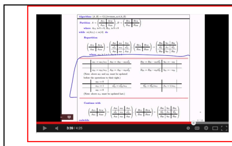
  Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

endwhile

Figure 8.6: Algorithm that transforms diagonal matrix A to an identity matrix and updates an identity matrix stored in B accordingly.

8.2.5 Computing A^{-1} via Gauss-Jordan Elimination, Alternative



☛ YouTube

We now motivate a slight alternative to the Gauss Jordan method, which is easiest to program.

Homework 8.2.5.1

- Determine $\delta_{0,0}$, $\lambda_{1,0}$, $\lambda_{2,0}$ so that

$$\left(\begin{array}{c|cc} \delta_{0,0} & 0 & 0 \\ \hline \lambda_{1,0} & 1 & 0 \\ \lambda_{2,0} & 0 & 1 \end{array}\right)\left(\begin{array}{c|cc||c|cc} -1 & -4 & -2 & 1 & 0 & 0 \\ \hline 2 & 6 & 2 & 0 & 1 & 0 \\ -1 & 0 & 3 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{c|cc||c|cc} 1 & 4 & 2 & -1 & 0 & 0 \\ \hline 0 & -2 & -2 & 2 & 1 & 0 \\ 0 & 4 & 5 & -1 & 0 & 1 \end{array}\right)$$

- Determine $\upsilon_{0,1}$, $\delta_{1,1}$, and $\lambda_{2,1}$ so that

$$\left(\begin{array}{c|c|c} 1 & \upsilon_{0,1} & 0 \\ \hline 0 & \delta_{1,1} & 0 \\ \hline 0 & \lambda_{2,1} & 1 \end{array}\right)\left(\begin{array}{c|cc||c|c|c} 1 & 4 & 2 & -1 & 0 & 0 \\ \hline 0 & -2 & -2 & 2 & 1 & 0 \\ \hline 0 & 4 & 5 & -1 & 0 & 1 \end{array}\right) = \left(\begin{array}{c|c|c||c|c|c} 1 & 0 & -2 & 3 & 2 & 0 \\ \hline 0 & 1 & 1 & -1 & -\frac{1}{2} & 0 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right)$$

- Determine $\upsilon_{0,2}$, $\upsilon_{0,2}$, and $\delta_{2,2}$ so that

$$\left(\begin{array}{cc|c} 1 & 0 & \upsilon_{0,2} \\ 0 & 1 & \upsilon_{1,2} \\ \hline 0 & 0 & \delta_{2,2} \end{array}\right)\left(\begin{array}{cc|c||cc|c} 1 & 0 & -2 & 3 & 2 & 0 \\ 0 & 1 & 1 & -1 & -\frac{1}{2} & 0 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right) = \left(\begin{array}{cc|c||cc|c} 1 & 0 & 0 & 9 & 6 & 2 \\ 0 & 1 & 0 & -4 & -\frac{5}{2} & -1 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right)$$

- Evaluate

$$\left(\begin{array}{ccc} -1 & -4 & -2 \\ 2 & 6 & 2 \\ -1 & 0 & 3 \end{array}\right)\left(\begin{array}{ccc} 9 & 6 & 2 \\ -4 & -\frac{5}{2} & -1 \\ 3 & 2 & 1 \end{array}\right) =$$

☛ SEE ANSWER

Homework 8.2.5.2 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & \delta_{11} & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} I & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21} & A_{22} & B_{20} & 0 & I \end{array}\right)$$

$$= \left(\begin{array}{c|c|c||c|c|c} I & a_{01}-\alpha_{11}u_{01} & A_{02}-u_{01}a_{12}^T & B_{00}-u_{01}b_{10}^T & -u_{01} & 0 \\ \hline 0 & \delta_{11}\alpha_{11} & \delta_{11}a_{12}^T & \delta_{11}b_{10}^T & \delta_{11} & 0 \\ \hline 0 & a_{21}-\alpha_{11}l_{21} & A_{22}-l_{21}a_{12}^T & B_{20}-l_{21}b_{10}^T & -l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER

Homework 8.2.5.3 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$;
- $l_{21} := a_{21}/\alpha_{11}$; and
- $\delta_{11} := 1/\alpha_{11}$.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & \delta_{11} & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} I & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21} & A_{22} & B_{20} & 0 & I \end{array}\right)$$
$$= \left(\begin{array}{c|c|c||c|c|c} I & 0 & A_{02}-u_{01}a_{12}^T & B_{00}-u_{01}b_{10}^T & -u_{01} & 0 \\ \hline 0 & 1 & a_{12}^T/\alpha_{11} & b_{10}^T/\alpha_{11} & 1/\alpha_{11} & 0 \\ \hline 0 & 0 & A_{22}-l_{21}a_{12}^T & B_{20}-l_{21}b_{10}^T & -l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

☛ SEE ANSWER

The last homework motivates the algorithm in Figure 8.7

Homework 8.2.5.4 Implement the algorithm in Figure 8.7 yielding the function

- `[ A_out ] = GJ_Inverse_alt_unb( A, B )`. Assume that it is called as

```
Ainv = GJ_Inverse_alt_unb( A, B )
```

  Matrices A and B must be square and of the same size.

Check that it computes correctly with the script

- `test_GJ_Inverse_alt_unb.m`.

☛ SEE ANSWER

Homework 8.2.5.5 If you are very careful, you can overwrite matrix A with its inverse without requiring the matrix B.
Modify the algorithm in Figure 8.7 so that it overwrites A with its inverse without the use of matrix B yielding the function

- `[ A_out ] = GJ_Inverse_inplace_unb( A )`.

Check that it computes correctly with the script

- `test_GJ_Inverse_inplace_unb.m`.

☛ SEE ANSWER

Algorithm: $[B] := \text{GJ\_INVERSE\_ALT}(A, B)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), B \rightarrow \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right)$

where A_{TL} is 0×0, B_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

where α_{11} is 1×1, β_{11} is 1×1

$$\left(\begin{array}{c|c|c} & a_{01} := a_{01}/\alpha_{11} & A_{02} := A_{02} - a_{01}a_{12}^T \\ \hline & & \\ \hline & a_{21} := a_{21}/\alpha_{11} & A_{22} := A_{22} - a_{21}a_{12}^T \end{array}\right) \quad \left(\begin{array}{c|c|c} B_{00} := B_{00} - a_{01}b_{10}^T & b_{01} := -a_{01} & \\ \hline & & \\ \hline B_{20} := B_{20} - a_{21}b_{10}^T & b_{21} := -a_{21} & \end{array}\right)$$

(Note: above a_{01} and a_{21} must be updated before the operations to their right.)

$$\left(\begin{array}{c|c|c} & a_{01} := 0 & \\ \hline & \alpha_{11} := 1 & a_{12}^T := a_{12}^T/\alpha_{11} \\ \hline & a_{21} := 0 & \end{array}\right) \quad \left(\begin{array}{c|c|c} & & \\ \hline b_{10}^T := b_{10}^T/\alpha_{11} & \beta_{11} = 1/\alpha_{11} & \\ \hline & & \end{array}\right)$$

(Note: above α_{11} must be updated last.)

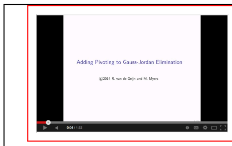
Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

endwhile

Figure 8.7: Algorithm that simultaneously transforms matrix A to an identity and matrix B from the identity to A^{-1}.

8.2.6 Pivoting

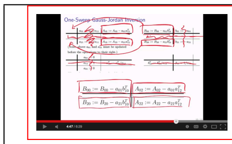


☛ YouTube

Adding pivoting to any of the discussed Gauss-Jordan methods is straight forward. It is a matter of recognizing that if a zero is found on the diagonal during the process at a point where a divide by zero will happen, one will need to swap the current row with another row below it to overcome this. If such a row cannot be found, then the matrix does not have an inverse.

We do not further discuss this in this course.

8.2.7 Cost of Matrix Inversion



☛ YouTube

Let us now discuss the cost of matrix inversion via various methods. In our discussion, we will ignore pivoting. In other words, we will assume that no zero pivot is encountered. We wil start with an $n \times n$ matrix A.

A very naive approach

Here is a very naive approach. Let X be the matrix in which we will compute the inverse. We have argued several times that $AX = I$ means that

$$A \left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} e_0 & e_1 & \cdots & e_{n-1} \end{array} \right)$$

so that $Ax_j = e_j$. So, *for each column* x_j, we can perform the operations

- Compute the LU factorization of A so that $A = LU$. We argued in Week 6 that the cost of this is approximately $\frac{2}{3}n^3$ flops.
- Solve $Lz = e_j$. This is a lower (unit) triangular solve with cost of approximately n^2 flops.
- Solve $Ux_j = z$. This is an upper triangular solve with cost of approximately n^2 flops.

So, for *each* column of X the cost is approximately $\frac{2}{3}n^3 + n^2 + n^2 = \frac{2}{3}n^3 + 2n^2$. There are n columns of X to be computed for a total cost of approximately

$$n(\frac{2}{3}n^3 + 2n^2) = \frac{2}{3}n^4 + 2n^3 \text{ flops.}$$

To put this in perspective: A relatively small problem to be solved on a current supercomputer involves a $100,000 \times 100,000$ matrix. The fastest current computer can perform approximately 55,000 Teraflops, meaning 55×10^{15} floating point operations per second. On this machine, inverting such a matrix would require approximately a third of an hour of compute time.

(Note: such a supercomputer would not attain the stated peak performance. But let's ignore that in our discussions.)

A less naive approach

The problem with the above approach is that A is redundantly factored into L and U for every column of X. Clearly, we only need to do that once. Thus, a less naive approach is given by

- Compute the LU factorization of A so that $A = LU$ at a cost of approximately $\frac{2}{3}n^3$ flops.
- For each column x_j
  - Solve $Lz = e_j$. This is a lower (unit) triangular solve with cost of approximately n^2 flops.
  - Solve $Ux_j = z$. This is an upper triangular solve with cost of approximately n^2 flops.

There are n columns of X to be computed for a total cost of approximately

$$n(n^2 + n^2) = 2n^3 \text{ flops.}$$

Thus, the total cost is now approximately

$$\frac{2}{3}n^3 + 2n^3 = \frac{8}{3}n^3 \text{ flops.}$$

Returning to our relatively small problem of inverting a $100,000 \times 100,000$ matrix on the fastest current computer that can perform approximately 55,000 Teraflops, inverting such a matrix with this alternative approach would require approximately 0.05 seconds. Clearly an improvement.

The cost of the discussed Gauss-Jordan matrix inversion

Now let's consider the Gauss-Jordan matrix inversion algorithm that we developed in the last unit:

Algorithm: $[B] := \text{GJ\_INVERSE\_ALT}(A, B)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), B \rightarrow \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right)$

where A_{TL} is 0×0, B_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

where α_{11} is 1×1, β_{11} is 1×1

$$\begin{array}{c|c|c} & a_{01} := a_{01}/\alpha_{11} & A_{02} := A_{02} - a_{01}a_{12}^T \\ \hline & & \\ \hline & a_{21} := a_{21}/\alpha_{11} & A_{22} := A_{22} - a_{21}a_{12}^T \end{array} \qquad \begin{array}{c|c|c} B_{00} := B_{00} - a_{01}b_{10}^T & b_{01} := -a_{01} & \\ \hline & & \\ \hline B_{20} := B_{20} - a_{21}b_{10}^T & b_{21} := -a_{21} & \end{array}$$

(Note: above a_{01} and a_{21} must be updated before the operations to their right.)

$$\begin{array}{c|c|c} & a_{01} := 0 & \\ \hline & \alpha_{11} := 1 & a_{12}^T := a_{12}^T/\alpha_{11} \\ \hline & a_{21} := 0 & \end{array} \qquad \begin{array}{c|c|c} & & \\ \hline b_{10}^T := b_{10}^T/\alpha_{11} & \beta_{11} = 1/\alpha_{11} & \\ \hline & & \end{array}$$

(Note: above α_{11} must be updated last.)

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c|c} B_{TL} & B_{TR} \\ \hline B_{BL} & B_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} B_{00} & b_{01} & B_{02} \\ \hline b_{10}^T & \beta_{11} & b_{12}^T \\ \hline B_{20} & b_{21} & B_{22} \end{array}\right)$$

endwhile

During the kth iteration, A_{TL} and B_{TL} are $k \times k$ (starting with $k = 0$). After repartitioning, the sizes of the different submatrices are

$$
\begin{array}{r}
\\
k\{ \\
1\{ \\
n-k-1\{
\end{array}
\begin{array}{c|c|c}
\overbrace{\phantom{A_{00}}}^{k} & \overbrace{\phantom{a_{01}}}^{1} & \overbrace{\phantom{A_{02}}}^{n-k-1} \\
A_{00} & a_{01} & A_{02} \\ \hline
a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline
A_{20} & a_{21} & A_{02}
\end{array}
$$

The following operations are performed (we ignore the other operations since they are clearly "cheap" relative to the ones we do count here):

- $A_{02} := A_{02} - a_{01}a_{12}^T$. This is a rank-1 update. The cost is $2k \times (n-k-1)$ flops.
- $A_{22} := A_{22} - a_{21}a_{12}^T$. This is a rank-1 update. The cost is $2(n-k-1) \times (n-k-1)$ flops.
- $B_{00} := B_{00} - a_{01}b_{10}^T$. This is a rank-1 update. The cost is $2k \times k$ flops.
- $B_{02} := B_{02} - a_{21}b_{12}^T$. This is a rank-1 update. The cost is $2(n-k-1) \times k$ flops.

For a total of, approximately,

$$
\begin{aligned}
\underbrace{2k(n-k-1) + 2(n-k-1)(n-k-1)}_{2(n-1)(n-k-1)} + \underbrace{2k^2 + 2(n-k-1)k}_{2(n-1)k} &= 2(n-1)(n-k-1) + 2(n-1)k \\
&= 2(n-1)^2 \text{ flops.}
\end{aligned}
$$

Now, we do this for n iterations, so the total cost of the Gauss-Jordan inversion algorithms is, approximately,

$$n(2(n-1)^2) \approx 2n^3 \text{ flops.}$$

Barring any special properties of matrix A, or high-trapeze heroics, this turns out to be the cost of matrix inversion. Notice that this cost is less than the cost of the (less) naive algorithm given before.

A simpler analysis is as follows: The bulk of the computation in each iteration is in the updates

$$
\begin{array}{c||c}
B_{00} := B_{00} - a_{01}b_{10}^T & A_{02} := A_{02} - a_{01}a_{12}^T \\ \hline\hline
B_{20} := B_{20} - a_{21}b_{10}^T & A_{22} := A_{22} - a_{21}a_{12}^T
\end{array}
$$

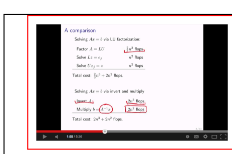
Here we try to depict that the elements being updated occupy almost an entire $n \times n$ matrix. Since there are rank-1 updates being performed, this means that essentially every element in this matrix is being updated with one multiply and one add. Thus, in this iteration, approximately $2n^2$ flops are being performed. The total for n iterations is then, approximately, $2n^3$ flops.

Returning one last time to our relatively small problem of inverting a $100,000 \times 100,000$ matrix on the fastest current computer that can perform approximately 55,000 Teraflops, inverting such a matrix with this alternative approach is further reduced from approximately 0.05 seconds to approximately 0.036 seconds. Not as dramatic a reduction, but still worthwhile.

Interestingly, the cost of matrix inversion is approximately the same as the cost of matrix-matrix multiplication.

8.3 (Almost) Never, Ever Invert a Matrix

8.3.1 Solving $Ax = b$



☛ YouTube

Solving $Ax = b$ via LU Factorization

Homework 8.3.1.1 Let $A \in \mathbb{R}^{n \times n}$ and $x, b \in \mathbb{R}^n$. What is the cost of solving $Ax = b$ via LU factorization (assuming there is nothing special about A)? You may ignore the need for pivoting.

☛ SEE ANSWER

Solving $Ax = b$ by Computing A^{-1}

Homework 8.3.1.2 Let $A \in \mathbb{R}^{n \times n}$ and $x, b \in \mathbb{R}^n$. What is the cost of solving $Ax = b$ if you first invert matrix A and than compute $x = A^{-1}b$? (Assume there is nothing special about A and ignore the need for pivoting.)

☛ SEE ANSWER

Just Don't Do It!

The bottom line is: LU factorization followed by two triangular solves is cheaper!

Now, some people would say "What if we have many systems $Ax = b$ where A is the same, but b differs? Then we can just invert A once and for each of the bs multiply $x = A^{-1}b$."

Homework 8.3.1.3 What is wrong with the above argument?

☛ SEE ANSWER

There are other arguments why computing A^{-1} is a bad idea that have to do with floating point arithmetic and the roundoff error that comes with it. This is a subject called "numerical stability", which goes beyond the scope of this course.

So.... You should be very suspicious if someone talks about computing the inverse of a matrix. There are very, very few applications where one legitimately needs the inverse of a matrix.
However, realize that often people use the term "inverting a matrix" interchangeably with "solving $Ax = b$", where they don't mean to imply that they explicitly invert the matrix. So, be careful before you start arguing with such a person! They may simply be using awkward terminology.

Of course, the above remarks are for general matrices. For small matrices and/or matrices with special structure, inversion may be a reasonable option.

8.3.2 But...

No Video for this Unit

Inverse of a general matrix

Ironically, one of the instructors of this course has written a paper about high-performance inversion of a matrix, which was then published by a top journal:

> Xiaobai Sun, Enrique S. Quintana, Gregorio Quintana, and Robert van de Geijn.
> A Note on Parallel Matrix Inversion.
> *SIAM Journal on Scientific Computing*, Vol. 22, No. 5, pp. 1762–1771.
> Available from http://www.cs.utexas.edu/users/flame/pubs/SIAMMatrixInversion.pdf.
> (This was the first journal paper in which the FLAME notation was introduced.)

The algorithm developed for that paper is a blocked algorithm that incorporates pivoting that is a direct extension of the algorithm we introduce in Unit 8.2.5. It was developed for use in a specific algorithm that required the explicit inverse of a general matrix.

Inverse of a symmetric positive definite matrix

Inversion of a special kind of symmetric matrix called a symmetric positive definite (SPD) matrix is sometimes needed in statistics applications. The inverse of the so-called covariance matrix (which is typically a SPD matrix) is called the precision matrix, which for some applications is useful to compute. We talk about how to compute a factorization of such matrices in this week's enrichment.

If you go to wikipedia and seach for "precision matrix" you will end up on this page:

Precision (statistics)

that will give you more information.

We have a paper on how to compute the inverse of a SPD matrix:

> Paolo Bientinesi, Brian Gunter, Robert A. van de Geijn.
> Families of algorithms related to the inversion of a Symmetric Positive Definite matrix.
> *ACM Transactions on Mathematical Software (TOMS)*, 2008
> Available from `http://www.cs.utexas.edu/~flame/web/FLAMEPublications.html`.

Welcome to the frontier!

Try reading the papers above (as an enrichment)! You will find the notation very familiar.

8.4 (Very Important) Enrichment

8.4.1 Symmetric Positive Definite Matrices

Symmetric positive definite (SPD) matrices are an important class of matrices that occur naturally as part of applications. We will see SPD matrices come up later in this course, when we discuss how to solve overdetermined systems of equations:

$$Bx = y \quad \text{where} \quad B \in \mathbb{R}^{m\times n} \text{ and } m > n.$$

In other words, when there are more equations than there are unknowns in our linear system of equations. When B has "linearly independent columns," a term with which you will become very familiar later in the course, the best solution to $Bx = y$ satisfies $B^T Bx = B^T y$. If we set $A = B^T B$ and $b = B^T y$, then we need to solve $Ax = b$, and now A is square and nonsingular (which we will prove later in the course). Now, we could solve $Ax = b$ via any of the methods we have discussed so far. However, these methods ignore the fact that A is symmetric. So, the question becomes how to take advantage of symmetry.

Definition 8.1 *Let $A \in \mathbb{R}^{n\times n}$. Matrix A is said to be symmetric positive definite (SPD) if*

- *A is symmetric; and*
- *$x^T Ax > 0$ for all nonzero vectors $x \in \mathbb{R}^n$.*

A nonsymmetric matrix can also be positive definite and there are the notions of a matrix being negative definite or indefinite. We won't concern ourselves with these in this course.

Here is a way to relate what a positive definite matrix is to something you may have seen before. Consider the quadratic polynomial

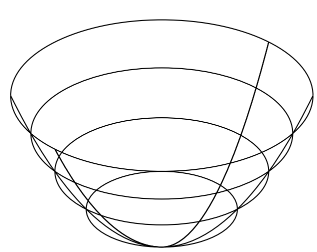
$$p(\chi) = \alpha\chi^2 + \beta\chi + \gamma = \chi\alpha\chi + \beta\chi + \gamma.$$

The graph of this function is a parabola that is "concaved up" if $\alpha > 0$. In that case, it attains a minimum at a unique value χ.

Now consider the vector function $f : \mathbb{R}^n \to \mathbb{R}$ given by

$$f(x) = x^T Ax + b^T x + \gamma$$

where $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$, and $\gamma \in \mathbb{R}$ are all given. If A is a SPD matrix, then this equation is minimized for a unique vector x. If $n = 2$, plotting this function when A is SPD yields a paraboloid that is concaved up:



8.4.2 Solving $Ax = b$ when A is Symmetric Positive Definite

We are going to concern ourselves with how to solve $Ax = b$ when A is SPD. What we will notice is that by taking advantage of symmetry, we can factor A akin to how we computed the LU factorization, but at roughly half the computational cost. This new factorization is known as the Cholesky factorization.

Cholesky factorization theorem

Theorem 8.2 *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix. Then there exists a lower triangular matrix $L \in \mathbb{R}^{n\times n}$ such that $A = LL^T$. If the diagonal elements of L are chosen to be positive, this factorization is unique.*

We will not prove this theorem.

Unblocked Cholesky factorization

We are going to closely mimic the derivation of the LU factorization algorithm from Unit 6.3.1.

Partition

$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & \star \\ \hline a_{21} & A_{22} \end{array}\right), \quad \text{and} \quad L \rightarrow \left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right).$$

Here we use $\star$ to indicate that we are not concerned with that part of the matrix because A is symmetric and hence we should be able to just work with the lower triangular part of it.

We want L to satisfy $A = LL^T$. Hence

$$\begin{aligned}
\overbrace{\left(\begin{array}{c|c} \alpha_{11} & \star \\ \hline a_{21} & A_{22} \end{array}\right)}^{A} &= \overbrace{\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right)}^{L} \overbrace{\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right)^T}^{L^T} \\
&= \overbrace{\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right)}^{L} \overbrace{\left(\begin{array}{c|c} \lambda_{11} & l_{21}^T \\ \hline 0 & L_{22}^T \end{array}\right)}^{L^T} \\
&= \overbrace{\left(\begin{array}{c|c} \lambda_{11}^2 + 0 \times 0 & \star \\ \hline l_{21}\lambda_{11} + L_{22} \times 0 & l_{21}l_{21}^T + L_{22}L_{22}^T \end{array}\right)}^{LL^T}. \\
&= \overbrace{\left(\begin{array}{c|c} \lambda_{11}^2 & \star \\ \hline l_{21}\lambda_{11} & l_{21}l_{21}^T + L_{22}L_{22}^T \end{array}\right)}^{LL^T}.
\end{aligned}$$

where, again, the $\star$ refers to part of the matrix in which we are not concerned because of symmetry.

For two matrices to be equal, their elements must be equal, and therefore, if they are partitioned conformally, their submatrices must be equal:

$$\begin{array}{c|c} \alpha_{11} = \lambda_{11}^2 & \star \\ \hline a_{21} = l_{21}\lambda_{11} & A_{22} = l_{21}l_{21}^T + L_{22}L_{22}^T \end{array}$$

or, rearranging,

$$\begin{array}{c|c} \lambda_{11} = \sqrt{\alpha_{11}} & \star \\ \hline l_{21} = a_{21}/\lambda_{11} & L_{22}L_{22}^T = A_{22} - l_{21}l_{21}^T \end{array}.$$

This suggests the following steps for **overwriting** a matrix A with its Cholesky factorization:

Algorithm: $[A] := \text{CHOL\_UNB\_VAR3}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

$\alpha_{11} := \sqrt{\alpha_{11}}$

$a_{21} := a_{21}/\alpha_{11}$

$A_{22} := A_{22} - a_{21}a_{21}^T$

(updating only the lower triangular part)

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 8.8: Algorithm for overwriting the lower triangular part of A with its Cholesky factor.

- Partition
$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & \star \\ \hline a_{21} & A_{22} \end{array}\right).$$
- $\alpha_{11} = \sqrt{\alpha_{11}} \quad (= \lambda_{11})$.
- Update $a_{21} = a_{21}/\alpha_{11} \quad (= l_{21})$.
- Update $A_{22} = A_{22} - a_{21}a_{12}^T (= A_{22} - l_{21}l_{21}^T)$
Here we use a "symmetric rank-1 update" since A_{22} and $l_{21}l_{21}^T$ are both symmetric and hence only the lower triangular part needs to be updated. This is where we save flops.
- Overwrite A_{22} with L_{22} by repeating with $A = A_{22}$.

This overwrites the lower triangular part of A with L.

The above can be summarized in Figure 8.8. The suspicious reader will notice that $\alpha_{11} := \sqrt{\alpha_{11}}$ is only legal if $\alpha_{11} > 0$ and $a_{21} := a_{21}/\alpha_{11}$ is only legal if $\alpha_{11} \neq 0$. It turns out that *if* A is SPD, then

- $\alpha_{11} > 0$ in the first iteration and hence $\alpha_{11} := \sqrt{\alpha_{11}}$ and $a_{21} := a_{21}/\alpha_{11}$ are legal; and
- $A_{22} := A_{22} - a_{21}a_{21}^T$ is again a SPD matrix.

The proof of these facts goes beyond the scope of this course. The net result is that the algorithm will compute L if it is executed starting with a matrix A that is SPD. It is useful to compare and contrast the derivations of the unblocked LU factorization and the unblocked Cholesky factorization, in Figure 8.9.

<table>
<tr><th>LU factorization</th><th>Cholesky factorization</th></tr>
<tr>
<td>$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right), L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)$</td>
<td>$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & \star \\ \hline a_{21} & A_{22} \end{array}\right), L \rightarrow \left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right).$</td>
</tr>
<tr>
<td>$$\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \underbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)}_{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline l_{21}\upsilon_{11} & l_{21}u_{12}^T + L_{22}U_{22} \end{array}\right)}$$</td>
<td>$$\left(\begin{array}{c|c} \alpha_{11} & \star \\ \hline a_{21} & A_{22} \end{array}\right) = \underbrace{\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right)\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right)^T}_{\left(\begin{array}{c|c} \lambda_{11}^2 & \star \\ \hline l_{21}\lambda_{11} & l_{21}l_{12}^T + L_{22}L_{22}^T \end{array}\right)}.$$</td>
</tr>
<tr>
<td>$$\begin{array}{c|c} \alpha_{11} = \upsilon_{11} & a_{12}^T = u_{12}^T \\ \hline a_{21} = l_{21}\upsilon_{11} & A_{22} = l_{21}u_{12}^T + L_{22}U_{22} \end{array}$$</td>
<td>$$\begin{array}{c|c} \alpha_{11} = \lambda_{11}^2 & \star \\ \hline a_{21} = l_{21}\lambda_{11} & A_{22} = l_{21}l_{12}^T + L_{22}L_{22}^T \end{array}$$</td>
</tr>
<tr>
<td>$\alpha_{11} := \alpha_{11}$
$a_{12}^T := a_{12}^T$
$a_{21} := a_{21}/\alpha_{11}$
$A_{22} := A_{22} - a_{21}a_{12}^T$</td>
<td>$\alpha_{11} := \sqrt{\alpha_{11}}$
$a_{21} := a_{21}/\alpha_{11}$
$A_{22} := A_{22} - a_{21}a_{12}^T$
(update only lower triangular part)</td>
</tr>
<tr>
<td>Algorithm: $[A] := \text{LU\_UNB\_VAR5}(A)$
Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
where A_{TL} is 0×0
while $m(A_{TL}) < m(A)$ do
Repartition
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$

$a_{21} := a_{21}/\alpha_{11}$
$A_{22} := A_{22} - a_{21}a_{12}^T$

Continue with
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$
endwhile</td>
<td>Algorithm: $[A] := \text{CHOL\_UNB\_VAR3}(A)$
Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
where A_{TL} is 0×0
while $m(A_{TL}) < m(A)$ do
Repartition
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$

$\alpha_{11} := \sqrt{\alpha_{11}}$
$a_{21} := a_{21}/\alpha_{11}$
$A_{22} := A_{22} - a_{21}a_{21}^T$
(updating only the lower triangular part)

Continue with
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$
endwhile</td>
</tr>
</table>

Figure 8.9: Side-by-side derivations of the unblocked LU factorization and Cholesky factorization algorithms.

Once one has computed the Cholesky factorization of A, one can solve $Ax = b$ by substituting

$$\overbrace{LL^T}^{A} x = b$$

and first solving $Lz = b$ after which solving $L^T x = z$ computes the desired solution x. Of course, as you learned in Weeks 3 and 4, you need not transpose the matrix!

Blocked (and other) algorithms

If you are interested in blocked algorithms for computing the Cholesky factorization, you may want to look at some notes we wrote:

> Robert van de Geijn.
> Notes on Cholesky Factorization
> http://www.cs.utexas.edu/users/flame/Notes/NotesOnCholReal.pdf

These have since become part of the notes Robert wrote for his graduate class on Numerical Linear Algebra:

> Robert van de Geijn.
> Linear Algebra: Foundations to Frontiers - Notes on Numerical Linear Algebra, Chapter 12.

Systematic derivation of Cholesky factorization algorithms

☛ YouTube

The above video was created when Robert was asked to give an online lecture for a class at Carnegie Mellon University. It shows how algorithms can be systematically derived (as we discussed already in Week 2) using goal-oriented programming. It includes a demonstration by Prof. Paolo Bientinesi (RWTH Aachen University) of a tool that performs the derivation automatically. It is when a process is systematic to the point where it can be automated that a computer scientist is at his/her happiest!

More materials

You will find materials related to the implementation of this operations, including a video that demonstrates this, at

http://www.cs.utexas.edu/users/flame/Movies.html#Chol

Unfortunately, some of the links don't work (we had a massive failure of the wiki that hosted the material).

8.4.3 Other Factorizations

We have now encountered the LU factorization,

$$A = LU,$$

the LU factorization with row pivoting,

$$PA = LU,$$

and the Cholesky factorization,

$$A = LL^T.$$

Later in this course you will be introduced to the QR factorization,

$$A = QR,$$

where Q has the special property that $Q^T Q = I$ and R is an upper triangular matrix.

When a matrix is *indefinite symmetric*, there is a factorization called the LDL^T (pronounce as L D L transpose) factorization,

$$A = LDL^T,$$

where L is unit lower triangular and D is diagonal. You may want to see if you can modify the derivation of the Cholesky factorization to yield an algorithm for the LDL^T factorization.

8.4.4 Welcome to the Frontier

Building on the material to which you have been exposed so far in this course, you should now be able to fully understand significant parts of many of our publications. (When we write our papers, we try to target a broad audience.) Many of these papers can be found at

http://www.cs.utexas.edu/~flame/web/publications.

If not there, then Google!

Here is a small sampling:

- The paper I consider our most significant contribution to science to date:

  Paolo Bientinesi, John A. Gunnels, Margaret E. Myers, Enrique S. Quintana-Orti, Robert A. van de Geijn.
  The science of deriving dense linear algebra algorithms.
  ACM Transactions on Mathematical Software (TOMS), 2005.

- The book that explains the material in that paper at a more leisurely pace:

  Robert A. van de Geijn and Enrique S. Quintana-Orti.
  The Science of Programming Matrix Computations.
  www.lulu.com, 2008.

- The journal paper that first introduced the FLAME notation:

  Xiaobai Sun, Enrique S. Quintana, Gregorio Quintana, and Robert van de Geijn.
  A Note on Parallel Matrix Inversion.
  SIAM Journal on Scientific Computing, Vol. 22, No. 5, pp. 1762–1771.
  http://www.cs.utexas.edu/~flame/pubs/SIAMMatrixInversion.pdf.

- The paper that discusses many operations related to the inversion of a SPD matrix:

  Paolo Bientinesi, Brian Gunter, Robert A. van de Geijn.
  Families of algorithms related to the inversion of a Symmetric Positive Definite matrix.
  ACM Transactions on Mathematical Software (TOMS), 2008.

- The paper that introduced the FLAME APIs:

  Paolo Bientinesi, Enrique S. Quintana-Orti, Robert A. van de Geijn.
  Representing linear algebra algorithms in code: the FLAME application program interfaces.
  ACM Transactions on Mathematical Software (TOMS), 2005.

- Our papers on high-performance implementation of BLAS libraries:

  Kazushige Goto, Robert A. van de Geijn.
  Anatomy of high-performance matrix multiplication.
  ACM Transactions on Mathematical Software (TOMS), 2008.

  Kazushige Goto, Robert van de Geijn.
  High-performance implementation of the level-3 BLAS.
  ACM Transactions on Mathematical Software (TOMS), 2008

  Field G. Van Zee, Robert A. van de Geijn.
  BLIS: A Framework for Rapid Instantiation of BLAS Functionality.
  ACM Transactions on Mathematical Software, to appear.

- A classic paper on how to parallelize matrix-matrix multiplication:

  Robert A van de Geijn, Jerrell Watts.
  SUMMA: Scalable universal matrix multiplication algorithm.
  Concurrency Practice and Experience, 1997.

For that paper, and others on parallel computing on large distributed memory computers, it helps to read up on collective communication on massively parallel architectures:

> Ernie Chan, Marcel Heimlich, Avi Purkayastha, Robert van de Geijn.
> Collective communication: theory, practice, and experience.
> Concurrency and Computation: Practice & Experience , Volume 19 Issue 1, September 2007

- A paper that gives you a peek at how to parallelize for massively parallel architectures:

> Jack Poulson, Bryan Marker, Robert A. van de Geijn, Jeff R. Hammond, Nichols A. Romero.
> Elemental: A New Framework for Distributed Memory Dense Matrix Computations.
> ACM Transactions on Mathematical Software (TOMS), 2013.

Obviously, there are many people who work in the area of dense linear algebra operations and algorithms. We cite our papers here because you will find the notation used in those papers to be consistent with the slicing and dicing notation that you have been taught in this course. Much of the above cite work builds on important results of others. We stand on the shoulders of giants.

8.5 Wrap Up

8.5.1 Homework

8.5.2 Summary

Equivalent conditions

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.
- LU with partial pivoting does not break down.

Algorithm for inverting a matrix

See Figure 8.10.

Cost of inverting a matrix

Via Gauss-Jordan, taking advantage of zeroes in the appended identity matrix, requires approximately

$$2n^3 \text{ floating point operations.}$$

Algorithm: $[A] := \text{GJ\_INVERSE\_INPLACE}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

$$\left(\begin{array}{c|c|c} & a_{01} := a_{01}/\alpha_{11} & A_{02} := A_{02} - a_{01}a_{12}^T \\ \hline & & \\ \hline & a_{21} := a_{21}/\alpha_{11} & A_{22} := A_{22} - a_{21}a_{12}^T \end{array}\right) \quad \left(\begin{array}{c|c|c} A_{00} := A_{00} - a_{01}a_{10}^T & a_{01} := -a_{01} & \\ \hline & & \\ \hline A_{20} := A_{20} - a_{21}a_{10}^T & a_{21} := -a_{21} & \end{array}\right)$$

(Note: above a_{01} and a_{21} must be updated before the operations to their right.)

$$\left(\begin{array}{c|c|c} & & \\ \hline & & a_{12}^T := a_{12}^T/\alpha_{11} \\ \hline & & \end{array}\right) \quad \left(\begin{array}{c|c|c} & & \\ \hline a_{10}^T := a_{10}^T/\alpha_{11} & \alpha_{11} = 1/\alpha_{11} & \\ \hline & & \end{array}\right)$$

(Note: above α_{11} must be updated last.)

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 8.10: Algorithm for inplace inversion of a matrix (when pivoting is not needed).

(Almost) never, ever invert a matrix

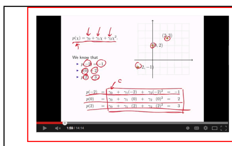
Solving $Ax = b$ should be accomplished by first computing its LU factorization (possibly with partial pivoting) and then solving with the triangular matrices.

Week 9

Vector Spaces

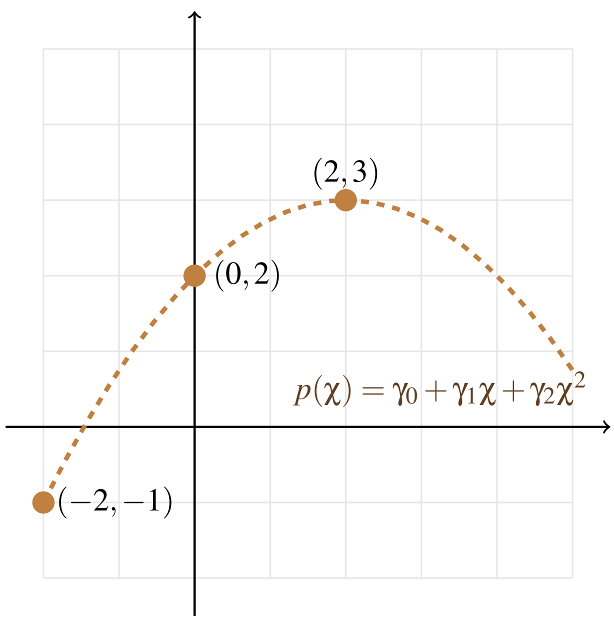
9.1 Opening Remarks

9.1.1 Solvable or not solvable, that's the question



☛ YouTube

Consider the picture



depicting three points in $\mathbb{R}^2$ and a quadratic polynomial (polynomial of degree two) that passes through those points. We say that this polynomial *interpolates* these points. Let's denote the polynomial by

$$p(\chi) = \gamma_0 + \gamma_1\chi + \gamma_2\chi^2.$$

How can we find the coefficients γ_0, γ_1, and γ_2 of this polynomial? We know that $p(-2) = -1$, $p(0) = 2$, and $p(2) = 3$. Hence

$$\begin{array}{lcccccccc}
p(-2) & = & \gamma_0 & + & \gamma_1(-2) & + & \gamma_2(-2)^2 & = & -1 \\
p(0) & = & \gamma_0 & + & \gamma_1\ \ (0) & + & \gamma_2\ \ (0)^2 & = & 2 \\
p(2) & = & \gamma_0 & + & \gamma_1\ \ (2) & + & \gamma_2\ \ (2)^2 & = & 3
\end{array}$$

In matrix notation we can write this as

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}.$$

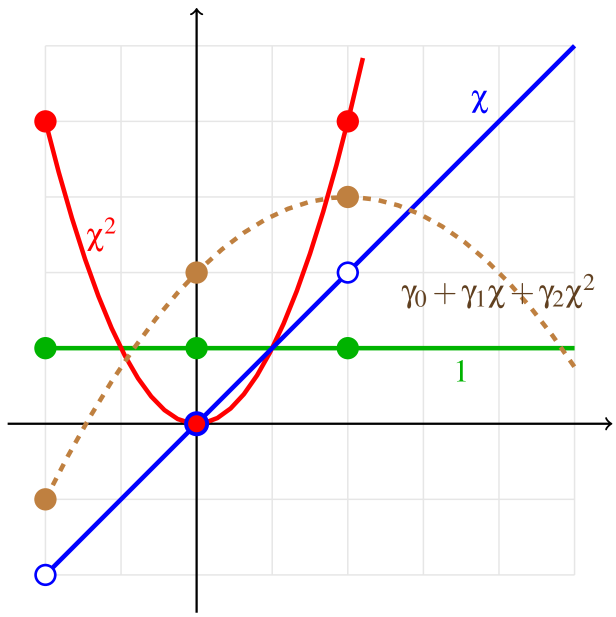
By now you have learned a number of techniques to solve this linear system, yielding

$$\begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$$

so that

$$p(\chi) = 2 + \chi - \frac{1}{4}\chi^2.$$

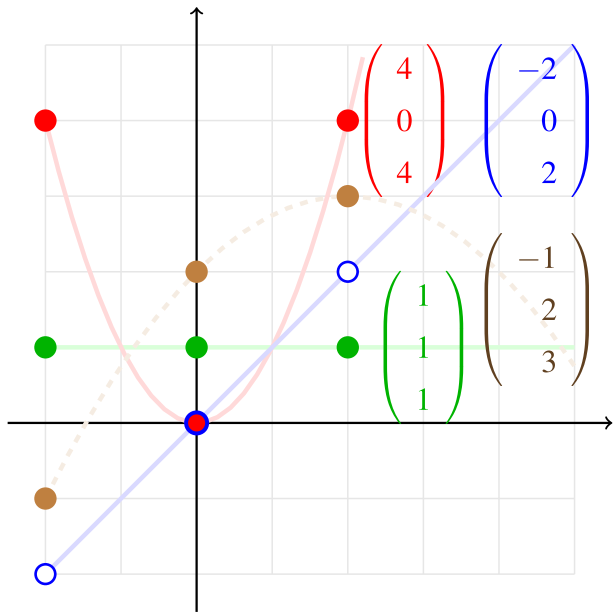
Now, let's look at this problem a little differently. $p(\chi)$ is a linear combination (a word you now understand well) of the polynomials $p_0(\chi) = 1$, $p_1(\chi) = \chi$, and $p_2(\chi) = \chi^2$. These basic polynomials are called "parent functions".



Now, notice that

$$\begin{aligned}
\begin{pmatrix} p(-2) \\ p(0) \\ p(2) \end{pmatrix} &= \begin{pmatrix} \gamma_0 & + & \gamma_1(-2) & + & \gamma_2(-2)^2 \\ \gamma_0 & + & \gamma_1(0) & + & \gamma_2(0)^2 \\ \gamma_0 & + & \gamma_1(2) & + & \gamma_2(2)^2 \end{pmatrix} \\
&= \gamma_0 \begin{pmatrix} p_0(-2) \\ p_0(0) \\ p_0(2) \end{pmatrix} + \gamma_1 \begin{pmatrix} p_1(-2) \\ p_1(0) \\ p_1(2) \end{pmatrix} + \gamma_2 \begin{pmatrix} p_2(-2) \\ p_2(0) \\ p_2(2) \end{pmatrix} \\
&= \gamma_0 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \gamma_1 \begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix} + \gamma_2 \begin{pmatrix} (-2)^2 \\ 0^2 \\ 2^2 \end{pmatrix} \\
&= \gamma_0 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \gamma_1 \begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix} + \gamma_2 \begin{pmatrix} 4 \\ 0 \\ 4 \end{pmatrix}.
\end{aligned}$$

You need to think of the three vectors $\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$, $\begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix}$, and $\begin{pmatrix} 4 \\ 0 \\ 4 \end{pmatrix}$ as vectors that capture the polynomials p_0, p_1, and p_2 at the values -2, 0, and 2. Similarly, the vector $\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$ captures the polynomial p that interpolates the given points.



What we notice is that this last vector must equal a linear combination of the first three vectors:

$$\gamma_0 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \gamma_1 \begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix} + \gamma_2 \begin{pmatrix} 4 \\ 0 \\ 4 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$$

Again, this gives rise to the matrix equation

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$$

with the solution

$$\begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}.$$

The point is that one can think of finding the coefficients of a polynomial that interpolates points as either solving a system of linear equations that come from the constraint imposed by the fact that the polynomial must go through a given set of points, or as finding the linear combination of the vectors that represent the parent functions at given values so that this linear combination equals the vector that represents the polynomial that is to be found.

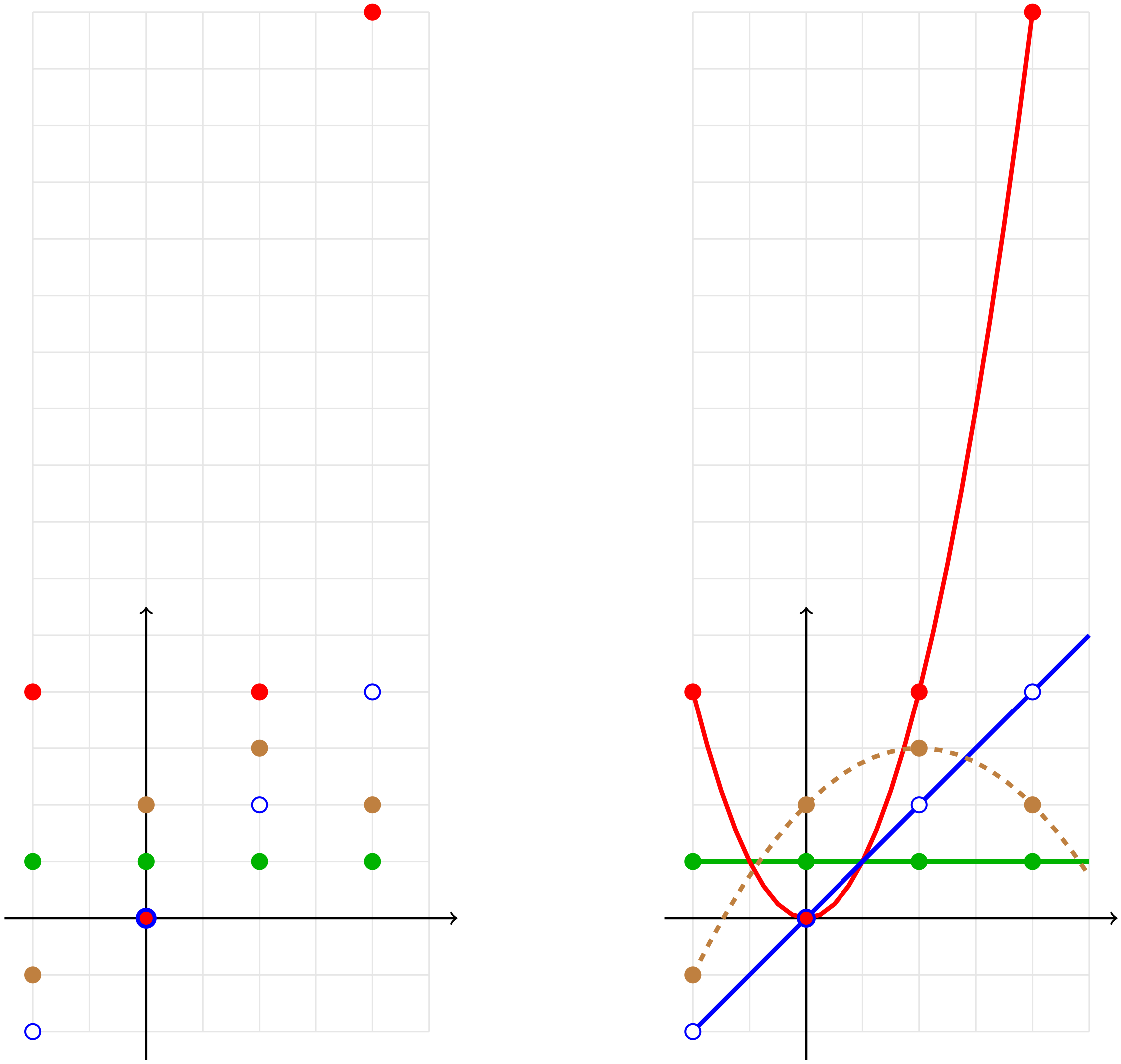


Figure 9.1: Interpolating with at second degree polynomial at $\chi = -2, 0, 2, 4$.

To be or not to be (solvable), that's the question

Next, consider the picture in Figure 9.1 (left), which accompanies the matrix equation

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \\ 1 & 4 & 16 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \\ 2 \end{pmatrix}.$$

Now, this equation is also solved by

$$\begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}.$$

The picture in Figure 9.1 (right) explains why: The new brown point that was added happens to lie on the overall quadratic polynomial $p(\chi)$.

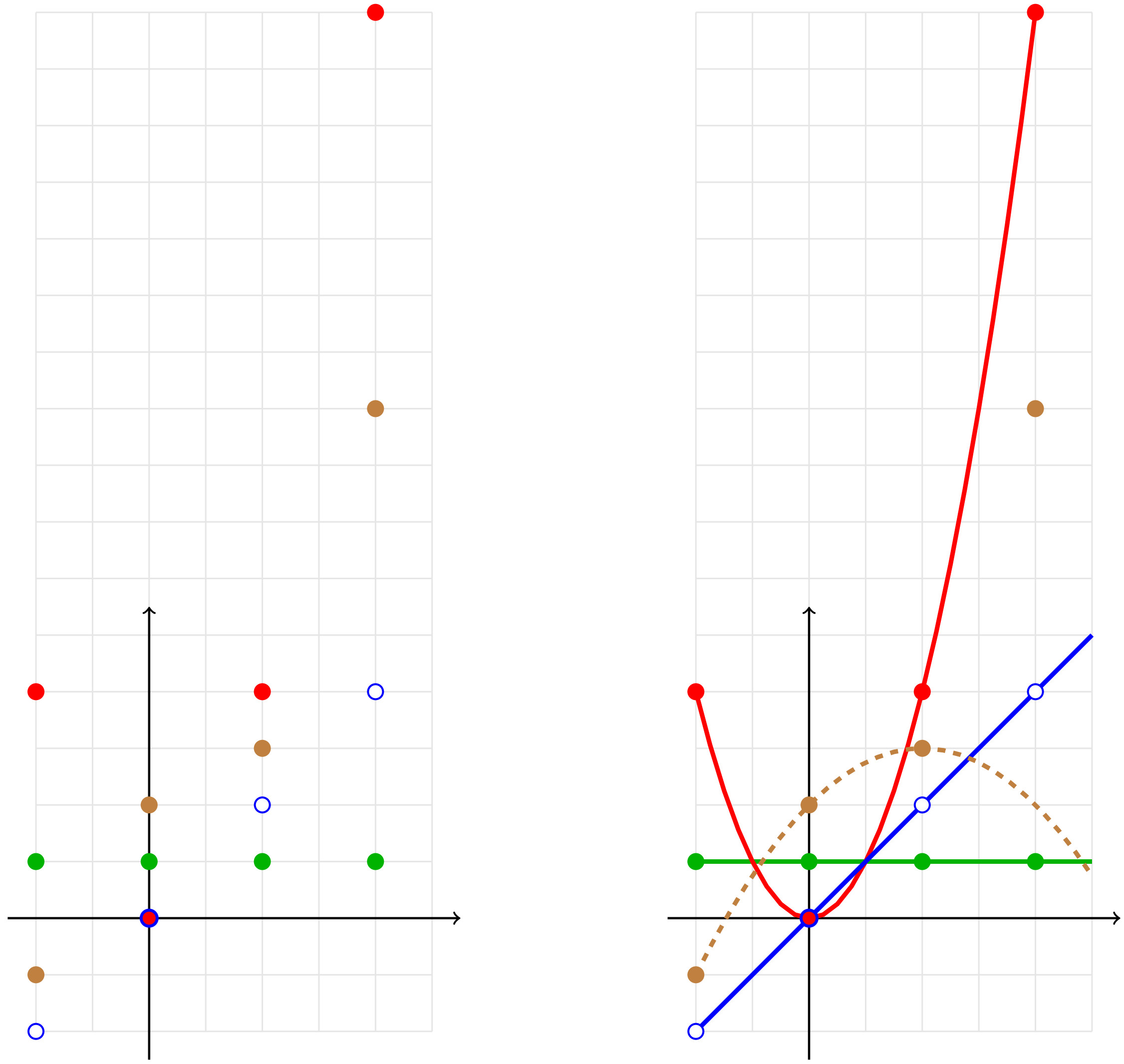


Figure 9.2: Interpolating with at second degree polynomial at $\chi = -2, 0, 2, 4$: when the fourth point doesn't fit.

Finally, consider the picture in Figure 9.2 (left) which accompanies the matrix equation

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \\ 1 & 4 & 16 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \\ 9 \end{pmatrix}.$$

It turns out that this matrix equation (system of linear equations) does not have a solution. The picture in Figure 9.2 (right) explains why: The new brown point that was added does not lie on the quadratic polynomial $p_2(\chi)$.

This week, you will learn that the system $Ax = b$ for an $m \times n$ matrix A sometimes has a unique solution, sometimes has no solution at all, and sometimes has an infinite number of solutions. Clearly, it does not suffice to only look at the matrix A. It is how the columns of A are related to the right-hand side vector that is key to understanding with which situation we are dealing. And the key to understanding how the columns of A are related to those right-hand sides for which $Ax = b$ has a solution is to understand a concept called vector spaces.

9.1.2 Outline

9.1.3 What you will learn

Upon completion of this unit, you should be able to

- Determine when systems do not have a unique solution and recognize the general solution for a system.
- Use and understand set notation.
- Determine if a given subset of $\mathbb{R}^n$ is a subspace.
- For simple examples, determine the null space and column space for a given matrix.
- Identify, apply, and prove simple properties of sets, vector spaces, subspaces, null spaces and column spaces.
- Recognize for simple examples when the span of two sets of vectors is the same.
- Determine when a set of vectors is linearly independent by exploiting special structures. For example, relate the rows of a matrix with the columns of its transpose to determine if the matrix has linearly independent rows.
- For simple examples, find a basis for a subspace and recognize that while the basis is not unique, the number of vectors in the basis is.

Track your progress in Appendix B.

9.2 When Systems Don't Have a Unique Solution

9.2.1 When Solutions Are Not Unique

☛ YouTube

Up until this week, we looked at linear systems that had exactly one solution. The reason was that some variant of Gaussian elimination (with row exchanges, if necessary and/or Gauss-Jordan elimination) completed, which meant that there was exactly one solution.

What we will look at this week are linear systems that have either no solution or many solutions (indeed an infinite number).

Example 9.1 Consider

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix}$$

Does $Ax = b_0$ have a solution? The answer is yes:

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix}. \quad \checkmark$$

But this is not the only solution:

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} \frac{3}{2} \\ 0 \\ \frac{3}{2} \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix} \quad \checkmark$$

and

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix}. \quad \checkmark$$

Indeed, later we will see there are an infinite number of solutions!

Example 9.2 Consider

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 4 \end{pmatrix}.$$

We will show that this equation does not have a solution in the next unit.

Homework 9.2.1.1 Evaluate

1. $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} =$

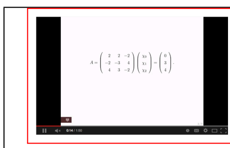
2. $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ -1 \end{pmatrix} =$

3. $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix} =$

Does the system $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}$ have multiple solutions? Yes/No

☛ SEE ANSWER

9.2.2 When Linear Systems Have No Solutions



☛ YouTube

Consider

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 4 \end{pmatrix}.$$

- Set this up as an appended system

$$\left(\begin{array}{rrr|r} 2 & 2 & -2 & 0 \\ -2 & -3 & 4 & 3 \\ 4 & 3 & -2 & 4 \end{array}\right).$$

Now, start applying Gaussian elimination (with row exchanges).

- Use the first row to eliminate the coefficients in the first column below the diagonal:

$$\left(\begin{array}{rrr|r} 2 & 2 & -2 & 0 \\ 0 & -1 & 2 & 3 \\ 0 & -1 & 2 & 4 \end{array}\right).$$

- Use the second row to eliminate the coefficients in the second column below the diagonal:

$$\left(\begin{array}{rrr|r} 2 & 2 & -2 & 0 \\ 0 & -1 & 2 & 3 \\ 0 & 0 & 0 & 1 \end{array}\right).$$

- At this point, we have encountered a zero on the diagonal of the matrix that cannot be fixed by exchanging with rows below the row that has the zero on the diagonal.

Now we have a problem: The last line of the appended system represents

$$0 \times \chi_0 + 0 \times \chi_1 + 0 \times \chi_2 = 1,$$

or,

$$0 = 1$$

which is a contradiction. Thus, the original linear system represented three equations with three unknowns in which a contradiction was hidden. As a result this system does not have a solution.

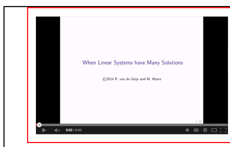
> Anytime you execute Gaussian elimination (with row exchanges) or Gauss-Jordan (with row exchanges) and at some point encounter a row in the appended system that has zeroes to the left of the vertical bar and a nonzero to its right, the process fails and the system has no solution.

Homework 9.2.2.1 The system $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 3 \end{pmatrix}$ has no solution.

True/False

☛ SEE ANSWER

9.2.3 When Linear Systems Have Many Solutions



☛ **YouTube**

Now, let's learn how to find one solution to a system $Ax = b$ that has an infinite number of solutions. Not surprisingly, the process is remarkably like Gaussian elimination:

Consider again

$$A = \begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix}.$$

Set this up as an appended systems

$$\left(\begin{array}{rrr|r} 2 & 2 & -2 & 0 \\ -2 & -3 & 4 & 3 \\ 4 & 3 & -2 & 3 \end{array}\right) \tag{9.1}$$

Now, apply Gauss-Jordan elimination. (Well, something that closely resembles what we did before, anyway.)

- Use the first row to eliminate the coefficients in the first column below the diagonal:

$$\left(\begin{array}{rrr|r} 2 & 2 & -2 & 0 \\ 0 & -1 & 2 & 3 \\ 0 & -1 & 2 & 3 \end{array}\right).$$

- Use the second row to eliminate the coefficients in the second column below the diagonal and use the second row to eliminate the coefficients in the second column above the diagonal:

$$\left(\begin{array}{rrr|r} 2 & 0 & 2 & 6 \\ 0 & -1 & 2 & 3 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

- Divide the first and second row by the diagonal element:

$$\left(\begin{array}{ccc|c} 1 & 0 & 1 & 3 \\ 0 & 1 & -2 & -3 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

Now, what does this mean? Up until this point, we have not encountered a situation in which the system, upon completion of either Gaussian elimination or Gauss-Jordan elimination, an entire zero row. Notice that the difference between this situation and the situation of no solution in the previous section is that the entire row of the final appended system is zero, including the part to the right of the vertical bar.

So, let's translate the above back into a system of linear equations:

$$\begin{array}{rcrcrcr} \chi_0 & & & + & \chi_2 & = & 3 \\ & & \chi_1 & - & 2\chi_2 & = & -3 \\ & & & & 0 & = & 0 \end{array}$$

Notice that we really have two equations and three unknowns, plus an equation that says that "$0 = 0$", which *is true*, but doesn't help much!

Two equations with three unknowns does not give us enough information to find a unique solution. What we are going to do is to make χ_2 a "free variable", meaning that it can take on any value in $\mathbb{R}$ and we will see how the "bound variables" χ_0 and χ_1 now depend on the free variable. To so so, we introduce β to capture this "any value" that χ_2 can take on. We introduce this as the third equation

$$\begin{array}{rcrcrcr} \chi_0 & & & + & \chi_2 & = & 3 \\ & & \chi_1 & - & 2\chi_2 & = & -3 \\ & & & & \chi_2 & = & \beta \end{array}$$

and then substitute β in for χ_2 in the other equations:

$$\begin{array}{rcrcrcr} \chi_0 & & & + & \beta & = & 3 \\ & & \chi_1 & - & 2\beta & = & -3 \\ & & & & \chi_2 & = & \beta \end{array}$$

Next, we bring the terms that involve β to the right

$$\begin{array}{rcrcrcrcr} \chi_0 & & & & & = & 3 & - & \beta \\ & & \chi_1 & & & = & -3 & + & 2\beta \\ & & & & \chi_2 & = & & & \beta \end{array}$$

Finally, we write this as vectors:

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}$$

We now claim that this captures *all* solutions of the system of linear equations. We will call this the *general* solution.

Let's check a few things:

- Let's multiply the original matrix times the first vector in the general solution:

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix}. \checkmark$$

Thus the first vector in the general solution is a solution to the linear system, corresponding to the choice $\beta = 0$. We will call this vector a *specific* solution and denote it by x_s. Notice that there are many (indeed an infinite number of) specific solutions for this problem.

- Next, let's multiply the original matrix times the second vector in the general solution, the one multiplied by β:

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \checkmark$$

And what about the other solutions that we saw two units ago? Well,

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix}. \quad \checkmark$$

and

$$\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} 3/2 \\ 0 \\ 3/2 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix} \quad \checkmark$$

But notice that these are among the infinite number of solutions that we identified:

$$\begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} + (1) \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 3/2 \\ 0 \\ 3/2 \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} + (3/2) \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}.$$

9.2.4 What is Going On?

☛ **YouTube**

Consider $Ax = b$ and assume that we have

- One solution to the system $Ax = b$, the specific solution we denote by x_s so that $Ax_s = b$.
- One solution to the system $Ax = 0$ that we denote by x_n so that $Ax_n = 0$.

Then

$$\begin{array}{l} A(x_s + x_n) \\ \quad = \qquad < \text{Distribute } A > \\ Ax_s + Ax_n \\ \quad = \qquad < Ax_s = b \text{ and } Ax_n = 0 > \\ b + 0 \\ \quad = \qquad < \text{algebra} > \\ b \end{array}$$

So, $x_s + x_n$ is *also* a solution.

Now,

$$
\begin{aligned}
& A(x_s + \beta x_n) \\
& \quad = \qquad < \text{Distribute } A > \\
& Ax_s + A(\beta x_n) \\
& \quad = \qquad < \text{Constant can be brought out} > \\
& Ax_s + \beta A x_n \\
& \quad = \qquad < Ax_s = b \text{ and } Ax_n = 0 > \\
& b + 0 \\
& \quad = \qquad < \text{algebra} > \\
& b
\end{aligned}
$$

So $A(x_s + \beta x_n)$ is a solution for *every* $\beta \in \mathbb{R}$.

Given a linear system $Ax = b$, the strategy is to first find a specific solution, x_s such that $Ax_s = b$. If this is clearly a unique solution (Gauss-Jordan completed successfully with no zero rows), then you are done. Otherwise, find vector(s) x_n such that $Ax_n = 0$ and use it (these) to specify the general solution.

We will make this procedure more precise later this week.

Homework 9.2.4.1 Let $Ax_s = b$, $Ax_{n_0} = 0$ and $Ax_{n_1} = 0$. Also, let $\beta_0, \beta_1 \in \mathbb{R}$. Then $A(x_s + \beta_0 x_{n_0} + \beta_1 x_{n_1}) = b$.

Always/Sometimes/Never

☛ SEE ANSWER

9.2.5 Toward a Systematic Approach to Finding All Solutions

☛ YouTube

Let's focus on finding nontrivial solutions to $Ax = 0$, for the same example as in Unit 9.2.3. (The trivial solution to $Ax = 0$ is $x = 0$.)

Recall the example

$$
\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix}
$$

which had the general solution

$$
\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}.
$$

We will again show the steps of Gaussian elimination, except that this time we **also** solve

$$
\begin{pmatrix} 2 & 2 & -2 \\ -2 & -3 & 4 \\ 4 & 3 & -2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
$$

- Set both of these up as an appended systems

$$\left(\begin{array}{rrr|r} 2 & 2 & -2 & 0 \\ -2 & -3 & 4 & 3 \\ 4 & 3 & -2 & 3 \end{array}\right) \qquad \left(\begin{array}{rrr|r} 2 & 2 & -2 & 0 \\ -2 & -3 & 4 & 0 \\ 4 & 3 & -2 & 0 \end{array}\right)$$

- Use the first row to eliminate the coefficients in the first column below the diagonal:

$$\left(\begin{array}{rrr|r} \boxed{2} & 2 & -2 & 0 \\ 0 & -1 & 2 & 3 \\ 0 & -1 & 2 & 3 \end{array}\right) \qquad \left(\begin{array}{rrr|r} \boxed{2} & 2 & -2 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & -1 & 2 & 0 \end{array}\right).$$

- Use the second row to eliminate the coefficients in the second column below the diagonal

$$\left(\begin{array}{rrr|r} \boxed{2} & 2 & -2 & 0 \\ 0 & \boxed{-1} & 2 & 3 \\ 0 & 0 & 0 & 0 \end{array}\right) \qquad \left(\begin{array}{rrr|r} \boxed{2} & 2 & -2 & 0 \\ 0 & \boxed{-1} & 2 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

Some terminology

The form of the transformed equations that we have now reached on the left is known as the **row-echelon form**. Let's examine it:

$$\left(\begin{array}{rrr|r} \boxed{2} & 2 & -2 & 0 \\ 0 & \boxed{-1} & 2 & 3 \\ 0 & 0 & 0 & 0 \end{array}\right)$$

The boxed values are known as the *pivots*. In each row to the left of the vertical bar, the left-most nonzero element is the pivot for that row. Notice that the pivots in later rows appear to the right of the pivots in earlier rows.

Continuing on

- Use the second row to eliminate the coefficients in the second column above the diagonal:

$$\left(\begin{array}{rrr|r} \boxed{2} & 0 & 2 & 6 \\ 0 & \boxed{-1} & 2 & 3 \\ 0 & 0 & 0 & 0 \end{array}\right) \qquad \left(\begin{array}{rrr|r} \boxed{2} & 0 & 2 & 0 \\ 0 & \boxed{-1} & 2 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

  In this way, all elements above pivots are eliminated. (Notice we could have done this as part of the previous step, as part of the Gauss-Jordan algorithm from Week 8. However, we broke this up into two parts to be able to introduce the term **row echelon form**, which is a term that some other instructors may expect you to know.)

- Divide the first and second row by the diagonal element to normalize the pivots:

$$\left(\begin{array}{rrr|r} \boxed{1} & 0 & 1 & 3 \\ 0 & \boxed{1} & -2 & -3 \\ 0 & 0 & 0 & 0 \end{array}\right) \qquad \left(\begin{array}{rrr|r} \boxed{1} & 0 & 1 & 0 \\ 0 & \boxed{1} & -2 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

Some more terminology

The form of the transformed equations that we have now reached on the left is known as the **reduced row-echelon form**. Let's examine it:

$$\left(\begin{array}{ccc|c} \boxed{1} & 0 & 1 & 3 \\ 0 & \boxed{1} & -2 & -3 \\ 0 & 0 & 0 & 0 \end{array}\right) \qquad \left(\begin{array}{ccc|c} \boxed{1} & 0 & 1 & 0 \\ 0 & \boxed{1} & -2 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

In each row, the pivot is now equal to one. All elements above pivots have been zeroed.

Continuing on again

- Observe that there was no need to perform all the transformations with the appended system on the right. One could have simply applied them only to the appended system on the left. Then, to obtain the results on the right we simply set the right-hand side (the appended vector) equal to the zero vector.

So, let's translate the left appended system back into a system of linear systems:

$$\begin{array}{rcrcr} \chi_0 & & + & \chi_2 & = & 3 \\ & \chi_1 & - & 2\chi_2 & = & -3 \\ & & & 0 & = & 0 \end{array}$$

As before, we have two equations and three unknowns, plus an equation that says that "$0 = 0$", which is *true*, but doesn't help much! We are going to find *one solution* (a specific solution), by choosing the free variable $\chi_2 = 0$. We can set it to equal anything, but zero is an easy value with which to compute. Substituting $\chi_2 = 0$ into the first two equations yields

$$\begin{array}{rcrcr} \chi_0 & & + & 0 & = & 3 \\ & \chi_1 & - & 2(0) & = & -3 \\ & & & 0 & = & 0 \end{array}$$

We conclude that a specific solution is given by

$$x_s = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix}.$$

Next, let's look for *one* non-trivial solution to $Ax = 0$ by translating the right appended system back into a system of linear equations:

$$\begin{array}{rcrcr} \chi_0 & & + & \chi_2 & = & 0 \\ & \chi_1 & - & 2\chi_2 & = & 0 \end{array}$$

Now, if we choose the free variable $\chi_2 = 0$, then it is easy to see that $\chi_0 = \chi_1 = 0$, and we end up with the trivial solution, $x = 0$. So, instead choose $\chi_2 = 1$. (We, again, can choose any value, but it is easy to compute with 1.) Substituting this into the first two equations yields

$$\begin{array}{rcrcr} \chi_0 & & + & 1 & = & 0 \\ & \chi_1 & - & 2(1) & = & 0 \end{array}$$

Solving for χ_0 and χ_1 gives us the following non-trivial solution to $Ax = 0$:

$$x_n = \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}.$$

But if $Ax_n = 0$, then $A(\beta x_n) = 0$. This means that all vectors

$$x_s + \beta x_n = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}$$

solve the linear system. This is the general solution that we saw before.

In this particular example, it was not necessary to exchange (pivot) rows.

Homework 9.2.5.1 Find the general solution (an expression for all solutions) for

$$\begin{pmatrix} 2 & -2 & -4 \\ -2 & 1 & 4 \\ 2 & 0 & -4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}.$$

☛ SEE ANSWER

Homework 9.2.5.2 Find the general solution (an expression for all solutions) for

$$\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}.$$

☛ SEE ANSWER

9.3 Review of Sets

9.3.1 Definition and Notation

☛ **YouTube**

We very quickly discuss what a set is and some properties of sets. As part of discussing vector spaces, we will see lots of examples of sets and hence we keep examples down to a minimum.

Definition 9.3 *In mathematics, a set is defined as a collection of distinct objects.*

The objects that are members of a set are said to be its elements. If S is used to denote a given set and x is a member of that set, then we will use the notation $x \in S$ which is pronounced x is an element of S.

If x, y, and z are distinct objects that together are the collection that form a set, then we will often use the notation $\{x, y, z\}$ to describe that set. It is extremely important to realize that **order does not matter**: $\{x, y, z\}$ is the same set as $\{y, z, x\}$, and this is true for all ways in which you can order the objects.

A set itself is an object and hence once can have a set of sets, which has elements that are sets.

Definition 9.4 *The size of a set equals the number of distinct objects in the set.*

This size can be finite or infinite. If S denotes a set, then its size is denoted by $|S|$.

Definition 9.5 *Let S and T be sets. Then S is a subset of T if all elements of S are also elements of T. We use the notation $S \subset T$ or $T \supset S$ to indicate that S is a subset of T.*

Mathematically, we can state this as

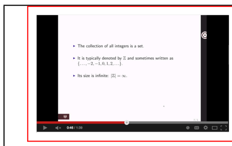
$$(S \subset T) \Leftrightarrow (x \in S \Rightarrow x \in T).$$

(S is a subset of T if and only if every element in S is also an element in T.)

Definition 9.6 *Let S and T be sets. Then S is a proper subset of T if all S is a subset of T and there is an element in T that is not in S. We use the notation $S \subsetneq T$ or $T \supsetneq S$ to indicate that S is a proper subset of T.*

Some texts will use the symbol $\subset$ to mean "proper subset" and $\subseteq$ to mean "subset". Get used to it! You'll have to figure out from context what they mean.

9.3.2 Examples



☛ YouTube

Examples

Example 9.7 The integers $1,2,3$ are a collection of three objects (the given integers). The set formed by these three objects is given by $\{1,2,3\}$ (again, emphasizing that order doesn't matter). The size of this set is $|\{1,2,3,\}| = 3$.

Example 9.8 The collection of all integers is a set. It is typically denoted by $\mathbb{Z}$ and sometimes written as $\{\ldots,-2,-1,0,1,2,\ldots\}$. Its size is infinite: $|\mathbb{Z}| = \infty$.

Example 9.9 The collection of all real numbers is a set that we have already encountered in our course. It is denoted by $\mathbb{R}$. Its size is infinite: $|\mathbb{R}| = \infty$. We cannot enumerate it (it is uncountably infinite, which is the subject of other courses).

Example 9.10 The set of all vectors of size n whose components are real valued is denoted by $\mathbb{R}^n$.

9.3.3 Operations with Sets

☛ YouTube

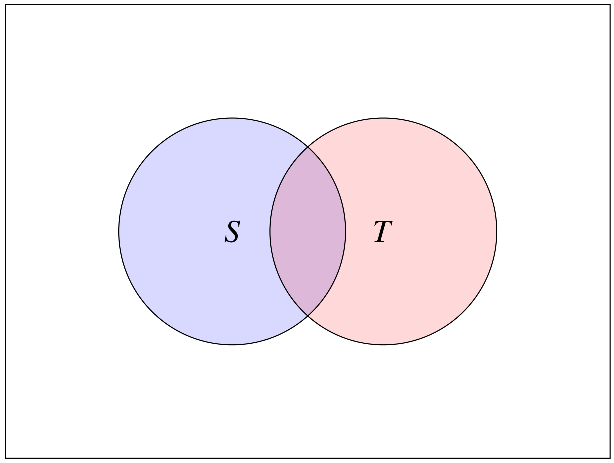
There are three operations on sets that will be of interest:

Definition 9.11 *The* **union** *of two sets S and T is the set of all elements that are in S* **or** *in T. This union is denoted by $S \cup T$.*

Formally, we can give the union as

$$S \cup T = \{x | x \in S \vee x \in T\}$$

which is read as "The union of S and T equals the set of all elements x such that x is in S or x is in T." (The "|" (vertical bar) means "such that" and the $\vee$ is the logical "or" operator.) It can be depicted by the shaded area (blue, pink, and purple) in the following Venn diagram:



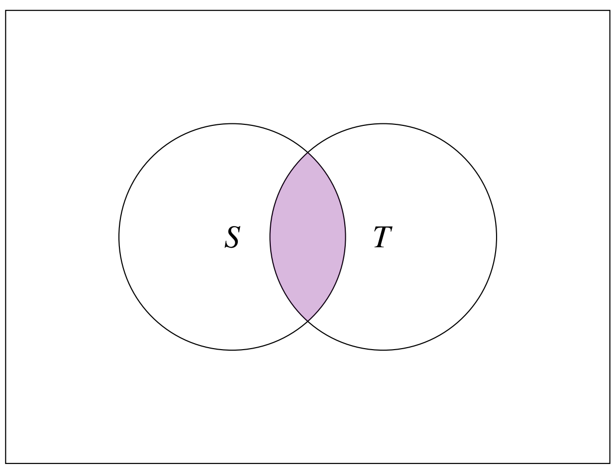
Example 9.12 Let $S = \{1,2,3\}$ and $T = \{2,3,5,8,9\}$. Then $S \cup T = \{1,2,3,5,8,9\}$.
What this example shows is that the size of the union is not necessarily the sum of the sizes of the individual sets.

Definition 9.13 *The* **intersection** *of two sets S and T is the set of all elements that are in S* **and** *in T. This intersection is denoted by $S \cap T$.*

Formally, we can give the intersection as

$$S \cap T = \{x | x \in S \wedge x \in T\}$$

which is read as "The intersection of S and T equals the set of all elements x such that x is in S and x is in T." (The "|" (vertical bar) means "such that" and the $\wedge$ is the logical "and" operator.) It can be depicted by the shaded area in the following Venn diagram:



Example 9.14 Let $S = \{1,2,3\}$ and $T = \{2,3,5,8,9\}$. Then $S \cap T = \{2,3\}$.

Example 9.15 Let $S = \{1,2,3\}$ and $T = \{5,8,9\}$. Then $S \cap T = \varnothing$ ($\varnothing$ is read as "the empty set").

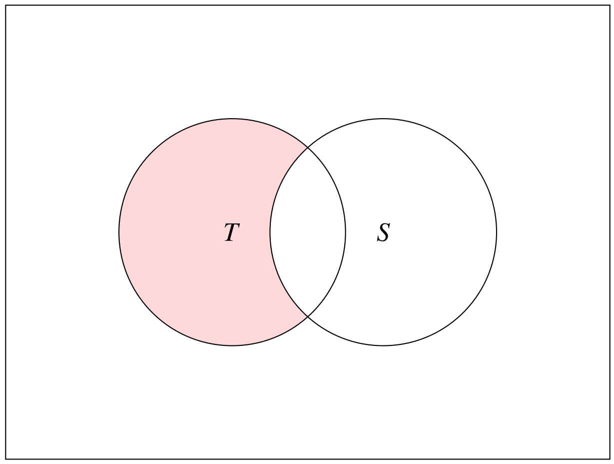
Definition 9.16 *The* **complement** *of set S with respect to set T is the set of all elements that are in T but are not in S. This complement is denoted by $T \backslash S$.*

Example 9.17 Let $S = \{1,2,3\}$ and $T = \{2,3,5,8,9\}$. Then $T \backslash S = \{5,8,9\}$ and $S \backslash T = \{1\}$.

Formally, we can give the complement as

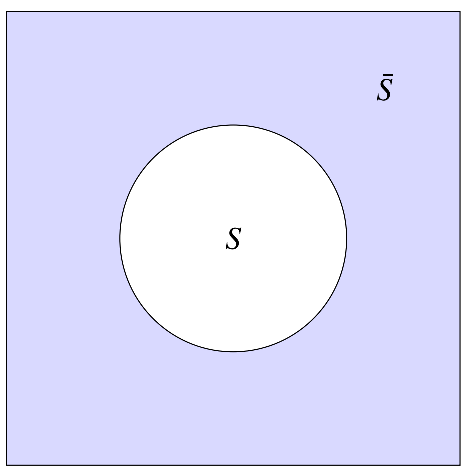
$$T \backslash S = \{x | x \notin S \wedge x \in T\}$$

which is read as "The complement of S with respect to T equals the set of all elements x such that x is not in S and x is in T." (The "|" (vertical bar) means "such that", $\wedge$ is the logical "and" operator, and the $\notin$ means "is not an element in".) It can be depicted by the shaded area in the following Venn diagram:



Sometimes, the notation $\bar{S}$ or S^c is used for the complement of set S. Here, the set with respect to which the complement is taken is "obvious from context".

For a single set S, the complement, $\bar{S}$ is shaded in the diagram below.



Homework 9.3.3.1 Let S and T be two sets. Then $S \subset S \cup T$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 9.3.3.2 Let S and T be two sets. Then $S \cap T \subset S$.

Always/Sometimes/Never

☛ SEE ANSWER

9.4 Vector Spaces

9.4.1 What is a Vector Space?

☛ YouTube

For our purposes, a vector space is a subset, S, of $\mathbb{R}^n$ with the following properties:

- $0 \in S$ (the zero vector of size n is in the set S); and
- If $v, w \in S$ then $(v + w) \in S$; and
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$.

A mathematician would describe the last two properties as "S is closed under addition and scalar multiplication." All the results that we will encounter for such vector spaces carry over to the case where the components of vectors are complex valued.

Example 9.18 The set $\mathbb{R}^n$ is a vector space:

- $0 \in \mathbb{R}^n$.
- If $v, w \in \mathbb{R}^n$ then $v + w \in \mathbb{R}^n$.
- If $v \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$ then $\alpha v \in \mathbb{R}^n$.

9.4.2 Subspaces

☛ YouTube

So, the question now becomes: "What subsets of $\mathbb{R}^n$ are vector spaces?" We will call such sets **subspaces** of $\mathbb{R}^n$.

Homework 9.4.2.1 Which of the following subsets of $\mathbb{R}^3$ are subspaces of $\mathbb{R}^3$?

1. The plane of vectors $x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$ such that $\chi_0 = 0$. In other words, the set of all vectors $\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} 0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \right\}$.

2. Similarly, the plane of vectors x with $\chi_0 = 1$: $\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} 1 \\ \chi_1 \\ \chi_2 \end{pmatrix} \right\}$.

3. $\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \wedge \chi_0\chi_1 = 0 \right\}$. (Recall, $\wedge$ is the logical "and" operator.)

4. $\left\{ x \in \mathbb{R}^3 \middle| x = \beta_0 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} \text{ where } \beta_0, \beta_1 \in \mathbb{R} \right\}$.

5. $\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \wedge \chi_0 - \chi_1 + 3\chi_2 = 0 \right\}$.

☛ SEE ANSWER

Homework 9.4.2.2 The empty set, $\varnothing$, is a subspace of $\mathbb{R}^n$.

True/False

☛ SEE ANSWER

Homework 9.4.2.3 The set $\{0\}$ where 0 is a vector of size n is a subspace of $\mathbb{R}^n$.

True/False

☛ SEE ANSWER

Homework 9.4.2.4 The set $S \subset \mathbb{R}^n$ described by

$$\{x \mid \|x\|_2 < 1\}.$$

is a subspace of $\mathbb{R}^n$. (Recall that $\|x\|_2$ is the Euclidean length of vector x so this describes all elements with length less than or equal to one.)

True/False

☛ SEE ANSWER

Homework 9.4.2.5 The set $S \subset \mathbb{R}^n$ described by

$$\left\{ \left(\begin{array}{c} \nu_0 \\ 0 \\ \vdots \\ 0 \end{array} \right) \middle| \, \nu_0 \in \mathbb{R} \right\}$$

is a subspace of $\mathbb{R}^n$.

True/False

☛ SEE ANSWER

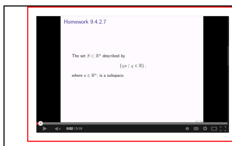
Homework 9.4.2.6 The set $S \subset \mathbb{R}^n$ described by

$$\{\nu e_j \mid \nu \in \mathbb{R}\},$$

where e_j is a unit basis vector, is a subspace.

True/False

☛ SEE ANSWER



☛ YouTube

Homework 9.4.2.7 The set $S \subset \mathbb{R}^n$ described by

$$\{\chi a \mid \chi \in \mathbb{R}\},$$

where $a \in \mathbb{R}^n$, is a subspace.

True/False

☛ SEE ANSWER

Homework 9.4.2.8 The set $S \subset \mathbb{R}^n$ described by

$$\{\chi_0 a_0 + \chi_1 a_1 \mid \chi_0, \chi_1 \in \mathbb{R}\},$$

where $a_0, a_1 \in \mathbb{R}^n$, is a subspace.

True/False

☛ SEE ANSWER

Homework 9.4.2.9 The set $S \subset \mathbb{R}^m$ described by

$$\left\{ \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \left(\begin{array}{c} \chi_0 \\ \chi_1 \end{array} \right) \middle| \, \chi_0, \chi_1 \in \mathbb{R} \right\},$$

where $a_0, a_1 \in \mathbb{R}^m$, is a subspace.

True/False

☛ SEE ANSWER

Homework 9.4.2.10 The set $S \subset \mathbb{R}^m$ described by

$$\left\{ Ax \mid x \in \mathbb{R}^2 \right\},$$

where $A \in \mathbb{R}^{m \times 2}$, is a subspace.

True/False

☛ SEE ANSWER

9.4.3 The Column Space

☛ **YouTube**

Homework 9.4.3.1 The set $S \subset \mathbb{R}^m$ described by

$$\left\{ Ax \mid x \in \mathbb{R}^n \right\},$$

where $A \in \mathbb{R}^{m \times n}$, is a subspace.

True/False

☛ SEE ANSWER

This last exercise very precisely answers the question of when a linear system of equation, expressed as the matrix equation $Ax = b$, has a solution: it has a solution only if b is an element of the space S in this last exercise.

Definition 9.19 *Let $A \in \mathbb{R}^{m \times n}$. Then the* **column space** *of A equals the set*

$$\left\{ Ax \mid x \in \mathbb{R}^n \right\}.$$

It is denoted by $\mathcal{C}(A)$.

The name "column space" comes from the observation (which we have made many times by now) that

$$Ax = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}.$$

Thus $\mathcal{C}(A)$ equals the set of all linear combinations of the columns of matrix A.

Theorem 9.20 *The column space of $A \in \mathbb{R}^{m \times n}$ is a subspace of $\mathbb{R}^m$.*

Proof: The last exercise proved this.

Theorem 9.21 *Let $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, and $b \in \mathbb{R}^m$. Then $Ax = b$ has a solution if and only if $b \in \mathcal{C}(A)$.*

Proof: Recall that to prove an "if and only if" statement $P \Leftrightarrow Q$, you may want to instead separately prove $P \Rightarrow Q$ and $P \Leftarrow Q$.

($\Rightarrow$) Assume that $Ax = b$. Then $b \in \{Ax | x \in \mathbb{R}^n\}$. Hence b is in the column space of A.

($\Leftarrow$) Assume that b is in the column space of A. Then $b \in \{Ax | x \in \mathbb{R}^n\}$. But this means there exists a vector x such that $Ax = b$.

Homework 9.4.3.2 Match the matrices on the left to the column space on the right. (You should be able to do this "by examination.")

1. $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$
2. $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$
3. $\begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}$
4. $\begin{pmatrix} 0 & 0 \\ 1 & -2 \end{pmatrix}$
5. $\begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}$
6. $\begin{pmatrix} 1 & 0 \\ 2 & 3 \end{pmatrix}$
7. $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$
8. $\begin{pmatrix} 1 & -2 \\ 2 & -4 \end{pmatrix}$
9. $\begin{pmatrix} 1 & -2 & -1 \\ 2 & -4 & -2 \end{pmatrix}$

a. $\mathbb{R}^2$.

b. $\left\{ \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \middle| \chi_0 = 0 \vee \chi_1 = 0 \right\}$

c. $\left\{ \begin{pmatrix} \alpha \\ 0 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

d. $\left\{ \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

e. $\left\{ \alpha \begin{pmatrix} 1 \\ 2 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

f. $\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix} \right\}$

(Recall that $\vee$ is the logical "or" operator.)

☛ SEE ANSWER

Homework 9.4.3.3 Which of the following matrices have a FINITE number of elements in their column space? (Mark all that apply.)

1. The identity matrix.
2. The zero matrix.
3. All matrices.
4. None of the above.

☛ SEE ANSWER

9.4.4 The Null Space

☛ YouTube

Recall:

- We are interested in the solutions of $Ax = b$.
- We have already seen that if $Ax_s = b$ and $Ax_n = 0$ then $x_s + x_n$ is also a solution:

$$A(x_s + x_n) = b.$$

Definition 9.22 *Let $A \in \mathbb{R}^{m \times n}$. Then the set of all vectors $x \in \mathbb{R}^n$ that have the property that $Ax = 0$ is called the null space of A and is denoted by*

$$\mathcal{N}(A) = \{x | Ax = 0\}.$$

Homework 9.4.4.1 Let $A \in \mathbb{R}^{m \times n}$. The null space of A, $\mathcal{N}(A)$, is a subspace

True/False

☛ SEE ANSWER

Homework 9.4.4.2 For each of the matrices on the left match the set of vectors on the right that describes its null space. (You should be able to do this "by examination.")

1. $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$
2. $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$
3. $\begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}$
4. $\begin{pmatrix} 0 & 0 \\ 1 & -2 \end{pmatrix}$
5. $\begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}$
6. $\begin{pmatrix} 1 & 0 \\ 2 & 3 \end{pmatrix}$
7. $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$
8. $\begin{pmatrix} 1 & -2 \\ 2 & -4 \end{pmatrix}$

a. $\mathbb{R}^2$.

b. $\left\{ \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \middle| \chi_0 = 0 \vee \chi_1 = 0 \right\}$

c. $\left\{ \begin{pmatrix} \alpha \\ 0 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

d. $\emptyset$

e. $\left\{ \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

f. $\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix} \right\}$

g. $\left\{ \begin{pmatrix} 0 \end{pmatrix} \right\}$

h. $\left\{ \alpha \begin{pmatrix} 1 \\ 2 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

i. $\left\{ \alpha \begin{pmatrix} 2 \\ 1 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

(Recall that $\vee$ is the logical "or" operator.)

☛ SEE ANSWER

9.5 Span, Linear Independence, and Bases

9.5.1 Span

☛ YouTube

What is important about vector (sub)spaces is that if you have one or more vectors in that space, then it is possible to generate other vectors in the subspace by taking linear combinations of the original known vectors.

Example 9.23

$$\left\{ \alpha_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \alpha_1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \middle| \alpha_0, \alpha_1 \in \mathbb{R} \right\}$$

is the set of all linear combinations of the unit basis vectors $e_0, e_1 \in \mathbb{R}^2$. Notice that all of $\mathbb{R}^2$ (an uncountable infinite set) can be described with just these two vectors.

We have already seen that, given a set of vectors, the set of all linear combinations of those vectors is a subspace. We now give a name to such a set of linear combinations.

Definition 9.24 *Let $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$. Then the span of these vectors, Span$\{v_0, v_1, \cdots, v_{n-1}\}$, is said to be the set of all vectors that are a linear combination of the given set of vectors.*

Example 9.25

$$\text{Span}\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \mathbb{R}^2.$$

Example 9.26 Consider the equation $\chi_0 + 2\chi_1 - \chi_2 = 0$. It defines a subspace. In particular, that subspace is the null space of the matrix $\begin{pmatrix} 1 & 2 & -1 \end{pmatrix}$. We know how to find two vectors in that nullspace:

$$\left(\begin{array}{ccc|c} \boxed{1} & 2 & -1 & 0 \end{array} \right)$$

The box identifies the pivot. Hence, the free variables are χ_1 and χ_2. We first set $\chi_1 = 1$ and $\chi_2 = 0$ and solve for χ_0. Then we set $\chi_1 = 0$ and $\chi_2 = 1$ and again solve for χ_0. This gives us the vectors

$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} -2 \\ 1 \\ 0 \end{pmatrix}.$$

We know that any linear combination of these vectors also satisfies the equation (is also in the null space). Hence, we know that any vector in

$$\text{Span}\left(\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} -2 \\ 1 \\ 0 \end{pmatrix} \right)$$

is also in the null space of the matrix. Thus, any vector in that set satisfies the equation given at the start of this example.

We will later see that the vectors in this last example "span" the entire null space for the given matrix. But we are not quite ready to claim that.

We have learned three things in this course that relate to this discussion:

- Given a set of vectors $\{v_0, v_1, \ldots, v_{n-1}\} \subset \mathbb{R}^n$, we can create a matrix that has those vectors as its columns:

$$V = \left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_{n-1} \end{array} \right).$$

- Given a matrix $V \in \mathbb{R}^{m \times n}$ and vector $x \in \mathbb{R}^n$,

$$Vx = \chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1}.$$

In other words, Vx takes a linear combination of the columns of V.

- The column space of V, $\mathcal{C}(V)$, is the set (subspace) of all linear combinations of the columns of V:

$$\mathcal{C}(V) = \{Vx | x \in \mathbb{R}^n\} = \{\chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1} | \chi_0, \chi_1, \ldots, \chi_{n-1} \in \mathbb{R}\}.$$

We conclude that

If $V = \left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_{n-1} \end{array} \right)$, then $\text{Span}(v_0, v_1, \ldots, v_{n-1}) = \mathcal{C}(V)$.

Definition 9.27 *A spanning set of a subspace S is a set of vectors $\{v_0, v_1, \ldots, v_{n-1}\}$ such that Span$(\{v_0, v_1, \ldots, v_{n-1}\}) = S$.*

9.5.2 Linear Independence

☛ **YouTube**

Example 9.28 We show that $\text{Span}\left(\left\{\begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}\right\}\right) = \text{Span}\left(\left\{\begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)$. One can either simply recognize that both sets equal all of $\mathbb{R}^2$, or one can reason it by realizing that in order to show that sets S and T are equal one can just show that both $S \subset T$ and $T \subset S$:

- $S \subset T$: Let $x \in \text{Span}\left(\begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}\right)$ Then there exist α_0 and α_1 such that $x = \alpha_0 \begin{pmatrix}1\\0\\0\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\1\\0\end{pmatrix}$. This in turn means that $x = \alpha_0 \begin{pmatrix}1\\0\\0\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\1\\0\end{pmatrix} + (0) \begin{pmatrix}1\\1\\0\end{pmatrix}$. Hence

$$x \in \text{Span}\left(\left\{\begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right).$$

- $T \subset S$: Let $x \in \text{Span}\left(\left\{\begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}1\\1\\0\end{pmatrix}\right\}\right)$. Then there exist α_0, α_1, and α_2 such that $x = \alpha_0 \begin{pmatrix}1\\0\\0\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\1\\0\end{pmatrix} + \alpha_2 \begin{pmatrix}1\\1\\0\end{pmatrix}$. But $\begin{pmatrix}1\\1\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\end{pmatrix} + \begin{pmatrix}0\\1\\0\end{pmatrix}$. Hence

$$x = \alpha_0 \begin{pmatrix}1\\0\\0\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\1\\0\end{pmatrix} + \alpha_2 \left(\begin{pmatrix}1\\0\\0\end{pmatrix} + \begin{pmatrix}0\\1\\0\end{pmatrix}\right) = (\alpha_0 + \alpha_2) \begin{pmatrix}1\\0\\0\end{pmatrix} + (\alpha_1 + \alpha_2) \begin{pmatrix}0\\1\\0\end{pmatrix}.$$

Therefore $x \in \text{Span}\left(\left\{\begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}\right\}\right)$.

Homework 9.5.2.1

$$\text{Span}\left(\left\{\begin{pmatrix}1\\0\\1\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix}\right\}\right)=\text{Span}\left(\left\{\begin{pmatrix}1\\0\\1\end{pmatrix},\begin{pmatrix}0\\0\\1\end{pmatrix},\begin{pmatrix}1\\0\\3\end{pmatrix}\right\}\right)$$

True/False

☛ SEE ANSWER

You might be thinking that needing fewer vectors to describe a subspace is better than having more, and we'd agree with you!

In both examples and in the homework, the set on the right of the equality sign identifies three vectors to identify the subspace rather than the two required for the equivalent set to its left. The issue is that at least one (indeed all) of the vectors can be written as linear combinations of the other two. Focusing on the exercise, notice that

$$\begin{pmatrix}1\\1\\0\end{pmatrix}=\begin{pmatrix}1\\0\\0\end{pmatrix}+\begin{pmatrix}0\\1\\0\end{pmatrix}.$$

Thus, any linear combination

$$\alpha_0\begin{pmatrix}1\\0\\1\end{pmatrix}+\alpha_1\begin{pmatrix}0\\0\\1\end{pmatrix}+\alpha_2\begin{pmatrix}1\\0\\3\end{pmatrix}$$

can also be generated with only the first two vectors:

$$\alpha_0\begin{pmatrix}1\\0\\1\end{pmatrix}+\alpha_1\begin{pmatrix}0\\0\\1\end{pmatrix}+\alpha_2\begin{pmatrix}1\\0\\3\end{pmatrix}=(\alpha_0+\alpha_2)\begin{pmatrix}1\\0\\1\end{pmatrix}+(\alpha_0+2\alpha_2)\begin{pmatrix}0\\0\\1\end{pmatrix}$$

We now introduce the concept of linear (in)dependence to cleanly express when it is the case that a set of vectors has elements that are redundant in this sense.

Definition 9.29 *Let $\{v_0,\ldots,v_{n-1}\}\subset\mathbb{R}^m$. Then this set of vectors is said to be linearly independent if $\chi_0 v_0+\chi_1 v_1+\cdots+\chi_{n-1}v_{n-1}=0$ implies that $\chi_0=\cdots=\chi_{n-1}=0$. A set of vectors that is not linearly independent is said to be linearly dependent.*

Homework 9.5.2.2 Let the set of vectors $\{a_0,a_1,\ldots,a_{n-1}\}\subset\mathbb{R}^m$ be linearly dependent. Then at least one of these vectors can be written as a linear combination of the others.

True/False

☛ SEE ANSWER

This last exercise motivates the term *linearly independent* in the definition: none of the vectors can be written as a linear combination of the other vectors.

Example 9.30 The set of vectors

$$\left\{\begin{pmatrix}1\\0\\0\end{pmatrix},\begin{pmatrix}0\\1\\0\end{pmatrix},\begin{pmatrix}1\\1\\0\end{pmatrix}\right\}$$

is linearly dependent:

$$\begin{pmatrix}1\\0\\0\end{pmatrix}+\begin{pmatrix}0\\1\\0\end{pmatrix}-\begin{pmatrix}1\\1\\0\end{pmatrix}=\begin{pmatrix}0\\0\\0\end{pmatrix}.$$

Theorem 9.31 *Let $\{a_0,\ldots,a_{n-1}\} \subset \mathbb{R}^m$ and let $A = \left(\begin{array}{c|c|c} a_0 & \cdots & a_{n-1} \end{array} \right)$. Then the vectors $\{a_0,\ldots,a_{n-1}\}$ are linearly independent if and only if $\mathcal{N}(A) = \{0\}$.*

Proof:

(⇒) Assume $\{a_0,\ldots,a_{n-1}\}$ are linearly independent. We need to show that $\mathcal{N}(A) = \{0\}$. Assume $x \in \mathcal{N}(A)$. Then $Ax = 0$ implies that

$$\begin{array}{rcl} 0 & = & Ax = \left(\begin{array}{c|c|c} a_0 & \cdots & a_{n-1} \end{array} \right) \left(\begin{array}{c} \chi_0 \\ \vdots \\ \chi_{n-1} \end{array} \right) \\ & = & \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} \end{array}$$

and hence $\chi_0 = \cdots = \chi_{n-1} = 0$. Hence $x = 0$.

(⇐) Notice that we are trying to prove $P \Leftarrow Q$, where P represents "the vectors $\{a_0,\ldots,a_{n-1}\}$ are linearly independent" and Q represents "$\mathcal{N}(A) = \{0\}$". It suffices to prove the **contrapositive**: $\neg P \Rightarrow \neg Q$. (Note that $\neg$ means "not") Assume that $\{a_0,\ldots,a_{n-1}\}$ are *not* linearly independent. Then there exist $\{\chi_0,\cdots,\chi_{n-1}\}$ with at least one $\chi_j \neq 0$ such that $\chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} = 0$. Let $x = (\chi_0,\ldots,\chi_{n-1})^T$. Then $Ax = 0$ which means $x \in \mathcal{N}(A)$ and hence $\mathcal{N}(A) \neq \{0\}$.

Example 9.32 In the last example, we could have taken the three vectors to be the columns of a 3×3 matrix A and checked if $Ax = 0$ has a solution:

$$\left(\begin{array}{ccc} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{array} \right) \left(\begin{array}{r} 1 \\ 1 \\ -1 \end{array} \right) = \left(\begin{array}{c} 0 \\ 0 \\ 0 \end{array} \right)$$

Because there is a non-trivial solution to $Ax = 0$, the nullspace of A has more than just the zero vector in it, and the columns of A are linearly dependent.

Example 9.33 The columns of an identity matrix $I \in \mathbb{R}^{n \times n}$ form a linearly independent set of vectors.

Proof: Since I has an inverse (I itself) we know that $\mathcal{N}(I) = \{0\}$. Thus, by Theorem 9.31, the columns of I are linearly independent.

Example 9.34 The columns of $L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & -1 & 0 \\ 1 & 2 & 3 \end{pmatrix}$ are linearly independent. If we consider

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & -1 & 0 \\ 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

and simply solve this, we find that $\chi_0 = 0/1 = 0$, $\chi_1 = (0 - 2\chi_0)/(-1) = 0$, and $\chi_2 = (0 - \chi_0 - 2\chi_1)/(3) = 0$. Hence, $\mathcal{N}(L) = \{0\}$ (the zero vector) and we conclude, by Theorem 9.31, that the columns of L are linearly independent.

The last example motivates the following theorem:

Theorem 9.35 *Let $L \in \mathbb{R}^{n \times n}$ be a lower triangular matrix with nonzeroes on its diagonal. Then its columns are linearly independent.*

Proof: Let L be as indicated and consider $Lx = 0$. If one solves this via whatever method one pleases, the solution $x = 0$ will emerge as the only solution. Thus $\mathcal{N}(L) = \{0\}$ and by Theorem 9.31, the columns of L are linearly independent.

Homework 9.5.2.3 Let $U \in \mathbb{R}^{n \times n}$ be an upper triangular matrix with nonzeroes on its diagonal. Then its columns are linearly independent.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 9.5.2.4 Let $L \in \mathbb{R}^{n \times n}$ be a lower triangular matrix with nonzeroes on its diagonal. Then its rows are linearly independent. (Hint: How do the rows of L relate to the columns of L^T?)

Always/Sometimes/Never

☛ SEE ANSWER

Example 9.36 The columns of $L = \left(\begin{array}{ccc} 1 & 0 & 0 \\ 2 & -1 & 0 \\ 1 & 2 & 3 \\ \hline -1 & 0 & -2 \end{array}\right)$ are linearly independent. If we consider

$$\left(\begin{array}{ccc} 1 & 0 & 0 \\ 2 & -1 & 0 \\ 1 & 2 & 3 \\ \hline -1 & 0 & -2 \end{array}\right) \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

and simply solve this, we find that $\chi_0 = 0/1 = 0$, $\chi_1 = (0 - 2\chi_0)/(-1) = 0$, $\chi_2 = (0 - \chi_0 - 2\chi_1)/(3) = 0$. Hence, $\mathcal{N}(L) = \{0\}$ (the zero vector) and we conclude, by Theorem 9.31, that the columns of L are linearly independent.

Next, we observe that if one has a set of more than m vectors in $\mathbb{R}^m$, then they must be linearly dependent:

Theorem 9.37 *Let $\{a_0, a_1, \ldots, a_{n-1}\} \in \mathbb{R}^m$ and $n > m$. Then these vectors are linearly dependent.*

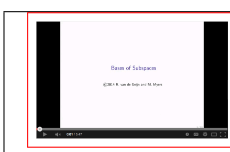
Proof: Consider the matrix $A = \left(\begin{array}{c|c|c} a_0 & \cdots & a_{n-1} \end{array} \right)$. If one applies the Gauss-Jordan method to this matrix in order to get it to upper triangular form, at most m columns with pivots will be encountered. The other $n-m$ columns correspond to free variables, which allow us to construct nonzero vectors x so that $Ax = 0$.

The observations in this unit allows us to add to our conditions related to the invertibility of matrix A:

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.
- LU with partial pivoting does not break down.
- $\mathcal{C}(A) = \mathbb{R}^n$.
- A has linearly independent columns.
- $\mathcal{N}(A) = \{0\}$.

9.5.3 Bases for Subspaces



☛ YouTube

In the last unit, we started with an example and then an exercise that showed that if we had three vectors and one of the three vectors could be written as a linear combination of the other two, then the span of the three vectors was equal to the span of the other two vectors.

It turns out that this can be generalized:

Definition 9.38 *Let S be a subspace of $\mathbb{R}^m$. Then the set $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ is said to be a basis for S if (1) $\{v_0, v_1, \cdots, v_{n-1}\}$ are linearly independent and (2) $Span\{v_0, v_1, \cdots, v_{n-1}\} = S$.*

Homework 9.5.3.1 The vectors $\{e_0, e_1, \ldots, e_{n-1}\} \subset \mathbb{R}^n$ are a basis for $\mathbb{R}^n$.

True/False

☛ SEE ANSWER

Example 9.39 Let $\{a_0,\ldots,a_{n-1}\} \subset \mathbb{R}^n$ and let $A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$ be invertible. Then $\{a_0,\ldots,a_{n-1}\} \subset \mathbb{R}^n$ form a basis for $\mathbb{R}^n$.
Note: The fact that A is invertible means there exists A^{-1} such that $A^{-1}A = I$. Since $Ax = 0$ means $x = A^{-1}Ax = A^{-1}0 = 0$, the columns of A are linearly independent. Also, given any vector $y \in \mathbb{R}^n$, there exists a vector $x \in \mathbb{R}^n$ such that $Ax = y$ (namely $x = A^{-1}y$). Letting $x = \begin{pmatrix} \chi_0 \\ \vdots \\ \chi_{n-1} \end{pmatrix}$ we find that $y = \chi_0 a_0 + \cdots + \chi_{n-1}a_{n-1}$ and hence every vector in $\mathbb{R}^n$ is a linear combination of the set $\{a_0,\ldots,a_{n-1}\} \subset \mathbb{R}^n$.

Lemma 9.40 *Let $S \subset \mathbb{R}^m$. Then S contains at most m linearly independent vectors.*

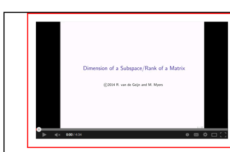
Proof: Proof by contradiction. We will assume that S contains more than m linearly independent vectors and show that this leads to a contradiction.

Since S contains more than m linearly independent vectors, it contains at least $m+1$ linearly independent vectors. Let us label $m+1$ such vectors $v_0, v_1, \ldots, v_{m-1}, v_m$. Let $V = \left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_m \end{array} \right)$. This matrix is $m \times (m+1)$ and hence there exists a nontrivial x_n such that $Vx_n = 0$. (This is an equation with m equations and $m+1$ unknowns.) Thus, the vectors $\{v_0, v_1, \cdots, v_m\}$ are linearly dependent, which is a contradiction.

Theorem 9.41 *Let S be a nontrivial subspace of $\mathbb{R}^m$. (In other words, $S \neq \{0\}$.) Then there exists a basis $\{v_0, v_1, \ldots, v_{n-1}\} \subset \mathbb{R}^m$ such that $Span(v_0, v_1, \ldots, v_{n-1}) = S$.*

Proof: Notice that we have already established that $m \leq n$. We will construct the vectors. Let S be a nontrivial subspace. Then S contains at least one nonzero vector. Let v_0 equal such a vector. Now, either $\text{Span}(v_0) = S$ in which case we are done or $S \backslash \text{Span}(v_0)$ is not empty, in which case we can pick some vector in $S \backslash \text{Span}(v_0)$ as v_1. Next, either $\text{Span}(v_0, v_1) = S$ in which case we are done or $S \backslash \text{Span}(v_0, v_1)$ is not empty, in which case we pick some vector in $S \backslash \text{Span}(v_0, v_1)$ as v_2. This process continues until we have a basis for S. It can be easily shown that the vectors are all linearly independent.

9.5.4 The Dimension of a Subspace



☛ YouTube

We have established that every nontrivial subspace of $\mathbb{R}^m$ has a basis with n vectors. This basis is not unique. After all, we can simply multiply all the vectors in the basis by a nonzero constant and contruct a new basis. What we'll establish now is that the number of vectors in a basis for a given subspace is always the same. This number then becomes the dimension of the subspace.

Theorem 9.42 *Let S be a subspace of $\mathbb{R}^m$ and let $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ and $\{w_0, w_1, \cdots, w_{k-1}\} \subset \mathbb{R}^m$ both be bases for S. Then $k = n$. In other words, the number of vectors in a basis is unique.*

Proof: Proof by contradiction. Without loss of generality, let us assume that $k > n$. (Otherwise, we can switch the roles of the two sets.) Let $V = \left(\begin{array}{c|c|c} v_0 & \cdots & v_{n-1} \end{array} \right)$ and $W = \left(\begin{array}{c|c|c} w_0 & \cdots & w_{k-1} \end{array} \right)$. Let x_j have the property that $w_j = V x_j$. (We know such a vector x_j exists because V spans $\mathbf{V}$ and $w_j \in \mathbf{V}$.) Then $W = VX$, where $X = \left(\begin{array}{c|c|c} x_0 & \cdots & x_{k-1} \end{array} \right)$. Now, $X \in \mathbb{R}^{n \times k}$ and recall that $k > n$. This means that $\mathcal{N}(X)$ contains nonzero vectors (why?). Let $y \in \mathcal{N}(X)$. Then $Wy = VXy = V(Xy) = V(0) = 0$, which contradicts the fact that $\{w_0, w_1, \cdots, w_{k-1}\}$ are linearly independent, and hence this set cannot be a basis for $\mathbf{V}$.

Definition 9.43 *The dimension of a subspace S equals the number of vectors in a basis for that subspace.*

A basis for a subspace S can be derived from a spanning set of a subspace S by, one-to-one, removing vectors from the set that are dependent on other remaining vectors until the remaining set of vectors is linearly independent , as a consequence of the following observation:

Definition 9.44 *Let $A \in \mathbb{R}^{m \times n}$. The rank of A equals the number of vectors in a basis for the column space of A. We will let* $\text{rank}(A)$ *denote that rank.*

Theorem 9.45 *Let $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ be a spanning set for subspace S and assume that v_i equals a linear combination of the other vectors. Then $\{v_0, v_1, \cdots, v_{i-1}, v_{i+1}, \cdots, v_{n-1}\}$ is a spanning set of S.*

Similarly, a set of linearly independent vectors that are in a subspace S can be "built up" to be a basis by successively adding vectors that are in S to the set while maintaining that the vectors in the set remain linearly independent until the resulting is a basis for S.

Theorem 9.46 *Let $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ be linearly independent and assume that $\{v_0, v_1, \cdots, v_{n-1}\} \subset S$ where S is a subspace. Then this set of vectors is either a spanning set for S or there exists $w \in S$ such that $\{v_0, v_1, \cdots, v_{n-1}, w\}$ are linearly independent.*

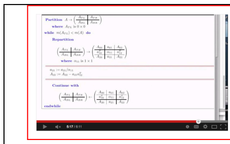
We can add some more conditions regarding the invertibility of matrix A:

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.
- LU with partial pivoting does not break down.
- $\mathcal{C}(A) = \mathbb{R}^n$.
- A has linearly independent columns.
- $\mathcal{N}(A) = \{0\}$.
- $\text{rank}(A) = n$.

9.6 Enrichment

9.6.1 Typesetting algorithms with the FLAME notation



☛ YouTube

9.7 Wrap Up

9.7.1 Homework

No additional homework this week.

9.7.2 Summary

Solution(s) to linear systems

Whether a linear system of equations $Ax = b$ has a unique solution, no solution, or multiple solutions can be determined by writing the system as an appended system

$$\left(\begin{array}{c|c} A & b \end{array} \right)$$

and transforming this appended system to row echelon form, swapping rows if necessary.

When A is square, conditions for the solution to be unique were discussed in Weeks 6-8.

Examples of when it has a unique solution, no solution, or multiple solutions when $m \neq n$ were given in this week, but this will become more clear in Week 10. Therefore, we won't summarize it here.

Sets

Definition 9.47 *In mathematics, a set is defined as a collection of distinct objects.*

- The objects that are members of a set are said to be its elements.
- The notation $x \in S$ is used to indicate that x is an element in set S.

Definition 9.48 *The size of a set equals the number of distinct objects in the set. It is denoted by $|S|$.*

Definition 9.49 *Let S and T be sets. Then S is a subset of T if all elements of S are also elements of T. We use the notation $S \subset T$ to indicate that S is a subset of T:*

$$(S \subset T) \Leftrightarrow (x \in S \Rightarrow x \in T).$$

Definition 9.50 *The* **union** *of two sets S and T is the set of all elements that are in S* **or** *in T. This union is denoted by $S \cup T$:*

$$S \cup T = \{x | x \in S \vee x \in T.\}$$

Definition 9.51 *The* **intersection** *of two sets S and T is the set of all elements that are in S* **and** *in T. This intersection is denoted by $S \cap T$:*

$$S \cap T = \{x | x \in S \wedge x \in T.\}$$

Definition 9.52 *The* **complement** *of set S with respect to set T is the set of all elements that are in T but are not in S. This complement is denoted by $T \backslash S$:*

$$T \backslash S = \{x | x \notin S \wedge x \in T\}$$

Vector spaces

For our purposes, a vector space is a subset, S, of $\mathbb{R}^n$ with the following properties:

- $0 \in S$ (the zero vector of size n is in the set S); and
- If $v, w \in S$ then $(v+w) \in S$; and
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$.

Definition 9.53 *A subset of $\mathbb{R}^n$ is said to be a subspace of $\mathbb{R}^n$ is it a vector space.*

Definition 9.54 *Let $A \in \mathbb{R}^{m \times n}$. Then the* **column space** *of A equals the set*

$$\{Ax \mid x \in \mathbb{R}^n\}.$$

It is denoted by $\mathcal{C}(A)$.

The name "column space" comes from the observation (which we have made many times by now) that

$$Ax = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}.$$

Thus $\mathcal{C}(A)$ equals the set of all linear combinations of the columns of matrix A.

Theorem 9.55 *The column space of $A \in \mathbb{R}^{m \times n}$ is a subspace of $\mathbb{R}^m$.*

Theorem 9.56 *Let $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, and $b \in \mathbb{R}^m$. Then $Ax = b$ has a solution if and only if $b \in \mathcal{C}(A)$.*

Definition 9.57 *Let $A \in \mathbb{R}^{m \times n}$. Then the set of all vectors $x \in \mathbb{R}^n$ that have the property that $Ax = 0$ is called the null space of A and is denoted by*

$$\mathcal{N}(A) = \{x | Ax = 0\}.$$

Span, Linear Dependence, Bases

Definition 9.58 *Let $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$. Then the span of these vectors, Span$\{v_0, v_1, \cdots, v_{n-1}\}$, is said to be the set of all vectors that are a linear combination of the given set of vectors.*

If $V = \left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_{n-1} \end{array} \right)$, then $\mathrm{Span}(v_0, v_1, \ldots, v_{n-1}) = \mathcal{C}(V)$.

Definition 9.59 *A spanning set of a subspace S is a set of vectors $\{v_0, v_1, \ldots, v_{n-1}\}$ such that Span$(\{v_0, v_1, \ldots, v_{n-1}\}) = S$.*

Definition 9.60 *Let $\{v_0, \ldots, v_{n-1}\} \subset \mathbb{R}^m$. Then this set of vectors is said to be linearly independent if $\chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1} = 0$ implies that $\chi_0 = \cdots = \chi_{n-1} = 0$. A set of vectors that is not linearly independent is said to be linearly dependent.*

Theorem 9.61 *Let the set of vectors $\{a_0, a_1, \ldots, a_{n-1}\} \subset \mathbb{R}^m$ be linearly dependent. Then at least one of these vectors can be written as a linear combination of the others.*

This last theorem motivates the term *linearly independent* in the definition: none of the vectors can be written as a linear combination of the other vectors.

Theorem 9.62 *Let $\{a_0, \ldots, a_{n-1}\} \subset \mathbb{R}^m$ and let $A = \left(\begin{array}{c|c|c} a_0 & \cdots & a_{n-1} \end{array} \right)$. Then the vectors $\{a_0, \ldots, a_{n-1}\}$ are linearly independent if and only if $\mathcal{N}(A) = \{0\}$.*

Theorem 9.63 *Let* $\{a_0, a_1, \ldots, a_{n-1}\} \in \mathbb{R}^m$ *and* $n > m$. *Then these vectors are linearly dependent.*

Definition 9.64 *Let* S *be a subspace of* $\mathbb{R}^m$. *Then the set* $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ *is said to be a basis for* S *if (1)* $\{v_0, v_1, \cdots, v_{n-1}\}$ *are linearly independent and (2) Span*$\{v_0, v_1, \cdots, v_{n-1}\} = S$.

Theorem 9.65 *Let* S *be a subspace of* $\mathbb{R}^m$ *and let* $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ *and* $\{w_0, w_1, \cdots, w_{k-1}\} \subset \mathbb{R}^m$ *both be bases for* S. *Then* $k = n$. *In other words, the number of vectors in a basis is unique.*

Definition 9.66 *The dimension of a subspace* S *equals the number of vectors in a basis for that subspace.*

Definition 9.67 *Let* $A \in \mathbb{R}^{m \times n}$. *The rank of* A *equals the number of vectors in a basis for the column space of* A. *We will let* $\text{rank}(A)$ *denote that rank.*

Theorem 9.68 *Let* $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ *be a spanning set for subspace* S *and assume that* v_i *equals a linear combination of the other vectors. Then* $\{v_0, v_1, \cdots, v_{i-1}, v_{i+1}, \cdots, v_{n-1}\}$ *is a spanning set of* S.

Theorem 9.69 *Let* $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ *be linearly independent and assume that* $\{v_0, v_1, \cdots, v_{n-1}\} \subset S$ *where* S *is a subspace. Then this set of vectors is either a spanning set for* S *or there exists* $w \in S$ *such that* $\{v_0, v_1, \cdots, v_{n-1}, w\}$ *are linearly independent.*

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.
- LU with partial pivoting does not break down.
- $\mathcal{C}(A) = \mathbb{R}^n$.
- A has linearly independent columns.
- $\mathcal{N}(A) = \{0\}$.
- $\text{rank}(A) = n$.

Week 10

Vector Spaces, Orthogonality, and Linear Least Squares

10.1 Opening Remarks

10.1.1 Visualizing Planes, Lines, and Solutions

Consider the following system of linear equations from the opener for Week 9:

$$\begin{array}{rcrcrcr} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \\ \chi_0 & & & & & = & 2 \\ \chi_0 & + & 2\chi_1 & + & 4\chi_2 & = & 3 \end{array}$$

We solved this to find the (unique) solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$$

Let us look at each of these equations one at a time, and then put them together.

Example 10.1 Find the general solution to

$$\begin{array}{ccccccc} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \end{array}$$

We can write this as an appended system:

$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & -1 \end{array} \right).$$

Now, we would perform Gaussian or Gauss-Jordan elimination with this, except that there really isn't anything to do, other than to identify the pivot, the free variables, and the dependent variables:

$$\begin{array}{c} \left(\begin{array}{ccc|c} \boxed{1} & -2 & 4 & -1 \end{array} \right). \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ \text{dependent variable} & \text{free variable} & \text{free variable} \end{array} \end{array}$$

Here the pivot is highlighted with the box. There are two free variables, χ_1 and χ_2, and there is one dependent variable, χ_0. To find a specific solution, we can set χ_1 and χ_2 to any value, and solve for χ_0. Setting $\chi_1 = \chi_2 = 0$ is particularly convenient, leaving us with $\chi_0 - 2(0) + 4(0) = -1$, or $\chi_0 = -1$, so that the specific solution is given by

$$x_s = \begin{pmatrix} \boxed{\text{-1}} \\ 0 \\ 0 \end{pmatrix}.$$

To find solutions (a basis) in the null space, we look for solutions of $\left(\begin{array}{ccc|c} \boxed{1} & -2 & 4 & 0 \end{array} \right)$ in the form

$$x_{n_0} = \begin{pmatrix} \boxed{\chi_0} \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad x_{n_1} = \begin{pmatrix} \boxed{\chi_0} \\ 0 \\ 1 \end{pmatrix}$$

which yields the vectors

$$x_{n_0} = \begin{pmatrix} \boxed{2} \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad x_{n_1} = \begin{pmatrix} \boxed{\text{-4}} \\ 0 \\ 1 \end{pmatrix}.$$

This then gives us the general solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = x_s + \beta_0 x_{n_0} + \beta_1 x_{n_1} = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}.$$

Homework 10.1.1.1 Consider, again, the equation from the last example:

$$\chi_0 \quad - \quad 2\chi_1 \quad + \quad 4\chi_2 \quad = \quad -1$$

Which of the following represent(s) a general solution to this equation? (Mark all)

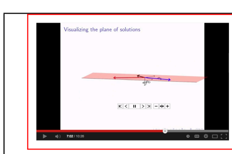
- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$.

- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta_0 \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$.

- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -5 \\ 0 \\ 1 \end{pmatrix} + \beta_0 \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$.

☛ SEE ANSWER

The following video helps you visualize the results from the above exercise:



☛ YouTube

Homework 10.1.1.2 Now you find the general solution for the **second** equation in the system of linear equations with which we started this unit. Consider

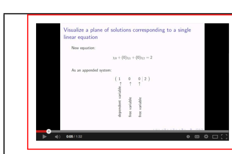
$$\chi_0 \qquad = \quad 2$$

Which of the following is a true statement about this equation:

- $\begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta_0 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}$ is a general solution.

☛ SEE ANSWER

The following video helps you visualize the message in the above exercise:



☛ YouTube

Homework 10.1.1.3 Now you find the general solution for the **third** equation in the system of linear equations with which we started this unit. Consider

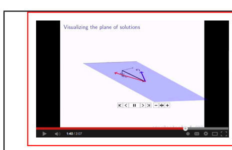
$$\chi_0 \;+\; 2\chi_1 \;+\; 4\chi_2 \;=\; 3$$

Which of the following is a true statement about this equation:

- $\begin{pmatrix} 3 \\ 0 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 3 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} -2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta_0 \begin{pmatrix} -2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 3 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} -4 \\ 2 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.

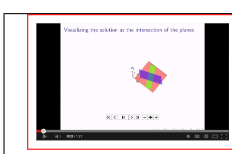
☛ SEE ANSWER

The following video helps you visualize the message in the above exercise:



☛ YouTube

Now, let's put the three planes together in one visualization.



☛ YouTube

Homework 10.1.1.4 We notice that it would be nice to put lines where planes meet. Now, let's start by focusing on the first two equations: Consider

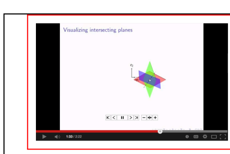
$$\begin{array}{rcrcrcr} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \\ \chi_0 & & & & & = & 2 \end{array}$$

Compute the general solution of this system with two equations in three unknowns and indicate which of the following is true about this system?

- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 3/2 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 3/2 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}$ is a general solution.

☛ SEE ANSWER

The following video helps you visualize the message in the above exercise:



☛ YouTube

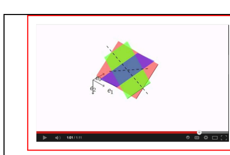
Homework 10.1.1.5 Similarly, consider

$$\begin{array}{rcrcrcl} \chi_0 & & & & & = & 2 \\ \chi_0 & + & 2\chi_1 & + & 4\chi_2 & = & 3 \end{array}$$

Compute the general solution of this system that has two equations with three unknowns and indicate which of the following is true about this system?

- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 1/2 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 1/2 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ -2 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ -2 \\ 1 \end{pmatrix}$ is a general solution.

☛ SEE ANSWER



☛ YouTube

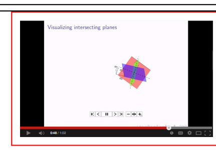
Homework 10.1.1.6 Finally consider

$$\begin{array}{rcrcrcr} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \\ \chi_0 & + & 2\chi_1 & + & 4\chi_2 & = & 3 \end{array}$$

Compute the general solution of this system with two equations in three unknowns and indicate which of the following is true about this system? UPDATE

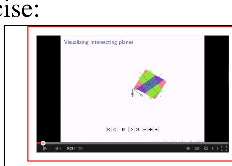
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.

☛ SEE ANSWER



☛ YouTube

The following video helps you visualize the message in the above exercise:



☛ YouTube

10.1.2 Outline

10.1.3 What You Will Learn

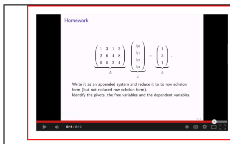
Upon completion of this unit, you should be able to

- Determine when linear systems of equations have a unique solution, an infinite number of solutions, or only approximate solutions.
- Determine the row-echelon form of a system of linear equations or matrix and use it to
  - find the pivots,
  - decide the free and dependent variables,
  - establish specific (particular) and general (complete) solutions,
  - find a basis for the column space, the null space, and row space of a matrix,
  - determine the rank of a matrix, and/or
  - determine the dimension of the row and column space of a matrix.
- Picture and interpret the fundamental spaces of matrices and their dimensionalities.
- Indicate whether vectors are orthogonal and determine whether subspaces are orthogonal.
- Determine the null space and column space for a given matrix and connect the row space of A with the column space of A^T.
- Identify, apply, and prove simple properties of vector spaces, subspaces, null spaces and column spaces.
- Determine when a set of vectors is linearly independent by exploiting special structures. For example, relate the rows of a matrix with the columns of its transpose to determine if the matrix has linearly independent rows.
- Approximate the solution to a system of linear equations of small dimension using the method of normal equations to solve the linear least-squares problem.

Track your progress in Appendix B.

10.2 How the Row Echelon Form Answers (Almost) Everything

10.2.1 Example



☛ YouTube

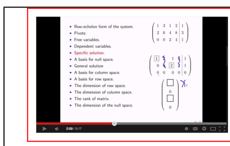
Homework 10.2.1.1 Consider the linear system of equations

$$\underbrace{\begin{pmatrix} 1 & 3 & 1 & 2 \\ 2 & 6 & 4 & 8 \\ 0 & 0 & 2 & 4 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}}_{b}.$$

Write it as an appended system and reduce it to to row echelon form (but not reduced row echelon form). Identify the pivots, the free variables and the dependent variables.

☛ SEE ANSWER

10.2.2 The Important Attributes of a Linear System



☛ YouTube

We now discuss how questions about subspaces can be answered once it has been reduced to its row echelon form. In particular, you can identify:

- The row-echelon form of the system.
- The pivots.
- The free variables.
- The dependent variables.
- A specific solution
  Often called a particular solution.
- A general solution
  Often called a complete solution.
- A basis for the column space.
  Something we should have mentioned before: The column space is often called the *range* of the matrix.
- A basis for the null space.
  Something we should have mentioned before: The null space is often called the *kernel* of the matrix.
- A basis for the row space.
  The row space is the subspace of all vectors that can be created by taking linear combinations of the rows of a matrix. In other words, the row space of A equals $\mathcal{C}(A^T)$ (the column space of A^T).
- The dimension of the row and column space.

- The rank of the matrix.
- The dimension of the null space.

Motivating example

Consider the example from the last unit.

$$\underbrace{\begin{pmatrix} 1 & 3 & 1 & 2 \\ 2 & 6 & 4 & 8 \\ 0 & 0 & 2 & 4 \end{pmatrix}}_{A} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}$$

which, when reduced to row echelon form, yields

$$\left(\begin{array}{cccc|c} 1 & 3 & 1 & 2 & 1 \\ 2 & 6 & 4 & 8 & 3 \\ 0 & 0 & 2 & 4 & 1 \end{array}\right) \quad \rightarrow \quad \left(\begin{array}{cccc|c} \boxed{1} & 3 & 1 & 2 & 1 \\ 0 & 0 & \boxed{2} & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

Here the boxed entries are the pivots (the first nonzero entry in each row) and they identify that the corresponding variables (χ_0 and χ_2) are dependent variables while the other variables (χ_1 and χ_3) are free variables.

Various dimensions

Notice that inherently the matrix is $m \times n$. In this case

- $m = 3$ (the number of rows in the matrix which equals the number of equations in the linear system); and
- $n = 4$ (the number of columns in the matrix which equals the number of unknowns in the linear system).

Now

- There are two pivots. Let's say that in general there are k pivots, where here $k = 2$.
- There are two free variables. In general, there are $n - k$ free variables, corresponding to the columns in which no pivot reside. **This means that the null space dimension equals $n - k$, or two in this case.**
- There are two dependent variables. In general, there are k dependent variables, corresponding to the columns in which the pivots reside. **This means that the column space dimension equals k, or also two in this case. This also means that the row space dimension equals k, or also two in this case.**
- The dimension of the row space always equals the dimension of the column space which always equals the number of pivots in the row echelon form of the equation. This number, k, is called the **rank** of matrix A, $\text{rank}(A)$.

Format of a general solution

To find a general solution to problem, you recognize that there are two free variables (χ_1 and χ_3) and a general solution can be given by

$$\begin{pmatrix} \square \\ 0 \\ \square \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} \square \\ 1 \\ \square \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} \square \\ 0 \\ \square \\ 1 \end{pmatrix}.$$

Computing a specific solution

The specific (particular or special) solution is given by $x_s = \begin{pmatrix} \boxed{} \\ 0 \\ \boxed{} \\ 0 \end{pmatrix}$. It solves the system. To obtain it, you set the free variables to zero and solve the row echelon form of the system for the values in the boxes:

$$\begin{pmatrix} 1 & 3 & 1 & 2 \\ 0 & 0 & 2 & 4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ 0 \\ \chi_2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

or

$$\begin{array}{cccc} \chi_0 & +\chi_2 & = & 1 \\ & 2\chi_2 & = & 1 \end{array}$$

so that $\chi_2 = 1/2$ and $\chi_0 = 1/2$ yielding a specific solution $x_p = \begin{pmatrix} 1/2 \\ 0 \\ 1/2 \\ 0 \end{pmatrix}$.

Computing a basis for the null space

Next, we have to find two linearly independent vectors in the null space of the matrix. (There are two because there are two free variables. In general, there are $n-k$.)

To obtain the first, we set the first free variable to one and the other(s) to zero, and solve the row echelon form of the system *with the right-hand side set to zero*:

$$\begin{pmatrix} 1 & 3 & 1 & 2 \\ 0 & 0 & 2 & 4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ 1 \\ \chi_2 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

or

$$\begin{array}{ccccc} \chi_0 & +3\times 1 & +\chi_2 & = & 0 \\ & & 2\chi_2 & = & 0 \end{array}$$

so that $\chi_2 = 0$ and $\chi_0 = -3$, yielding the first vector in the null space $x_{n_0} = \begin{pmatrix} -3 \\ 1 \\ 0 \\ 0 \end{pmatrix}$.

To obtain the second, we set the second free variable to one and the other(s) to zero, and solve the row echelon form of the system *with the right-hand side set to zero*:

$$\begin{pmatrix} 1 & 3 & 1 & 2 \\ 0 & 0 & 2 & 4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ 0 \\ \chi_2 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

or

$$\begin{array}{ccccc} \chi_0 & +\chi_2 & +2\times 1 & = & 0 \\ & 2\chi_2 & +4\times 1 & = & 0 \end{array}$$

so that $\chi_2 = -4/2 = -2$ and $\chi_0 = -\chi_2 - 2 = 0$, yielding the second vector in the null space $x_{n_1} = \begin{pmatrix} 0 \\ 0 \\ -2 \\ 1 \end{pmatrix}$.

Thus,

$$\mathcal{N}(A) = \text{Span}\left(\left\{\begin{pmatrix} -3 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -2 \\ 1 \end{pmatrix}\right\}\right).$$

A general solution

Thus, a general solution is given by

$$\begin{pmatrix} 1/2 \\ 0 \\ 1/2 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} -3 \\ 1 \\ 0 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ -2 \\ 1 \end{pmatrix},$$

where $\beta_0, \beta_1 \in \mathbb{R}$.

Finding a basis for the column space of the original matrix

To find the linearly independent columns, you look at the row echelon form of the matrix:

$$\begin{pmatrix} \boxed{1} & 3 & 1 & 2 \\ 0 & 0 & \boxed{2} & 4 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

with the pivots highlighted. The columns that have pivots in them are linearly independent. The corresponding columns in the original matrix are also linearly independent:

$$\left(\begin{array}{|c|c|c|c} 1 & 3 & 1 & 2 \\ 2 & 6 & 4 & 8 \\ 0 & 0 & 2 & 4 \end{array}\right).$$

Thus, in our example, the answer is $\begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 4 \\ 2 \end{pmatrix}$ (the first and third column).

Thus,

$$\mathcal{C}(A) = \text{Span}\left(\left\{\begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 4 \\ 2 \end{pmatrix}\right\}\right).$$

Find a basis for the row space of the matrix.

The row space (we will see in the next chapter) is the space spanned by the rows of the matrix (viewed as column vectors). Reducing a matrix to row echelon form merely takes linear combinations of the rows of the matrix. What this means is that the space spanned by the rows of the original matrix is the same space as is spanned by the rows of the matrix in row echelon form. Thus, all you need to do is list the rows in the matrix in row echelon form, as column vectors.

For our example this means a basis for the row space of the matrix is given by

$$\mathcal{R}(A) = \text{Span}\left(\left\{ \begin{pmatrix} 1 \\ 3 \\ 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 2 \\ 4 \end{pmatrix} \right\} \right).$$

Summary observation

The following are all equal:

- The dimension of the column space.
- The rank of the matrix.
- The number of dependent variables.
- The number of nonzero rows in the upper echelon form.
- The number of columns in the matrix minus the number of free variables.
- The number of columns in the matrix minus the dimension of the null space.
- The number of linearly independent columns in the matrix.
- The number of linearly independent rows in the matrix.

Homework 10.2.2.1 Consider $\begin{pmatrix} 1 & 2 & 2 \\ 2 & 4 & 5 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \end{pmatrix}$.

- Reduce the system to row echelon form (but not reduced row echelon form).
- Identify the free variables.
- Identify the dependent variables.
- What is the dimension of the column space?
- What is the dimension of the row space?
- What is the dimension of the null space?
- Give a set of linearly independent vectors that span the column space
- Give a set of linearly independent vectors that span the row space.
- What is the rank of the matrix?
- Give a general solution.

☛ SEE ANSWER

Homework 10.2.2.2 Which of these statements is a correct definition of the rank of a given matrix $A \in \mathbb{R}^{m\times n}$?

1. The number of nonzero rows in the reduced row echelon form of A. **True/False**
2. The number of columns minus the number of rows, $n-m$. **True/False**
3. The number of columns minus the number of free columns in the row reduced form of A. (Note: a free column is a column that does not contain a pivot.) **True/False**
4. The number of 1s in the row reduced form of A. **True/False**

☛ SEE ANSWER

Homework 10.2.2.3 Compute

$$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} \begin{pmatrix} 3 & -1 & 2 \end{pmatrix}.$$

Reduce it to row echelon form. What is the rank of this matrix?

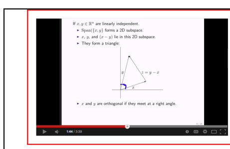
☛ SEE ANSWER

Homework 10.2.2.4 Let $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$ so that uv^T is a $m \times n$ matrix. What is the rank, k, of this matrix?

☛ SEE ANSWER

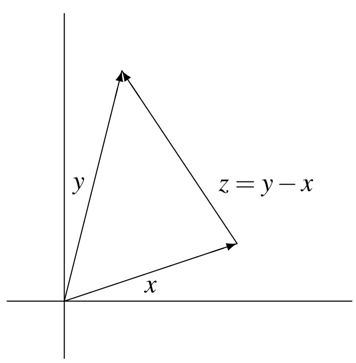
10.3 Orthogonal Vectors and Spaces

10.3.1 Orthogonal Vectors



☛ YouTube

If nonzero vectors $x, y \in \mathbb{R}^n$ are linearly independent then the subspace of all vectors $\alpha x + \beta y$, $\alpha, \beta \in \mathbb{R}$ (the space spanned by x and y) form a plane. All three vectors x, y, and $(x-y)$ lie in this plane and they form a triangle:



where this page represents the plane in which all of these vectors lie.

Vectors x and y are considered to be orthogonal (perpendicular) if they meet at a right angle. Using the Euclidean length

$$\|x\|_2 = \sqrt{\chi_0^2 + \cdots + \chi_{n-1}^2} = \sqrt{x^T x},$$

we find that the Pythagorean Theorem dictates that *if* the angle in the triangle where x and y meet is a right angle, then $\|z\|_2^2 = \|x\|_2^2 + \|y\|_2^2$. In this case,

$$\|z\|_2^2 = \|x\|_2^2 + \|y\|_2^2 \quad = \quad \|y-x\|_2^2$$

$$
\begin{aligned}
&= (y-x)^T(y-x) \\
&= (y^T - x^T)(y-x) \\
&= (y^T - x^T)y - (y^T - x^T)x \\
&= \underbrace{y^T y}_{\|y\|_2^2} - \underbrace{(x^T y + y^T x)}_{2x^T y} + \underbrace{x^T x}_{\|x\|_2^2} \\
&= \|x\|_2^2 - 2x^T y + \|y\|_2^2.
\end{aligned}
$$

In other words, when x and y are perpendicular (orthogonal)

$$\|x\|_2^2 + \|y\|_2^2 = \|x\|_2^2 - 2x^T y + \|y\|_2^2.$$

Cancelling terms on the left and right of the equality, this implies that $x^T y = 0$. This motivates the following definition:

Definition 10.2 *Two vectors $x, y \in \mathbb{R}^n$ are said to be orthogonal if and only if $x^T y = 0$.*

Sometimes we will use the notation $x \perp y$ to indicate that x is perpendicular to y.

Homework 10.3.1.1 For each of the following, indicate whether the vectors are orthogonal:

$\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$	True/False
$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$	True/False
The unit basis vectors e_i and e_j.	Always/Sometimes/Never
$\begin{pmatrix} c \\ s \end{pmatrix}$ and $\begin{pmatrix} -s \\ c \end{pmatrix}$	Always/Sometimes/Never

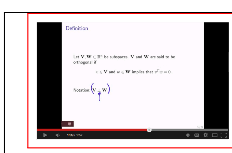
☛ SEE ANSWER

Homework 10.3.1.2 Let $A \in \mathbb{R}^{m \times n}$. Let a_i^T be a row of A and $x \in \mathcal{N}(A)$. Then a_i is orthogonal to x.

Always/Sometimes/Never

☛ SEE ANSWER

10.3.2 Orthogonal Spaces



☛ YouTube

We can extend this to define orthogonality of two subspaces:

Definition 10.3 *Let $\mathbf{V}, \mathbf{W} \subset \mathbb{R}^n$ be subspaces. Then $\mathbf{V}$ and $\mathbf{W}$ are said to be orthogonal if and only if $v \in \mathbf{V}$ and $w \in \mathbf{W}$ implies that $v^T w = 0$.*

We will use the notation $\mathbf{V} \perp \mathbf{W}$ to indicate that subspace $\mathbf{V}$ is orthogonal to subspace $\mathbf{W}$.

In other words: Two subspaces are orthogonal if all the vectors from one of the subspaces are orthogonal to all of the vectors from the other subspace.

Homework 10.3.2.1 Let $\mathbf{V} = \{0\}$ where 0 denotes the zero vector of size n. Then $\mathbf{V} \perp \mathbb{R}^n$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 10.3.2.2 Let

$$\mathbf{V} = \mathrm{Span}\left(\left\{\begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}\right\}\right) \quad \text{and} \quad \mathbf{W} = \mathrm{Span}\left(\left\{\begin{pmatrix}0\\0\\1\end{pmatrix}\right\}\right)$$

Then $\mathbf{V} \perp \mathbf{W}$.

True/False

☛ SEE ANSWER

The above can be interpreted as: the "x-y" plane is orthogonal to the z axis.

Homework 10.3.2.3 Let $\mathbf{V}, \mathbf{W} \subset \mathbb{R}^n$ be subspaces. If $\mathbf{V} \perp \mathbf{W}$ then $\mathbf{V} \cap \mathbf{W} = \{0\}$, the zero vector.

Always/Sometimes/Never

☛ SEE ANSWER

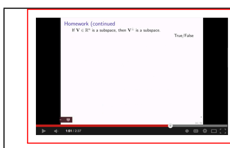
Whenever $S \cap T = \{0\}$ we will sometimes call this the *trivial intersection* of two subspaces. Trivial in the sense that it only contains the zero vector.

Definition 10.4 *Given subspace $\mathbf{V} \subset \mathbb{R}^n$, the set of all vectors in $\mathbb{R}^n$ that are orthogonal to $\mathbf{V}$ is denoted by $\mathbf{V}^\perp$ (pronounced as "V-perp").*

Homework 10.3.2.4 If $\mathbf{V} \subset \mathbb{R}^n$ is a subspace, then $\mathbf{V}^\perp$ is a subspace.

True/False

☛ SEE ANSWER



☛ YouTube

10.3.3 Fundamental Spaces

☛ YouTube

Let us recall some definitions:

- The column space of a matrix $A \in \mathbb{R}^{m \times n}$, $\mathcal{C}(\mathcal{A})$, equals the set of all vectors in $\mathbb{R}^m$ that can be written as Ax: $\{y \mid y = Ax\}$.
- The null space of a matrix $A \in \mathbb{R}^{m \times n}$, $\mathcal{N}(A)$, equals the set of all vectors in $\mathbb{R}^n$ that map to the zero vector: $\{x \mid Ax = 0\}$.
- The row space of a matrix $A \in \mathbb{R}^{m \times n}$, $\mathcal{R}(A)$, equals the set of all vectors in $\mathbb{R}^n$ that can be written as $A^T x$: $\{y \mid y = A^T x\}$.

Theorem 10.5 *Let $A \in \mathbb{R}^{m \times n}$. Then $\mathcal{R}(A) \perp \mathcal{N}(A)$.*

Proof: Let $y \in \mathcal{R}(A)$ and $z \in \mathcal{N}(A)$. We need to prove that $y^T z = 0$.

$$
\begin{array}{l}
y^T z \\
\quad = \quad < y \in \mathcal{R}(A) \text{ implies that } y = A^T x \text{ for some } x \in \mathbb{R}^m > \\
(A^T x)^T z \\
\quad = \quad < (AB)^T = B^T A^T > \\
x^T (A^T)^T z \\
\quad = \quad < (A^T)^T = A > \\
x^T A z \\
\quad = \quad < z \in \mathcal{N}(A) \text{ implies that } Az = 0 > \\
x^T 0 \\
\quad = \quad < \text{algebra} > \\
0
\end{array}
$$

Theorem 10.6 *Let $A \in \mathbb{R}^{m \times n}$. Then every $x \in \mathbb{R}^n$ can be written as $x = x_r + x_n$ where $x_r \in \mathcal{R}(A)$ and $x_n \in \mathcal{N}(A)$.*

Proof: Recall that if $\dim(\mathcal{R}(A) = k$, then $\dim(\mathcal{N}(A)) = n - k$. Let $\{v_0, \ldots, v_{k-1}$ be a basis for $\mathcal{R}(A)$ and $\{v_k, \ldots, v_{n-1}$ be a basis for $\mathcal{N}(A)$. It can be argued, via a proof by contradiction that is beyond this course, that the set of vectors $\{v_0, \ldots, v_{n-1}\}$ are linearly independent.

Let $x \in \mathbb{R}^n$. This is then a basis for $\mathbb{R}^n$, which in turn means that $x = \sum_{i=0} \alpha_i v_i$, some linear combination. But then

$$
x = \underbrace{\sum_{i=0}^{k-1} \alpha_i v_i}_{x_r} + \underbrace{\sum_{i=k}^{n-1} \alpha_i v_i}_{x_n},
$$

where by construction $x_r \in \mathcal{R}(A)$ and $x_n \in \mathcal{N}(A)$.

Let $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, and $b \in \mathbb{R}^m$, with $Ax = b$. Then there exist $x_r \in \mathcal{R}(A)$ and $x_n \in \mathcal{N}(A)$ such that $x = x_r + x_n$. But then

$$\begin{aligned}
& Ax_r \\
& \quad = \quad < 0 \text{ of size } n > \\
& Ax_r + 0 \\
& \quad = \quad < Ax_n = 0 > \\
& Ax_r + Ax_n \\
& \quad = \quad < A(y+z) = Ay + Az > \\
& A(x_r + x_n) \\
& \quad = \quad < x = x_r + x_n > \\
& Ax \\
& \quad = \quad < Ax = b > \\
& b.
\end{aligned}$$

We conclude that if $Ax = b$ has a solution, then there is a $x_r \in \mathcal{R}(A)$ such that $Ax_r = b$.

Theorem 10.7 *Let $A \in \mathbb{R}^{m\times n}$. Then A is a one-to-one, onto mapping from $\mathcal{R}(A)$ to $\mathcal{C}(\mathcal{A})$.*

Proof: Let $A \in \mathbb{R}^{m\times n}$. We need to show that

- A maps $\mathcal{R}(A)$ to $\mathcal{C}(\mathcal{A})$. This is trivial, since any vector $x \in \mathbb{R}^n$ maps to $\mathcal{C}(\mathcal{A})$.
- Uniqueness: We need to show that if $x, y \in \mathcal{R}(A)$ and $Ax = Ay$ then $x = y$. Notice that $Ax = Ay$ implies that $A(x-y) = 0$, which means that $(x-y)$ is both in $\mathcal{R}(A)$ (since it is a linear combination of x and y, both of which are in $\mathcal{R}(A)$) and in $\mathcal{N}(A)$. Since we just showed that these two spaces are orthogonal, we conclude that $(x-y) = 0$, the zero vector. Thus $x = y$.
- Onto: We need to show that for any $b \in \mathcal{C}(\mathcal{A})$ there exists $x_r \in \mathcal{R}(A)$ such that $Ax_r = b$. Notice that if $b \in \mathcal{C}$, then there exists $x \in \mathbb{R}^n$ such that $Ax = b$. By Theorem 10.6, $x = x_r + x_n$ where $x_r \in \mathcal{R}(A)$ and $x_n \in \mathcal{N}(A)$. Then $b = Ax = A(x_r + x_n) = Ax_r + Ax_n = Ax_r$.

We define one more subpace:

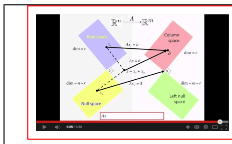
Definition 10.8 *Given $A \in \mathbb{R}^{m\times n}$ the left null space of A is the set of all vectors x such that $x^T A = 0$.*

Clearly, the left null space of A equals the null space of A^T.

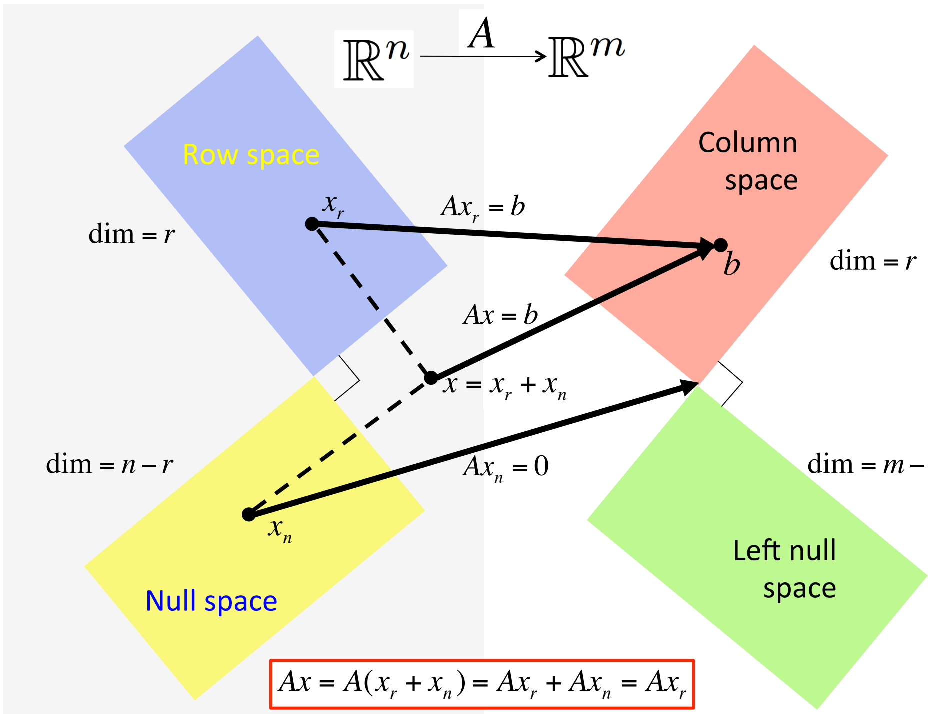
Theorem 10.9 *Let $A \in \mathbb{R}^{m\times n}$. Then the left null space of A is orthogonal to the column space of A and the dimension of the left null space of A equals $m - r$, where r is the dimension of the column space of A.*

Proof: This follows trivially by applying the previous theorems to A^T.

The observations in this unit are summarized by the following video and subsequent picture:

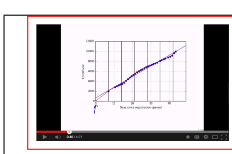


☛ YouTube



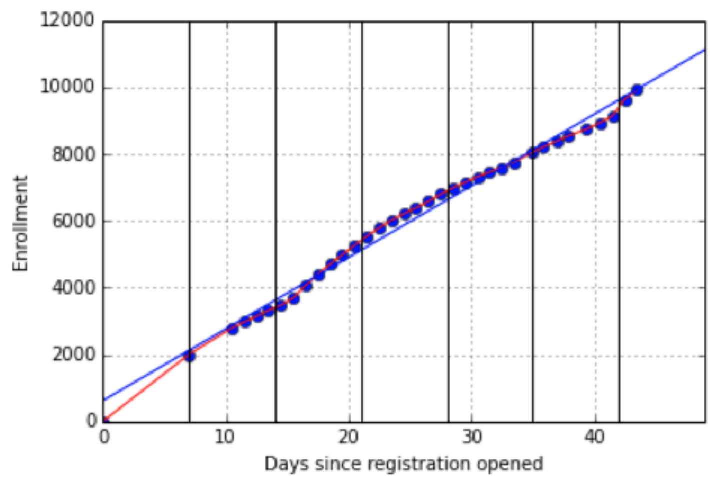
10.4 Approximating a Solution

10.4.1 A Motivating Example



☛ YouTube

Consider the following graph:

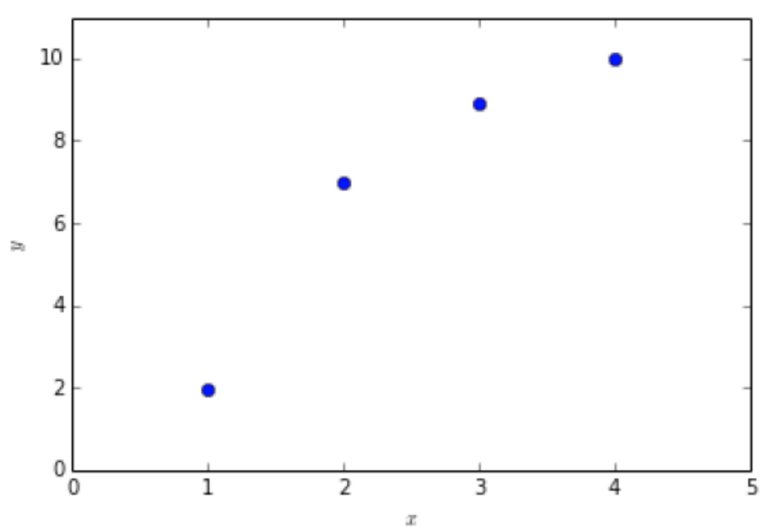


It plots the number of registrants for our "Linear Algebra - Foundations to Frontiers" course as a function of days that have passed since registration opened (data for the first offering of LAFF in Spring 2014), for the first 45 days or so (the course opens after 107 days). The blue dots represent the measured data and the blue line is the best straight line fit (which we will later call the linear least-squares fit to the data). By fitting this line, we can, for example, extrapolate that we will likely have more than 20,000 participants by the time the course commences.

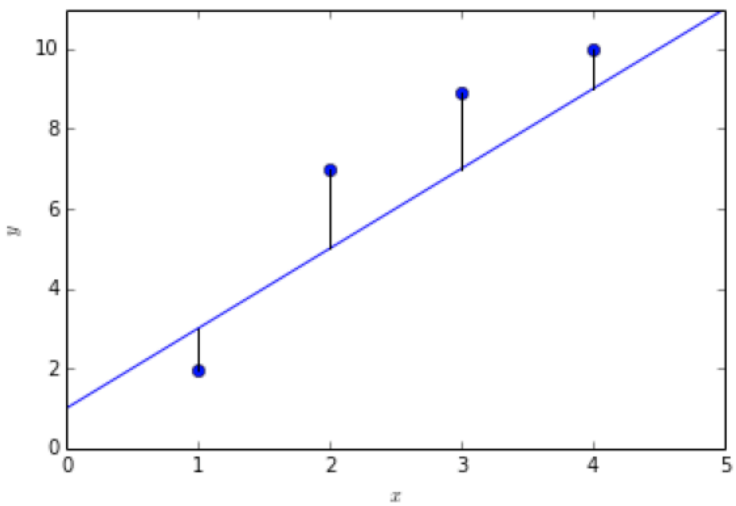
Let us illustrate the basic principles with a simpler, artificial example. Consider the following set of points:

$$(\chi_0,\psi_0) = (1, 1.97), (\chi_1,\psi_1) = (2, 6.97), (\chi_2,\psi_2) = (3, 8.89), (\chi_3,\psi_3) = (4, 10.01),$$

which we plot in the following figure:



What we would like to do is to find a line that interpolates these points. Here is a rough approximation for such a line:



Here we show with the vertical lines the distance from the points to the line that was chosen. The question becomes, what is the best line? We will see that "best" is defined in terms of minimizing the sum of the square of the distances to the line. The above line does **not** appear to be "best", and it isn't.

Let us express this with matrices and vectors. Let

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \quad \text{and} \quad y = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} = \begin{pmatrix} 1.97 \\ 6.97 \\ 8.89 \\ 10.01 \end{pmatrix}.$$

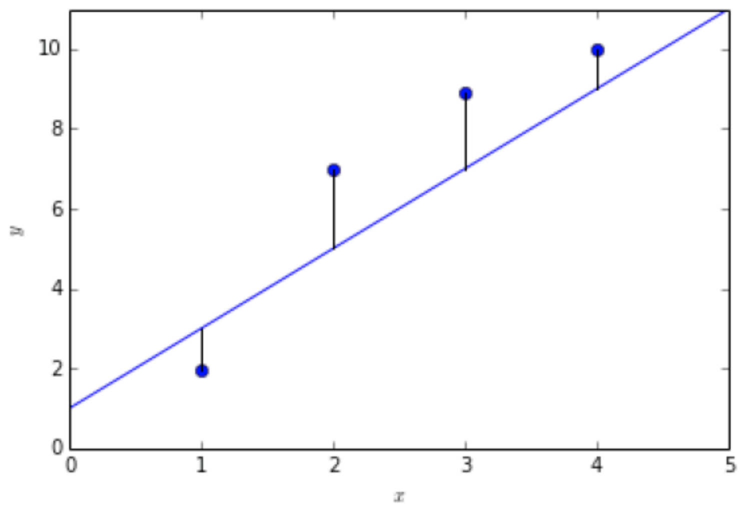
If we give the equation of the line as $y = \gamma_0 + \gamma_1 x$ then, **IF** this line **COULD** go through all these points **THEN** the following equations would have to be simultaneously satified:

$$\begin{array}{rcl} \psi_0 & = & \gamma_0 + \gamma_1 \chi_1 \\ \psi_1 & = & \gamma_0 + \gamma_1 \chi_2 \\ \psi_2 & = & \gamma_0 + \gamma_1 \chi_3 \\ \psi_3 & = & \gamma_0 + \gamma_1 \chi_4 \end{array} \quad \text{or, specifically,} \quad \begin{array}{rcl} 1.97 & = & \gamma_0 + \gamma_1 \\ 6.97 & = & \gamma_0 + 2\gamma_1 \\ 8.89 & = & \gamma_0 + 3\gamma_1 \\ 10.01 & = & \gamma_0 + 4\gamma_1 \end{array}$$

which can be written in matrix notation as

$$\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} = \begin{pmatrix} 1 & \chi_0 \\ 1 & \chi_1 \\ 1 & \chi_2 \\ 1 & \chi_3 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix} \quad \text{or, specifically,} \quad \begin{pmatrix} 1.97 \\ 6.97 \\ 8.89 \\ 10.01 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix}.$$

Now, just looking at



it is obvious that these points do not lie on the same line and that therefore all these equations cannot be simultaneously satified. **So, what do we do now?**

How does it relate to column spaces?

The first question we ask is "For what right-hand sides could we have solved all four equations simultaneously?" We would have had to choose y so that $Ac = y$, where

$$A = \begin{pmatrix} 1 & \chi_0 \\ 1 & \chi_1 \\ 1 & \chi_2 \\ 1 & \chi_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{pmatrix} \quad \text{and} \quad c = \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix}.$$

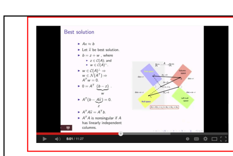
This means that y must be in the column space of A. It must be possible to express it as $y = \gamma_0 a_0 + \gamma_1 a_1$, where $A = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right)$! What does this mean if we relate this back to the picture? Only if $\{\psi_0, \cdots, \psi_3\}$ have the property that $\{(1,\psi_0), \cdots, (4,\psi_3)\}$ lie on a line can we find coefficients γ_0 and γ_1 such that $Ac = y$.

How does this problem relate to orthogonality?

The problem is that the given y does **not** lie in the column space of A. So a question is, what vector z, that **does** lie in the column space should we use to solve $Ac = z$ instead so that we end up with a line that best interpolates the given points?

If z solves $Ac = z$ exactly, then $z = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix} = \gamma_0 a_0 + \gamma_1 a_1$, which is of course just a repeat of the observation that z is in the column space of A. Thus, what we want is $y = z + w$, where w is as small (in length) as possible. This happens when w is orthogonal to z! So, $y = \gamma_0 a_0 + \gamma_1 a_1 + w$, with $a_0^T w = a_1^T w = 0$. The vector z in the column space of A that is closest to y is known as the **projection** of y onto the column space of A. So, it would be nice to have a way of finding a way to compute this projection.

10.4.2 Finding the Best Solution



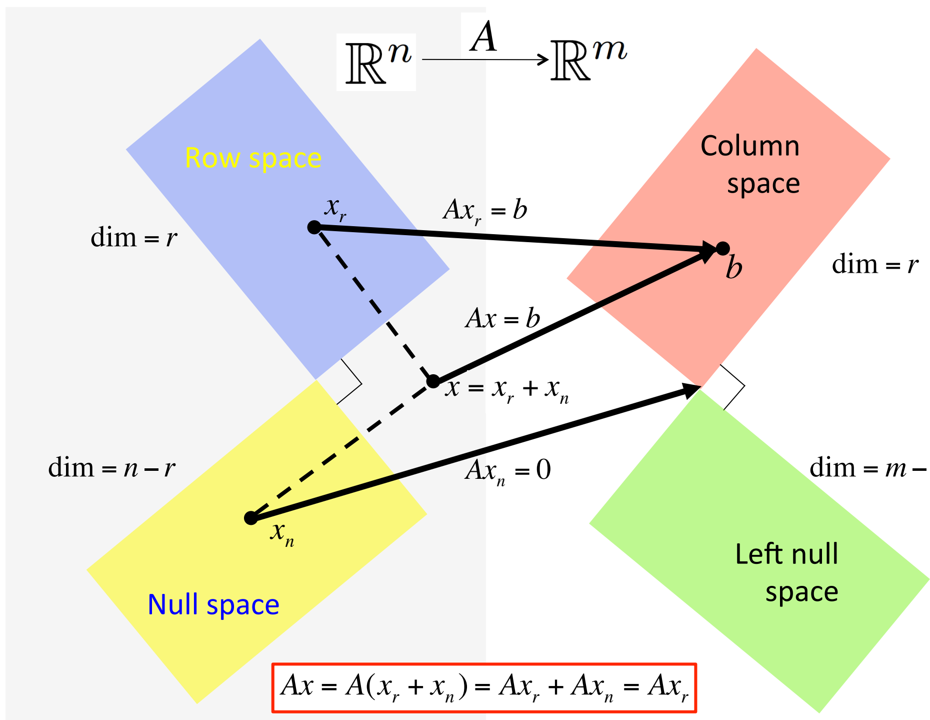
☛ YouTube

The last problem motivated the following general problem: Given m equations in n unknowns, we end up with a system $Ax = b$ where $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, and $b \in \mathbb{R}^m$.

- This system of equations may have no solutions. This happens when b is not in the column space of A.
- This system may have a unique solution. This happens only when $r = m = n$, where r is the rank of the matrix (the dimension of the column space of A). Another way of saying this is that it happens only if A is square and nonsingular (it has an inverse).
- This system may have many solutions. This happens when b is in the column space of A and $r < n$ (the columns of A are linearly dependent, so that the null space of A is nontrivial).

Let us focus on the first case: b is not in the column space of A.

In the last unit, we argued that what we want is an approximate solution $\hat{x}$ such that $A\hat{x} = z$, where z is the vector in the column space of A that is "closest" to b: $b = z + w$ where $w^T v = 0$ for all $v \in \mathcal{C}(\mathcal{A})$. From



we conclude that this means that w is in the left null space of A. So, $A^T w = 0$. But that means that

$$0 = A^T w = A^T(b - z) = A^T(b - A\hat{x})$$

which we can rewrite as

$$A^T A\hat{x} = A^T b. \tag{10.1}$$

This is known as the **normal equation** associated with the problem $A\hat{x} \approx b$.

Theorem 10.10 *If $A \in \mathbb{R}^{m\times n}$ has linearly independent columns, then $A^T A$ is nonsingular (equivalently, has an inverse, $A^T A\hat{x} = A^T b$ has a solution for all b, etc.).*

Proof: Proof by contradiction.

- Assume that $A \in \mathbb{R}^{m\times n}$ has linearly independent columns and $A^T A$ is singular.
- Then there exists $x \neq 0$ such that $A^T Ax = 0$.

- Hence, there exists $y = Ax \neq 0$ such that $A^T y = 0$ (because A has linearly independent columns and $x \neq 0$).
- This means y is in the left null space of A.
- But y is also in the column space of A, since $Ax = y$.
- Thus, $y = 0$, since the intersection of the column space of A and the left null space of A only contains the zero vector.
- This contradicts the fact that A has linearly independent columns.

Therefore $A^T A$ cannot be singular.

This means that if A has linearly independent columns, then the desired $\hat{x}$ that is the best approximate solution is given by

$$\hat{x} = (A^T A)^{-1} A^T b$$

and the vector $z \in \mathcal{C}(\mathcal{A})$ closest to b is given by

$$z = A\hat{x} = A(A^T A)^{-1} A^T b.$$

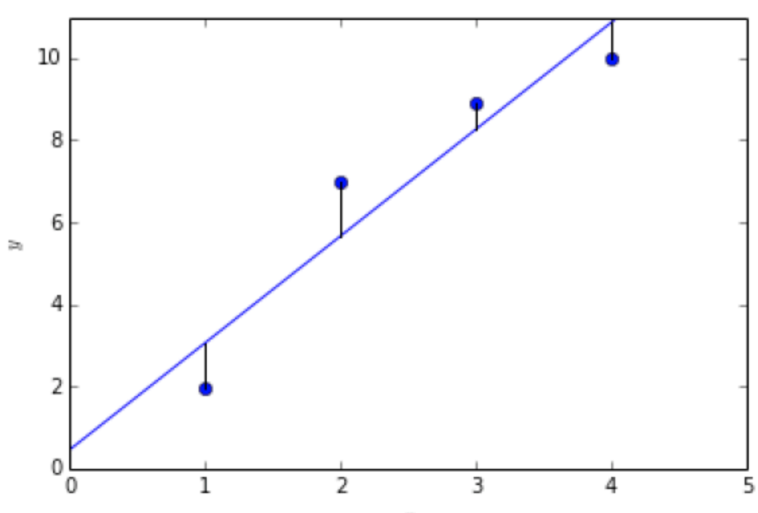
This shows that *if* A has linearly independent columns, then $z = A(A^T A)^{-1} A^T b$ is the vector in the columns space closest to b. **This is the projection of b onto the column space of A.**

Let us now formulate the above observations as a special case of a *linear least-squares* problem:

Theorem 10.11 *Let $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $x \in \mathbb{R}^n$ and assume that A has linearly independent columns. Then the solution that minimizes the length of the vector $b - Ax$ is given by $\hat{x} = (A^T A)^{-1} A^T b$.*

Definition 10.12 *Let $A \in \mathbb{R}^{m \times n}$. If A has linearly independent columns, then $A^\dagger = (A^T A)^{-1} A^T$ is called the (left) pseudo inverse. Note that this means $m \geq n$ and $A^\dagger A = (A^T A)^{-1} A^T A = I$.*

If we apply these insights to the motivating example from the last unit, we get the following approximating line



Homework 10.4.2.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$.

1. Is b in the column space of A?

   True/False

2. $A^T b =$
3. $A^T A =$
4. $(A^T A)^{-1} =$
5. $A^\dagger =$.
6. $A^\dagger A =$.
7. Compute the approximate solution, in the least squares sense, of $Ax \approx b$.

   $$x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} =$$

8. What is the project of b onto the column space of A?

   $$\widehat{b} = \begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_1 \\ \widehat{\beta}_2 \end{pmatrix} =$$

☛ SEE ANSWER

Homework 10.4.2.2 Consider $A = \begin{pmatrix} 1 & -1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}$ and $b = \begin{pmatrix} 4 \\ 5 \\ 9 \end{pmatrix}$.

1. b is in the column space of A, $\mathcal{C}(A)$.

   True/False

2. Compute the approximate solution, in the least squares sense, of $Ax \approx b$.

   $$x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} =$$

3. What is the project of b onto the column space of A?

   $$\widehat{b} = \begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_1 \\ \widehat{\beta}_2 \end{pmatrix} =$$

4. $A^\dagger =$.
5. $A^\dagger A =$.

☛ SEE ANSWER

Homework 10.4.2.3 What 2×2 matrix B projects the x-y plane onto the line $x + y = 0$?

☛ SEE ANSWER

Homework 10.4.2.4 Find the line that best fits the following data:

x	y
-1	2
1	-3
0	0
2	-5

☛ SEE ANSWER

Homework 10.4.2.5 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 2 \\ 7 \end{pmatrix}$.

1. b is in the column space of A, $\mathcal{C}(A)$.

   True/False

2. Compute the approximate solution, in the least squares sense, of $Ax \approx b$.

   $x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} =$

3. What is the projection of b onto the column space of A?

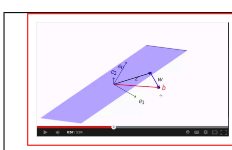
   $\widehat{b} = \begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_1 \\ \widehat{\beta}_2 \end{pmatrix} =$

4. $A^\dagger =$.

5. $A^\dagger A =$.

☛ SEE ANSWER

10.4.3 Why It is Called Linear Least-Squares



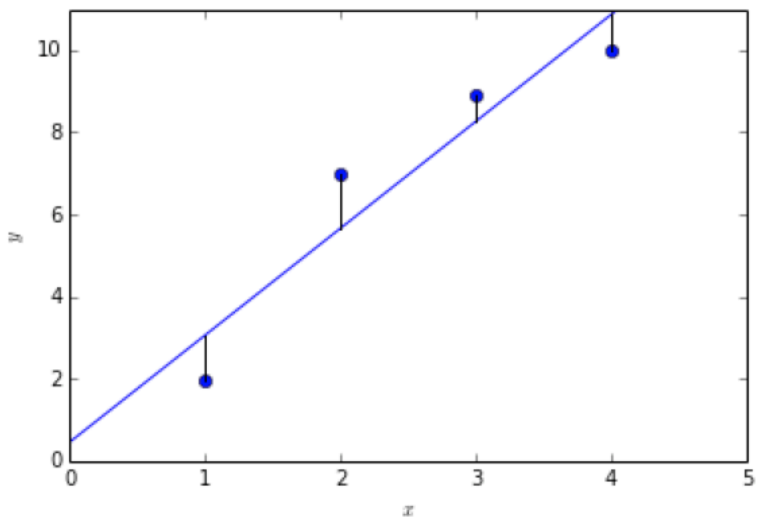
☛ YouTube

The “best” solution discussed in the last unit is known as the “linear least-squares” solution. Why?

Notice that we are trying to find $\hat{x}$ that minimizes the length of the vector $b - Ax$. In other words, we wish to find $\hat{x}$ that minimizes $\min_x \|b - Ax\|_2$. Now, if $\hat{x}$ minimizes $\min_x \|b - Ax\|_2$, it also minimizes the function $\|b - Ax\|_2^2$. Let $y = A\hat{x}$. Then

$$\|b - A\hat{x}\|^2 = \|b - y\|^2 = \sum_{i=0}^{n-1} (\beta_i - \psi_i)^2.$$

Thus, we are trying to minimize the sum of the squares of the differences. If you consider, again,



then this translates to minimizing the sum of the lenghts of the vertical lines that connect the linear approximation to the original points.

10.5 Enrichment

10.5.1 Solving the Normal Equations

In our examples and exercises, we solved the normal equations

$$A^T A x = A^T b,$$

where $A \in \mathbb{R}^{m \times n}$ has linear independent columns, via the following steps:

- Form $y = A^T b$
- Form $A^T A$.
- Invert $A^T A$ to compute $B = (A^T A)^{-1}$.
- Compute $\hat{x} = By = (A^T A)^{-1} A^T b$.

This involves the inversion of a matrix, and we claimed in Week 8 that one should (almost) never, ever invert a matrix.

In practice, this is not how it is done for larger systems of equations. Instead, one uses either the Cholesky factorization (which was discussed in the enrichment for Week 8), the QR factorization (to be discussed in Week 11), or the Singular Value Decomposition (SVD, which is briefly mentioned in Week 11).

Let us focus on how to use the Cholesky factorization. Here are the steps:

- Compute $C = A^T A$.
- Compute the Cholesky factorization $C = LL^T$, where L is lower triangular.
  This allows us to take advantage of symmetry in C.
- Compute $y = A^T b$.
- Solve $Lz = y$.
- Solve $L^T \hat{x} = z$.

The vector $\hat{x}$ is then the best solution (in the linear least-squares sense) to $Ax \approx b$.

The Cholesky factorization of a matrix, C, exists if and only if C has a special property. Namely, it must be symmetric positive definite (SPD).

Definition 10.13 *A symmetric matrix $C \in \mathbb{R}^{m \times m}$ is said to be symmetric positive definite if $x^T C x \geq 0$ for all nonzero vectors $x \in \mathbb{R}^m$.*

We started by assuming that A has linearly independent columns and that $C = A^T A$. Clearly, C is symmetric: $C^T = (A^T A)^T = A^T (A^T)^T = A^T A = C$. Now, let $x \neq 0$. Then

$$x^T C x = x^T (A^T A) x = (x^T A^T)(Ax) = (Ax)^T (Ax) = \|Ax\|_2^2.$$

We notice that $Ax \neq 0$ because the columns of A are linearly independent. But that means that its length, $\|Ax\|_2$, is not equal to zero and hence $\|Ax\|_2^2 > 0$. We conclude that $x \neq 0$ implies that $x^T C x > 0$ and that therefore C is symmetric positive definite.

10.6 Wrap Up

10.6.1 Homework

No additional homework this week.

10.6.2 Summary

Solving underdetermined systems

Important attributes of a linear system $Ax = b$ and associated matrix A:

- The row-echelon form of the system.
- The pivots.
- The free variables.
- The dependent variables.
- A specific solution
  Also called a *particular* solution.
- A general solution
  Also called a *complete* solution.
- A basis for the null space.
  Also called the *kernel* of the matrix. This is the set of all vectors that are mapped to the zero vector by A.
- A basis for the column space, $\mathcal{C}(A)$.
  Also called the *range* of the matrix. This is the set of linear combinations of the columns of A.
- A basis for the row space, $\mathcal{R}(A) = \mathcal{C}(A^T)$.
  This is the set of linear combinations of the columns of A^T.
- The dimension of the row and column space.
- The rank of the matrix.
- The dimension of the null space.

Various dimensions Notice that, in general, a matrix is $m \times n$. In this case

- Start the linear system of equations $Ax = y$.
- Reduce this to row echelon form $Bx = \hat{y}$.
- If any of the equations are inconsistent ($0 \neq \hat{\psi}_i$, for some row i in the row echelon form $Bx = \hat{y}$), then the system does not have a solution, and y is not in the column space of A.

- If this is not the case, assume there are k pivots in the row echelon reduced form.
- Then there are $n-k$ free variables, corresponding to the columns in which no pivots reside. **This means that the null space dimension equals** $n-k$
- There are k dependent variables corresponding to the columns in which the pivots reside. **This means that the column space dimension equals** k **and the row space dimension equals** k**.**
- The dimension of the row space always equals the dimension of the column space which always equals the number of pivots in the row echelon form of the equation, k. This number, k, is called the **rank** of matrix A, $\text{rank}(A)$.
- To find a specific (particular) solution to system $Ax = b$, set the free variables to zero and solve $Bx = \hat{y}$ for the dependent variables. Let us call this solution x_s.
- To find $n-k$ linearly independent vectors in $\mathcal{N}(A)$, follow the following procedure, assuming that $n_0, \ldots n_{n-k-1}$ equal the indices of the free variables. (In other words: $\chi_{n_0}, \ldots, \chi_{n_{n-k-1}}$ equal the free variables.)
  - Set χ_{n_j} equal to one and χ_{n_k} with $n_k \neq n_j$ equal to zero. Solve for the dependent variables.

  This yields $n-k$ linearly independent vectors that are a basis for $\mathcal{N}(A)$. Let us call these $x_{n_0}, \ldots, x_{n_{n-k-1}}$.
- The general (complete) solution is then given as

$$x_s + \gamma_0 x_{n_0} + \gamma_1 x_{n_1} + \cdots + \gamma_{n-k-1} x_{n_{n-k-1}}.$$

- To find a basis for the column space of A, $\mathcal{C}(A)$, you take the columns of A that correspond to the columns with pivots in B.
- To find a basis for the row space of A, $\mathcal{R}(A)$, you take the rows of B that contain pivots, and transpose those into the vectors that become the desired basis. (Note: you take the rows of B, not A.)
- The following are all equal:
  - The dimension of the column space.
  - The rank of the matrix.
  - The number of dependent variables.
  - The number of nonzero rows in the upper echelon form.
  - The number of columns in the matrix minus the number of free variables.
  - The number of columns in the matrix minus the dimension of the null space.
  - The number of linearly independent columns in the matrix.
  - The number of linearly independent rows in the matrix.

Orthogonal vectors

Definition 10.14 *Two vectors $x, y \in \mathbb{R}^m$ are orthogonal if and only if $x^T y = 0$.*

Orthogonal subspaces

Definition 10.15 *Two subspaces $\mathbf{V}, \mathbf{W} \subset \mathbb{R}^m$ are orthogonal if and only if $v \in \mathbf{V}$ and $w \in \mathbf{W}$ implies $v^T w = 0$.*

Definition 10.16 *Let $\mathbf{V} \subset \mathbb{R}^m$ be a subspace. Then $\mathbf{V}^\perp \subset \mathbb{R}^m$ equals the set of all vectors that are orthogonal to $\mathbf{V}$.*

Theorem 10.17 *Let $\mathbf{V} \subset \mathbb{R}^m$ be a subspace. Then $\mathbf{V}^\perp$ is a subspace of $\mathbb{R}^m$.*

The Fundamental Subspaces

- The column space of a matrix $A \in \mathbb{R}^{m \times n}$, $\mathcal{C}(\mathcal{A})$, equals the set of all vectors in $\mathbb{R}^m$ that can be written as Ax: $\{y \mid y = Ax\}$.
- The null space of a matrix $A \in \mathbb{R}^{m \times n}$, $\mathcal{N}(A)$, equals the set of all vectors in $\mathbb{R}^n$ that map to the zero vector: $\{x \mid Ax = 0\}$.
- The row space of a matrix $A \in \mathbb{R}^{m \times n}$, $\mathcal{R}(A)$, equals the set of all vectors in $\mathbb{R}^n$ that can be written as $A^T x$: $\{y \mid y = A^T x\}$.
- The left null space of a matrix $A \in \mathbb{R}^{m \times n}$, $\mathcal{N}(A^T)$, equals the set of all vectors in $\mathbb{R}^m$ described by $\{x \mid x^T A = 0\}$.

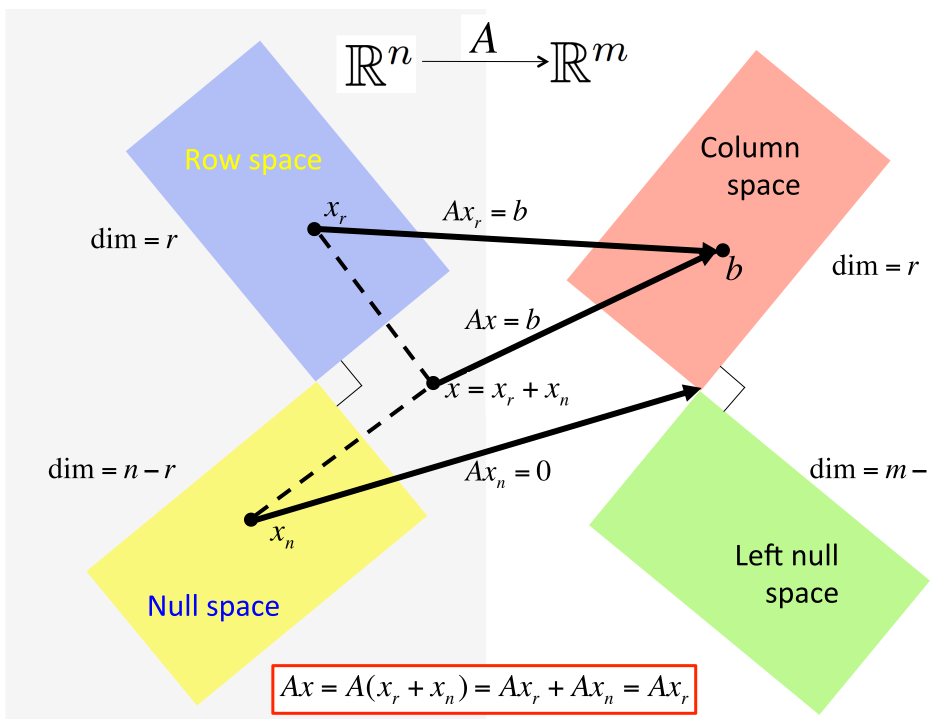
Theorem 10.18 *Let $A \in \mathbb{R}^{m \times n}$. Then $\mathcal{R}(A) \perp \mathcal{N}(A)$.*

Theorem 10.19 *Let $A \in \mathbb{R}^{m \times n}$. Then every $x \in \mathbb{R}^n$ can be written as $x = x_r + x_n$ where $x_r \in \mathcal{R}(A)$ and $x_n \in \mathcal{N}(A)$.*

Theorem 10.20 *Let $A \in \mathbb{R}^{m \times n}$. Then A is a one-to-one, onto mapping from $\mathcal{R}(A)$ to $\mathcal{C}(\mathcal{A})$.*

Theorem 10.21 *Let $A \in \mathbb{R}^{m \times n}$. Then the left null space of A is orthogonal to the column space of A and the dimension of the left null space of A equals $m - r$, where r is the dimension of the column space of A.*

An important figure:



Overdetermined systems

- $Ax = b$ has a solution if and only if $b \in \mathcal{C}(A)$.
- Let us assume that A has linearly independent columns and we wish to solve $Ax \approx b$. Then
  - The solution of the normal equations
  $$A^T A x = A^T b$$
  is the best solution (in the linear least-squares sense) to $Ax \approx b$.
  - The pseudo inverse of A is given by $A^\dagger = (A^T A)^{-1} A^T$.

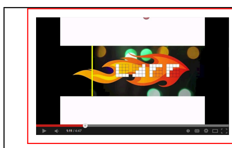
- The best solution (in the linear least-squares sense) of $Ax = b$ is given by $\hat{x} = A^{\dagger} b = (A^T A)^{-1} A^T b$.
- The orthogonal projection of b onto $\mathcal{C}(A)$ is given by $\hat{b} = A(A^T A)^{-1} A^T b$.
- The vector $(b - \hat{b})$ is the component of b orthogonal to $\mathcal{C}(A)$.
- The orthogonal projection of b onto $\mathcal{C}(A)^{\perp}$ is given by $b - \hat{b} = [I - A(A^T A)^{-1} A^T] b$.

Week 11

Orthogonal Projection, Low Rank Approximation, and Orthogonal Bases

11.1 Opening Remarks

11.1.1 Low Rank Approximation



☛ YouTube

11.1.2 Outline

11.1.3 What You Will Learn

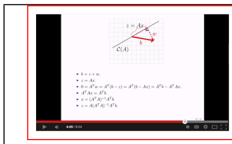
Upon completion of this unit, you should be able to

- Given vectors a and b in $\mathbb{R}^m$, find the component of b in the direction of a and the component of b orthogonal to a.
- Given a matrix A with linear independent columns, find the matrix that projects any given vector b onto the column space A and the matrix that projects b onto the space orthogonal to the column space of A, which is also called the left null space of A.
- Understand low rank approximation, projecting onto columns to create a rank-k approximation.
- Identify, apply, and prove simple properties of orthonormal vectors.
- Determine if a set of vectors is orthonormal.
- Transform a set of basis vectors into an orthonormal basis using Gram-Schmidt orthogonalization.
- Compute an orthonormal basis for the column space of A.
- Apply Gram-Schmidt orthogonalization to compute the QR factorization.
- Solve the Linear Least-Squares Problem via the QR Factorization.
- Make a change of basis.
- Be aware of the existence of the Singular Value Decomposition and that it provides the “best” rank-k approximation.

Track your progress in Appendix B.

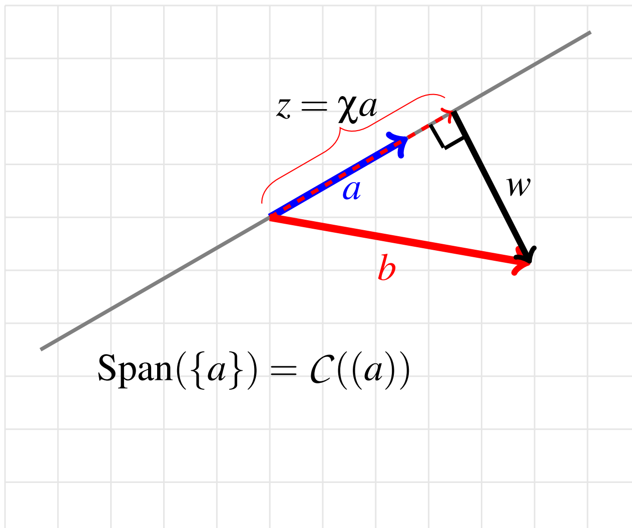
11.2 Projecting a Vector onto a Subspace

11.2.1 Component in the Direction of ...



☛ YouTube

Consider the following picture:



Here, we have two vectors, $a, b \in \mathbb{R}^m$. They exist in the plane defined by $\mathrm{Span}(\{a, b\})$ which is a two dimensional space (unless a and b point in the same direction). From the picture, we can also see that b can be thought of as having a component z in the direction of a and another component w that is orthogonal (perpendicular) to a. The component in the direction of a lies in the $\mathrm{Span}(\{a\}) = \mathcal{C}((a))$ (here (a) denotes the matrix with only once column, a) while the component that is orthogonal to a lies in $\mathrm{Span}(\{a\})^\perp$. Thus,

$$b = z + w,$$

where

- $z = \chi a$ with $\chi \in \mathbb{R}$; and
- $a^T w = 0$.

Noting that $w = b - z$ we find that

$$0 = a^T w = a^T (b - z) = a^T (b - \chi a)$$

or, equivalently,

$$a^T a \chi = a^T b.$$

We have seen this before. Recall that when you want to approximately solve $Ax = b$ where b is not in $\mathcal{C}(A)$ via Linear Least Squares, the "best" solution satisfies $A^T A x = A^T b$. The equation that we just derived is the exact same, except that A has one column: $A = (a)$.

Then, provided $a \neq 0$,

$$\chi = (a^T a)^{-1}(a^T b).$$

Thus, the component of b in the direction of a is given by

$$u = \chi a = (a^T a)^{-1}(a^T b) a = a (a^T a)^{-1}(a^T b) = \left[a (a^T a)^{-1} a^T \right] b.$$

Note that we were able to move a to the left of the equation because $(a^T a)^{-1}$ and $a^T b$ are both scalars. The component of b orthogonal (perpendicular) to a is given by

$$w = b - z = b - \left(a(a^T a)^{-1} a^T\right) b = Ib - \left(a(a^T a)^{-1} a^T\right) b = \left(I - a(a^T a)^{-1} a^T\right) b.$$

Summarizing:

$$\begin{array}{rcll} z & = & \left(a(a^T a)^{-1} a^T\right) b & \text{is the component of } b \text{ in the direction of } a\text{; and} \\ w & = & \left(I - a(a^T a)^{-1} a^T\right) b & \text{is the component of } b \text{ perpendicular (orthogonal) to } a. \end{array}$$

We say that, given vector a, the matrix that *projects* any given vector b onto the space spanned by a is given by

$$a(a^T a)^{-1} a^T \qquad (= \frac{1}{a^T a} a a^T)$$

since $a(a^T a)^{-1} a^T b$ is the component of b in $\mathrm{Span}(\{a\})$. Notice that this is an outer product:

$$a \underbrace{(a^T a)^{-1} a^T}_{v^T}.$$

We say that, given vector a, the matrix that *projects* any given vector b onto the space orthogonal to the space spanned by a is given by

$$I - a(a^T a)^{-1} a^T \qquad (= I - \frac{1}{a^T a} a a^T = I - a v^T),$$

since $\left(I - a(a^T a)^{-1} a^T\right) b$ is the component of b in $\mathrm{Span}(\{a\})^\perp$.

Notice that $I - \frac{1}{a^T a} a a^T = I - a v^T$ is a rank-1 update to the identity matrix.

Homework 11.2.1.1 Let $a = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $P_a(x)$ and $P_a^\perp(x)$ be the projection of vector x onto $\mathrm{Span}(\{a\})$ and $\mathrm{Span}(\{a\})^\perp$, respectively. Compute

1. $P_a(\begin{pmatrix} 2 \\ 0 \end{pmatrix}) =$
2. $P_a^\perp(\begin{pmatrix} 2 \\ 0 \end{pmatrix}) =$
3. $P_a(\begin{pmatrix} 4 \\ 2 \end{pmatrix}) =$
4. $P_a^\perp(\begin{pmatrix} 4 \\ 2 \end{pmatrix}) =$
5. Draw a picture for each of the above.

☛ SEE ANSWER

Homework 11.2.1.2 Let $a = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$ and $P_a(x)$ and $P_a^{\perp}(x)$ be the projection of vector x onto $\text{Span}(\{a\})$ and $\text{Span}(\{a\})^{\perp}$, respectively. Compute

1. $P_a(\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}) =$

2. $P_a^{\perp}(\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}) =$

3. $P_a(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}) =$

4. $P_a^{\perp}(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}) =$

☛ SEE ANSWER

Homework 11.2.1.3 Let $a, v, b \in \mathbb{R}^m$.
What is the approximate cost of computing $(av^T)b$, obeying the order indicated by the parentheses?

- $m^2 + 2m$.
- $3m^2$.
- $2m^2 + 4m$.

What is the approximate cost of computing $(v^T b)a$, obeying the order indicated by the parentheses?

- $m^2 + 2m$.
- $3m$.
- $2m^2 + 4m$.

☛ SEE ANSWER

For computational efficiency, it is important to compute $a(a^T a)^{-1} a^T b$ according to order indicated by the following parentheses:

$$((a^T a)^{-1}(a^T b))a.$$

Similarly, $(I - a(a^T a)^{-1} a^T)b$ should be computed as

$$b - (((a^T a)^{-1}(a^T b))a).$$

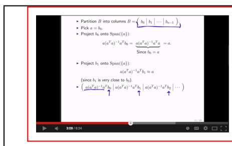
Homework 11.2.1.4 Given $a, x \in \mathbb{R}^m$, let $P_a(x)$ and $P_a^\perp(x)$ be the projection of vector x onto $\mathrm{Span}(\{a\})$ and $\mathrm{Span}(\{a\})^\perp$, respectively. Then which of the following are true:

1. $P_a(a) = a$. True/False
2. $P_a(\chi a) = \chi a$. True/False
3. $P_a^\perp(\chi a) = 0$ (the zero vector). True/False
4. $P_a(P_a(x)) = P_a(x)$. True/False
5. $P_a^\perp(P_a^\perp(x)) = P_a^\perp(x)$. True/False
6. $P_a(P_a^\perp(x)) = 0$ (the zero vector). True/False

(Hint: Draw yourself a picture.)

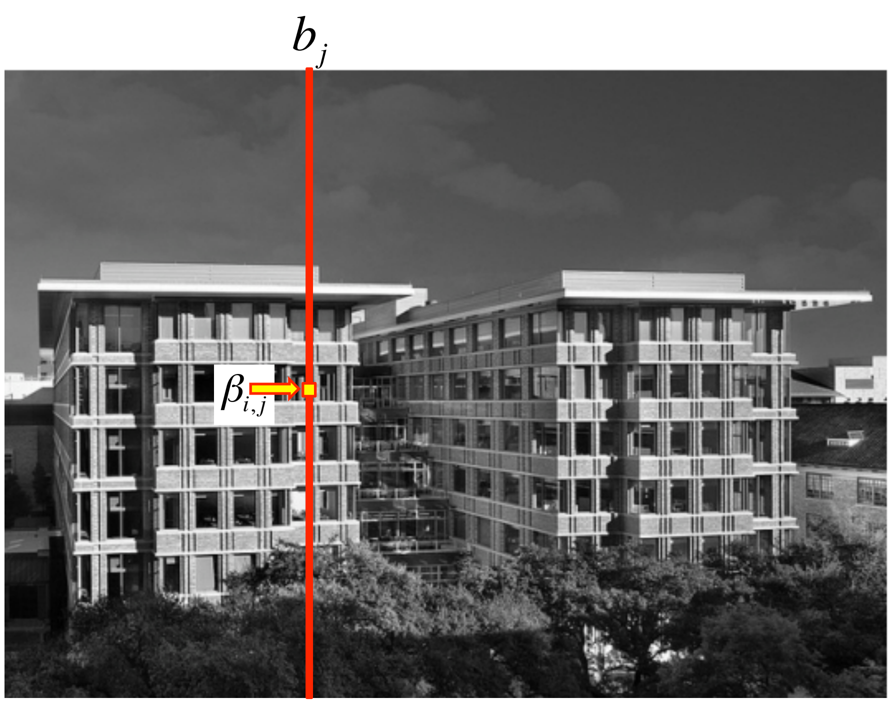
☛ SEE ANSWER

11.2.2 An Application: Rank-1 Approximation



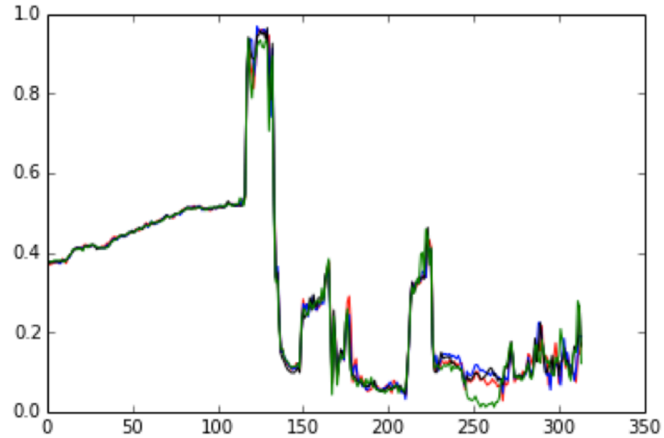
☛ YouTube

Consider the picture



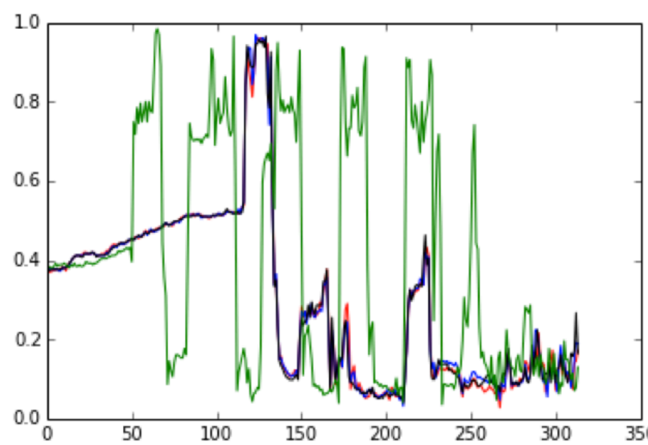
This picture can be thought of as a matrix $B \in \mathbb{R}^{m \times n}$ where each element in the matrix encodes a pixel in the picture. The jth column of B then encodes the jth column of pixels in the picture.

Now, let's focus on the first few columns. Notice that there is a lot of similarity in those columns. This can be illustrated by plotting the values in the column as a function of the element in the column:

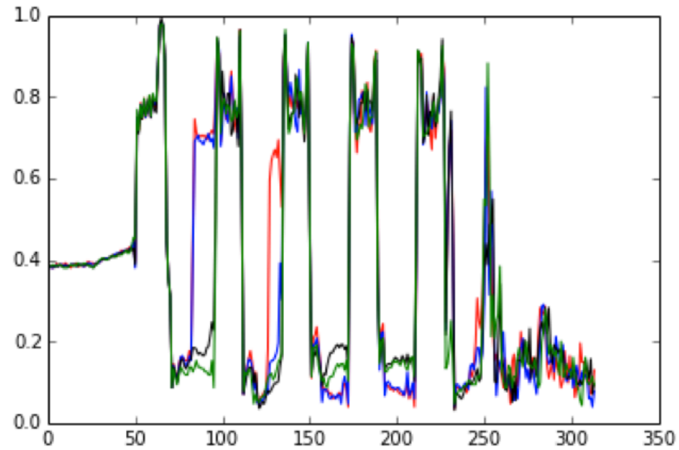


In the graph on the left, we plot $\beta_{i,j}$, the value of the (i, j) pixel, for $j = 0, 1, 2, 3$ in different colors. The picture on the right highlights the columns for which we are doing this. The green line corresponds to $j = 3$ and you notice that it is starting to deviate some for i near 250.

If we now instead look at columns $j = 0, 1, 2, 100$, where the green line corresponds to $j = 100$, we see that that column in the picture is dramatically different:



Changing this to plotting $j = 100, 101, 102, 103$ and we notice a lot of similarity again:



Now, let's think about this from the point of view taking one vector, say the first column of B, b_0, and projecting the other columns onto the span of that column. What does this mean?

- Partition B into columns $B = \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right)$.
- Pick $a = b_0$.
- Focus on projecting b_0 onto $\mathrm{Span}(\{a\})$:

$$a(a^Ta)^{-1}a^Tb_0 = \underbrace{a(a^Ta)^{-1}a^Ta}_{\text{Since } b_0 = a} = a.$$

  Of course, this is what we expect when projecting a vector onto itself.

- Next, focus on projecting b_1 onto $\mathrm{Span}(\{a\})$:

$$a(a^Ta)^{-1}a^Tb_1$$

  since b_1 is very close to b_0.

- Do this for all columns, and create a picture with all of the projected vectors:

$$\left(\begin{array}{c|c|c|c} a(a^Ta)^{-1}a^Tb_0 & a(a^Ta)^{-1}a^Tb_1 & a(a^Ta)^{-1}a^Tb_2 & \cdots \end{array} \right)$$

- Now, remember that if T is some matrix, then

$$TB = \left(\begin{array}{c|c|c|c} Tb_0 & Tb_1 & Tb_2 & \cdots \end{array} \right).$$

  If we let $T = a(a^Ta)^{-1}a^T$ (the matrix that projects onto $\mathrm{Span}(\{a\})$, then

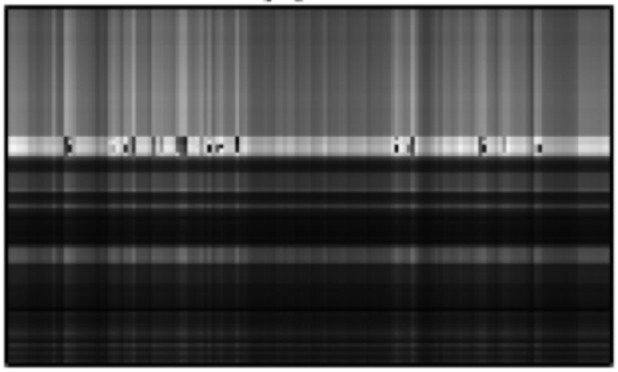
$$a(a^Ta)^{-1}a^T \left(\begin{array}{c|c|c|c} b_0 & b_1 & b_2 & \cdots \end{array} \right) = a(a^Ta)^{-1}a^TB.$$

- We can manipulate this further by recognizing that $y^T = (a^Ta)^{-1}a^TB$ can be computed as $y = (a^Ta)^{-1}B^Ta$:

$$a(a^Ta)^{-1}a^TB = a\ (\underbrace{(a^Ta)^{-1}B^Ta}_{y})^T = ay^T$$

- We now recognize ay^T as an outer product (a column vector times a row vector).

- If we do this for our picture, we get the picture on the left:



Notice how it seems like each column is the same, except with some constant change in the gray-scale. The same is true for rows. Why is this? If you focus on the left-most columns in the picture, they almost look correct (comparing to the left-most columns in the picture on the right). Why is this?

- The benefit of the approximation on the left is that it can be described with two vectors: a and y ($n+m$ floating point numbers) while the original matrix on the right required an entire matrix ($m \times n$ floating point numbers).

- The disadvantage of the approximation on the left is that it is hard to recognize the original picture...

Homework 11.2.2.1 Let $\mathbf{S}$ and $\mathbf{T}$ be subspaces of $\mathbb{R}^m$ and $\mathbf{S} \subset \mathbf{T}$.
$\dim(\mathbf{S}) \leq \dim(\mathbf{T})$.

Always/Sometimes/Never

☛ SEE ANSWER

Homework 11.2.2.2 Let $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. Then the $m \times n$ matrix uv^T has a rank of at most one.

True/False

☛ SEE ANSWER

Homework 11.2.2.3 Let $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. Then uv^T has rank equal to zero if
(Mark all correct answers.)

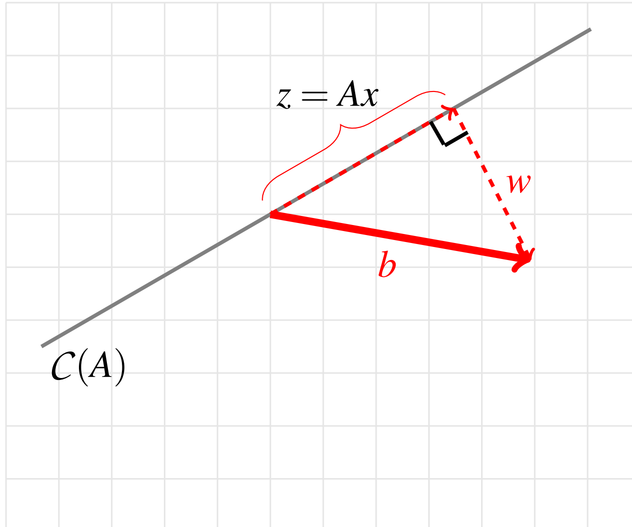
1. $u = 0$ (the zero vector in $\mathbb{R}^m$).
2. $v = 0$ (the zero vector in $\mathbb{R}^n$).
3. Never.
4. Always.

☛ SEE ANSWER

11.2.3 Projection onto a Subspace

No video this section

Next, consider the following picture:



What we have here are

- Matrix $A \in \mathbb{R}^{m \times n}$.
- The space spanned by the columns of A: $\mathcal{C}(A)$.
- A vector $b \in \mathbb{R}^m$.
- Vector z, the component of b in $\mathcal{C}(A)$ which is also the vector in $\mathcal{C}(A)$ closest to the vector b. Since this vector is in the column space of A, $z = Ax$ for some vector $x \in \mathbb{R}^n$.
- The vector w which is the component of b orthogonal to $\mathcal{C}(A)$.

The vectors b, z, w, all exist in the same planar subspace since $b = z + w$, which is the page on which these vectors are drawn in the above picture.

Thus,

$$b = z + w,$$

where

- $z = Ax$ with $x \in \mathbb{R}^n$; and
- $A^T w = 0$ since w is orthogonal to the column space of A and hence in $\mathcal{N}(A^T)$.

Noting that $w = b - z$ we find that

$$0 = A^T w = A^T(b - z) = A^T(b - Ax)$$

or, equivalently,

$$A^T Ax = A^T b.$$

This should look familiar!

Then, provided $(A^T A)^{-1}$ exists (which, we saw before happens when A has linearly independent columns),

$$x = (A^T A)^{-1} A^T b.$$

Thus, the component of b in $\mathcal{C}(A)$ is given by

$$z = Ax = A(A^T A)^{-1} A^T b$$

while the component of b orthogonal (perpendicular) to $\mathcal{C}(A)$ is given by

$$w = b - z = b - A(A^T A)^{-1} A^T b = Ib - A(A^T A)^{-1} A^T b = \left(I - A(A^T A)^{-1} A^T\right) b.$$

Summarizing:

$$\begin{array}{rcl} z & = & A(A^T A)^{-1} A^T b \\ w & = & \left(I - A(A^T A)^{-1} A^T\right) b. \end{array}$$

We say that, given matrix A with linearly independent columns, the matrix that *projects* a given vector b onto the column space of A is given by

$$A(A^TA)^{-1}A^T$$

since $A(A^TA)^{-1}A^Tb$ is the component of b in $\mathcal{C}(A)$.

We say that, given matrix A with linearly independent columns, the matrix that *projects* a given vector b onto the space orthogonal to the column space of A (which, recall, is the *left null space* of A) is given by

$$I - A(A^TA)^{-1}A^T$$

since $\left(I - A(A^TA)^{-1}A^T\right)b$ is the component of b in $\mathcal{C}(A)^\perp = \mathcal{N}(A^T)$.

Homework 11.2.3.1 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 2 \\ 7 \end{pmatrix}$.

1. Find the projection of b onto the column space of A.
2. Split b into $z + w$ where z is in the column space and w is perpendicular (orthogonal) to that space.
3. Which of the four subspaces ($C(A)$, $R(A)$, $\mathcal{N}(A)$, $\mathcal{N}(A^T)$) contains w?

☛ SEE ANSWER

For computational reasons, it is important to compute $A(A^TA)^{-1}A^Tx$ according to order indicated by the following parentheses:

$$A[(A^TA)^{-1}[A^Tx]]$$

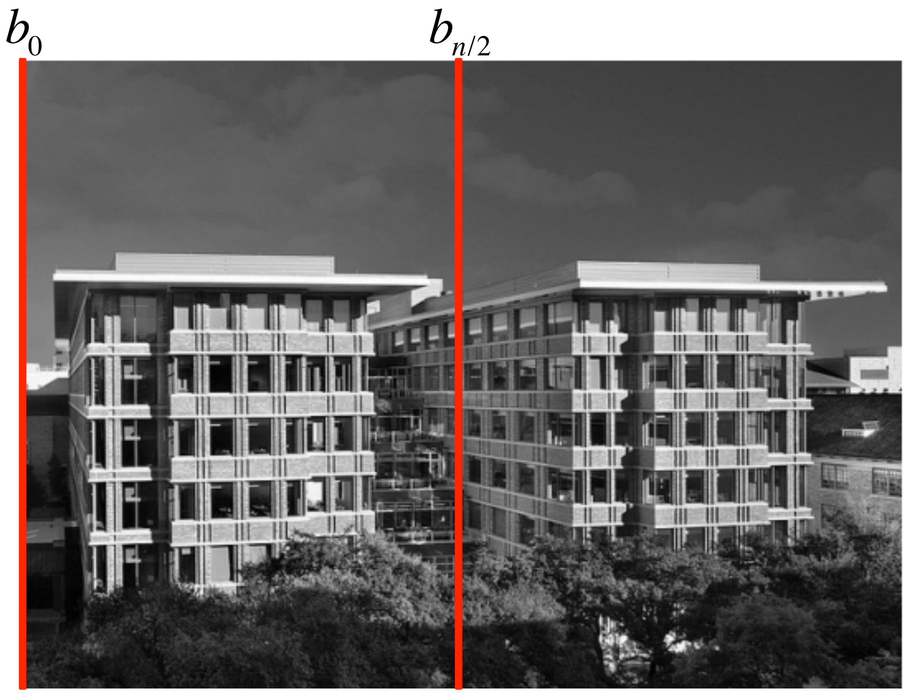
Similarly, $(I - A(A^TA)^{-1}A^T)x$ should be computed as

$$x - [A[(A^TA)^{-1}[A^Tx]]]$$

11.2.4 An Application: Rank-2 Approximation

☛ **YouTube**

Earlier, we took the first column as being representative of all columns of the picture. Looking at the picture, this is clearly not the case. But what if we took two columns instead, say column $j = 0$ and $j = n/2$, and projected each of the columns onto the subspace spanned by those two columns:



- Partition B into columns $B = \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right)$.
- Pick $A = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) = \left(\begin{array}{c|c} b_0 & b_{n/2} \end{array} \right)$.
- Focus on projecting b_0 onto $\mathrm{Span}(\{a_0, a_1\}) = \mathcal{C}(A)$:

$$A(A^TA)^{-1}A^Tb_0 = a = b_0$$

  because a is in $\mathcal{C}(A)$ and a is therefore the best vector in $\mathcal{C}(A)$.
- Next, focus on projecting b_1 onto $\mathrm{Span}(\{a\})$:

$$A(A^TA)^{-1}A^Tb_1 \approx b_1$$

  since b_1 is very close to a.
- Do this for all columns, and create a picture with all of the projected vectors:

$$\left(\begin{array}{c|c|c|c} A(A^TA)^{-1}A^Tb_0 & A(A^TA)^{-1}A^Tb_1 & A(A^TA)^{-1}A^Tb_2 & \cdots \end{array} \right)$$

- Now, remember that if T is some matrix, then

$$TB = \left(\begin{array}{c|c|c|c} Tb_0 & Tb_1 & Tb_2 & \cdots \end{array} \right).$$

  If we let $T = A(A^TA)^{-1}A^T$ (the matrix that projects onto $\mathcal{C}(A)$, then

$$A(A^TA)^{-1}A^T \left(\begin{array}{c|c|c|c} b_0 & b_1 & b_2 & \cdots \end{array} \right) = A(A^TA)^{-1}A^TB.$$

- We can manipulate this by letting $W = B^T A (A^T A)^{-1}$ so that

$$A \underbrace{(A^T A)^{-1} A^T B}_{W^T} = A W^T.$$

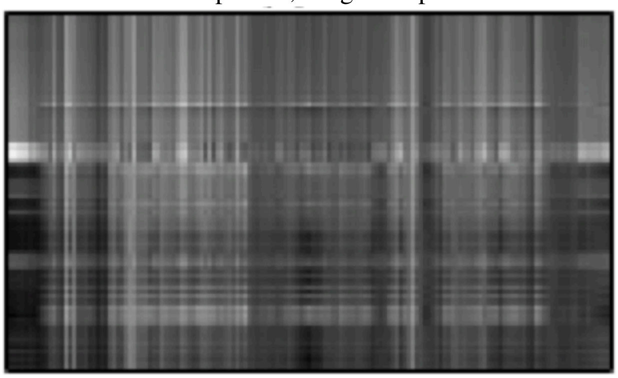
  Notice that A and W each have two columns.

- We now recognize AW^T is the sum of two outer products:

$$AW^T = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \left(\begin{array}{c|c} w_0 & w_1 \end{array} \right)^T = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \left(\begin{array}{c} w_0^T \\ w_1^T \end{array} \right) = a_0 w_0^T + a_1 w_1^T.$$

  It can be easily shown that this matrix has rank of at most two, which is why this would be called a rank-2 approximation of B.

- If we do this for our picture, we get the picture on the left:




  We are starting to see some more detail.

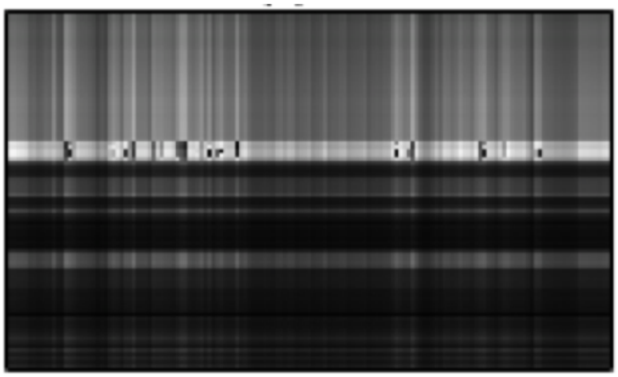
- We now have to store only a $n \times 2$ and $m \times 2$ matrix (A and W).

11.2.5 An Application: Rank-k Approximation

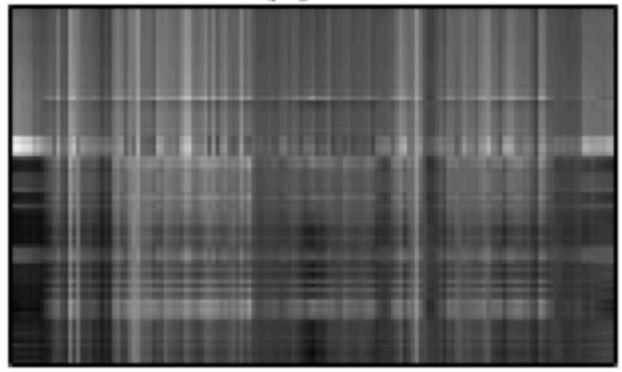
☛ YouTube

Rank-k approximations

We can improve the approximations above by picking progressively more columns for A. The following progression of pictures shows the improvement as more and more columns are used, where k indicates the number of columns:



$k = 1$



$k = 2$

$k = 10$

$k = 25$

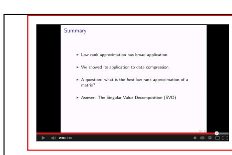
$k = 50$

original

Homework 11.2.5.1 Let $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$. Then the $m \times n$ matrix UV^T has rank at most k.

True/False

☛ SEE ANSWER



☛ **YouTube**

Homework 11.2.5.2 We discussed in this section that the projection of B onto the column space of A is given by $A(A^TA)^{-1}A^TB$. So, if we compute $V = (A^TA)^{-1}A^TB$, then AV is an approximation to B that requires only $m \times k$ matrix A and $k \times n$ matrix V.
To compute V, we can perform the following steps:

- Form $C = A^TA$.
- Compute the LU factorization of C, overwriting C with the resulting L and U.
- Compute $V = A^TB$.
- Solve $LX = V$, overwriting V with the solution matrix X.
- Solve $UX = V$, overwriting V with the solution matrix X.
- Compute the approximation of B as $A \cdot V$ (A times V). In practice, you would not compute this approximation, but store A and V instead, which typically means less data is stored.

To experiments with this, download Week11.zip, place it in

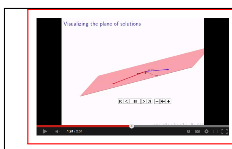
```
LAFF-2.0xM -> Programming
```

and unzip it. Then examine the file `Week11/CompressPicture.m`, look for the comments on what operations need to be inserted, and insert them. Execute the script in the Command Window and see how the picture in file `building.png` is approximated. Play with the number of columns used to approximate. Find your own picture! (It will have to be a black-and-white picture for what we discussed to work.
Notice that A^TA is a symmetric matrix, and it can be shown to be symmetric positive definite under most circumstances (when A has linearly independent columns). This means that instead of the LU factorization, one can use the Cholesky factorization (see the enrichment in Week 8). In `Week11.zip` you will also find a function for computing the Cholesky factorization. Try to use it to perform the calculations.

☛ SEE ANSWER

11.3 Orthonormal Bases

11.3.1 The Unit Basis Vectors, Again



☛ YouTube

Recall the unit basis vectors in $\mathbb{R}^3$:

$$e_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad e_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad e_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

This set of vectors forms a basis for $\mathbb{R}^3$; they are linearly independent and any vector $x \in \mathbb{R}^3$ can be written as a linear combination of these three vectors.

Now, the set

$$v_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad v_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad v_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

is also a basis for $\mathbb{R}^3$, but is not nearly as nice:

- Two of the vectors are not of length one.

- They are not orthogonal to each other.

There is something pleasing about a basis that is **orthonormal**. By this we mean that each vector in the basis is of length one, and any pair of vectors is orthogonal to each other.

A question we are going to answer in the next few units is how to take a given basis for a subspace and create an orthonormal basis from it.

Homework 11.3.1.1 Consider the vectors

$$v_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad v_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad v_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

1. Compute
   (a) $v_0^T v_1 =$
   (b) $v_0^T v_2 =$
   (c) $v_1^T v_2 =$
2. These vectors are orthonormal. True/False

☛ SEE ANSWER

11.3.2 Orthonormal Vectors

☛ YouTube

Definition 11.1 *Let $q_0, q_1, \ldots, q_{k-1} \in \mathbb{R}^m$. Then these vectors are (mutually) orthonormal if for all $0 \leq i, j < k$:*

$$q_i^T q_j = \begin{cases} 1 & \textit{if } i = j \\ 0 & \textit{otherwise}. \end{cases}$$

Homework 11.3.2.1

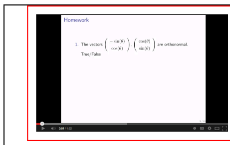
1. $\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} =$

2. $\begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} =$

3. The vectors $\begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$ are orthonormal. True/False

4. The vectors $\begin{pmatrix} \sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ -\sin(\theta) \end{pmatrix}$ are orthonormal. True/False

☛ SEE ANSWER



☛ YouTube

Homework 11.3.2.2 Let $q_0, q_1, \ldots, q_{k-1} \in \mathbb{R}^m$ be a set of orthonormal vectors. Let

$$Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right).$$

Then $Q^T Q = I$.

TRUE/FALSE

☛ SEE ANSWER

☛ YouTube

Homework 11.3.2.3 Let $Q \in \mathbb{R}^{m \times k}$ (with $k \leq m$) and $Q^T Q = I$. Partition

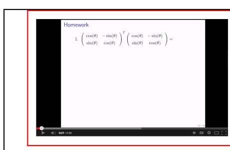
$$Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right).$$

Then $q_0, q_1, \ldots, q_{k-1}$ are orthonormal vectors.

TRUE/FALSE

☛ SEE ANSWER

☛ YouTube

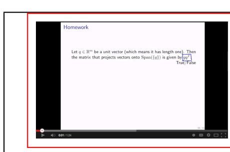


☛ YouTube

Homework 11.3.2.4 Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Then the matrix that projects vectors onto $\text{Span}(\{q\})$ is given by qq^T.

True/False

☛ SEE ANSWER

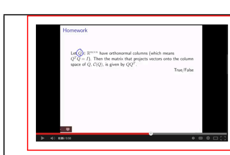


☛ YouTube

Homework 11.3.2.5 Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Let $x \in \mathbb{R}^m$. Then the component of x in the direction of q (in $\text{Span}(\{q\})$) is given by $q^T xq$.

True/False

☛ SEE ANSWER

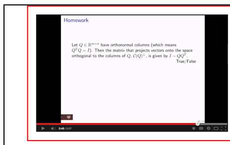


☛ YouTube

Homework 11.3.2.6 Let $Q \in \mathbb{R}^{m \times n}$ have orthonormal columns (which means $Q^T Q = I$). Then the matrix that projects vectors onto the column space of Q, $\mathcal{C}(Q)$, is given by QQ^T.

True/False

☛ SEE ANSWER

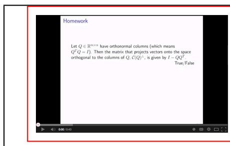


☛ YouTube

Homework 11.3.2.7 Let $Q \in \mathbb{R}^{m \times n}$ have orthonormal columns (which means $Q^T Q = I$). Then the matrix that projects vectors onto the space orthogonal to the columns of Q, $\mathcal{C}(Q)^{\perp}$, is given by $I - QQ^T$.

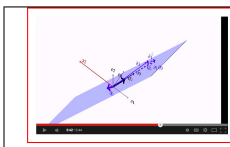
True/False

☛ SEE ANSWER

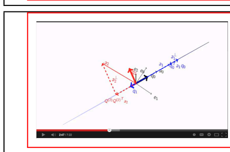


☛ YouTube

11.3.3 Orthogonal Bases



☛ YouTube



☛ YouTube

The fundamental idea for this unit is that it is convenient for a basis to be orthonormal. The question is: how do we transform a given set of basis vectors (e.g., the columns of a matrix A with linearly independent columns) into a set of orthonormal vectors that form a basis for the same space? The process we will described is known as **Gram-Schmidt orthogonalization** (GS orthogonalization).

The idea is very simple:

- Start with a set of n linearly independent vectors, $a_0, a_1, \ldots, a_{n-1} \in \mathbb{R}^m$.
- Take the first vector and make it of unit length:

$$q_0 = a_0 / \underbrace{\|a_0\|_2}_{\rho_{0,0}},$$

  where $\rho_{0,0} = \|a_0\|_2$, the length of a_0.

  Notice that $\text{Span}(\{a_0\}) = \text{Span}(\{q_0\})$ since q_0 is simply a scalar multiple of a_0.

  This gives us one orthonormal vector, q_0.
- Take the second vector, a_1, and compute its component *orthogonal* to q_0:

$$a_1^{\perp} = (I - q_0 q_0^T) a_1 = a_1 - q_0 q_0^T a_1 = a_1 - \underbrace{q_0^T a_1}_{\rho_{0,1}} q_0.$$

- Take $a_1^\perp$, the component of a_1 *orthogonal* to q_0, and make it of unit length:

$$q_1 = a_1^\perp / \underbrace{\|a_1^\perp\|_2}_{\rho_{1,1}},$$

We will see later that $\mathrm{Span}(\{a_0, a_1\}) = \mathrm{Span}(\{q_0, q_1\})$.

This gives us two orthonormal vectors, q_0, q_1.

- Take the third vector, a_2, and compute its component *orthogonal* to $Q^{(2)} = \left(\begin{array}{cc} q_0 & q_1 \end{array} \right)$ (orthogonal to both q_0 and q_1 and hence $\mathrm{Span}(\{q_0, q_1\}) = \mathcal{C}(Q^{(2)})$:

$$\begin{array}{rcl}
a_2^\perp & = & \underbrace{(I - Q^{(2)} Q^{(2)\,T}) a_2}_{\text{Projection onto } \mathcal{C}(Q^{(2)})^\perp} = a_2 - \underbrace{Q^{(2)} Q^{(2)\,T} a_2}_{\text{Component in } \mathcal{C}(Q^{(2)})} = a_2 - \left(\begin{array}{cc} q_0 & q_1 \end{array} \right) \left(\begin{array}{cc} q_0 & q_1 \end{array} \right)^T a_2 \\
& = & a_2 - \left(\begin{array}{cc} q_0 & q_1 \end{array} \right) \left(\begin{array}{c} q_0^T \\ q_1^T \end{array} \right) a_2 = a_2 - \left(\begin{array}{cc} q_0 & q_1 \end{array} \right) \left(\begin{array}{c} q_0^T a_2 \\ q_1^T a_2 \end{array} \right) \\
& = & a_2 - (q_0^T a_2 q_0 + q_1^T a_2 q_1) \\
& = & a_2 - \underbrace{q_0^T a_2 q_0}_{\text{Component in direction of } q_0} - \underbrace{q_1^T a_2 q_1.}_{\text{Component in direction of } q_1}
\end{array}$$

Notice:

  - $a_2 - q_0^T a_2 q_0$ equals the vector a_2 with the component in the direction of q_0 subtracted out.
  - $a_2 - q_0^T a_2 q_0 - q_1^T a_2 q_1$ equals the vector a_2 with the components in the direction of q_0 **and** q_1 subtracted out.
  - Thus, $a_2^\perp$ equals component of a_2 that is orthogonal to both q_0 **and** q_1.

- Take $a_2^\perp$, the component of a_2 *orthogonal* to q_0 and q_1, and make it of unit length:

$$q_2 = a_2^\perp / \underbrace{\|a_2^\perp\|_2}_{\rho_{2,2}},$$

We will see later that $\mathrm{Span}(\{a_0, a_1, a_2\}) = \mathrm{Span}(\{q_0, q_1, q_2\})$.

This gives us three orthonormal vectors, q_0, q_1, q_2.

- (Continue repeating the process)

- Take vector a_k, and compute its component *orthogonal* to $Q^{(k)} = \left(\begin{array}{cccc} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)$ (orthogonal to all vectors $q_0, q_1, \ldots, q_{k-1}$ and hence $\mathrm{Span}(\{q_0, q_1, \ldots, q_{k-1}\}) = \mathcal{C}(Q^{(k)})$:

$$\begin{array}{rcl}
a_k^\perp & = & (I - Q^{(k)} Q^{(k)\,T}) a_k = a_k - Q^{(k)} Q^{(k)\,T} a_k = a_k - \left(\begin{array}{cccc} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) \left(\begin{array}{cccc} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)^T a_k \\
& = & a_k - \left(\begin{array}{cccc} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) \left(\begin{array}{c} q_0^T \\ q_1^T \\ \vdots \\ q_{k-1}^T \end{array} \right) a_k = a_k - \left(\begin{array}{cccc} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) \left(\begin{array}{c} q_0^T a_k \\ q_1^T a_k \\ \vdots \\ q_{k-1}^T a_k \end{array} \right) \\
& = & a_k - q_0^T a_k q_0 - q_1^T a_k q_1 - \cdots q_{k-1}^T a_k q_{k-1}.
\end{array}$$

Notice:

- $a_k - q_0^T a_k q_0$ equals the vector a_k with the component in the direction of q_0 subtracted out.
- $a_k - q_0^T a_k q_0 - q_1^T a_k q_1$ equals the vector a_k with the components in the direction of q_0 **and** q_1 subtracted out.
- $a_k - q_0^T a_k q_0 - q_1^T a_k q_1 - \cdots - q_{k-1}^T a_k q_{k-1}$ equals the vector a_k with the components in the direction of $q_0, q_1, \ldots, q_{k-1}$ subtracted out.
- Thus, $a_k^\perp$ equals component of a_k that is orthogonal to all vectors q_j that have already been computed.

- Take $a_k^\perp$, the component of a_k *orthogonal* to $q_0, q_1, \ldots q_{k-1}$, and make it of unit length:

$$q_k = a_k^\perp / \underbrace{\|a_k^\perp\|_2}_{\rho_{k,k}},$$

We will see later that $\text{Span}(\{a_0, a_1, \ldots, a_k\}) = \text{Span}(\{q_0, q_1, \ldots, q_k\})$.

This gives us $k+1$ orthonormal vectors, $q_0, q_1, \ldots, q_k$.

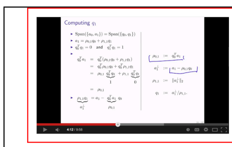
- Continue this process to compute $q_0, q_1, \ldots, q_{n-1}$.

The following result is the whole point of the Gram-Schmidt process, namely to find an orthonormal basis for the span of a given set of linearly independent vectors.

Theorem 11.2 *Let $a_0, a_1, \ldots, a_{k-1} \in \mathbb{R}^m$ be linearly independent vectors and let $q_0, q_1, \ldots, q_{k-1} \in \mathbb{R}^m$ be the result of Gram-Schmidt orthogonalization. Then Span$(\{a_0, a_1, \ldots, a_{k-1}\}) =$ Span$(\{q_0, q_1, \ldots, q_{k-1}\})$.*

The proof is a bit tricky (and in some sense stated in the material in this unit) so we do not give it here.

11.3.4 Orthogonal Bases (Alternative Explanation)



☛ **YouTube**

We now give an alternate explanation for Gram-Schmidt orthogonalization.

We are given linearly independent vectors $a_0, a_1, \ldots, a_{n-1} \in \mathbb{R}^m$ and would like to compute orthonormal vectors $q_0, q_1, \ldots, q_{n-1} \in \mathbb{R}^m$ such that $\text{Span}(\{a_0, a_1, \ldots, a_{n-1}\})$ equals $\text{Span}(\{q_0, q_1, \ldots, q_{n-1}\})$.

Let's put one more condition on the vectors q_k: $\text{Span}(\{a_0, a_1, \ldots, a_{k-1}\}) = \text{Span}(\{q_0, q_1, \ldots, q_{k-1}\})$ for $k = 0, 1, \ldots, n$. In other words,

$$\begin{array}{rcl} \text{Span}(\{a_0\}) & = & \text{Span}(\{q_0\}) \\ \text{Span}(\{a_0, a_1\}) & = & \text{Span}(\{q_0, q_1\}) \\ & \vdots & \\ \text{Span}(\{a_0, a_1, \ldots, a_{k-1}\}) & = & \text{Span}(\{q_0, q_1, \ldots, q_{k-1}\}) \\ & \vdots & \\ \text{Span}(\{a_0, a_1, \ldots, a_{n-1}\}) & = & \text{Span}(\{q_0, q_1, \ldots, q_{n-1}\}) \end{array}$$

Computing q_0

Now, $\text{Span}(\{a_0\}) = \text{Span}(\{q_0\})$ means that $a_0 = \rho_{0,0} q_0$ for some scalar $\rho_{0,0}$. Since q_0 has to be of length one, we can choose

$$\begin{array}{rcl} \rho_{0,0} & := & \|a_0\|_2 \\ q_0 & := & a_0 / \rho_{0,0}. \end{array}$$

Notice that q_0 is not unique: we could have chosen $\rho_{0,0} = -\|a_0\|_2$ and $q_0 = a_0/\rho_{0,0}$. This non-uniqueness is recurring in the below discussion, and we will ignore it since we are merely interested in *a single* orthonormal basis.

Computing q_1

Next, we note that $\text{Span}(\{a_0, a_1\}) = \text{Span}(\{q_0, q_1\})$ means that $a_1 = \rho_{0,1} q_0 + \rho_{1,1} q_1$ for some scalars $\rho_{0,1}$ and $\rho_{1,1}$. We also know that $q_0^T q_1 = 0$ and $q_1^T q_1 = 1$ since these vectors are orthonormal. Now

$$q_0^T a_1 = q_0^T(\rho_{0,1} q_0 + \rho_{1,1} q_1) = q_0^T \rho_{0,1} q_0 + q_0^T \rho_{1,1} q_1 = \rho_{0,1} \underbrace{q_0^T q_0}_{1} + \rho_{1,1} \underbrace{q_0^T q_1}_{0} = \rho_{0,1}$$

so that

$$\rho_{0,1} = q_0^T a_1.$$

Once $\rho_{0,1}$ has been computed, we can compute the component of a_1 orthogonal to q_0:

$$\underbrace{\rho_{1,1} q_1}_{a_1^\perp} = a_1 - \underbrace{q_0^T a_1}_{\rho_{0,1}} q_0$$

after which $a_1^\perp = \rho_{1,1} q_1$. Again, we can now compute $\rho_{1,1}$ as the length of $a_1^\perp$ and normalize to compute q_1:

$$\begin{array}{rcl} \rho_{0,1} & := & q_0^T a_1 \\ a_1^\perp & := & a_1 - \rho_{0,1} q_0 \\ \rho_{1,1} & := & \|a_1^\perp\|_2 \\ q_1 & := & a_1^\perp / \rho_{1,1}. \end{array}$$

Computing q_2

We note that $\text{Span}(\{a_0, a_1, a_2\}) = \text{Span}(\{q_0, q_1, q_2\})$ means that $a_2 = \rho_{0,2} q_0 + \rho_{1,2} q_1 + \rho_{2,2} q_2$ for some scalars $\rho_{0,2}$, $\rho_{1,2}$ and $\rho_{2,2}$. We also know that $q_0^T q_2 = 0$, $q_1^T q_2 = 0$ and $q_2^T q_2 = 1$ since these vectors are orthonormal. Now

- $$q_0^T a_2 = q_0^T(\rho_{0,2} q_0 + \rho_{1,2} q_1 + \rho_{2,2} q_2) = \rho_{0,2} \underbrace{q_0^T q_0}_{1} + \rho_{1,2} \underbrace{q_0^T q_1}_{0} + \rho_{2,2} \underbrace{q_0^T q_2}_{0} = \rho_{0,2}$$

  so that

  $$\rho_{0,2} = q_0^T a_2.$$

- $$q_1^T a_2 = q_1^T(\rho_{0,2} q_0 + \rho_{1,2} q_1 + \rho_{2,2} q_2) = \rho_{0,2} \underbrace{q_1^T q_0}_{0} + \rho_{1,2} \underbrace{q_1^T q_1}_{1} + \rho_{2,2} \underbrace{q_1^T q_2}_{0} = \rho_{1,2}$$

  so that

  $$\rho_{1,2} = q_1^T a_2.$$

Once $\rho_{0,2}$ and $\rho_{1,2}$ have been computed, we can compute the component of a_2 orthogonal to q_0 and q_1:

$$\underbrace{\rho_{2,2} q_2}_{a_2^\perp} = a_2 - \underbrace{q_0^T a_2}_{\rho_{0,2}} q_0 - \underbrace{q_1^T a_2}_{\rho_{1,2}} q_1$$

after which $a_2^\perp = \rho_{2,2} q_2$. Again, we can now compute $\rho_{2,2}$ as the length of $a_2^\perp$ and normalize to compute q_2:

$$\begin{array}{rcl} \rho_{0,2} & := & q_0^T a_2 \\ \rho_{1,2} & := & q_1^T a_2 \\ a_2^\perp & := & a_2 - \rho_{0,2} q_0 - \rho_{1,2} q_1 \\ \rho_{2,2} & := & \|a_2^\perp\|_2 \\ q_2 & := & a_2^\perp / \rho_{2,2}. \end{array}$$

Computing q_k

Let's generalize this: $\mathrm{Span}(\{a_0, a_1, \ldots, a_k\}) = \mathrm{Span}(\{q_0, q_1, \ldots, q_k\})$ means that

$$a_k = \rho_{0,k} q_0 + \rho_{1,k} q_1 + \cdots + \rho_{k-1,k} q_{k-1} + \rho_{k,k} q_k = \sum_{j=0}^{k-1} \rho_{j,k} q_j + \rho_{k,k} q_k$$

for some scalars $\rho_{0,k}, \rho_{1,k}, \ldots, \rho_{k,k}$. We also know that

$$q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

Now, if $p < k$,

$$q_p^T a_k = q_p^T \left(\sum_{j=0}^{k-1} \rho_{j,k} q_j + \rho_{k,k} q_k \right) = \sum_{j=0}^{k-1} \rho_{j,k} q_p^T q_j + \rho_{k,k} q_p^T q_k = \rho_{p,k} q_p^T q_p = \rho_{p,k}$$

so that

$$\rho_{p,k} = q_p^T a_k.$$

Once the scalars $\rho_{p,k}$ have been computed, we can compute the component of a_k orthogonal to $q_0, \ldots, q_{k-1}$:

$$\underbrace{\rho_{k,k} q_k}_{a_k^\perp} = a_k - \sum_{j=0}^{k-1} \underbrace{q_j^T a_k}_{\rho_{j,k}} q_j$$

after which $a_k^\perp = \rho_{k,k} q_k$. Once again, we can now compute $\rho_{k,k}$ as the length of $a_k^\perp$ and normalize to compute q_k:

$$\begin{array}{rcl} \rho_{0,k} & := & q_0^T a_k \\ & \vdots & \\ \rho_{k-1,k} & := & q_{k-1}^T a_k \\ a_k^\perp & := & a_k - \sum_{j=0}^{k-1} \rho_{j,k} q_j \\ \rho_{k,k} & := & \|a_k^\perp\|_2 \\ q_k & := & a_k^\perp / \rho_{k,k}. \end{array}$$

An algorithm

The above discussion yields an algorithm for Gram-Schmidt orthogonalization, computing $q_0, \ldots, q_{n-1}$ (and all the $\rho_{i,j}$'s as a side product). This is not a FLAME algorithm so it may take longer to comprehend:

for $k = 0, \ldots, n-1$

$$\left.\begin{array}{l} \textbf{for } p = 0, \ldots, k-1 \\ \quad \rho_{p,k} := q_p^T a_k \\ \textbf{endfor} \end{array}\right\} \left(\begin{array}{c} \rho_{0,k} \\ \hline \rho_{1,k} \\ \hline \vdots \\ \hline \rho_{k-1,k} \end{array}\right) = \left(\begin{array}{c} q_0^T a_k \\ \hline q_1^T a_k \\ \hline \vdots \\ \hline q_{k-1}^T a_k \end{array}\right) = \left(\begin{array}{c} q_0^T \\ \hline q_1^T \\ \hline \vdots \\ \hline q_{k-1}^T \end{array}\right) a_k = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array}\right)^T a_k$$

$$\left.\begin{array}{l} a_k^\perp := a_k \\ \textbf{for } j = 0, \ldots, k-1 \\ \quad a_k^\perp := a_k^\perp - \rho_{j,k} q_j \\ \textbf{endfor} \end{array}\right\} a_k^\perp = a_k - \sum_{j=0}^{k-1} \rho_{j,k} q_j = a_k - \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array}\right) \left(\begin{array}{c} \rho_{0,k} \\ \hline \rho_{1,k} \\ \hline \vdots \\ \hline \rho_{k-1,k} \end{array}\right)$$

$$\left.\begin{array}{l} \rho_{k,k} := \|a_k^\perp\|_2 \\ q_k := a_k^\perp / \rho_{k,k} \end{array}\right\} \text{Normalize } a_k^\perp \text{ to be of length one.}$$

endfor

Homework 11.3.4.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ Compute an orthonormal basis for $\mathcal{C}(A)$.

☛ SEE ANSWER

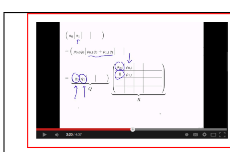
Homework 11.3.4.2 Consider $A = \begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 2 \end{pmatrix}$. Compute an orthonormal basis for $\mathcal{C}(A)$.

☛ SEE ANSWER

Homework 11.3.4.3 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$. Compute an orthonormal basis for $\mathcal{C}(A)$.

☛ SEE ANSWER

11.3.5 The QR Factorization



☛ YouTube

Given linearly independent vectors $a_0, a_1, \ldots, a_{n-1} \in \mathbb{R}^m$, the last unit computed the orthonormal basis $q_0, q_1, \ldots, q_{n-1}$ such that $\text{Span}(\{a_1, a_2, \ldots, a_{n-1}\})$ equals $\text{Span}(\{q_1, q_2, \ldots, q_{n-1}\})$. As a side product, the scalars $\rho_{i,j} = q_i^T a_j$ were computed, for $i \leq j$. We now show that in the process we computed what's known as the **QR factorization** of the matrix

$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$:

$$\underbrace{\left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)}_{A} = \underbrace{\left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right)}_{Q} \underbrace{\left(\begin{array}{c|c|c|c} \rho_{0,0} & \rho_{0,1} & \cdots & \rho_{0,n-1} \\ \hline 0 & \rho_{1,1} & \cdots & \rho_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \rho_{n-1,n-1} \end{array} \right)}_{R}.$$

Notice that $Q^T Q = I$ (since its columns are orthonormal) and R is upper triangular.

In the last unit, we noticed that

$$\begin{array}{ccccccccc} a_0 & = & \rho_{0,0} q_0 & & & & & & \\ a_1 & = & \rho_{0,1} q_0 & + & \rho_{1,1} q_1 & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & \\ a_{n-1} & = & \rho_{0,n-1} q_0 & + & \rho_{1,n-1} q_1 & + & \cdots & + & \rho_{n-1,n-1} q_{n-1} \end{array}$$

If we write the vectors on the left of the equal signs as the columns of a matrix, and do the same for the vectors on the right of the equal signs, we get

$$\begin{aligned} \underbrace{\left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)}_{A} &= \left(\begin{array}{c|c|c|c} \rho_{0,0} q_0 & \rho_{0,1} q_0 + \rho_{1,1} q_1 & \cdots & \rho_{0,n-1} q_0 + \rho_{1,n-1} q_1 + \cdots + \rho_{n-1,n-1} q_{n-1} \end{array} \right) \\ &= \underbrace{\left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right)}_{Q} \underbrace{\left(\begin{array}{c|c|c|c} \rho_{0,0} & \rho_{0,1} & \cdots & \rho_{0,n-1} \\ \hline 0 & \rho_{1,1} & \cdots & \rho_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \rho_{n-1,n-1} \end{array} \right)}_{R}. \end{aligned}$$

Bingo, we have shown how Gram-Schmidt orthogonalization computes the QR factorization of a matrix A.

Homework 11.3.5.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$.

- Compute the QR factorization of this matrix.
  (Hint: Look at Homework 11.3.4.1)
- Check that $QR = A$.

☛ SEE ANSWER

Homework 11.3.5.2 Considerx !m

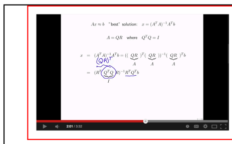
$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$. Compute the QR factorization of this matrix.

(Hint: Look at Homework 11.3.4.3)

Check that $A = QR$.

☛ SEE ANSWER

11.3.6 Solving the Linear Least-Squares Problem via QR Factorization



☛ YouTube

Now, let's look at how to use the QR factorization to solve $Ax \approx b$ when b is not in the column space of A but A has linearly independent columns. We know that the linear least-squares solution is given by

$$x = (A^T A)^{-1} A^T b.$$

Now $A = QR$ where $Q^T Q = I$. Then

$$\begin{aligned} x &= (A^T A)^{-1} A^T b = ((\underbrace{QR}_{A})^T (\underbrace{QR}_{A}))^{-1} (\underbrace{QR}_{A})^T b \\ &= (R^T \underbrace{Q^T Q}_{I} R)^{-1} R^T Q^T b = (R^T R)^{-1} R^T Q^T b = R^{-1} \underbrace{R^{-T} R^T}_{I} Q^T b \\ &= R^{-1} Q^T b. \end{aligned}$$

Thus, the linear least-square solution, x, for $Ax \approx b$ when A has linearly independent columns solves $Rx = Q^T b$.

Homework 11.3.6.1 In Homework 11.3.4.1 you were asked to consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ and compute an orthonormal basis for $\mathcal{C}(A)$.

In Homework 11.3.5.1 you were then asked to compute the QR factorization of that matrix. Of course, you could/should have used the results from Homework 11.3.4.1 to save yourself calculations. The result was the following factorization $A = QR$:

$$\left(\begin{array}{c|c} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{array}\right) = \left(\begin{array}{c|c} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} & \frac{\sqrt{2}}{\sqrt{3}} \begin{pmatrix} -\frac{1}{2} \\ 1 \\ \frac{1}{2} \end{pmatrix} \end{array}\right) \left(\begin{array}{c|c} \sqrt{2} & \frac{1}{\sqrt{2}} \\ \hline 0 & \frac{\sqrt{6}}{2} \end{array}\right)$$

Now, compute the "best" solution (in the linear least-squares sense), $\hat{x}$, to

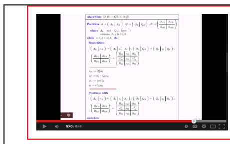
$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}.$$

(This is the same problem as in Homework 10.4.2.1.)

- $u = Q^T b =$
- The solution to $R\hat{x} = u$ is $\hat{x} =$

☛ SEE ANSWER

11.3.7 The QR Factorization (Again)



☛ YouTube

We now give an explanation of how to compute the QR factorization that yields an algorithm in FLAME notation.

We wish to compute $A = QR$ where $A, Q \in \mathbb{R}^{m\times n}$ and $R \in \mathbb{R}^{n\times n}$. Here $Q^T Q = I$ and R is upper triangular. Let's partition these matrices:

$$A = \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array} \right), \quad Q = \left(\begin{array}{c|c|c} Q_0 & q_1 & Q_2 \end{array} \right), \quad \text{and} \quad \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline 0 & \rho_{11} & r_{12}^T \\ \hline 0 & 0 & R_{22} \end{array} \right),$$

where $A_0, Q_0 \in \mathbb{R}^{m\times k}$ and $R_{00} \in \mathbb{R}^{k\times k}$. Now, $A = QR$ means that

$$\left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array} \right) = \left(\begin{array}{c|c|c} Q_0 & q_1 & Q_2 \end{array} \right) \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline 0 & \rho_{11} & r_{12}^T \\ \hline 0 & 0 & R_{22} \end{array} \right)$$

so that

$$\left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array} \right) = \left(\begin{array}{c|c|c} Q_0 R_{00} & Q_0 r_{01} + \rho_{11} q_1 & Q_0 R_{02} + q_1 r_{12}^T + Q_2 R_{22} \end{array} \right).$$

Now, assume that Q_0 and R_{00} have already been computed so that $A_0 = Q_0 R_{00}$. Let's focus on how to compute the next column of Q, q_1, and the next column of R, $\left(\begin{array}{c} r_{01} \\ \hline \rho_{11} \end{array} \right)$:

$$a_1 = Q_0 r_{01} + \rho_{11} q_1$$

implies that

$$Q_0^T a_1 = Q_0^T (Q_0 r_{01} + \rho_{11} q_1) = \underbrace{Q_0^T Q_0}_{I} r_{01} + \rho_{11} \underbrace{Q_0^T q_1}_{0} = r_{01},$$

since $Q_0^T Q_0 = I$ (the columns of Q_0 are orthonormal) and $Q_0^T q_1 = 0$ (q_1 is orthogonal to all the columns of Q_0). So, we can compute r_{01} as

$$r_{01} := Q_0^T a_1.$$

Now we can compute $a_1^\perp$, the component of a_1 orthogonal to the columns of Q_0:

$$\begin{array}{rcl} a_1^\perp & := & a_1 - Q_0 r_{01} \\ & = & a_1 - Q_0 Q_0^T a_1 \\ & = & (I - Q_0 Q_0^T) a_1, \text{ the component of } a_1 \text{ orthogonal to } \mathcal{C}(Q_0). \end{array}$$

Rearranging $a_1 = Q_0 r_{01} + \rho_{11} q_1$ yields $\rho_{11} q_1 = a_1 - Q_0 r_{01} = a_1^\perp$. Now, q_1 is simply the vector of length *one* in the direction of $a_1^\perp$. Hence we can choose

$$\begin{array}{rcl} \rho_{11} & := & \|a_1^\perp\|_2 \\ q_1 & := & a_1^\perp / \rho_{11}. \end{array}$$

All of these observations are summarized in the algorithm in Figure 11.1

Algorithm: $[Q,R] := \mathrm{QR}(A,Q,R)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$, $Q \rightarrow \left(Q_L \,\middle|\, Q_R \right)$, $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right)$

where A_L and Q_L have 0 columns, R_{TL} is 0×0

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right), \left(Q_L \,\middle|\, Q_R \right) \rightarrow \left(Q_0 \,\middle|\, q_1 \,\middle|\, Q_2 \right),$$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array} \right)$$

$r_{01} := Q_0^T a_1$

$a_1^\perp := a_1 - Q_0 r_{01}$

$\rho_{11} := \|a_1^\perp\|_2$

$q_1 = a_1^\perp / \rho_{11}$

Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,\middle|\, a_1 \,\middle|\, A_2 \right), \left(Q_L \,\middle|\, Q_R \right) \leftarrow \left(Q_0 \,\middle|\, q_1 \,\middle|\, Q_2 \right),$$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array} \right)$$

endwhile

Figure 11.1: QR facorization via Gram-Schmidt orthogonalization.

Homework 11.3.7.1 Implement the algorithm for computing the QR factorization of a matrix in Figure 11.1

```
[ Q_out, R_out ] = QR_unb( A, Q, R )
```

where `A` and `Q` are $m \times n$ matrices and `R` is an $n \times n$ matrix. You will want to use the routines `laff_gemv`, `laff_norm`, and `laff_invscal`. (Alternatively, use native MATLAB operations.) Store the routine in

```
LAFF-2.0xM -> Programming -> Week11 -> QR_unb.m
```

Test the routine with

```
A = [  1 -1  2
       2  1 -3
      -1  3  2
       0 -2 -1 ];

Q = zeros( 4, 3 );
R = zeros( 3, 3 );
[ Q_out, R_out ] = QR_unb( A, Q, R );
```

Next, see if $A = QR$:

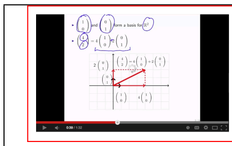
```
A - Q_out * R_out
```

This should equal, approximately, the zero matrix. Check if Q has mutually orthogonal columns:

```
Q_out' * Q_out
```

11.4 Change of Basis

11.4.1 The Unit Basis Vectors, One More Time



☛ YouTube

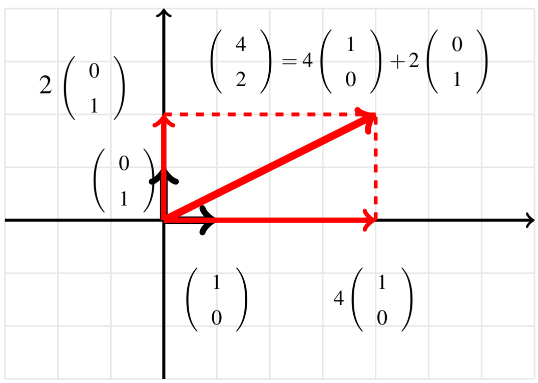
Once again, recall the unit basis vectors in $\mathbb{R}^2$:

$$e_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad e_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

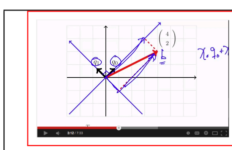
Now,

$$\begin{pmatrix} 4 \\ 2 \end{pmatrix} = 4 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

by which we illustrate the fact that $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ form a basis for $\mathbb{R}^2$ and the vector $\begin{pmatrix} 4 \\ 2 \end{pmatrix}$ can then be written as a linear combination of these basis vectors, with coefficients 4 and 2. We can illustrate this with



11.4.2 Change of Basis



☛ YouTube

Similar to the example from the last unit, we could have created an alternate coordinate system with basis vectors

$$q_0 = \frac{\sqrt{2}}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad q_1 = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.$$

What are the coefficients for the linear combination of these two vectors (q_0 and q_1) that produce the vector $\begin{pmatrix} 4 \\ 2 \end{pmatrix}$? First let's look at a few exercises demonstrating how special these vectors that we've chosen are.

Homework 11.4.2.1 The vectors

$$q_0 = \frac{\sqrt{2}}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad q_1 = \frac{\sqrt{2}}{2}\begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.$$

are mutually orthonormal.

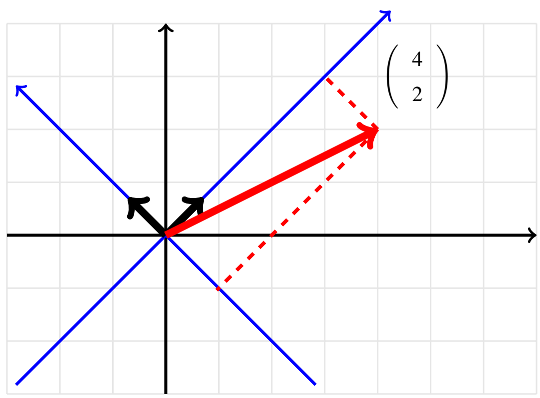
True/False

☛ SEE ANSWER

Homework 11.4.2.2 If $Q \in \mathbb{R}^{n \times n}$ has mutually orthonormal columns then which of the following are true:

1. $Q^T Q = I$ True/False
2. $QQ^T = I$ True/False
3. $QQ^{-1} = I$ True/False
4. $Q^{-1} = Q^T$ True/False

☛ SEE ANSWER



What we would like to determine are the coefficients χ_0 and χ_1 such that

$$\chi_0 \frac{\sqrt{2}}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix} + \chi_1 \frac{\sqrt{2}}{2}\begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}.$$

This can be alternatively written as

$$\underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

In Homework 11.4.2.1 we noticed that

$$\underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

and hence

$$\underbrace{\underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q}}_{I} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

or, equivalently,

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \begin{pmatrix} 4 \\ 2 \end{pmatrix} = \begin{pmatrix} 4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \\ -4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} 3\sqrt{2} \\ -\sqrt{2} \end{pmatrix}$$

so that

$$3\sqrt{2}\begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix} - \sqrt{2}\begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}.$$

In other words: In the new basis, the coefficients are $3\sqrt{2}$ and $-\sqrt{2}$.

Another way of thinking of the above discussion is that

$$\begin{aligned}
4\begin{pmatrix} 1 \\ 0 \end{pmatrix} + 2\begin{pmatrix} 0 \\ 1 \end{pmatrix} &= \begin{pmatrix} 4 \\ 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 4 \\ 2 \end{pmatrix} \\
&= \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q} \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \begin{pmatrix} 4 \\ 2 \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q} \begin{pmatrix} 4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \\ -4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \end{pmatrix} \\
&= \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q} \begin{pmatrix} 3\sqrt{2} \\ -\sqrt{2} \end{pmatrix} = 3\sqrt{2}\begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix} - \sqrt{2}\begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.
\end{aligned}$$

This last way of looking at the problem suggest a way of finding the coefficients for any basis, $a_0, a_1, \ldots, a_{n-1} \in \mathbb{R}^n$. Let $b \in \mathbb{R}^n$ and let $A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$. Then

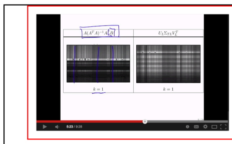
$$b = \underbrace{AA^{-1}}_{I} b = Ax = \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}.$$

So, when the basis is changed from the unit basis vectors to the vectors $a_0, a_1, \ldots, a_{n-1}$, the coefficients change from $\beta_0, \beta_1, \ldots, \beta_{n-1}$ (the components of the vector b) to $\chi_0, \chi_1, \ldots, \chi_{n-1}$ (the components of the vector x).

Obviously, instead of computing $A^{-1}b$, one can instead solve $Ax = b$.

11.5 Singular Value Decomposition

11.5.1 The Best Low Rank Approximation



☛ YouTube

Earlier this week, we showed that by taking a few columns from matrix B (which encoded the picture), and projecting onto those columns we could create a rank-k approximation, AW^T, that approximated the picture. The columns in A were chosen from the columns of B.

Now, what if we could choose the columns of A to be the *best* colums onto which to project? In other words, what if we could choose the columns of A so that the subspace spanned by them minimized the error in the approximation AW^T when we choose $W = (A^TA)^{-1}A^TB$?

The answer to how to obtain the answers the above questions go beyond the scope of an introductory undergraduate linear algebra course. But let us at least look at some of the results.

One of the most important results in linear algebra is the **Singular Value Decomposition Theorem** which says that any matrix $B \in \mathbb{R}^{m\times n}$ can be written as the product of three matrices, the Singular Value Decomposition (SVD):

$$B = U\Sigma V^T$$

where

- $U \in \mathbb{R}^{m\times r}$ and $U^TU = I$ (U has orthonormal columns).
- $\Sigma \in \mathbb{R}^{r\times r}$ is a diagonal matrix with positive diagonal elements that are ordered so that $\sigma_{0,0} \geq \sigma_{1,1} \geq \cdots \geq \sigma_{(r-1),(r-1)} > 0$.
- $V \in \mathbb{R}^{n\times r}$ and $V^TV = I$ (V has orthonormal columns).
- r equals the rank of matrix B.

If we partition

$$U = \left(\begin{array}{c|c} U_L & U_R \end{array} \right), V = \left(\begin{array}{c|c} V_L & V_R \end{array} \right), \text{ and } \Sigma = \left(\begin{array}{c|c} \Sigma_{TL} & 0 \\ \hline 0 & \Sigma_{BR} \end{array} \right),$$

where U_L and V_L have k columns and Σ_{TL} is $k \times k$, then $U_L\Sigma_{TL}V_L^T$ is the "best" rank-k approximation to matrix B. So, the "best" rank-k approximation $B = AW^T$ is given by the choices $A = U_L$ and $W^T = \Sigma_{TL}V_L^T$.

The sequence of pictures in Figures 11.2 and 11.3 illustrate the benefits of using a rank-k update based on the SVD.

Homework 11.5.1.1 Let $B = U\Sigma V^T$ be the SVD of B, with $U \in \mathbb{R}^{m\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$, and $V \in \mathbb{R}^{n\times r}$. Partition

$$U = \left(\begin{array}{c|c|c|c} u_0 & u_1 & \cdots & u_{r-1} \end{array} \right), \quad \Sigma = \left(\begin{array}{c|c|c|c} \sigma_0 & 0 & \cdots & 0 \\ \hline 0 & \sigma_1 & \cdots & 0 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \sigma_{r-1} \end{array} \right), V = \left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_{r-1} \end{array} \right).$$

$U\Sigma V^T = \sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \cdots + \sigma_{r-1} u_{r-1} v_{r-1}^T$.

Always/Sometimes/Never

☛ SEE ANSWER

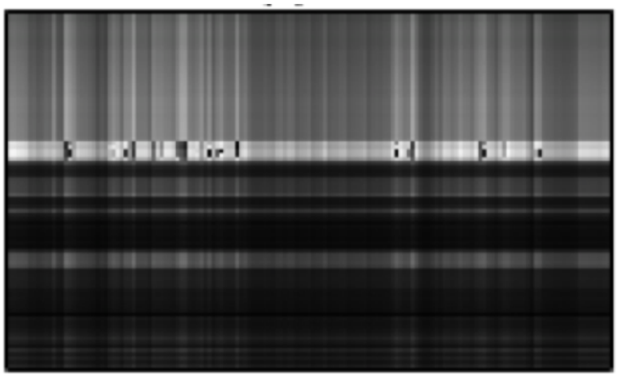
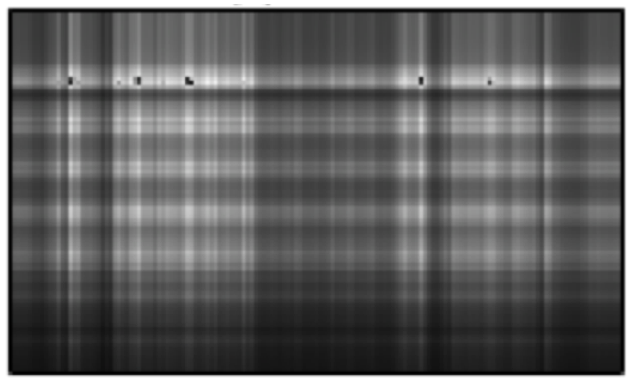
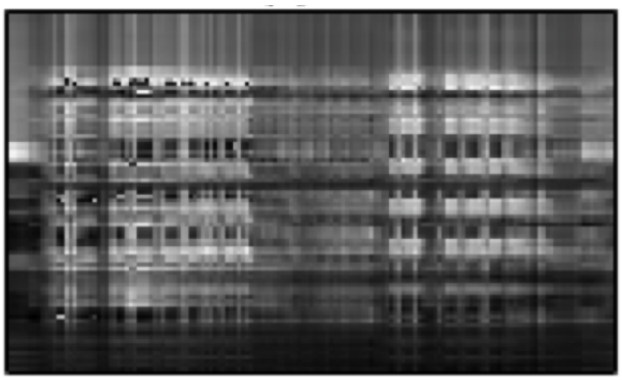
$A(A^TA)^{-1}A^TB$	$U_L\Sigma_{TL}V_L^T$
 $k = 1$	 $k = 1$
 $k = 5$	 $k = 5$
 $k = 10$	 $k = 10$

Figure 11.2: Rank-k approximation using columns from the picture versus using the SVD. (Part 1)

$A(A^TA)^{-1}A^TB$	$U_L\Sigma_{TL}V_L^T$
 $k = 25$	 $k = 25$
 $k = 50$	 $k = 50$

Figure 11.3: Rank-k approximation using columns from the picture versus using the SVD. (Continued)

Homework 11.5.1.2 Let $B = U\Sigma V^T$ be the SVD of B with $U \in \mathbb{R}^{m\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$, and $V \in \mathbb{R}^{n\times r}$.

- $\mathcal{C}(B) = \mathcal{C}(U)$

  Always/Sometimes/Never

- $\mathcal{R}(B) = \mathcal{C}(V)$

  Always/Sometimes/Never

☛ SEE ANSWER

Given $A \in \mathbb{R}^{m\times n}$ with linearly independent columns, and $b \in \mathbb{R}^m$, we can solve $Ax \approx b$ for the "best" solution (in the linear least-squares sense) via its SVD, $A = U\Sigma V^T$, by observing that

$$\begin{aligned}
\hat{x} &= (A^T A)^{-1} A^T b \\
&= ((U\Sigma V^T)^T (U\Sigma V^T))^{-1} (U\Sigma V^T)^T b \\
&= (V\Sigma^T U^T U\Sigma V^T)^{-1} V\Sigma^T U^T b \\
&= (V\Sigma\Sigma V^T)^{-1} V\Sigma U^T b \\
&= ((V^T)^{-1}(\Sigma\Sigma)^{-1} V^{-1}) V\Sigma U^T b \\
&= V\Sigma^{-1}\Sigma^{-1}\Sigma U^T b \\
&= V\Sigma^{-1} U^T b.
\end{aligned}$$

Hence, the "best" solution is given by

$$\hat{x} = V\Sigma^{-1} U^T b.$$

Homework 11.5.1.3 You will now want to revisit exercise 11.2.5.2 and compare an approximation by projecting onto a few columns of the picture versus using the SVD to approximate. You can do so by executing the script `Week11/CompressPictureWithSVD.m` that you downloaded in `Week11.zip`. That script creates three figures: the first is the original picture. The second is the approximation as we discussed in Section 11.2.5. The third uses the SVD. Play with the script, changing variable `k`.

☛ SEE ANSWER

11.6 Enrichment

11.6.1 The Problem with Computing the QR Factorization

Modified Gram-Schmidt

In theory, the Gram-Schmidt process, started with a set of linearly independent vectors, yields an orthonormal basis for the span of those vectors. In practice, due to round-off error, the process can result in a set of vectors that are far from mutually orhonormal. A minor modification of the Gram-Schmidt process, known as Modified Gram-Schmidt, partially fixes this.

A more advanced treatment of Gram-Schmidt orthonalization, including the Modified Gram-Schmidt process, can be found in Robert's notes for his graduate class on Numerical Linear Algebra, available from `http://www.ulaff.net`.

Many linear algebra texts also treat this material.

11.6.2 QR Factorization Via Householder Transformations (Reflections)

If orthogonality is important, an alternative algorithm for computing the QR factorization is employed, based on Householder transformations (reflections). This approach resembles LU factorization with Gauss transforms, except that at each step a reflection is used to zero elements below the current diagonal.

QR factorization via Householder transformations is discussed in Robert's notes for his graduate class on Numerical Linear Algebra, available from `http://www.ulaff.net`.

Graduate level texts on numerical linear algebra usually treat this topic, as may some more advanced undergraduate texts.

11.6.3 More on SVD

The SVD is possibly the most important topic in linear algebra. Graduate level texts on numerical linear algebra usually treat this topic, as may some more advanced undergraduate texts.

A thorough treatment of the SVD can be found early in our MOOC titled "Advanced Linear Algebra: Foundations to Frontiers", which covers the content of a typical graduate class on Numerical Linear Algebra. For details, visit `http://www.ulaff.net`.

11.7 Wrap Up

11.7.1 Homework

No additional homework this week.

11.7.2 Summary

Projection

Given $a, b \in \mathbb{R}^m$:

- Component of b in direction of a:
$$u = \frac{a^T b}{a^T a} a = a(a^T a)^{-1} a^T b.$$
- Matrix that projects onto $\text{Span}(\{a\})$:
$$a(a^T a)^{-1} a^T$$
- Component of b orthogonal to a:
$$w = b - \frac{a^T b}{a^T a} a = b - a(a^T a)^{-1} a^T b = (I - a(a^T a)^{-1} a^T) b.$$
- Matrix that projects onto $\text{Span}(\{a\})^\perp$:
$$I - a(a^T a)^{-1} a^T$$

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns and vector $b \in \mathbb{R}^m$:

- Component of b in $\mathcal{C}(A)$:
$$u = A(A^T A)^{-1} A^T b.$$
- Matrix that projects onto $\mathcal{C}(A)$:
$$A(A^T A)^{-1} A^T.$$
- Component of b in $\mathcal{C}(A)^\perp = \mathcal{N}(A^T)$:
$$w = b - A(A^T A)^{-1} A^T b = (I - A(A^T A)^{-1} A^T) b.$$
- Matrix that projects onto $\mathcal{C}(A)^\perp = \mathcal{N}(A^T)$:
$$(I - A(A^T A)^{-1} A^T).$$

"Best" rank-k approximation of $B \in \mathbb{R}^{m \times n}$ using the column space of $A \in \mathbb{R}^{m \times k}$ with linearly independent columns:

$$A(A^T A)^{-1} A^T B = AV^T, \quad \text{where } V^T = (A^T A)^{-1} A^T B.$$

Orthonormal vectors and spaces

Definition 11.3 *Let $q_0, q_1, \ldots, q_{k-1} \in \mathbb{R}^m$. Then these vectors are (mutually) orthonormal if for all $0 \leq i, j < k$:*

$$q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 11.4 *A matrix $Q \in \mathbb{R}^{m \times n}$ has mutually orthonormal columns if and only if $Q^T Q = I$.*

Given $q, b \in \mathbb{R}^m$, with $\|q\|_2 = 1$ (q of length one):

- Component of b in direction of q:
$$u = q^T b q = q q^T b.$$

- Matrix that projects onto Span($\{q\}$):

$$qq^T$$

- Component of b orthogonal to q:

$$w = b - q^T bq = (I - qq^T)b.$$

- Matrix that projects onto Span($\{q\}$)$^\perp$:

$$I - qq^T$$

Given matrix $Q \in \mathbb{R}^{m\times n}$ with mutually orthonormal columns and vector $b \in \mathbb{R}^m$:

- Component of b in $\mathcal{C}(Q)$:

$$u = QQ^T b.$$

- Matrix that projects onto $\mathcal{C}(Q)$:

$$QQ^T.$$

- Component of b in $\mathcal{C}(Q)^\perp = \mathcal{N}(Q^T)$:

$$w = b - QQ^T b = (I - QQ^T)b.$$

- Matrix that projects onto $\mathcal{C}(Q)^\perp = \mathcal{N}(Q^T)$:

$$(I - QQ^T).$$

"Best" rank-k approximation of $B \in \mathbb{R}^{m\times n}$ using the column space of $Q \in \mathbb{R}^{m\times k}$ with mutually orthonormal columns:

$$QQ^T B = QV^T, \quad \text{where } V^T = Q^T B.$$

Gram-Schmidt orthogonalization

Starting with linearly independent vectors $a_0, a_1, \ldots, a_{n-1} \in \mathbb{R}^m$, the following algorithm computes the mutually orthonormal vectors $q_0, q_1, \ldots, q_{n-1} \in \mathbb{R}^m$ such that Span($\{a_0, a_1, \ldots, a_{n-1}\}$) = Span($\{q_0, q_1, \ldots, q_{n-1}\}$):

for $k = 0, \ldots, n-1$

  for $p = 0, \ldots, k-1$
    $\rho_{p,k} := q_p^T a_k$
  endfor

$$\left.\right\} \begin{pmatrix} \rho_{0,k} \\ \hline \rho_{1,k} \\ \hline \vdots \\ \hline \rho_{k-1,k} \end{pmatrix} = \begin{pmatrix} q_0^T a_k \\ \hline q_1^T a_k \\ \hline \vdots \\ \hline q_{k-1}^T a_k \end{pmatrix} = \begin{pmatrix} q_0^T \\ \hline q_1^T \\ \hline \vdots \\ \hline q_{k-1}^T \end{pmatrix} a_k = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)^T a_k$$

  $a_k^\perp := a_k$
  for $j = 0, \ldots, k-1$
    $a_k^\perp := a_k^\perp - \rho_{j,k} q_j$
  endfor

$$\left.\right\} a_k^\perp = a_k - \textstyle\sum_{j=0}^{k-1} \rho_{j,k} q_j = a_k - \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) \begin{pmatrix} \rho_{0,k} \\ \hline \rho_{1,k} \\ \hline \vdots \\ \hline \rho_{k-1,k} \end{pmatrix}$$

  $\rho_{k,k} := \|a_k^\perp\|_2$
  $q_k := a_k^\perp / \rho_{k,k}$

  } Normalize $a_k^\perp$ to be of length one.

endfor

The QR factorization

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns, there exists a matrix $Q \in \mathbb{R}^{m \times n}$ with mutually orthonormal columns and upper triangular matrix $R \in \mathbb{R}^{n \times n}$ such that $A = QR$.

If one partitions

$$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right), \quad Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right), \quad \text{and} \quad R = \left(\begin{array}{c|c|c|c} \rho_{0,0} & \rho_{0,1} & \cdots & \rho_{0,n-1} \\ \hline 0 & \rho_{1,1} & \cdots & \rho_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \rho_{n-1,n-1} \end{array} \right)$$

then

$$\underbrace{\left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)}_{A} = \underbrace{\left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right)}_{Q} \underbrace{\left(\begin{array}{c|c|c|c} \rho_{0,0} & \rho_{0,1} & \cdots & \rho_{0,n-1} \\ \hline 0 & \rho_{1,1} & \cdots & \rho_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \rho_{n-1,n-1} \end{array} \right)}_{R}$$

and Gram-Schmidt orthogonalization (the Gram-Schmidt process) in the above algorithm computes the columns of Q and elements of R.

Solving the linear least-squares problem via the QR factorization

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns, there exists a matrix $Q \in \mathbb{R}^{m \times n}$ with mutually orthonormal columns and upper triangular matrix $R \in \mathbb{R}^{n \times n}$ such that $A = QR$. The vector $\hat{x}$ that is the best solution (in the linear least-squares sense) to $Ax \approx b$ is given by

- $\hat{x} = (A^T A)^{-1} A^T b$ (as shown in Week 10) computed by solving the normal equations

$$A^T A x = A^T b.$$

- $\hat{x} = R^{-1} Q^T b$ computed by solving

$$Rx = Q^T b.$$

An algorithm for computing the QR factorization (presented in FLAME notation) is given by

Algorithm: $[Q,R] := \mathrm{QR}(A,Q,R)$

Partition $A \rightarrow \left(A_L \,\middle|\, A_R \right)$, $Q \rightarrow \left(Q_L \,\middle|\, Q_R \right)$, $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right)$

where A_L and Q_L have 0 columns, R_{TL} is 0×0

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(A_L \,\middle|\, A_R \right) \rightarrow \left(A_0 \,\middle|\, a_1 \,|\, A_2 \right), \left(Q_L \,\middle|\, Q_R \right) \rightarrow \left(Q_0 \,\middle|\, q_1 \,|\, Q_2 \right),$$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array} \right)$$

$r_{01} := Q_0^T a_1$

$a_1^\perp := a_1 - Q_0 r_{01}$

$\rho_{11} := \|a_1^\perp\|_2$

$q_1 = a_1^\perp / \rho_{11}$

Continue with

$$\left(A_L \,\middle|\, A_R \right) \leftarrow \left(A_0 \,|\, a_1 \,\middle|\, A_2 \right), \left(Q_L \,\middle|\, Q_R \right) \leftarrow \left(Q_0 \,|\, q_1 \,\middle|\, Q_2 \right),$$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array} \right)$$

endwhile

Singular Value Decomposition

Any matrix $B \in \mathbb{R}^{m \times n}$ can be written as the product of three matrices, the Singular Value Decomposition (SVD):

$$B = U \Sigma V^T$$

where

- $U \in \mathbb{R}^{m \times r}$ and $U^T U = I$ (U has orthonormal columns).
- $\Sigma \in \mathbb{R}^{r \times r}$ is a diagonal matrix with positive diagonal elements that are ordered so that $\sigma_{0,0} \geq \sigma_{1,1} \geq \cdots \geq \sigma_{(r-1),(r-1)} > 0$.
- $V \in \mathbb{R}^{n \times r}$ and $V^T V = I$ (V has orthonormal columns).
- r equals the rank of matrix B.

If we partition

$$U = \left(U_L \,\middle|\, U_R \right), V = \left(V_L \,\middle|\, V_R \right), \text{ and } \Sigma = \left(\begin{array}{c|c} \Sigma_{TL} & 0 \\ \hline 0 & \Sigma_{BR} \end{array} \right),$$

where U_L and V_L have k columns and Σ_{TL} is $k \times k$, then $U_L \Sigma_{TL} V_L^T$ is the "best" rank-k approximation to matrix B. So, the "best" rank-k approximation $B = AW^T$ is given by the choices $A = U_L$ and $W = \Sigma_{TL} V_L$.

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns, and $b \in \mathbb{R}^m$, the "best" solution to $Ax \approx b$ (in the linear least-squares sense) via its SVD, $A = U \Sigma V^T$, is given by

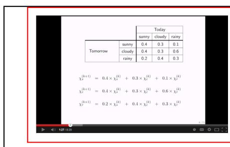
$$\hat{x} = V \Sigma^{-1} U^T b.$$

Week 12

Eigenvalues, Eigenvectors, and Diagonalization

12.1 Opening Remarks

12.1.1 Predicting the Weather, Again



☛ YouTube

Let us revisit the example from Week 4, in which we had a simple model for predicting the weather. Again, the following table tells us how the weather for any day (e.g., today) predicts the weather for the next day (e.g., tomorrow):

		Today		
		sunny	cloudy	rainy
Tomorrow	sunny	0.4	0.3	0.1
	cloudy	0.4	0.3	0.6
	rainy	0.2	0.4	0.3

This table is interpreted as follows: If today is rainy, then the probability that it will be cloudy tomorrow is 0.6, etc.

We introduced some notation:

- Let $\chi_s^{(k)}$ denote the probability that it will be sunny k days from now (on day k).
- Let $\chi_c^{(k)}$ denote the probability that it will be cloudy k days from now.
- Let $\chi_r^{(k)}$ denote the probability that it will be rainy k days from now.

We then saw that predicting the weather for day $k+1$ based on the prediction for day k was given by the system of linear equations

$$\begin{array}{rcrcrcr}
\chi_s^{(k+1)} & = & 0.4 \times \chi_s^{(k)} & + & 0.3 \times \chi_c^{(k)} & + & 0.1 \times \chi_r^{(k)} \\
\chi_c^{(k+1)} & = & 0.4 \times \chi_s^{(k)} & + & 0.3 \times \chi_c^{(k)} & + & 0.6 \times \chi_r^{(k)} \\
\chi_r^{(k+1)} & = & 0.2 \times \chi_s^{(k)} & + & 0.4 \times \chi_c^{(k)} & + & 0.3 \times \chi_r^{(k)}.
\end{array}$$

which could then be written in matrix form as

$$x^{(k)} = \begin{pmatrix} \chi_s^{(k)} \\ \chi_c^{(k)} \\ \chi_r^{(k)} \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix}.$$

so that

$$\begin{pmatrix} \chi_s^{(k+1)} \\ \chi_c^{(k+1)} \\ \chi_r^{(k+1)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(k)} \\ \chi_c^{(k)} \\ \chi_r^{(k)} \end{pmatrix}$$

or $x^{(k+1)} = Px^{(k)}$.

Now, if we start with day zero being cloudy, then the predictions for the first two weeks are given by

Day #	Sunny	Cloudy	Rainy
0	0.	1.	0.
1	0.3	0.3	0.4
2	0.25	0.45	0.3
3	0.265	0.415	0.32
4	0.2625	0.4225	0.315
5	0.26325	0.42075	0.316
6	0.263125	0.421125	0.31575
7	0.2631625	0.4210375	0.3158
8	0.26315625	0.42105625	0.3157875
9	0.26315813	0.42105188	0.31579
10	0.26315781	0.42105281	0.31578938
11	0.26315791	0.42105259	0.3157895
12	0.26315789	0.42105264	0.31578947
13	0.2631579	0.42105263	0.31578948
14	0.26315789	0.42105263	0.31578947

What you notice is that eventually

$$x^{(k+1)} \approx Px^{(k)}.$$

What this means is that there is a vector x such that $Px = x$. Such a vector (if it is non-zero) is known as an eigenvector. In this example, it represents the long-term prediction of the weather. Or, in other words, a description of "typical weather": approximately 26% of the time it is sunny, 42% of the time it is cloudy, and 32% of the time rainy.

The question now is: How can we compute such vectors?

Some observations:

- $Px = x$ means that $Px - x = 0$ which in turn means that $(P - I)x = 0$.
- This means that x is a vector in the null space of $P - I$: $x \in \mathcal{N}(P - I)$.
- But we know how to find vectors in the null space of a matrix. You reduce a system to row echelon form, identify the free variable(s), etc.
- But we also know that a nonzero vector in the null space is not unique.
- In this **particular** case, we know two more pieces of information:
  - The components of x must be nonnegative (a negative probability does not make sense).
  - The components of x must add to one (the probabilities must add to one).

The above example can be stated as a more general problem:

$$Ax = \lambda x,$$

which is known as the (algebraic) eigenvalue problem. Scalars λ that satisfy $Ax = \lambda x$ for nonzero vector x are known as **eigenvalues** while the nonzero vectors are known as **eigenvectors**.

From the table above we can answer questions like “what is the typical weather?” (Answer: Cloudy). An approach similar to what we demonstrated in this unit is used, for example, to answer questions like “what is the most frequently visited webpage on a given topic?”

12.1.2 Outline

12.1.3 What You Will Learn

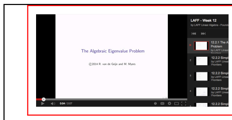
Upon completion of this unit, you should be able to

- Determine whether a given vector is an eigenvector for a particular matrix.
- Find the eigenvalues and eigenvectors for small-sized matrices.
- Identify eigenvalues of special matrices such as the zero matrix, the identity matrix, diagonal matrices, and triangular matrices.
- Interpret an eigenvector of A, as a direction in which the "action" of A, Ax, is equivalent to x being scaled without changing its direction. (Here scaling by a negative value still leaves the vector in the same direction.) Since this is true for any scalar multiple of x, it is the direction that is important, not the length of x.
- Compute the characteristic polynomial for 2×2 and 3×3 matrices.
- Know and apply the property that a matrix has an inverse if and only if its determinant is nonzero.
- Know and apply how the roots of the characteristic polynomial are related to the eigenvalues of a matrix.
- Recognize that if a matrix is real valued, then its characteristic polynomial has real valued coefficients but may still have complex eigenvalues that occur in conjugate pairs.
- Link diagonalization of a matrix with the eigenvalues and eigenvectors of that matrix.
- Make conjectures, reason, and develop arguments about properties of eigenvalues and eigenvectors.
- Understand practical algorithms for finding eigenvalues and eigenvectors such as the power method for finding an eigenvector associated with the largest eigenvalue (in magnitude).

Track your progress in Appendix B.

12.2 Getting Started

12.2.1 The Algebraic Eigenvalue Problem



☛ YouTube

The *algebraic eigenvalue problem* is given by

$$Ax = \lambda x.$$

where $A \in \mathbb{R}^{n \times n}$ is a square matrix, λ is a scalar, and x is a nonzero vector. Our goal is to, given matrix A, compute λ and x. It must be noted from the beginning that λ may be a complex number and that x will have complex components if λ is complex valued. If $x \neq 0$, then λ is said to be an *eigenvalue* and x is said to be an eigenvector associated with the eigenvalue λ. The tuple (λ, x) is said to be an *eigenpair*.

Here are some equivalent statements:

- $Ax = \lambda x$,, where $x \neq 0$.
  This is the statement of the (algebraic) eigenvalue problem.

- $Ax - \lambda x = 0$, where $x \neq 0$.
  This is merely a rearrangement of $Ax = \lambda x$.

- $Ax - \lambda I x = 0$, where $x \neq 0$.
  Early in the course we saw that $x = Ix$.

- $(A - \lambda I)x = 0$, where $x \neq 0$.
  This is a matter of fractoring' x out.

- $A - \lambda I$ is singular.
  Since there is a vector $x \neq 0$ such that $(A - \lambda I)x = 0$.

- $\mathcal{N}(A - \lambda I)$ contains a nonzero vector x.
  This is a consequence of there being a vector $x \neq 0$ such that $(A - \lambda I)x = 0$.

- $\dim(\mathcal{N}(A - \lambda I)) > 0$.
  Since there is a nonzero vector in $\mathcal{N}(A - \lambda I)$, that subspace must have dimension greater than zero.

If we find a vector $x \neq 0$ such that $Ax = \lambda x$, it is certainly not unique.

- For any scalar α, $A(\alpha x) = \lambda(\alpha x)$ also holds.

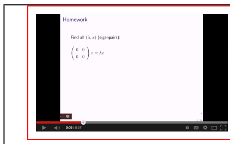
- If $Ax = \lambda x$ and $Ay = \lambda y$, then $A(x + y) = Ax + Ay = \lambda x + \lambda y = \lambda(x + y)$.

We conclude that the set of all vectors x that satisfy $Ax = \lambda x$ is a subspace.

It is not the case that the set of all vectors x that satisfy $Ax = \lambda x$ is the set of all eigenvectors associated with λ. After all, the zero vector is in that set, but is not considered an eigenvector.

It is important to think about eigenvalues and eigenvectors in the following way: If x is an eigenvector of A, then x is a direction in which the "action" of A (in other words, Ax) is equivalent to x being scaled in length without changing its direction other than changing sign. (Here we use the term "length" somewhat liberally, since it can be negative in which case the direction of x will be exactly the opposite of what it was before.) Since this is true for any scalar multiple of x, it is the direction that is important, not the magnitude of x.

12.2.2 Simple Examples



☛ YouTube

In this unit, we build intuition about eigenvalues and eigenvectors by looking at simple examples.

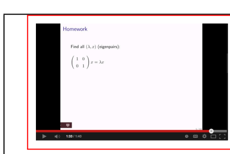
Homework 12.2.2.1 Which of the following are eigenpairs (λ, x) of the 2×2 zero matrix:

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} x = \lambda x,$$

where $x \neq 0$.
(Mark all correct answers.)

1. $(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix})$.
2. $(0, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
3. $(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix})$.
4. $(0, \begin{pmatrix} -1 \\ 1 \end{pmatrix})$.
5. $(0, \begin{pmatrix} 1 \\ 1 \end{pmatrix})$.
6. $(0, \begin{pmatrix} 0 \\ 0 \end{pmatrix})$.

☛ SEE ANSWER



☛ YouTube

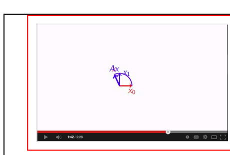
Homework 12.2.2.2 Which of the following are eigenpairs (λ, x) of the 2×2 zero matrix:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} x = \lambda x,$$

where $x \neq 0$.
(Mark all correct answers.)

1. $(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix})$.
2. $(1, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
3. $(1, \begin{pmatrix} 0 \\ 1 \end{pmatrix})$.
4. $(1, \begin{pmatrix} -1 \\ 1 \end{pmatrix})$.
5. $(1, \begin{pmatrix} 1 \\ 1 \end{pmatrix})$.
6. $(-1, \begin{pmatrix} 1 \\ -1 \end{pmatrix})$.

☛ SEE ANSWER



☛ YouTube

Homework 12.2.2.3 Let $A = \begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix}$.

- $\begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = 3\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ so that $(3, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$ is an eigenpair.

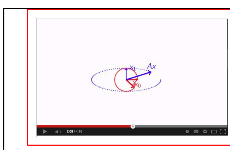
  True/False

- The set of all eigenvectors associated with eigenvalue 3 is characterized by (mark all that apply):
  - All vectors $x \neq 0$ that satisfy $Ax = 3x$.
  - All vectors $x \neq 0$ that satisfy $(A - 3I)x = 0$.
  - All vectors $x \neq 0$ that satisfy $\begin{pmatrix} 0 & 0 \\ 0 & -4 \end{pmatrix} x = 0$.
  - $\left\{ \begin{pmatrix} \chi_0 \\ 0 \end{pmatrix} \middle| \chi_0 \text{ is a scalar} \right\}$

- $\begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = -1\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ so that $(-1, \begin{pmatrix} 0 \\ 1 \end{pmatrix})$ is an eigenpair.

  True/False

☛ SEE ANSWER



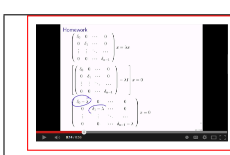
☛ YouTube

Homework 12.2.2.4 Consider the diagonal matrix $\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}$.

Eigenpairs for this matrix are given by $(\delta_0, e_0), (\delta_1, e_1), \cdots, (\delta_{n-1}, e_{n-1})$, where e_j equals the jth unit basis vector.

Always/Sometimes/Never

☛ SEE ANSWER



☛ YouTube

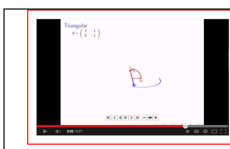
Homework 12.2.2.5 Which of the following are eigenpairs (λ, x) of the 2×2 triangular matrix:

$$\begin{pmatrix} 3 & 1 \\ 0 & -1 \end{pmatrix} x = \lambda x,$$

where $x \neq 0$.
(Mark all correct answers.)

1. $(-1, \begin{pmatrix} -1 \\ 4 \end{pmatrix})$.
2. $(1/3, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
3. $(3, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
4. $(-1, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
5. $(3, \begin{pmatrix} -1 \\ 0 \end{pmatrix})$.
6. $(-1, \begin{pmatrix} 3 \\ -1 \end{pmatrix})$.

☛ SEE ANSWER



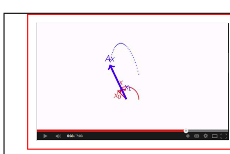
☛ YouTube

Homework 12.2.2.6 Consider the upper triangular matrix $U = \begin{pmatrix} \upsilon_{0,0} & \upsilon_{0,1} & \cdots & \upsilon_{0,n-1} \\ 0 & \upsilon_{1,1} & \cdots & \upsilon_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \upsilon_{n-1,n-1} \end{pmatrix}$.

The eigenvalues of this matrix are $\upsilon_{0,0}, \upsilon_{1,1}, \ldots, \upsilon_{n-1,n-1}$.

Always/Sometimes/Never

☛ SEE ANSWER



☛ YouTube

Below, on the left we discuss the general case, side-by-side with a specific example on the right.

General	Example
Consider $Ax = \lambda x$.	$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \lambda \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}.$
Rewrite as $Ax - \lambda x$	$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} - \lambda \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$
Rewrite as $Ax - \lambda I x = 0$.	$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$
Now $[A - \lambda I]x = 0$	$\left[\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$
$A - \lambda I$ is the matrix A with λ subtracted from its diagonal elements.	$\begin{pmatrix} 1-\lambda & -1 \\ 2 & 4-\lambda \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$

Now $A - \lambda I$ has a nontrivial vector x in its null space if that matrix does *not* have an inverse. Recall that

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}^{-1} = \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix}.$$

Here the scalar $\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}$ is known as the determinant of 2×2 matrix A, $\det(A)$.

This turns out to be a general statement:

> Matrix A has an inverse if and only if its determinant is nonzero.
> We have not yet defined the determinant of a matrix of size greater than 2.

So, the matrix $\begin{pmatrix} 1-\lambda & -1 \\ 2 & 4-\lambda \end{pmatrix}$ does not have an inverse if and only if

$$\det(\begin{pmatrix} 1-\lambda & -1 \\ 2 & 4-\lambda \end{pmatrix}) = (1-\lambda)(4-\lambda) - (2)(-1) = 0.$$

But

$$(1-\lambda)(4-\lambda) - (2)(-1) = 4 - 5\lambda + \lambda^2 + 2 = \lambda^2 - 5\lambda + 6$$

This is a quadratic (second degree) polynomial, which has at most two district roots. In particular, by examination,

$$\lambda^2 - 5\lambda + 6 = (\lambda - 2)(\lambda - 3) = 0$$

so that this matrix has two eigenvalues: $\lambda = 2$ and $\lambda = 3$.

If we now take $\lambda = 2$, then we can determine an eigenvector associated with that eigenvalue:

$$\begin{pmatrix} 1-(2) & -1 \\ 2 & 4-(2) \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

or

$$\begin{pmatrix} -1 & -1 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

By examination, we find that $\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$ is a vector in the null space and hence an eigenvector associated with the eigenvalue $\lambda = 2$. (This is not a unique solution. Any vector $\begin{pmatrix} \chi \\ -\chi \end{pmatrix}$ with $\chi \neq 0$ is an eigenvector.)

Similarly, if we take $\lambda = 3$, then we can determine an eigenvector associated with that second eigenvalue:

$$\begin{pmatrix} 1-(3) & -1 \\ 2 & 4-(3) \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

or

$$\begin{pmatrix} -2 & -1 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

By examination, we find that $\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ -2 \end{pmatrix}$ is a vector in the null space and hence an eigenvector associated with the eigenvalue $\lambda = 3$. (Again, this is not a unique solution. Any vector $\begin{pmatrix} \chi \\ -2\chi \end{pmatrix}$ with $\chi \neq 0$ is an eigenvector.)

The above discussion identifies a systematic way for computing eigenvalues and eigenvectors of a 2×2 matrix:

- Compute
$$\det\left(\begin{pmatrix} (\alpha_{0,0}-\lambda) & \alpha_{0,1} \\ \alpha_{1,0} & (\alpha_{1,1}-\lambda) \end{pmatrix}\right) = (\alpha_{0,0}-\lambda)(\alpha_{1,1}-\lambda) - \alpha_{0,1}\alpha_{1,0}.$$
- Recognize that this is a second degree polynomial in λ.
- It is called the *characteristic polynomial* of the matrix A, $p_2(\lambda)$.
- Compute the coefficients of $p_2(\lambda)$ so that
$$p_2(\lambda) = -\lambda^2 + \beta\lambda + \gamma.$$
- Solve
$$-\lambda^2 + \beta\lambda + \gamma = 0$$
for its roots. You can do this either by examination, or by using the quadratic formula:
$$\lambda = \frac{-\beta \pm \sqrt{\beta^2 + 4\gamma}}{-2}.$$
- For each of the roots, find an eigenvector that satisfies
$$\begin{pmatrix} (\alpha_{0,0}-\lambda) & \alpha_{0,1} \\ \alpha_{1,0} & (\alpha_{1,1}-\lambda) \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
The easiest way to do this is to subtract the eigenvalue from the diagonal, set one of the components of x to 1, and then solve for the other component.
- Check your answer! It is a matter of plugging it into $Ax = \lambda x$ and seeing if the computed λ and x satisfy the equation.

A 2×2 matrix yields a characteristic polynomial of degree at most two, and has at most two distinct eigenvalues.

Homework 12.2.2.7 Consider $A = \begin{pmatrix} 1 & 3 \\ 3 & 1 \end{pmatrix}$

- The eigenvalue *largest in magnitude* is
- Which of the following are eigenvectors associated with this largest eigenvalue (in magnitude):
  - $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$
  - $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$
  - $\begin{pmatrix} 2 \\ 2 \end{pmatrix}$
  - $\begin{pmatrix} -1 \\ 2 \end{pmatrix}$
- The eigenvalue *smallest in magnitude* is
- Which of the following are eigenvectors associated with this largest eigenvalue (in magnitude):
  - $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$
  - $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$
  - $\begin{pmatrix} 2 \\ 2 \end{pmatrix}$
  - $\begin{pmatrix} -1 \\ 2 \end{pmatrix}$

☛ SEE ANSWER

Homework 12.2.2.8 Consider $A = \begin{pmatrix} -3 & -4 \\ 5 & 6 \end{pmatrix}$

- The eigenvalue *largest in magnitude* is
- The eigenvalue *smallest in magnitude* is

☛ SEE ANSWER

Example 12.1 Consider the matrix $A = \begin{pmatrix} 3 & -1 \\ 2 & 1 \end{pmatrix}$. To find the eigenvalues and eigenvectors of this matrix, we form $A - \lambda I = \begin{pmatrix} 3-\lambda & -1 \\ 2 & 1-\lambda \end{pmatrix}$ and check when the characteristic polynomial is equal to zero:

$$\det\left(\begin{pmatrix} 3-\lambda & -1 \\ 2 & 1-\lambda \end{pmatrix}\right) = (3-\lambda)(1-\lambda) - (-1)(2) = \lambda^2 - 4\lambda + 5.$$

When is this equal to zero? We will use the quadratic formula:

$$\lambda = \frac{-(-4) \pm \sqrt{(-4)^2 - 4(5)}}{2} = 2 \pm i.$$

Thus, this matrix has complex valued eigenvalues in form of a conjugate pair: $\lambda_0 = 2+i$ and $\lambda_1 = 2-i$. To find the corresponding eigenvectors:

$\lambda_0 = 2+i$:

$$\begin{aligned} A - \lambda_0 I &= \begin{pmatrix} 3-(2+i) & -1 \\ 2 & 1-(2+i) \end{pmatrix} \\ &= \begin{pmatrix} 1-i & -1 \\ 2 & -1-i \end{pmatrix}. \end{aligned}$$

Find a nonzero vector in the null space:

$$\begin{pmatrix} 1-i & -1 \\ 2 & -1-i \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

By examination,

$$\begin{pmatrix} 1-i & -1 \\ 2 & -1-i \end{pmatrix} \begin{pmatrix} 1 \\ 1-i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Eigenpair: $(2+i, \begin{pmatrix} 1 \\ 1-i \end{pmatrix})$.

$\lambda_0 = 2-i$:

$$\begin{aligned} A - \lambda_1 I &= \begin{pmatrix} 3-(2-i) & -1 \\ 2 & 1-(2-i) \end{pmatrix} \\ &= \begin{pmatrix} 1+i & -1 \\ 2 & -1+i \end{pmatrix}. \end{aligned}$$

Find a nonzero vector in the null space:

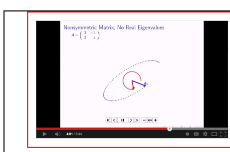
$$\begin{pmatrix} 1+i & -1 \\ 2 & -1+i \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

By examination,

$$\begin{pmatrix} 1+i & -1 \\ 2 & -1+i \end{pmatrix} \begin{pmatrix} 1 \\ 1+i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Eigenpair: $(2-i, \begin{pmatrix} 1 \\ 1+i \end{pmatrix})$.

If A is real valued, then its characteristic polynomial has real valued coefficients. However, a polynomial with real valued coefficients may still have complex valued roots. Thus, the eigenvalues of a real valued matrix may be complex.



☛ YouTube

Homework 12.2.2.9 Consider $A = \begin{pmatrix} 2 & 2 \\ -1 & 4 \end{pmatrix}$. Which of the following are the eigenvalues of A:

- 4 and 2.
- $3+i$ and 2.
- $3+i$ and $3-i$.
- $2+i$ and $2-i$.

☛ SEE ANSWER

12.2.3 Diagonalizing

☛ YouTube

Diagonalizing a square matrix $A \in \mathbb{R}^{n\times n}$ is closely related to the problem of finding the eigenvalues and eigenvectors of a matrix. In this unit, we illustrate this for some simple 2×2 examples. A more thorough treatment then follows when we talk about the eigenvalues and eigenvectors of $n\times n$ matrix, later this week.

In the last unit, we found eigenpairs for the matrix

$$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}.$$

Specifically,

$$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \end{pmatrix} = 2\begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} -1 \\ 2 \end{pmatrix} = 3\begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

so that eigenpairs are given by

$$(2, \begin{pmatrix} -1 \\ 1 \end{pmatrix}) \quad \text{and} \quad 3\begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

Now, let's put our understanding of matrix-matrix multiplication from Weeks 4 and 5 to good use:

	Comment (A here is 2×2)
$\underbrace{\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \end{pmatrix} = 2\begin{pmatrix} -1 \\ 1 \end{pmatrix}; \begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} -1 \\ 2 \end{pmatrix} = 3\begin{pmatrix} -1 \\ 2 \end{pmatrix}}$	$Ax_0 = \lambda_0 x_0; Ax_1 = \lambda_1 x_1$
$\underbrace{\left(\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \end{pmatrix} \middle\vert \begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}\begin{pmatrix} -1 \\ 2 \end{pmatrix}\right) = \left(2\begin{pmatrix} -1 \\ 1 \end{pmatrix} \middle\vert 3\begin{pmatrix} -1 \\ 2 \end{pmatrix}\right)}$	$\left(Ax_0 \middle\vert Ax_1 \right) = \left(\lambda_0 x_0 \middle\vert \lambda_1 x_1 \right)$
$\underbrace{\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}}_{A} \underbrace{\left(\begin{matrix} -1 \\ 1 \end{matrix} \middle\vert \begin{matrix} -1 \\ 2 \end{matrix}\right)}_{X} = \underbrace{\left(\begin{matrix} -1 \\ 1 \end{matrix} \middle\vert \begin{matrix} -1 \\ 2 \end{matrix}\right)}_{X} \underbrace{\begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}}_{\Lambda}$	$A \underbrace{\left(x_0 \middle\vert x_1 \right)}_{X} = \underbrace{\left(x_0 \middle\vert x_1 \right)}_{X} \underbrace{\begin{pmatrix} \lambda_0 & 0 \\ 0 & \lambda_1 \end{pmatrix}}_{\Lambda}$
$\underbrace{\left(\begin{matrix} -1 \\ 1 \end{matrix} \middle\vert \begin{matrix} -1 \\ 2 \end{matrix}\right)^{-1}}_{X^{-1}} \underbrace{\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix}}_{A} \underbrace{\left(\begin{matrix} -1 \\ 1 \end{matrix} \middle\vert \begin{matrix} -1 \\ 2 \end{matrix}\right)}_{X} = \underbrace{\begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}}_{\Lambda}$	$\underbrace{\left(x_0 \middle\vert x_1 \right)^{-1}}_{X^{-1}} A \underbrace{\left(x_0 \middle\vert x_1 \right)}_{X} = \underbrace{\begin{pmatrix} \lambda_0 & 0 \\ 0 & \lambda_1 \end{pmatrix}}_{\Lambda}$

What we notice is that if we take the two eigenvectors of matrix A, and create with them a matrix X that has those eigenvectors as its columns, then $X^{-1}AX = \Lambda$, where Λ is a diagonal matrix with the eigenvalues on its diagonal. The matrix X is said to *diagonalize* matrix A.

Defective matrices

Now, it is *not* the case that for every $A \in \mathbb{R}^{n\times n}$ there is a nonsingular matrix $X \in \mathbb{R}^{n\times n}$ such that $X^{-1}AX = \Lambda$, where Λ is diagonal. Matrices for which such a matrix X does not exists are called *defective* matrices.

Homework 12.2.3.1 The matrix

$$\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

can be diagonalized.

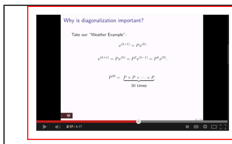
True/False

☛ SEE ANSWER

The matrix

$$\begin{pmatrix} \lambda & 1 \\ 0 & \lambda \end{pmatrix}$$

is a simple example of what is often called a *Jordan block*. It, too, is defective.



☛ YouTube

Homework 12.2.3.2 In Homework 12.2.2.7 you considered the matrix

$$A = \begin{pmatrix} 1 & 3 \\ 3 & 1 \end{pmatrix}$$

and computed the eigenpairs

$$(4, \begin{pmatrix} 1 \\ 1 \end{pmatrix}) \quad \text{and} \quad (-2, \begin{pmatrix} 1 \\ -1 \end{pmatrix}).$$

- Matrix A can be diagonalized by matrix $X =$. (Yes, this matrix is not unique, so please use the info from the eigenpairs, in order...)
- $AX =$
- $X^{-1} =$
- $X^{-1}AX =$

☛ SEE ANSWER

12.2.4 Eigenvalues and Eigenvectors of 3×3 Matrices

☛ YouTube

Homework 12.2.4.1 Let $A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix}$. Then which of the following are true:

- $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue 3.

  True/False

- $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue -1.

  True/False

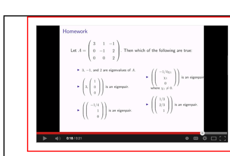
- $\begin{pmatrix} 0 \\ \chi_1 \\ 0 \end{pmatrix}$, where $\chi_1 \neq 0$ is a scalar, is an eigenvector associated with eigenvalue -1.

  True/False

- $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is an eigenvector associated with eigenvalue 2.

  True/False

☛ SEE ANSWER



☛ YouTube

Homework 12.2.4.2 Let $A = \begin{pmatrix} \alpha_{0,0} & 0 & 0 \\ 0 & \alpha_{1,1} & 0 \\ 0 & 0 & \alpha_{2,2} \end{pmatrix}$. Then which of the following are true:

- $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue $\alpha_{0,0}$.

  True/False

- $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue $\alpha_{1,1}$.

  True/False

- $\begin{pmatrix} 0 \\ \chi_1 \\ 0 \end{pmatrix}$ where $\chi_1 \neq 0$ is an eigenvector associated with eigenvalue $\alpha_{1,1}$.

  True/False

- $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is an eigenvector associated with eigenvalue $\alpha_{2,2}$.

  True/False

☛ SEE ANSWER

Homework 12.2.4.3 Let $A = \begin{pmatrix} 3 & 1 & -1 \\ 0 & -1 & 2 \\ 0 & 0 & 2 \end{pmatrix}$. Then which of the following are true:

- 3, -1, and 2 are eigenvalues of A.
- $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue 3.

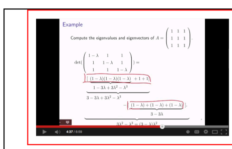
  True/False
- $\begin{pmatrix} -1/4 \\ 1 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue -1.

  True/False
- $\begin{pmatrix} -1/4\chi_1 \\ \chi_1 \\ 0 \end{pmatrix}$ where $\chi_1 \neq 0$ is an eigenvector associated with eigenvalue -1.

  True/False
- $\begin{pmatrix} 1/3 \\ 2/3 \\ 1 \end{pmatrix}$ is an eigenvector associated with eigenvalue 2.

  True/False

☛ SEE ANSWER



☛ YouTube

Homework 12.2.4.4 Let $A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} \\ 0 & \alpha_{1,1} & \alpha_{1,2} \\ 0 & 0 & \alpha_{2,2} \end{pmatrix}$. Then the eigenvalues of this matrix are $\alpha_{0,0}$, $\alpha_{1,1}$, and $\alpha_{2,2}$.

True/False

☛ SEE ANSWER

When we discussed how to find the eigenvalues of a 2×2 matrix, we saw that it all came down to the determinant of $A - \lambda I$, which then gave us the characteristic polynomial $p_2(\lambda)$. The roots of this polynomial were the eigenvalues of the matrix.

Similarly, there is a formula for the determinant of a 3×3 matrix:

$$\det\left(\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} \end{pmatrix}\right) =$$

$$\underbrace{(\alpha_{0,0}\alpha_{1,1}\alpha_{2,2}+\alpha_{0,1}\alpha_{1,2}\alpha_{2,0}+\alpha_{0,2}\alpha_{1,0}\alpha_{2,1})}_{\begin{array}{ccc|cc} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,0} & \alpha_{1,1} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,0} & \alpha_{2,1} \end{array}} - \underbrace{(\alpha_{2,0}\alpha_{1,1}\alpha_{0,2}+\alpha_{2,1}\alpha_{1,2}\alpha_{0,0}+\alpha_{2,2}\alpha_{1,0}\alpha_{0,1})}_{\begin{array}{ccc|cc} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,0} & \alpha_{1,1} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,0} & \alpha_{2,1} \end{array}}.$$

Thus, for a 3×3 matrix, the characteristic polynomial becomes

$$\begin{array}{l} p_3(\lambda) = \det(\left(\begin{array}{ccc} \alpha_{0,0}-\lambda & \alpha_{0,1} & \alpha_{0,2} \\ \alpha_{1,0} & \alpha_{1,1}-\lambda & \alpha_{1,2} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2}-\lambda \end{array}\right) = \\ \qquad [(\alpha_{0,0}-\lambda)(\alpha_{1,1}-\lambda)(\alpha_{2,2}-\lambda)+\alpha_{0,1}\alpha_{1,2}\alpha_{2,0}+\alpha_{0,2}\alpha_{1,0}\alpha_{2,1}] \\ \qquad -[\alpha_{2,0}(\alpha_{1,1}-\lambda)\alpha_{0,2}+\alpha_{2,1}\alpha_{1,2}(\alpha_{0,0}-\lambda)+(\alpha_{2,2}-\lambda)\alpha_{1,0}\alpha_{0,1}]. \end{array}$$

Multiplying this out, we get a third degree polynomial. The roots of this cubic polynomial are the eigenvalues of the 3×3 matrix. Hence, a 3×3 matrix has at most three distinct eigenvalues.

Example 12.2 Compute the eigenvalues and eigenvectors of $A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$.

$$\det\left(\begin{pmatrix} 1-\lambda & 1 & 1 \\ 1 & 1-\lambda & 1 \\ 1 & 1 & 1-\lambda \end{pmatrix}\right) = \underbrace{\underbrace{[\underbrace{(1-\lambda)(1-\lambda)(1-\lambda)}_{1-3\lambda+3\lambda^2-\lambda^3} + 1 + 1]}_{3-3\lambda+3\lambda^2-\lambda^3} - [\underbrace{(1-\lambda)+(1-\lambda)+(1-\lambda)}_{3-3\lambda}]}_{3\lambda^2-\lambda^3 = (3-\lambda)\lambda^2}.$$

So, $\lambda = 0$ is a double root, while $\lambda = 3$ is the third root.

<table>
<tr>
<td>

$\lambda_2 = 3$:

$$A - \lambda_2 I = \begin{pmatrix} -2 & 1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & -2 \end{pmatrix}$$

We wish to find a nonzero vector in the null space:

$$\begin{pmatrix} -2 & 1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

By examination, I noticed that

$$\begin{pmatrix} -2 & 1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Eigenpair:

$$(3, \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}).$$

</td>
<td>

$\lambda_0 = \lambda_1 = 0$:

$$A - 0I = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Reducing this to row-echelon form gives us the matrix

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

for which we find vectors in the null space

$$\begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} \text{ and } \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

Eigenpairs:

$$(0, \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix}) \text{ and } (0, \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix})$$

</td>
</tr>
</table>

What is interesting about this last example is that $\lambda = 0$ is a double root and yields two linearly independent eigenvectors.

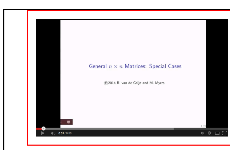
Homework 12.2.4.5 Consider $A = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$. Which of the following is true about this matrix:

- $(1, \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix})$ is an eigenpair.
- $(0, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix})$ is an eigenpair.
- $(0, \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix})$ is an eigenpair.
- This matrix is defective.

☛ SEE ANSWER

12.3 The General Case

12.3.1 Eigenvalues and Eigenvectors of $n \times n$ matrices: Special Cases



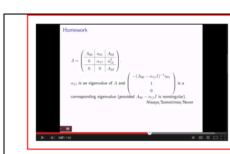
☛ YouTube

We are now ready to talk about eigenvalues and eigenvectors of arbitrary sized matrices.

Homework 12.3.1.1 Let $A \in \mathbb{R}^{n\times n}$ be a diagonal matrix: $A = \begin{pmatrix} \alpha_{0,0} & 0 & 0 & \cdots & 0 \\ 0 & \alpha_{1,1} & 0 & \cdots & 0 \\ 0 & 0 & \alpha_{2,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_{n-1,n-1} \end{pmatrix}$. Then e_i is an eigenvector associated with eigenvalue $\alpha_{i,i}$.

True/False

☛ SEE ANSWER

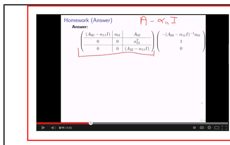


☛ YouTube

Homework 12.3.1.2 Let $A = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & 0 & A_{22} \end{array}\right)$, where A_{00} is square. Then α_{11} is an eigenvalue of A and $\left(\begin{array}{c} -(A_{00}-\alpha_{11}I)^{-1}a_{01} \\ 1 \\ 0 \end{array}\right)$ is a corresponding eigenvalue (provided $A_{00} - \alpha_{11}I$ is nonsingular).

True/False

☛ SEE ANSWER



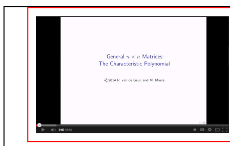
☛ YouTube

Homework 12.3.1.3 The eigenvalues of a triangular matrix can be found on its diagonal.

True/False

☛ SEE ANSWER

12.3.2 Eigenvalues of $n \times n$ Matrices



☛ YouTube

There is a formula for the determinant of a $n \times n$ matrix, which is a "inductively defined function", meaning that the formula for the determinant of an $n \times n$ matrix is defined in terms of the determinant of an $(n-1) \times (n-1)$ matrix. Other than as a theoretical tool, the determinant of a general $n \times n$ matrix is not particularly useful. We restrict our discussion to some facts and observations about the determinant that impact the characteristic polynomial, which is the polynomial that results when one computes the determinant of the matrix $A - \lambda I$, $\det(A - \lambda I)$.

Theorem 12.3 *A matrix $A \in \mathbb{R}^{n\times n}$ is nonsingular if and only if* $\det(A) \neq 0$.

Theorem 12.4 *Given $A \in \mathbb{R}^{n\times n}$,*

$$p_n(\lambda) = \det(A - \lambda I) = \lambda^n + \gamma_{n-1}\lambda^{n-1} + \cdots + \gamma_1\lambda + \gamma_0.$$

for some coefficients $\gamma_1, \ldots, \gamma_{n-1} \in \mathbb{R}$.

Since we don't give the definition of a determinant, we do not prove the above theorems.

Definition 12.5 *Given $A \in \mathbb{R}^{n\times n}$, $p_n(\lambda) = \det(A - \lambda I)$ is called the characteristic polynomial.*

Theorem 12.6 *Scalar λ satisfies $Ax = \lambda x$ for some nonzero vector x if and only if* $\det(A - \lambda I) = 0$.

Proof: This is an immediate consequence of the fact that $Ax = \lambda x$ is equivalent to $(A - \lambda I)x =$ and the fact that $A - \lambda I$ is singular (has a nontrivial null space) if and only if $\det(A - \lambda I) = 0$.

Roots of the characteristic polynomial

Since an eigenvalue of A is a root of $p_n(A) = \det(A - \lambda I)$ and vise versa, we can exploit what we know about roots of nth degree polynomials. Let us review, relating what we know to the eigenvalues of A.

- The characteristic polynomial of $A \in \mathbb{R}^{n\times n}$ is given by $p_n(\lambda) = \det(A - \lambda I) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n$

- Since $p_n(\lambda)$ is an nth degree polynomial, it has n roots, counting multiplicity. Thus, matrix A has n eigenvalues, counting multiplicity.
  - Let k equal the number of *distinct* roots of $p_n(\lambda)$. Clearly, $k \leq n$. Clearly, matrix A then has k distinct eigenvalues.
  - The set of all roots of $p_n(\lambda)$, which is the set of all eigenvalues of A, is denoted by $\Lambda(A)$ and is called the *spectrum* of matrix A.
  - The characteristic polynomial can be factored as $p_n(\lambda) = \det(A - \lambda I) = (\lambda - \lambda_0)^{n_0}(\lambda - \lambda_1)^{n_1} \cdots (\lambda - \lambda_{k-1})^{n_{k-1}}$, where $n_0 + n_1 + \cdots + n_{k-1} = n$ and n_j is the root λ_j, which is known as the (algebraic) multiplicity of eigenvalue λ_j.

- If $A \in \mathbb{R}^{n\times n}$, then the coefficients of the characteristic polynomial are real ($\gamma_0, \ldots, \gamma_{n-1} \in \mathbb{R}$), but
  - Some or all of the roots/eigenvalues may be complex valued and
  - Complex roots/eigenvalues come in "conjugate pairs": If $\lambda = \mathcal{R}\rceil(\lambda) + iI\Updownarrow(\lambda)$ is a root/eigenvalue, so is $\lambda = \mathcal{R}\rceil(\lambda) - iI\Updownarrow(\lambda)$

An inconvenient truth

Galois theory tells us that for $n \geq 5$, roots of arbitrary $p_n(\lambda)$ cannot be found in a finite number of computations.

Since we did not tell you how to compute the determinant of $A - \lambda I$, you will have to take the following for granted: For every $n$the degree polynomial

$$p_n(\lambda) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n,$$

there exists a matrix, C, called the companion matrix that has the property that

$$p_n(\lambda) = \det(C - \lambda I) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n.$$

In particular, the matrix

$$C = \begin{pmatrix} -\gamma_{n-1} & -\gamma_{n-2} & \cdots & -\gamma_1 & -\gamma_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}$$

is the companion matrix for $p_n(\lambda)$:

$$p_n(\lambda) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n = \det(\begin{pmatrix} -\gamma_{n-1} & -\gamma_{n-2} & \cdots & -\gamma_1 & -\gamma_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} - \lambda I).$$

Homework 12.3.2.1 If $A \in \mathbb{R}^{n\times n}$, then $\Lambda(A)$ has n distinct elements.

True/False

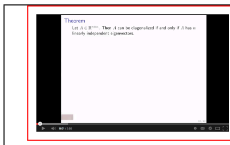
☛ SEE ANSWER

Homework 12.3.2.2 Let $A \in \mathbb{R}^{n \times n}$ and $\lambda \in \Lambda(A)$. Let S be the set of all vectors that satisfy $Ax = \lambda x$. (Notice that S is the set of all eigenvectors corresponding to λ *plus* the zero vector.) Then S is a subspace.

True/False

☛ SEE ANSWER

12.3.3 Diagonalizing, Again



☛ **YouTube**

We now revisit the topic of diagonalizing a square matrix $A \in \mathbb{R}^{n \times n}$, but for general n rather than the special case of $n = 2$ treated in Unit 12.2.3.

Let us start by assuming that matrix $A \in \mathbb{R}^{n \times n}$ has n eigenvalues, $\lambda_0, \ldots, \lambda_{n-1}$, where we simply repeat eigenvalues that have algebraic multiplicity greater than one. Let us also assume that x_j equals the eigenvector associated with eigenvalue λ_j and, importantly, that $x_0, \ldots, x_{n-1}$ are linearly independent. Below, we generalize the example from Unit 12.2.3.

$$
\begin{array}{l}
Ax_0 = \lambda_0 x_0; Ax_1 = \lambda_1 x_1; \cdots ; Ax_{n-1} = \lambda_{n-1} x_{n-1} \\
\qquad \text{if and only if} \qquad < \text{two matrices are equal if their columns are equal} > \\
\left(\begin{array}{c|c|c|c} Ax_0 & Ax_1 & \cdots & Ax_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} \lambda_0 x_0 & \lambda_1 x_1 & \cdots & \lambda_{n-1} x_{n-1} \end{array} \right) \\
\qquad \text{if and only if} \qquad < \text{partitioned matrix-matrix multiplication} > \\
A \left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} \lambda_0 x_0 & \lambda_1 x_1 & \cdots & \lambda_{n-1} x_{n-1} \end{array} \right) \\
\qquad \text{if and only if} \qquad < \text{multiplication on the right by a diagonal matrix} > \\
A \left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{n-1} \end{array} \right) \left(\begin{array}{cccc} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} \end{array} \right) \\
\qquad \text{if and only if} \qquad < \text{multiplication on the right by a diagonal matrix} > \\
AX = X\Lambda \text{ where } X = \left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{n-1} \end{array} \right) \text{ and } \Lambda = \left(\begin{array}{cccc} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} \end{array} \right) \\
\qquad \text{if and only if} \qquad < \text{columns of } X \text{ are linearly independent} > \\
X^{-1}AX = \Lambda \text{ where } X = \left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{n-1} \end{array} \right) \text{ and } \Lambda = \left(\begin{array}{cccc} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} \end{array} \right)
\end{array}
$$

The above argument motivates the following theorem:

Theorem 12.7 *Let $A \in \mathbb{R}^{n \times n}$. Then there exists a nonsingular matrix X such that $X^{-1}AX = \Lambda$ if and only if A has n linearly independent eigenvectors.*

If X is invertible (nonsingular, has linearly independent columns, etc.), then the following are equivalent

$$\begin{array}{rcl} X^{-1} A X & = & \Lambda \\ A X & = & X \Lambda \\ A & = & X \Lambda X^{-1} \end{array}$$

If Λ is in addition diagonal, then the diagonal elements of Λ are eigenvalues of A and the columns of X are eigenvectors of A.

Recognize that $\Lambda(A)$ denotes the spectrum (set of all eigenvalues) of matrix A while here we use it to denote the matrix Λ, which has those eigenvalues on its diagonal. This possibly confusing use of the same symbol for two different but related things is commonly encountered in the linear algebra literature. For this reason, you might as well get use to it!

Defective (deficient) matrices

We already saw in Unit 12.2.3, that it is *not* the case that for every $A \in \mathbb{R}^{n\times n}$ there is a nonsingular matrix $X \in \mathbb{R}^{n\times n}$ such that $X^{-1}AX = \Lambda$, where Λ is diagonal. In that unit, a 2×2 example was given that did not have two linearly independent eigenvectors.

In general, the $k \times k$ matrix $J_k(\lambda)$ given by

$$J_k(\lambda) = \begin{pmatrix} \lambda & 1 & 0 & \cdots & 0 & 0 \\ 0 & \lambda & 1 & \cdots & 0 & 0 \\ 0 & 0 & \lambda & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda & 1 \\ 0 & 0 & 0 & \cdots & 0 & \lambda \end{pmatrix}$$

has eigenvalue λ of algebraic multiplicity k, but *geometric multiplicity* one (it has only one linearly independent eigenvector). Such a matrix is known as a Jordan block.

Definition 12.8 *The geometric multiplicity of an eigenvalue λ equals the number of linearly independent eigenvectors that are associated with λ.*

The following theorem has theoretical significance, but little practical significance (which is why we do not dwell on it):

Theorem 12.9 *Let $A \in \mathbb{R}^{n\times n}$. Then there exists a nonsingular matrix $X \in \mathbb{R}^{n\times n}$ such that $A = XJX^{-1}$, where*

$$J = \begin{pmatrix} J_{k_0}(\lambda_0) & 0 & 0 & \cdots & 0 \\ 0 & J_{k_1}(\lambda_1) & 0 & \cdots & 0 \\ 0 & 0 & J_{k_2}(\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & J_{k_{m-1}}(\lambda_{m-1}) \end{pmatrix}$$

where each $J_{k_j}(\lambda_j)$ is a Jordan block of size $k_j \times k_j$.

The factorization $A = XJX^{-1}$ is known as the Jordan Canonical Form of matrix A.

A few comments are in order:

- It is *not* the case that $\lambda_0, \lambda_1, \ldots, \lambda_{m-1}$ are distinct. If λ_j appears in multiple Jordan blocks, the number of Jordan blocks in which λ_j appears equals the geometric multiplicity of λ_j (and the number of linearly independent eigenvectors associated with λ_j).

- The sum of the sizes of the blocks in which λ_j as an eigenvalue appears equals the algebraic multiplicity of λ_j.
- If each Jordan block is 1×1, then the matrix is diagonalized by matrix X.
- If any of the blocks is not 1×1, then the matrix cannot be diagonalized.

Homework 12.3.3.1 Consider $A = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{pmatrix}$.

- The algebraic multiplicity of $\lambda = 2$ is
- The geometric multiplicity of $\lambda = 2$ is
- The following vectors are linearly independent eigenvectors associated with $\lambda = 2$:

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \text{ and } \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}.$$

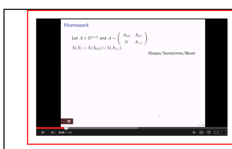
True/False

☛ SEE ANSWER

Homework 12.3.3.2 Let $A \in \mathbb{A}^{n \times n}$, $\lambda \in \Lambda(A)$, and S be the set of all vectors x such that $Ax = \lambda x$. Finally, let λ have algebraic multiplicity k (meaning that it is a root of multiplicity k of the characteristic polynomial).
The dimension of S is k ($\dim(S) = k$).

Always/Sometimes/Never
☛ SEE ANSWER

12.3.4 Properties of Eigenvalues and Eigenvectors



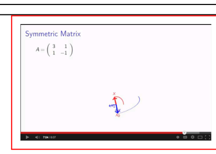
☛ **YouTube**

In this unit, we look at a few theoretical results related to eigenvalues and eigenvectors.

Homework 12.3.4.1 Let $A \in \mathbb{R}^{n \times n}$ and $A = \begin{pmatrix} A_{0,0} & A_{0,1} \\ 0 & A_{1,1} \end{pmatrix}$, where $A_{0,0}$ and $A_{1,1}$ are square matrices.

$\Lambda(A) = \Lambda(A_{0,0}) \cup \Lambda(A_{1,1})$.

Always/Sometimes/Never
☛ SEE ANSWER



☛ **YouTube**

The last exercise motives the following theorem (which we will not prove):

Theorem 12.10 *Let $A \in \mathbb{R}^{n \times n}$ and*

$$A = \begin{pmatrix} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ 0 & A_{1,1} & \cdots & A_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{N-1,N-1} \end{pmatrix}$$

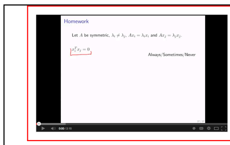
where all $A_{i,i}$ are a square matrices. Then $\Lambda(A) = \Lambda(A_{0,0}) \cup \Lambda(A_{1,1}) \cup \cdots \cup \Lambda(A_{N-1,N-1})$.

Homework 12.3.4.2 Let $A \in \mathbb{R}^{n \times n}$ be symmetric, $\lambda_i \neq \lambda_j$, $Ax_i = \lambda_i x_i$ and $Ax_j = \lambda_j x_j$.
$x_i^T x_j = 0$

Always/Sometimes/Never
☛ SEE ANSWER

The following theorem requires us to remember more about complex arithmetic than we have time to remember. For this reason, we will just state it:

Theorem 12.11 *Let $A \in \mathbb{R}^{n \times n}$ be symmetric. Then its eigenvalues are real valued.*



☛ YouTube

Homework 12.3.4.3 If $Ax = \lambda x$ then $AAx = \lambda^2 x$. (AA is often written as A^2.)

Always/Sometimes/Never
☛ SEE ANSWER

Homework 12.3.4.4 Let $Ax = \lambda x$ and $k \geq 1$. Recall that $A^k = \underbrace{AA \cdots A}_{k \text{ times}}$.

$A^k x = \lambda^k x$.

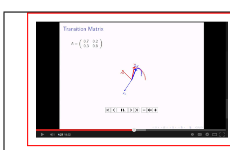
Always/Sometimes/Never
☛ SEE ANSWER

Homework 12.3.4.5 $A \in \mathbb{R}^{n \times n}$ is nonsingular if and only if $0 \notin \Lambda(A)$.

True/False
☛ SEE ANSWER

12.4 Practical Methods for Computing Eigenvectors and Eigenvalues

12.4.1 Predicting the Weather, One Last Time



☛ YouTube

If you think back about how we computed the probabilities of different types of weather for day k, recall that

$$x^{(k+1)} = Px^{(k)}$$

where $x^{(k)}$ is a vector with three components and P is a 3×3 matrix. We also showed that

$$x^{(k)} = P^k x^{(0)}.$$

We noticed that eventually

$$x^{(k+1)} \approx Px^{(k)}$$

and that therefore, eventually, $x^{(k+1)}$ came arbitrarily close to an eigenvector, x, associated with the eigenvalue 1 of matrix P:

$$Px = x.$$

Homework 12.4.1.1 If $\lambda \in \Lambda(A)$ then $\lambda \in \Lambda(A^T)$.

True/False

☛ SEE ANSWER

☛ YouTube

Homework 12.4.1.2 $\lambda \in \Lambda(A)$ if and only if $\lambda \in \Lambda(A^T)$.

True/False

☛ SEE ANSWER

Ah! It seems like we may have stumbled upon a possible method for computing an eigenvector for this matrix:

- Start with a first guess $x^{(0)}$.
- **for** $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  - $x^{(k+1)} := Px^{(k)}$.

Can we use what we have learned about eigenvalues and eigenvectors to explain this? In the video, we give one explanation. Below we give an alternative explanation that uses diagonalization.

Let's assume that P is diagonalizable:

$$P = V\Lambda V^{-1}, \quad \text{where } \Lambda = \begin{pmatrix} \lambda_0 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{pmatrix}.$$

Here we use the letter V rather than X since we already use $x^{(k)}$ in a different way.

Then we saw before that

$$\begin{aligned} x^{(k)} = P^k x^{(0)} &= (V\Lambda V^{-1})^k x^{(0)} = V\Lambda^k V^{-1} x^{(0)} \\ &= V \begin{pmatrix} \lambda_0 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{pmatrix}^k V^{-1} x^{(0)} \\ &= V \begin{pmatrix} \lambda_0^k & 0 & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{pmatrix} V^{-1} x^{(0)}. \end{aligned}$$

Now, let's assume that $\lambda_0 = 1$ (since we noticed that P has one as an eigenvalue), and that $|\lambda_1| < 1$ and $|\lambda_2| < 1$. Also, notice that $V = \begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix}$

where v_i equals the eigenvector associated with λ_i. Finally, notice that V has linearly independent columns and that therefore there exists a vector w such that $Vw = x^{(0)}$.

Then

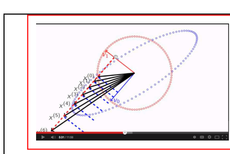
$$
\begin{aligned}
x^{(k)} &= V \begin{pmatrix} \lambda_0^k & 0 & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{pmatrix} V^{-1} x^{(0)} \\
&= V \begin{pmatrix} 1 & 0 & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{pmatrix} V^{-1} V w \\
&= \begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{pmatrix} w \\
&= \begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{pmatrix} \begin{pmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \end{pmatrix}.
\end{aligned}
$$

Now, what if k gets very large? We know that $\lim_{k\to\infty} \lambda_1^k = 0$, since $|\lambda_1| < 1$. Similarly, $\lim_{k\to\infty} \lambda_2^k = 0$. So,

$$
\begin{aligned}
\lim_{k\to\infty} x^{(k)} &= \lim_{k\to\infty} \left[\begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{pmatrix} \begin{pmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \end{pmatrix} \right] \\
&= \begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix} \lim_{k\to\infty} \left[\begin{pmatrix} 1 & 0 & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{pmatrix} \right] \begin{pmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \end{pmatrix} \\
&= \begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \lim_{k\to\infty} \lambda_1^k & 0 \\ 0 & 0 & \lim_{k\to\infty} \lambda_2^k \end{pmatrix} \begin{pmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \end{pmatrix} \\
&= \begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \end{pmatrix} \\
&= \begin{pmatrix} v_0 & v_1 & v_2 \end{pmatrix} \begin{pmatrix} \omega_0 \\ 0 \\ 0 \end{pmatrix} = \omega_0 v_0.
\end{aligned}
$$

Ah, so $x^{(k)}$ eventually becomes arbitrarily close (converges) to a multiple of the eigenvector associated with the eigenvalue 1 (provided $\omega_0 \neq 0$).

12.4.2 The Power Method



☛ YouTube

So, a question is whether the method we described in the last unit can be used in general. The answer is yes. The resulting method is known as the Power Method.

First, let's make some assumptions. Given $A \in \mathbb{R}^{n\times n}$,

- Let $\lambda_0, \lambda_1, \ldots, \lambda_{n-1} \in \Lambda(A)$. We list eigenvalues that have algebraic multiplicity k multiple (k) times in this list.

- Let us assume that $|\lambda_0| > |\lambda_1| \geq |\lambda_2| \geq \cdots \geq \lambda_{n-1}$. This implies that λ_0 is real, since complex eigenvalues come in conjugate pairs and hence there would have been two eigenvalues with equal greatest magnitude. It also means that there is a real valued eigenvector associated with λ_0.

- Let us assume that $A \in \mathbb{R}^{n\times n}$ is diagonalizable so that

$$A = V\Lambda V^{-1} = \left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1} \end{array}\right) \left(\begin{array}{cccc} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} \end{array}\right) \left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1} \end{array}\right)^{-1}.$$

  This means that v_i is an eigenvector associated with λ_i.

These assumptions set the stage.

Now, we start with some vector $x^{(0)} \in \mathbb{R}^n$. Since V is nonsingular, the vectors $v_0, \ldots, v_{n-1}$ form a linearly independent bases for $\mathbb{R}^n$. Hence,

$$x^{(0)} = \gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-1} v_{n-1} = \left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1} \end{array}\right) \left(\begin{array}{c} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-1} \end{array}\right) = Vc.$$

Now, we generate

$$\begin{array}{rcl} x^{(1)} & = & Ax^{(0)} \\ x^{(2)} & = & Ax^{(1)} \\ x^{(3)} & = & Ax^{(2)} \\ & \vdots & \end{array}$$

The following algorithm accomplishes this

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  $x^{(k+1)} := Ax^{(k)}$
endfor

Notice that then

$$x^{(k)} = Ax^{(k-1)} = A^2x^{(k-2)} = \cdots = A^kx^{(0)}.$$

But then

$$\begin{array}{rcl} A^kx^{(0)} & = & A^k(\underbrace{\gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-1} v_{n-1}}_{Vc}) \\ & = & A^k\gamma_0 v_0 + A^k\gamma_1 v_1 + \cdots + A^k\gamma_{n-1} v_{n-1} \\ & = & \gamma_0 A^k v_0 + \gamma_1 A^k v_1 + \cdots + \gamma_{n-1} A^k v_{n-1} \end{array}$$

$$= \underbrace{\underbrace{\gamma_0 \lambda_0^k v_0 + \gamma_1 \lambda_1^k v_1 + \cdots + \gamma_{n-1} \lambda_{n-1}^k v_{n-1}}\left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1} \end{array} \right) \left(\begin{array}{cccc} \lambda_0^k & 0 & \cdots & 0 \\ 0 & \lambda_1^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1}^k \end{array} \right) \left(\begin{array}{c} \gamma_0 \\ \gamma_1 \\ \cdots \\ \gamma_{n-1} \end{array} \right)}_{V \Lambda^k c}$$

Now, $\boxed{\text{if}}$ $\lambda_0 = 1$, then $|\lambda_j| < 1$ for $j > 0$ and hence

$$\lim_{k\to\infty} x^{(k)} = \underbrace{\underbrace{\underbrace{\underbrace{\lim_{k\to\infty}\left(\gamma_0 v_0 + \gamma_1 \lambda_1^k v_1 + \cdots + \gamma_{n-1} \lambda_{n-1}^k v_{n-1}\right)} \lim_{k\to\infty} \left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1} \end{array} \right) \left(\begin{array}{cccc} 1 & 0 & \cdots & 0 \\ 0 & \lambda_1^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1}^k \end{array} \right) \left(\begin{array}{c} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-1} \end{array} \right)} \left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1} \end{array} \right) \left(\begin{array}{cccc} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{array} \right) \left(\begin{array}{c} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-1} \end{array} \right)} \left(\begin{array}{cccc} v_0 & 0 & \cdots & 0 \end{array} \right) \left(\begin{array}{c} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-1} \end{array} \right)}_{\gamma_0 v_0} = \gamma_0 v_0$$

which means that $x^{(k)}$ eventually starts pointing towards the direction of v_0, the eigenvector associated with the eigenvalue that is largest in magnitude. (Well, as long as $\gamma_0 \neq 0$.)

Homework 12.4.2.1 Let $A \in \mathbb{R}^{n \times n}$ and $\mu \neq 0$ be a scalar. Then $\lambda \in \Lambda(A)$ if and only if $\lambda/\mu \in \Lambda(\frac{1}{\mu}A)$.

True/False

☛ SEE ANSWER

What this last exercise shows is that if $\lambda_0 \neq 1$, then we can instead iterate with the matrix $\frac{1}{\lambda_0}A$, in which case

$$1 = \frac{\lambda_0}{\lambda_0} > \left|\frac{\lambda_1}{\lambda_0}\right| \geq \cdots \geq \left|\frac{\lambda_{n-1}}{\lambda_0}\right|.$$

The iteration then becomes

$$\begin{aligned} x^{(1)} &= \frac{1}{\lambda_0} A x^{(0)} \\ x^{(2)} &= \frac{1}{\lambda_0} A x^{(1)} \\ x^{(3)} &= \frac{1}{\lambda_0} A x^{(2)} \\ &\vdots \end{aligned}$$

The following algorithm accomplishes this

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  $x^{(k+1)} := Ax^{(k)}/\lambda_0$
endfor

It is not hard to see that then

$$\lim_{k\to\infty} x^{(k)} = \underbrace{\underbrace{\underbrace{\underbrace{\lim_{k\to\infty}\left(\gamma_0\left(\frac{\lambda_0}{\lambda_0}\right)^k v_0 + \gamma_1\left(\frac{\lambda_1}{\lambda_0}\right)^k v_1 + \cdots + \gamma_{n-1}\left(\frac{\lambda_{n-1}}{\lambda_0}\right)^k v_{n-1}\right)}_{\lim_{k\to\infty}\left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1}\end{array}\right)\left(\begin{array}{cccc} 1 & 0 & \cdots & 0 \\ 0 & (\lambda_1/\lambda_0)^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (\lambda_{n-1}/\lambda_0)^k \end{array}\right)\left(\begin{array}{c}\gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-1}\end{array}\right)}}_{\left(\begin{array}{cccc} v_0 & v_1 & \cdots & v_{n-1}\end{array}\right)\left(\begin{array}{cccc} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{array}\right)\left(\begin{array}{c}\gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-1}\end{array}\right)}}_{\left(\begin{array}{cccc} v_0 & 0 & \cdots & 0\end{array}\right)\left(\begin{array}{c}\gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-1}\end{array}\right)}}_{\gamma_0 v_0} = \gamma_0 v_0$$

So, it seems that we have an algorithm that always works as long as

$$|\lambda_0| > |\lambda_1| \geq \cdots \geq |\lambda_{n-1}|.$$

Unfortunately, we are cheating... If we knew λ_0, then we could simply compute the eigenvector by finding a vector in the null space of $A - \lambda_0 I$. The key insight now is that, in $x^{(k+1)} = Ax^{(k)}/\lambda_0$, dividing by λ_0 is merely meant to keep the vector $x^{(k)}$ from getting progressively larger (if $|\lambda_0| > 1$) or smaller (if $|\lambda_0| < 1$). We can alternatively simply make $x^{(k)}$ of length one at each step, and that will have the same effect without requiring λ_0:

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  $x^{(k+1)} := Ax^{(k)}$
  $x^{(k+1)} := x^{(k+1)}/\|x^{(k+1)}\|_2$
endfor

This last algorithm is known as the *Power Method* for finding an eigenvector associated with the largest eigenvalue (in magnitude).

Homework 12.4.2.2 We now walk you through a simple implementation of the Power Method, referring to files in directory `LAFF-2.0xM/Programming/Week12`.

We want to work with a matrix A for which we know the eigenvalues. Recall that a matrix A is diagonalizable if and only if there exists a nonsingular matrix V and diagonal matrix Λ such that $A = V\Lambda V^{-1}$. The diagonal elements of Λ then equal the eigenvalues of A and the columns of V the eigenvectors.

Thus, given eigenvalues, we can create a matrix A by creating a diagonal matrix with those eigenvalues on the diagonal and a random nonsingular matrix V, after which we can compute A to equal $V\Lambda V^{-1}$. This is accomplished by the function

[A, V] = CreateMatrixForEigenvalueProblem(eigs)

(see file `CreateMatrixForEigenvalueProblem.m`).

The script in `PowerMethodScript.m` then illustrates how the Power Method, starting with a random vector, computes an eigenvector corresponding to the eigenvalue that is largest in magnitude, and via the Rayleigh quotient (a way for computing an eigenvalue given an eigenvector that is discussed in the next unit) an approximation for that eigenvalue.

To try it out, in the Command Window type

```
>> PowerMethodScript
input a vector of eigenvalues. e.g.: [ 4; 3; 2; 1 ]
[ 4; 3; 2; 1 ]
```

The script for each step of the Power Method reports for the current iteration the length of the component orthogonal to the eigenvector associated with the eigenvalue that is largest in magnitude. If this component becomes small, then the vector lies approximately in the direction of the desired eigenvector. The Rayleigh quotient slowly starts to get close to the eigenvalue that is largest in magnitude. The slow convergence is because the ratio of the second to largest and the largest eigenvalue is not much smaller than 1.

Try some other distributions of eigenvalues. For example, `[ 4; 1; 0.5; 0.25]`, which should converge faster, or `[ 4; 3.9; 2; 1 ]`, which should converge much slower.

You may also want to try `PowerMethodScript2.m`, which illustrates what happens if there are two eigenvalues that are equal in value and both largest in magnitude (relative to the other eigenvalues).

12.4.3 In Preparation for this Week's Enrichment

In the last unit we introduce a practical method for computing an eigenvector associated with the largest eigenvalue in magnitude. This method is known as the Power Method. The next homework shows how to compute an eigenvalue associated with an eigenvector. Thus, the Power Method can be used to first approximate that eigenvector, and then the below result can be used to compute the associated eigenvalue.

Given $A \in \mathbb{R}^{n\times n}$ and nonzero vector $x \in \mathbb{R}^n$, the scalar $x^T Ax/x^T x$ is known as the *Rayleigh quotient*.

Homework 12.4.3.1 Let $A \in \mathbb{R}^{n\times n}$ and x equal an eigenvector of A. Assume that x is real valued as is the eigenvalue λ with $Ax = \lambda x$.

$\lambda = \frac{x^T Ax}{x^T x}$ is the eigenvalue associated with the eigenvector x.

Always/Sometimes/Never

☛ SEE ANSWER

Notice that we are carefully avoiding talking about complex valued eigenvectors. The above results can be modified for the case where x is an eigenvector associated with a complex eigenvalue and the case where A itself is complex valued. However, this goes beyond the scope of this course.

The following result allows the Power Method to be extended so that it can be used to compute the eigenvector associated with the smallest eigenvalue (in magnitude). The new method is called the Inverse Power Method and is discussed in this week's enrichment section.

Homework 12.4.3.2 Let $A \in \mathbb{R}^{n \times n}$ be nonsingular, $\lambda \in \Lambda(A)$, and $Ax = \lambda x$. Then $A^{-1}x = \frac{1}{\lambda}x$.

True/False

☛ SEE ANSWER

The Inverse Power Method can be accelerated by "shifting" the eigenvalues of the matrix, as discussed in this week's enrichment, yielding the Rayleigh Quotient Iteration. The following exercise prepares the way.

Homework 12.4.3.3 Let $A \in \mathbb{R}^{n \times n}$ and $\lambda \in \Lambda(A)$. Then $(\lambda - \mu) \in \Lambda(A - \mu I)$.

True/False

☛ SEE ANSWER

12.5 Enrichment

12.5.1 The Inverse Power Method

The Inverse Power Method exploits a property we established in Unit 12.3.4: If A is nonsingular and $\lambda \in \Lambda(A)$ then $1/\lambda \in \Lambda(A^{-1})$.

Again, let's make some assumptions. Given nonsingular $A \in \mathbb{R}^{n \times n}$,

- Let $\lambda_0, \lambda_1, \ldots, \lambda_{n-2}, \lambda_{n-1} \in \Lambda(A)$. We list eigenvalues that have algebraic multiplicity k multiple (k) times in this list.
- Let us assume that $|\lambda_0| \geq |\lambda_1| \geq \cdots \geq |\lambda_{n-2}| > |\lambda_{n-1}| > 0$. This implies that λ_{n-1} is real, since complex eigenvalues come in conjugate pairs and hence there would have been two eigenvalues with equal smallest magnitude. It also means that there is a real valued eigenvector associated with λ_{n-1}.
- Let us assume that $A \in \mathbb{R}^{n \times n}$ is diagonalizable so that

$$A = V \Lambda V^{-1} = \left(\begin{array}{ccccc} v_0 & v_1 & \cdots & v_{n-2} & v_{n-1} \end{array} \right) \left(\begin{array}{ccccc} \lambda_0 & 0 & \cdots & 0 & 0 \\ 0 & \lambda_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-2} & 0 \\ 0 & 0 & \cdots & 0 & \lambda_{n-1} \end{array} \right) \left(\begin{array}{ccccc} v_0 & v_1 & \cdots & v_{n-2} & v_{n-1} \end{array} \right)^{-1}.$$

  This means that v_i is an eigenvector associated with λ_i.

These assumptions set the stage.

Now, we again start with some vector $x^{(0)} \in \mathbb{R}^n$. Since V is nonsingular, the vectors $v_0, \ldots, v_{n-1}$ form a linearly independent bases for $\mathbb{R}^n$. Hence,

$$x^{(0)} = \gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-2} v_{n-2} + \gamma_{n-1} v_{n-1} = \left(\begin{array}{ccccc} v_0 & v_1 & \cdots & v_{n-2} & v_{n-1} \end{array} \right) \left(\begin{array}{c} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1} \end{array} \right) = Vc.$$

Now, we generate

$$\begin{array}{rcl} x^{(1)} & = & A^{-1}x^{(0)} \\ x^{(2)} & = & A^{-1}x^{(1)} \\ x^{(3)} & = & A^{-1}x^{(2)} \\ & \vdots & \end{array}$$

The following algorithm accomplishes this

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  Solve $Ax^{(k+1)} := x^{(k)}$
endfor

(In practice, one would probably factor A once, and reuse the factors for the solve.) Notice that then

$$x^{(k)} = A^{-1}x^{(k-1)} = (A^{-1})^2 x^{(k-2)} = \cdots = (A^{-1})^k x^{(0)}.$$

But then

$$\begin{aligned}
(A^{-1})^k x^{(0)} &= (A^{-1})^k (\underbrace{\gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-2} v_{n-2} + \gamma_{n-1} v_{n-1}}_{Vc}) \\
&= (A^{-1})^k \gamma_0 v_0 + (A^{-1})^k \gamma_1 v_1 + \cdots + (A^{-1})^k \gamma_{n-2} v_{n-2} + (A^{-1})^k \gamma_{n-1} v_{n-1} \\
&= \gamma_0 (A^{-1})^k v_0 + \gamma_1 (A^{-1})^k v_1 + \cdots + \gamma_{n-2} (A^{-1})^k v_{n-2} + \gamma_{n-1} (A^{-1})^k v_{n-1} \\
&= \underbrace{\underbrace{\gamma_0 \left(\frac{1}{\lambda_0}\right)^k v_0 + \gamma_1 \left(\frac{1}{\lambda_1}\right)^k v_1 + \cdots + \gamma_{n-2} \left(\frac{1}{\lambda_{n-2}}\right)^k v_{n-2} + \gamma_{n-1} \left(\frac{1}{\lambda_{n-1}}\right)^k v_{n-1}}_{\left(\begin{array}{cccc} v_0 & \cdots & v_{n-2} & v_{n-1} \end{array}\right) \left(\begin{array}{cccc} \left(\frac{1}{\lambda_0}\right)^k & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \left(\frac{1}{\lambda_{n-2}}\right)^k & 0 \\ 0 & \cdots & 0 & \left(\frac{1}{\lambda_{n-1}}\right)^k \end{array}\right) \left(\begin{array}{c} \gamma_0 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1} \end{array}\right)}}_{V(\Lambda^{-1})^k c}
\end{aligned}$$

Now, $\boxed{\textbf{if}}$ $\lambda_{n-1} = 1$, then $\left|\frac{1}{\lambda_j}\right| < 1$ for $j < n-1$ and hence

$$\lim_{k\to\infty} x^{(k)} = \underbrace{\underbrace{\underbrace{\underbrace{\lim_{k\to\infty}\left(\gamma_0 \left(\frac{1}{\lambda_0}\right)^k v_0 + \cdots + \gamma_{n-2} \left(\frac{1}{\lambda_{n-2}}\right)^k v_{n-2} + \gamma_{n-1} v_{n-1}\right)}_{\lim_{k\to\infty}\left(\begin{array}{cccc} v_0 & \cdots & v_{n-2} & v_{n-1} \end{array}\right) \left(\begin{array}{cccc} \left(\frac{1}{\lambda_0}\right)^k & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \left(\frac{1}{\lambda_{n-2}}\right)^k & 0 \\ 0 & \cdots & 0 & 1 \end{array}\right) \left(\begin{array}{c} \gamma_0 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1} \end{array}\right)}}_{\left(\begin{array}{cccc} v_0 & \cdots & v_{n-2} & v_{n-1} \end{array}\right) \left(\begin{array}{cccc} 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 1 \end{array}\right) \left(\begin{array}{c} \gamma_0 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1} \end{array}\right)}}_{\left(\begin{array}{cccc} 0 & \cdots & 0 & v_{n-1} \end{array}\right) \left(\begin{array}{c} \gamma_0 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1} \end{array}\right)}}_{\gamma_{n-1} v_{n-1}} = \gamma_{n-1} v_{n-1}$$

which means that $x^{(k)}$ eventually starts pointing towards the direction of v_{n-1}, the eigenvector associated with the eigenvalue that is smallest in magnitude. (Well, as long as $\gamma_{n-1} \neq 0$.)

Similar to before, we can instead iterate with the matrix $\lambda_{n-1}A^{-1}$, in which case

$$\left|\frac{\lambda_{n-1}}{\lambda_0}\right| \le \cdots \le \left|\frac{\lambda_{n-1}}{\lambda_{n-2}}\right| < \left|\frac{\lambda_{n-1}}{\lambda_{n-1}}\right| = 1.$$

The iteration then becomes

$$\begin{array}{rcl} x^{(1)} & = & \lambda_{n-1}A^{-1}x^{(0)} \\ x^{(2)} & = & \lambda_{n-1}A^{-1}x^{(1)} \\ x^{(3)} & = & \lambda_{n-1}A^{-1}x^{(2)} \\ & \vdots & \end{array}$$

The following algorithm accomplishes this

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  Solve $Ax^{(k+1)} := x^{(k)}$
  $x^{(k+1)} := \lambda_{n-1}x^{(k+1)}$
endfor

It is not hard to see that then

$$\lim_{k\to\infty} x^{(k)} = \underbrace{\lim_{k\to\infty}\left(\gamma_0\left(\frac{\lambda_{n-1}}{\lambda_0}\right)^k v_0 + \cdots + \gamma_{n-2}\left(\frac{\lambda_{n-1}}{\lambda_{n-2}}\right)^k v_{n-2} + \gamma_{n-1}v_{n-1}\right)}_{\underbrace{\lim_{k\to\infty}\left(\begin{array}{cccc} v_0 & \cdots & v_{n-2} & v_{n-1}\end{array}\right)\left(\begin{array}{cccc} (\lambda_{n-1}/\lambda_0)^k & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & (\lambda_{n-1}/\lambda_{n-2})^k & 0 \\ 0 & \cdots & 0 & 1 \end{array}\right)\left(\begin{array}{c} \gamma_0 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1}\end{array}\right)}_{\underbrace{\left(\begin{array}{cccc} v_0 & \cdots & v_{n-2} & v_{n-1}\end{array}\right)\left(\begin{array}{cccc} 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 1 \end{array}\right)\left(\begin{array}{c} \gamma_0 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1}\end{array}\right)}_{\underbrace{\left(\begin{array}{cccc} 0 & \cdots & 0 & v_{n-1}\end{array}\right)\left(\begin{array}{c} \gamma_0 \\ \vdots \\ \gamma_{n-2} \\ \gamma_{n-1}\end{array}\right)}_{\gamma_{n-1}v_{n-1}}}}} = \gamma_{n-1}v_{n-1}$$

So, it seems that we have an algorithm that always works as long as

$$|\lambda_0| \ge \cdots \ge |\lambda_{n-1}| > |\lambda_{n-1}|.$$

Again, we are cheating... If we knew λ_{n-1}, then we could simply compute the eigenvector by finding a vector in the null space of $A - \lambda_{n-1}I$. Again, the key insight is that, in $x^{(k+1)} = \lambda_{n-1}Ax^{(k)}$, multiplying by λ_{n-1} is merely meant to keep the vector $x^{(k)}$ from getting progressively larger (if $|\lambda_{n-1}| < 1$) or smaller (if $|\lambda_{n-1}| > 1$). We can alternatively simply make $x^{(k)}$ of length one at each step, and that will have the same effect without requiring λ_{n-1}:

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  Solve $Ax^{(k+1)} := x^{(k)}$
  $x^{(k+1)} := x^{(k+1)}/\|x^{(k+1)}\|_2$
endfor

This last algorithm is known as the *Inverse Power Method* for finding an eigenvector associated with the smallest eigenvalue (in magnitude).

Homework 12.5.1.1 The script in `InversePowerMethodScript.m` illustrates how the Inverse Power Method, starting with a random vector, computes an eigenvector corresponding to the eigenvalue that is smallest in magnitude, and (via the Rayleigh quotient) an approximation for that eigenvalue.
To try it out, in the Command Window type

```
>> InversePowerMethodScript
input a vector of eigenvalues. e.g.: [ 4; 3; 2; 1 ]
[ 4; 3; 2; 1 ]
```

If you compare the script for the Power Method with this script, you notice that the difference is that we now use A^{-1} instead of A. To save on computation, we compute the LU factorization once, and solve $LUz = x$, overwriting x with z, to update $x := A^{-1}x$. You will notice that for this distribution of eigenvalues, the Inverse Power Method converges faster than the Power Method does.
Try some other distributions of eigenvalues. For example, `[ 4; 3; 1.25; 1 ]`, which should converge slower, or `[ 4; 3.9; 3.8; 1 ]`, which should converge faster.

Now, it is possible to accelerate the Inverse Power Method if one has a good guess of λ_{n-1}. The idea is as follows: Let μ be close to λ_{n-1}. Then we know that $(A-\mu I)x = (\lambda-\mu)x$. Thus, an eigenvector of A is an eigenvector of A^{-1} is an eigenvector of $A-\mu I$ is an eigenvector of $(A-muI)^{-1}$. Now, if μ is close to λ_{n-1}, then (hopefully)

$$|\lambda_0-\mu| \geq |\lambda_1-\mu| \geq \cdots \geq |\lambda_{n-2}-\mu| > |\lambda_{n-1}-\mu|.$$

The important thing is that if, as before,

$$x^{(0)} = \gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-2} v_{n-2} + \gamma_{n-1} v_{n-1}$$

where v_j equals the eigenvector associated with λ_j, then

$$\begin{array}{rcl} x^{(k)} & = & (\lambda_{n-1}-\mu)(A-\mu I)^{-1}x^{(k-1)} = \cdots = (\lambda_{n-1}-\mu)^k((A-\mu I)^{-1})^k x^{(0)} = \\ & = & \gamma_0(\lambda_{n-1}-\mu)^k((A-\mu I)^{-1})^k v_0 + \gamma_1(\lambda_{n-1}-\mu)^k((A-\mu I)^{-1})^k v_1 + \cdots \\ & & \quad + \gamma_{n-2}(\lambda_{n-1}-\mu)^k((A-\mu I)^{-1})^k v_{n-2} + \gamma_{n-1}(\lambda_{n-1}-\mu)^k((A-\mu I)^{-1})^k v_{n-1} \\ & = & \gamma_0 \left|\dfrac{\lambda_{n-1}-\mu}{\lambda_0-\mu}\right|^k v_0 + \gamma_1 \left|\dfrac{\lambda_{n-1}-\mu}{\lambda_1-\mu}\right|^k v_1 + \cdots + \gamma_{n-2}\left|\dfrac{\lambda_{n-1}-\mu}{\lambda_{n-2}-\mu}\right|^k v_{n-2} + \gamma_{n-1}v_{n-1} \end{array}$$

Now, how fast the terms involving $v_0, \ldots, v_{n-2}$ approx zero (become negligible) is dictated by the ratio

$$\left|\frac{\lambda_{n-1}-\mu}{\lambda_{n-2}-\mu}\right|.$$

Clearly, this can be made arbitrarily small by picking arbitrarily close to λ_{n-1}. Of course, that would require knowning λ_{n-1}...

The practical algorithm for this is given by

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  Solve $(A-\mu I)x^{(k+1)} := x^{(k)}$
  $x^{(k+1)} := x^{(k+1)}/\|x^{(k+1)}\|_2$
endfor

which is referred to as the Shifted Inverse Power Method. Obviously, we would want to only factor $A-\mu I$ once.

Homework 12.5.1.2 The script in `ShiftedInversePowerMethodScript.m` illustrates how shifting the matrix can improve how fast the Inverse Power Method, starting with a random vector, computes an eigenvector corresponding to the eigenvalue that is smallest in magnitude, and (via the Rayleigh quotient) an approximation for that eigenvalue.
To try it out, in the Command Window type

```
>> ShiftedInversePowerMethodScript
input a vector of eigenvalues. e.g.: [ 4; 3; 2; 1 ]
[ 4; 3; 2; 1 ]

<bunch of output>

enter a shift to use: (a number close to the smallest eigenvalue) 0.9
```

If you compare the script for the Inverse Power Method with this script, you notice that the difference is that we now iterate with $(A - sigmaI)^{-1}$, where σ is the shift, instead of A. To save on computation, we compute the LU factorization of $A - \sigma I$ once, and solve $LUz = x$, overwriting x with z, to update $x := (A^{-1} - \sigma I)x$. You will notice that if you pick the shift close to the smallest eigenvalue (in magnitude), this Shifted Inverse Power Method converges faster than the Inverse Power Method does. Indeed, pick the shift very close, and the convergence is very fast. See what happens if you pick the shift exactly equal to the smallest eigenvalue. See what happens if you pick it close to another eigenvalue.

12.5.2 The Rayleigh Quotient Iteration

In the previous unit, we explained that the Shifted Inverse Power Method converges quickly if only we knew a scalar μ close to λ_{n-1}.

The observation is that $x^{(k)}$ eventually approaches v_{n-1}. If we knew v_{n-1} but not λ_{n-1}, then we could compute the Rayleigh quotient:

$$\lambda_{n-1} = \frac{v_{n-1}^T A v_{n-1}}{v_{n-1}^T v_{n-1}}.$$

But we know an approximation of v_{n-1} (or at least its direction) and hence can pick

$$\mu = \frac{x^{(k)T} A x^{(k)}}{x^{(k)T} x^{(k)}} \approx \lambda_{n-1}$$

which will become a progressively better approximation to λ_{n-1} as k increases.

This then motivates the Rayleigh Quotient Iteration:

for $k = 0, \ldots,$ until $x^{(k)}$ doesn't change (much) anymore
  $\mu := \frac{x^{(k)T} A x^{(k)}}{x^{(k)T} x^{(k)}}$
  Solve $(A - \mu I)x^{(k+1)} := x^{(k)}$
  $x^{(k+1)} := x^{(k+1)} / \|x^{(k+1)}\|_2$
endfor

Notice that if $x^{(0)}$ has length one, then we can compute $\mu := x^{(k)T} A x^{(k)}$ instead, since $x^{(k)}$ will always be of length one.

The disadvantage of the Rayleigh Quotient Iteration is that one cannot factor $(A - \mu I)$ once before starting the loop. The advantage is that it converges dazingly fast. Obviously "dazingly" is not a very precise term. Unfortunately, quantifying how fast it converges is beyond this enrichment.

Homework 12.5.2.1 The script in `RayleighQuotientIterationScript.m` illustrates how shifting the matrix by the Rayleigh Quotient can greatly improve how fast the Shifted Inverse Power Method, starting with a random vector, computes an eigenvector. It could be that the random vector is close to an eigenvector associated with any of the eigenvalues, in which case the method will start converging towards an eigenvector associated with that eigenvalue. Pay close attention to how many digit are accurate from one iteration to the next.
To try it out, in the Command Window type

```
>> RayleighQuotientIterationScript
input a vector of eigenvalues. e.g.: [ 4; 3; 2; 1 ]
[ 4; 3; 2; 1 ]
```

12.5.3 More Advanced Techniques

The Power Method and its variants are the bases of algorithms that compute all eigenvalues and eigenvectors of a given matrix. Details, presented with notation similar to what you have learned in this class, can be found in our graduate level course "Advanced Linear Algebra: Foundations to Frontiers" (ALAFF). For details, visit ulaff.net.

12.6 Wrap Up

12.6.1 Homework

No additional homework this week.

12.6.2 Summary

The algebraic eigenvalue problem

The *algebraic eigenvalue problem* is given by

$$Ax = \lambda x.$$

where $A \in \mathbb{R}^{n \times n}$ is a square matrix, λ is a scalar, and x is a nonzero vector.

- If $x \neq 0$, then λ is said to be an *eigenvalue* and x is said to be an eigenvector associated with the eigenvalue λ.
- The tuple $(\lambda, x\ )$ is said to be an *eigenpair*.
- The set of all vectors that satisfy $Ax = \lambda x$ is a subspace.

Equivalent statements:

- $Ax = \lambda x,$, where $x \neq 0$.
- $(A - \lambda I)x = 0$, where $x \neq 0$.
  This is a matter of fractoring' x out.
- $A - \lambda I$ is singular.
- $\mathcal{N}(A - \lambda I)$ contains a nonzero vector x.
- $\dim(\mathcal{N}(A - \lambda I)) > 0$.
- $\det(A - \lambda I) = 0$.

If we find a vector $x \neq 0$ such that $Ax = \lambda x$, it is certainly not unique.

- For any scalar α, $A(\alpha x) = \lambda(\alpha x)$ also holds.
- If $Ax = \lambda x$ and $Ay = \lambda y$, then $A(x + y) = Ax + Ay = \lambda x + \lambda y = \lambda(x + y)$.

We conclude that the set of all vectors x that satisfy $Ax = \lambda x$ is a subspace.

Simple cases

- The eigenvalue of the zero matrix is the scalar $\lambda = 0$. All nonzero vectors are eigenvectors.
- The eigenvalue of the identity matrix is the scalar $\lambda = 1$. All nonzero vectors are eigenvectors.
- The eigenvalues of a diagonal matrix are its elements on the diagonal. The unit basis vectors are eigenvectors.
- The eigenvalues of a triangular matrix are its elements on the diagonal.
- The eigenvalues of a 2×2 matrix can be found by finding the roots of $p_2(\lambda) = \det(A - \lambda I) = 0$.
- The eigenvalues of a 3×3 matrix can be found by finding the roots of $p_3(\lambda) = \det(A - \lambda I) = 0$.

For 2×2 matrices, the following steps compute the eigenvalues and eigenvectors:

- Compute
$$\det\left(\begin{pmatrix} (\alpha_{0,0} - \lambda) & \alpha_{0,1} \\ \alpha_{1,0} & (\alpha_{1,1} - \lambda) \end{pmatrix}\right) = (\alpha_{0,0} - \lambda)(\alpha_{1,1} - \lambda) - \alpha_{0,1}\alpha_{1,0}.$$
- Recognize that this is a second degree polynomial in λ.
- It is called the *characteristic polynomial* of the matrix A, $p_2(\lambda)$.
- Compute the coefficients of $p_2(\lambda)$ so that
$$p_2(\lambda) = -\lambda^2 + \beta\lambda + \gamma.$$
- Solve
$$-\lambda^2 + \beta\lambda + \gamma = 0$$
for its roots. You can do this either by examination, or by using the quadratic formula:
$$\lambda = \frac{-\beta \pm \sqrt{\beta^2 + 4\gamma}}{-2}.$$
- For each of the roots, find an eigenvector that satisfies
$$\begin{pmatrix} (\alpha_{0,0} - \lambda) & \alpha_{0,1} \\ \alpha_{1,0} & (\alpha_{1,1} - \lambda) \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
The easiest way to do this is to subtract the eigenvalue from the diagonal, set one of the components of x to 1, and then solve for the other component.
- Check your answer! It is a matter of plugging it into $Ax = \lambda x$ and seeing if the computed λ and x satisfy the equation.

General case

Theorem 12.12 *A matrix $A \in \mathbb{R}^{n \times n}$ is nonsingular if and only if* $\det(A) \neq 0$.

Theorem 12.13 *Given $A \in \mathbb{R}^{n \times n}$,*
$$p_n(\lambda) = \det(A - \lambda I) = \lambda^n + \gamma_{n-1}\lambda^{n-1} + \cdots + \gamma_1\lambda + \gamma_0.$$
for some coefficients $\gamma_1, \ldots, \gamma_{n-1} \in \mathbb{R}$.

Definition 12.14 *Given $A \in \mathbb{R}^{n \times n}$, $p_n(\lambda) = \det(A - \lambda I)$ is called the characteristic polynomial.*

Theorem 12.15 *Scalar λ satisfies $Ax = \lambda x$ for some nonzero vector x if and only if* $\det(A - \lambda I) = 0$.

- The characteristic polynomial of $A \in \mathbb{R}^{n \times n}$ is given by
$$p_n(\lambda) = \det(A - \lambda I) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n.$$

- Since $p_n(\lambda)$ is an nth degree polynomial, it has n roots, counting multiplicity. Thus, matrix A has n eigenvalues, counting multiplicity.
  - Let k equal the number of *distinct* roots of $p_n(\lambda)$. Clearly, $k \leq n$. Clearly, matrix A then has k distinct eigenvalues.
  - The set of all roots of $p_n(\lambda)$, which is the set of all eigenvalues of A, is denoted by $\Lambda(A)$ and is called the *spectrum* of matrix A.
  - The characteristic polynomial can be factored as

$$p_n(\lambda) = \det(A - \lambda I) = (\lambda - \lambda_0)^{n_0}(\lambda - \lambda_1)^{n_1} \cdots (\lambda - \lambda_{k-1})^{n_{k-1}},$$

    where $n_0 + n_1 + \cdots + n_{k-1} = n$ and n_j is the root λ_j, which is known as that (algebraic) multiplicity of eigenvalue λ_j.

- If $A \in \mathbb{R}^{n \times n}$, then the coefficients of the characteristic polynomial are real ($\gamma_0, \ldots, \gamma_{n-1} \in \mathbb{R}$), but
  - Some or all of the roots/eigenvalues may be complex valued and
  - Complex roots/eigenvalues come in "conjugate pairs": If $\lambda = \mathrm{Re}(\lambda) + i\mathrm{Im}(\lambda)$ is a root/eigenvalue, so is $\lambda = \mathrm{Re}(\lambda) - i\mathrm{Im}(\lambda)$

Galois theory tells us that for $n \geq 5$, roots of arbitrary $p_n(\lambda)$ cannot be found in a finite number of computations.
For every $n$the degree polynomial

$$p_n(\lambda) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n,$$

there exists a matrix, C, called the companion matrix that has the property that

$$p_n(\lambda) = \det(C - \lambda I) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n.$$

In particular, the matrix

$$C = \begin{pmatrix} -\gamma_{n-1} & -\gamma_{n-2} & \cdots & -\gamma_1 & -\gamma_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}$$

is the companion matrix for $p_n(\lambda)$:

$$p_n(\lambda) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n = \det(\begin{pmatrix} -\gamma_{n-1} & -\gamma_{n-2} & \cdots & -\gamma_1 & -\gamma_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} - \lambda I).$$

Diagonalization

Theorem 12.16 *Let $A \in \mathbb{R}^{n \times n}$. Then there exists a nonsingular matrix X such that $X^{-1}AX = \Lambda$ if and only if A has n linearly independent eigenvectors.*

If X is invertible (nonsingular, has linearly independent columns, etc.), then the following are equivalent

$$\begin{array}{rcl} X^{-1}\,A\,X & = & \Lambda \\ A\,X & = & X\,\Lambda \\ A & = & X\,\Lambda\,X^{-1} \end{array}$$

If Λ is in addition diagonal, then the diagonal elements of Λ are eigenvalues of A and the columns of X are eigenvectors of A.

Defective matrices

It is *not* the case that for every $A \in \mathbb{R}^{n\times n}$ there is a nonsingular matrix $X \in \mathbb{R}^{n\times n}$ such that $X^{-1}AX = \Lambda$, where Λ is diagonal. In general, the $k \times k$ matrix $J_k(\lambda)$ given by

$$J_k(\lambda) = \begin{pmatrix} \lambda & 1 & 0 & \cdots & 0 & 0 \\ 0 & \lambda & 1 & \cdots & 0 & 0 \\ 0 & 0 & \lambda & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda & 1 \\ 0 & 0 & 0 & \cdots & 0 & \lambda \end{pmatrix}$$

has eigenvalue λ of algebraic multiplicity k, but *geometric multiplicity* one (it has only one linearly independent eigenvector). Such a matrix is known as a Jordan block.

Definition 12.17 *The geometric multiplicity of an eigenvalue λ equals the number of linearly independent eigenvectors that are associated with λ.*

Theorem 12.18 *Let $A \in \mathbb{R}^{n\times n}$. Then there exists a nonsingular matrix $X \in \mathbb{R}^{n\times n}$ such that $A = XJX^{-1}$, where*

$$J = \begin{pmatrix} J_{k_0}(\lambda_0) & 0 & 0 & \cdots & 0 \\ 0 & J_{k_1}(\lambda_1) & 0 & \cdots & 0 \\ 0 & 0 & J_{k_2}(\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & J_{k_{m-1}}(\lambda_{m-1}) \end{pmatrix}$$

where each $J_{k_j}(\lambda_j)$ is a Jordan block of size $k_j \times k_j$.

The factorization $A = XJX^{-1}$ is known as the Jordan Canonical Form of matrix A.

In the above theorem

- It is *not* the case that $\lambda_0, \lambda_1, \ldots, \lambda_{m-1}$ are distinct. If λ_j appears in multiple Jordan blocks, the number of Jordan blocks in which λ_j appears equals the geometric multiplicity of λ_j (and the number of linearly independent eigenvectors associated with λ_j).
- The sum of the sizes of the blocks in which λ_j as an eigenvalue appears equals the algebraic multiplicity of λ_j.
- If each Jordan block is 1×1, then the matrix is diagonalized by matrix X.
- If any of the blocks is not 1×1, then the matrix cannot be diagonalized.

Properties of eigenvalues and eigenvectors

Definition 12.19 *Given $A \in \mathbb{R}^{n\times n}$ and nonzero vector $x \in \mathbb{R}^n$, the scalar x^TAx/x^Tx is known as the Rayleigh quotient.*

Theorem 12.20 *Let $A \in \mathbb{R}^{n\times n}$ and x equal an eigenvector of A. Assume that x is real valued as is the eigenvalue λ with $Ax = \lambda x$. Then $\lambda = \frac{x^TAx}{x^Tx}$ is the eigenvalue associated with the eigenvector x.*

Theorem 12.21 *Let $A \in \mathbb{R}^{n\times n}$, β be a scalar, and $\lambda \in \Lambda(A)$. Then $(\beta\lambda) \in \Lambda(\beta A)$.*

Theorem 12.22 *Let $A \in \mathbb{R}^{n\times n}$ be nonsingular, $\lambda \in \Lambda(A)$, and $Ax = \lambda x$. Then $A^{-1}x = \frac{1}{\lambda}x$.*

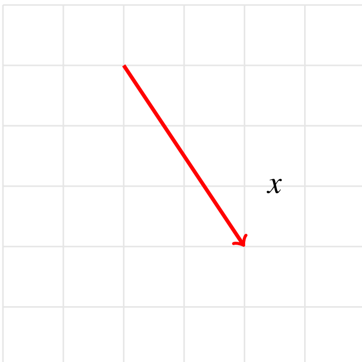
Theorem 12.23 *Let $A \in \mathbb{R}^{n\times n}$ and $\lambda \in \Lambda(A)$. Then $(\lambda - \mu) \in \Lambda(A - \mu I)$.*

Answers

Week 1: Vectors in Linear Algebra (Answers)

Notation

Homework 1.2.1.1 Consider the following picture:



Using the grid for units,

(a) $x = \begin{pmatrix} -2 \\ -3 \end{pmatrix}$ (b) $x = \begin{pmatrix} 3 \\ -2 \end{pmatrix}$

(c) $x = \begin{pmatrix} 2 \\ -3 \end{pmatrix}$ (d) $x = \begin{pmatrix} -3 \\ -2 \end{pmatrix}$

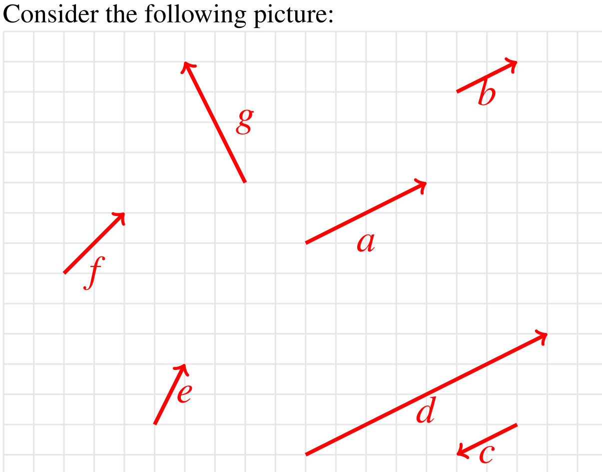
(e) None of these

Answer: (c)

☛ BACK TO TEXT

Homework 1.2.1.2

Consider the following picture:



Using the grid for units,

(a) $a = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$ (b) $b = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$

(c) $c = \begin{pmatrix} -2 \\ -1 \end{pmatrix}$ (d) $d = \begin{pmatrix} 8 \\ 4 \end{pmatrix}$

(e) $e = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ (f) $f = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$

(g) $g = \begin{pmatrix} -2 \\ 4 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.2.1.3 Write each of the following as a vector:

- The vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(-1,2)$ to point $(0,0)$.
- The vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(0,0)$ to point $(-1,2)$.
- The vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(-1,2,4)$ to point $(0,0,1)$.
- The vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(1,0,0)$ to point $(4,2,-1)$.

☛ BACK TO TEXT

Unit Basis Vectors

Homework 1.2.2.1 Which of the following is not a unit basis vector?

(a) $\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ (b) $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ (c) $\begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}$ (d) $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ (e) None of these are unit basis vectors.

☛ BACK TO TEXT

Equality (=), Assignment (:=), and Copy

Homework 1.3.1.1 Decide if the two vectors are equal.

- The vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(-1,2)$ to point $(0,0)$ and the vector represented geometrically in $\mathbb{R}^2$ by an arrow from point $(1,-2)$ to point $(2,-1)$ are equal.

  True/False

- The vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(1,-1,2)$ to point $(0,0,0)$ and the vector represented geometrically in $\mathbb{R}^3$ by an arrow from point $(1,1,-2)$ to point $(0,2,-4)$ are equal.

  True/False

☛ BACK TO TEXT

Vector Addition

Homework 1.3.2.1 $\begin{pmatrix} -1 \\ 2 \end{pmatrix} + \begin{pmatrix} -3 \\ -2 \end{pmatrix} = \begin{pmatrix} -4 \\ 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.3.2.2 $\begin{pmatrix} -3 \\ -2 \end{pmatrix} + \begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} -4 \\ 0 \end{pmatrix}$

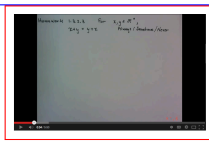
☛ BACK TO TEXT

Homework 1.3.2.3 For $x, y \in \mathbb{R}^n$,

$$x + y = y + x.$$

Always/Sometimes/Never

Answer:



☛ **YouTube**

☛ BACK TO TEXT

Homework 1.3.2.4 $\begin{pmatrix} -1 \\ 2 \end{pmatrix} + \left(\begin{pmatrix} -3 \\ -2 \end{pmatrix} + \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right) = \begin{pmatrix} -3 \\ 2 \end{pmatrix}$

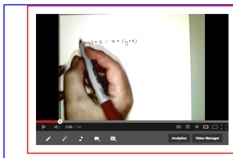
☛ BACK TO TEXT

Homework 1.3.2.5 $\left(\begin{pmatrix} -1 \\ 2 \end{pmatrix} + \begin{pmatrix} -3 \\ -2 \end{pmatrix} \right) + \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} -3 \\ 2 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.3.2.6 For $x, y, z \in \mathbb{R}^n$, $(x+y)+z = x+(y+z)$. Always/Sometimes/Never

Answer:



☛ YouTube

☛ BACK TO TEXT

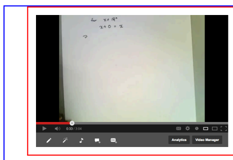
Homework 1.3.2.7 $\begin{pmatrix} -1 \\ 2 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.3.2.8 For $x \in \mathbb{R}^n$, $x + 0 = x$, where 0 is the zero vector of appropriate size.

Always/Sometimes/Never

Answer:



☛ YouTube

☛ BACK TO TEXT

Scaling

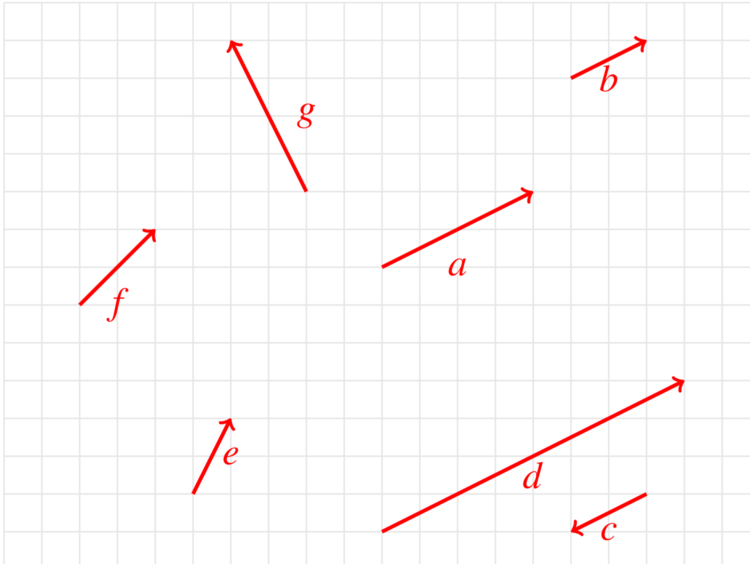
Homework 1.3.3.1 $\left(\begin{pmatrix} -1 \\ 2 \end{pmatrix} + \begin{pmatrix} -1 \\ 2 \end{pmatrix} \right) + \begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} -3 \\ 6 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.3.3.2 $3 \begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} -3 \\ 6 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.3.3.3 Consider the following picture:



Which vector equals $2a$?; $(1/2)a$? ; and $-(1/2)a$?

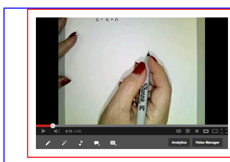
☛ BACK TO TEXT

Vector Subtraction

Homework 1.3.4.1 For $x \in \mathbb{R}^n$, $x - x = 0$.

Always/Sometimes/Never

Answer:



☛ **YouTube**

☛ BACK TO TEXT

Homework 1.3.4.2 For $x, y \in \mathbb{R}^n$, $x - y = y - x$.

Always/Sometimes/Never

Answer: Sometimes

Use the parallelogram method to represent both $x - y$ and $y - x$. Notice the resulting vectors are formed from the same diagonal of the parallelogram but their are opposite. Algebraically, you can come to the same conclusion by noticing that $x - y = (-1) \times (y - x)$.

It is equal ONLY if $x = y$! (This is why the answer is "sometimes" rather than "never".)

☛ BACK TO TEXT

Scaled Vector Addition

Homework 1.4.1.1 What is the cost of an axpy operation?

- How many memops?
- How many flops?

Answer:

The AXPY operation requires $3n+1$ memops and $2n$ flops. The reason is that α is only brought in from memory once and kept in a register for reuse. To fully understand this, you need to know a little bit about computer architecture. (Perhaps a video on this some day?)

- By combining the scaling and vector addition into one operation, there is the opportunity to reduce the number of memops that are incurred separately by the SCAL and ADD operations.
- "Among friends" we will say that the cost is $3n$ memops since the one extra memory operation (for bring α in from memory) is negligible.
- For those who understand "Big-O" notation, the cost of the AXPY operation is $O(n)$. However, we tend to want to be more exact than just saying $O(n)$. To us, the coefficient in front of n is important.

☛ BACK TO TEXT

Linear Combinations of Vectors

Homework 1.4.2.1

$$3\begin{pmatrix} 2 \\ 4 \\ -1 \\ 0 \end{pmatrix} + 2\begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 8 \\ 12 \\ -1 \\ 0 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 1.4.2.2

$$-3\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + 2\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + 4\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -3 \\ 2 \\ 4 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 1.4.2.3 Find α, β, γ such that

$$\alpha\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \beta\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \gamma\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \\ 3 \end{pmatrix}$$

$\alpha = 2$ $\beta = -1$ $\gamma = 3$

☛ BACK TO TEXT

Dot or Inner Product

Homework 1.4.3.1 $\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \text{Undefined}$

☛ BACK TO TEXT

Homework 1.4.3.2 $\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = 2$

☛ BACK TO TEXT

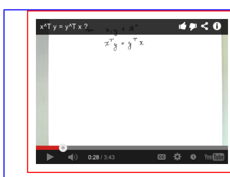
Homework 1.4.3.3 $\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} = 2$

☛ BACK TO TEXT

Homework 1.4.3.4 For $x, y \in \mathbb{R}^n$, $x^T y = y^T x$.

Always/Sometimes/Never

Answer:



☛ **YouTube**

☛ BACK TO TEXT

Homework 1.4.3.5 $\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \left(\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \right) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 3 \\ 7 \\ -3 \\ 5 \end{pmatrix} = 12$

☛ BACK TO TEXT

Homework 1.4.3.6 $\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} = 2 + 10 = 12$

☛ BACK TO TEXT

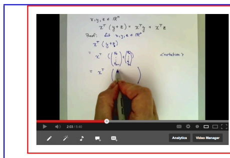
Homework 1.4.3.7 $\left(\begin{pmatrix} 2 \\ 5 \\ -6 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \right)^T \begin{pmatrix} 1 \\ 0 \\ 0 \\ 2 \end{pmatrix} =$

☛ BACK TO TEXT

Homework 1.4.3.8 For $x, y, z \in \mathbb{R}^n$, $x^T(y+z) = x^T y + x^T z$.

Always/Sometimes/Never

Answer:



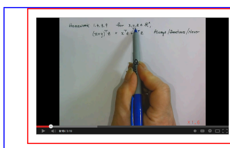
☛ YouTube

☛ BACK TO TEXT

Homework 1.4.3.9 For $x, y, z \in \mathbb{R}^n$, $(x+y)^T z = x^T z + y^T z$.

Always/Sometimes/Never

Answer:



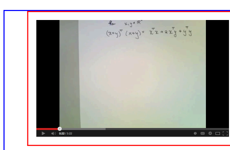
☛ YouTube

☛ BACK TO TEXT

Homework 1.4.3.10 For $x, y \in \mathbb{R}^n$, $(x+y)^T(x+y) = x^T x + 2x^T y + y^T y$.

Always/Sometimes/Never

Answer:



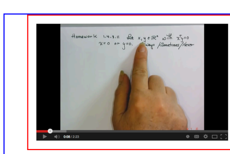
☛ YouTube

☛ BACK TO TEXT

Homework 1.4.3.11 Let $x, y \in \mathbb{R}^n$. When $x^T y = 0$, x or y is a zero vector.

Always/Sometimes/Never

Answer:



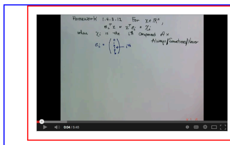
☛ YouTube

☛ BACK TO TEXT

Homework 1.4.3.12 For $x \in \mathbb{R}^n$, $e_i^T x = x^T e_i = \chi_i$, where χ_i equals the ith component of x.

Always/Sometimes/Never

Answer: ALWAYS.



☛ YouTube

Consider that $y^T x = \sum_{j=0}^{n-1} \psi_j \chi_j$. When $y = e_i$,

$$\psi_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}.$$

Hence

$$e_i^T x = \sum_{j=0}^{n-1} \psi_j \chi_j = \sum_{j=0}^{i-1} \psi_j \chi_j + \psi_i \chi_i + \sum_{j=i+1}^{n-1} \psi_j \chi_j = \sum_{j=0}^{i-1} 0 \times \chi_j + 1 \times \chi_i + \sum_{j=i+1}^{n-1} 0 \times \chi_j = \chi_i.$$

The fact that $x^T e_i = \chi_i$ is proved similarly.

☛ BACK TO TEXT

Homework 1.4.3.13 What is the cost of a dot product with vectors of size n?

Answer: Approximately $2n$ memops and $2n$ flops.

☛ BACK TO TEXT

Vector Length

Homework 1.4.4.1 Compute the lengths of the following vectors:

(a) $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ (b) $\begin{pmatrix} 1/2 \\ 1/2 \\ 1/2 \\ 1/2 \end{pmatrix}$ (c) $\begin{pmatrix} 1 \\ -2 \\ 2 \end{pmatrix}$ (d) $\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.4.4.2 Let $x \in \mathbb{R}^n$. The length of x is less than zero: $\|x\|_2 < 0$.

Always/Sometimes/Never

Answer: NEVER, since $\|x\|_2 = \sum_{i=0}^{n-1} \chi_i^2$, and the sum of squares is always positive.

☛ BACK TO TEXT

Homework 1.4.4.3 If x is a unit vector then x is a unit basis vector.

TRUE/FALSE

Answer: FALSE. A unit vector is any vector of length one (unit length). For example, the vector $\begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}$ has length one (check!) and is hence a unit vector. But it is not a unit basis vector.

☛ BACK TO TEXT

Homework 1.4.4.4 If x is a unit basis vector then x is a unit vector.

TRUE/FALSE

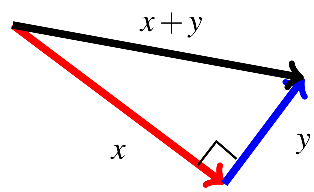
Answer: TRUE. A unit basis vector has unit length.

☛ BACK TO TEXT

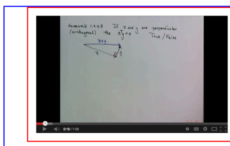
Homework 1.4.4.5 If x and y are perpendicular (orthogonal) then $x^T y = 0$.

TRUE/FALSE

Hint: Consider the picture



Answer:



☛ **YouTube**

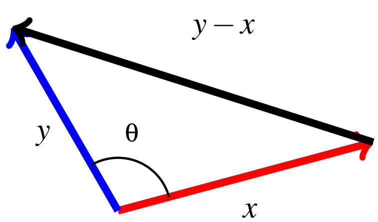
☛ BACK TO TEXT

Homework 1.4.4.6 Let $x, y \in \mathbb{R}^n$ be nonzero vectors and let the angle between them equal θ. Then

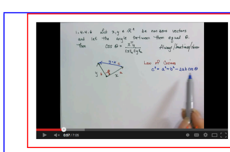
$$\cos\theta = \frac{x^T y}{\|x\|_2 \|y\|_2}.$$

Always/Sometimes/Never

Hint: Consider the picture and the "Law of Cosines" that you learned in high school. (Or look up this law!)



Answer:



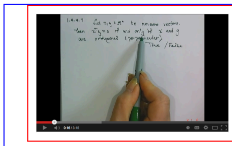
☛ **YouTube**

☛ BACK TO TEXT

Homework 1.4.4.7 Let $x, y \in \mathbb{R}^n$ be nonzero vectors. Then $x^T y = 0$ if and only if x and y are orthogonal (perpendicular).

True/False

Answer:



☛ YouTube

☛ BACK TO TEXT

Vector Functions

Homework 1.4.5.1 If $f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+\alpha \\ \chi_1+\alpha \\ \chi_2+\alpha \end{pmatrix}$, find

- $f(1, \begin{pmatrix} 6 \\ 2 \\ 3 \end{pmatrix}) = \begin{pmatrix} 6+1 \\ 2+1 \\ 3+1 \end{pmatrix} = \begin{pmatrix} 7 \\ 3 \\ 4 \end{pmatrix}$
- $f(\alpha, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}) = \begin{pmatrix} 0+\alpha \\ 0+\alpha \\ 0+\alpha \end{pmatrix} = \begin{pmatrix} \alpha \\ \alpha \\ \alpha \end{pmatrix}$
- $f(0, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+0 \\ \chi_1+0 \\ \chi_2+0 \end{pmatrix} = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$
- $f(\beta, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+\beta \\ \chi_1+\beta \\ \chi_2+\beta \end{pmatrix}$
- $\alpha f(\beta, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \alpha \begin{pmatrix} \chi_0+\beta \\ \chi_1+\beta \\ \chi_2+\beta \end{pmatrix} = \begin{pmatrix} \alpha(\chi_0+\beta) \\ \alpha(\chi_1+\beta) \\ \alpha(\chi_2+\beta) \end{pmatrix} = \begin{pmatrix} \alpha\chi_0+\alpha\beta \\ \alpha\chi_1+\alpha\beta \\ \alpha\chi_2+\alpha\beta) \end{pmatrix}$
- $f(\beta, \alpha \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = f(\beta, \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \\ \alpha\chi_2 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_0+\beta \\ \alpha\chi_1+\beta \\ \alpha\chi_2+\beta \end{pmatrix}$
- $f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = f(\alpha, \begin{pmatrix} \chi_0+\psi_0 \\ \chi_1+\psi_1 \\ \chi_2+\psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+\psi_0+\alpha \\ \chi_1+\psi_1+\alpha \\ \chi_2+\psi_2+\alpha \end{pmatrix}$
- $f(\alpha, \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + f(\alpha, \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0+\alpha \\ \chi_1+\alpha \\ \chi_2+\alpha \end{pmatrix} + \begin{pmatrix} \psi_0+\alpha \\ \psi_1+\alpha \\ \psi_2+\alpha \end{pmatrix} = \begin{pmatrix} \chi_0+\alpha+\psi_0+\alpha \\ \chi_1+\alpha+\psi_1+\alpha \\ \chi_2+\alpha+\psi_2+\alpha \end{pmatrix} = \begin{pmatrix} \chi_0+\psi_0+2\alpha \\ \chi_1+\psi_1+2\alpha \\ \chi_2+\psi_2+2\alpha \end{pmatrix}$

☛ BACK TO TEXT

Vector Functions that Map a Vector to a Vector

Homework 1.4.6.1 If $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + 1 \\ \chi_1 + 2 \\ \chi_2 + 3 \end{pmatrix}$, evaluate

- $f(\begin{pmatrix} 6 \\ 2 \\ 3 \end{pmatrix}) = \begin{pmatrix} 7 \\ 4 \\ 6 \end{pmatrix}$
- $f(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$
- $f(2\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} 2\chi_0 + 1 \\ 2\chi_1 + 2 \\ 2\chi_1 + 3 \end{pmatrix}$
- $2f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} 2\chi_0 + 2 \\ 2\chi_1 + 4 \\ 2\chi_1 + 6 \end{pmatrix}$
- $f(\alpha\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_0 + 1 \\ \alpha\chi_1 + 2 \\ \alpha\chi_1 + 3 \end{pmatrix}$
- $\alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \alpha(\chi_0 + 1) \\ \alpha(\chi_1 + 2) \\ \alpha(\chi_1 + 3) \end{pmatrix}$
- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \psi_0 + 1 \\ \chi_1 + \psi_1 + 2 \\ \chi_2 + \psi_2 + 3 \end{pmatrix}$
- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \psi_0 + 2 \\ \chi_1 + \psi_1 + 4 \\ \chi_2 + \psi_2 + 6 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.4.6.2 If $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 \\ \chi_0 + \chi_1 \\ \chi_0 + \chi_1 + \chi_2 \end{pmatrix}$, evaluate

- $f(\begin{pmatrix} 6 \\ 2 \\ 3 \end{pmatrix}) = \begin{pmatrix} 6 \\ 8 \\ 11 \end{pmatrix}$

- $f(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

- $f(2\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} 2\chi_0 \\ 2\chi_0 + 2\chi_1 \\ 2\chi_0 + 2\chi_1 + 2\chi_2 \end{pmatrix}$

- $2f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} 2\chi_0 \\ 2\chi_0 + 2\chi_1 \\ 2\chi_0 + 2\chi_1 + 2\chi_2 \end{pmatrix}$

- $f(\alpha\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_0 + \alpha\chi_1 \\ \alpha\chi_0 + \alpha\chi_1 + \alpha\chi_2 \end{pmatrix}$

- $\alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_0 + \alpha\chi_1 \\ \alpha\chi_0 + \alpha\chi_1 + \alpha\chi_2 \end{pmatrix}$

- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \psi_0 \\ \chi_0 + \chi_1 + \psi_0 + \psi_1 \\ \chi_0 + \chi_1 + \chi_2 + \psi_0 + \psi_1 + \psi_2 \end{pmatrix}$

- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \psi_0 \\ \chi_0 + \chi_1 + \psi_0 + \psi_1 \\ \chi_0 + \chi_1 + \chi_2 + \psi_0 + \psi_1 + \psi_2 \end{pmatrix}$

☛ BACK TO TEXT

Homework 1.4.6.3 If $f : \mathbb{R}^n \to \mathbb{R}^m$, then

$$f(0) = 0.$$

Always/Sometimes/Never

Answer: Sometimes. We have seen examples where $f(0) = 0$ and where $f(0) \neq 0$.

☛ BACK TO TEXT

Homework 1.4.6.4 If $f : \mathbb{R}^n \to \mathbb{R}^m$, $\lambda \in \mathbb{R}$, and $x \in \mathbb{R}^n$, then

$$f(\lambda x) = \lambda f(x).$$

Always/Sometimes/Never

Answer: Sometimes. We have seen examples where $f(\lambda x) = \lambda f(x)$ and where $f(\lambda x) \neq \lambda f(x)$.

☛ BACK TO TEXT

Homework 1.4.6.5 If $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ and $x, y \in \mathbb{R}^n$, then

$$f(x+y) = f(x) + f(y).$$

Always/Sometimes/Never

Answer: Sometimes. We have seen examples where $f(x+y) = f(x) + f(y)$ and where $f(x+y) \neq f(x) + f(y)$.

☛ BACK TO TEXT

Starting the Package

A Copy Routine (copy)

Homework 1.5.2.1 Implement the function `laff_copy` that copies a vector into another vector. The function is defined as

```
function [ y_out ] = laff_copy( x, y )
```

where

- `x` and `y` must each be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- `y_out` must be the same kind of vector as `y` (in other words, if `y` is a column vector, so is `y_out` and if `y` is a row vector, so is `y_out`).
- The function should "transpose" the vector if `x` and `y` do not have the same "shape" (if one is a column vector and the other one is a row vector).
- If `x` and/or `y` are not vectors or if the size of (row or column) vector `x` does not match the size of (row or column) vector `y`, the output should be `'FAILED'`.

☛ Additional instructions. If link does not work, open `LAFF-2.0xM/1521Instructions.pdf`.

Answer: See file `http://www.cs.utexas.edu/users/flame/-> Programming -> laff -> vecvec -> laff_copy.m`.

☛ BACK TO TEXT

A Routine that Scales a Vector (scal)

Homework 1.5.3.1 Implement the function `laff_scal` that scales a vector x by a scalar α. The function is defined as

```
function [ x_out ] = laff_scal( alpha, x )
```

where

- `x` must be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- `x_out` must be the same kind of vector as `x`; and
- If `x` or `alpha` are not a (row or column) vector and scalar, respectively, the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_scal.m`.

Answer: See file `LAFF-2.0xM/Programming/laff/vecvec/laff_scal.m`.

☛ BACK TO TEXT

An Inner Product Routine (dot)

Homework 1.5.4.1 Implement the function `laff_axpy` that computes $\alpha x + y$ given scalar α and vectors x and y. The function is defined as

```
function [ y_out ] = laff_axpy( alpha, x, y )
```

where

- `x` and `y` must each be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- `y_out` must be the same kind of vector as `y`; and
- If `x` and/or `y` are not vectors or if the size of (row or column) vector `x` does not match the size of (row or column) vector `y`, the output should be `'FAILED'`.
- If `alpha` is not a scalar, the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_axpy.m`.

Answer: See file `http://www.cs.utexas.edu/users/flame/-> Programming -> laff -> vecvec -> laff_axpy.m`.

☛ BACK TO TEXT

An Inner Product Routine (dot)

Homework 1.5.5.1 Implement the function `laff_dot` that computes the dot product of vectors x and y. The function is defined as

```
function [ alpha ] = laff_dot( x, y )
```

where

- `x` and `y` must each be either an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- If `x` and/or `y` are not vectors or if the size of (row or column) vector `x` does not match the size of (row or column) vector `y`, the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_dot.m`.

Answer: See file `LAFF-2.0xM/Programming/laff/vecvec/laff_dot.m`.

☛ BACK TO TEXT

A Vector Length Routine (norm2)

Homework 1.5.6.1 Implement the function `laff_norm2` that computes the length of vector x. The function is defined as

```
function [ alpha ] = laff_norm2( x )
```

where

- `x` is an $n \times 1$ array (column vector) or a $1 \times n$ array (row vector);
- If `x` is not a vector the output should be `'FAILED'`.

Check your implementation with the script in `LAFF-2.0xM/Programming/Week01/test_norm2.m.`.

Answer: See file `LAFF-2.0xM/Programming/laff/vecvec/laff_norm2.m`.

☛ BACK TO TEXT

Homework

Homework 1.8.1.1 Let

$$x = \begin{pmatrix} 2 \\ -1 \end{pmatrix}, \quad y = \begin{pmatrix} \alpha \\ \beta - \alpha \end{pmatrix}, \quad \text{and} \quad x = y.$$

Indicate which of the following must be true (there may be multiple correct answers):

(a) $\alpha = 2$

(b) $\beta = (\beta - \alpha) + \alpha = (-1) + 2 = 1$

(c) $\beta - \alpha = -1$

(d) $\beta - 2 = -1$

(e) $x = 2e_0 - e_1$

Answer: All are *true*

☛ BACK TO TEXT

Homework 1.8.1.2 A displacement vector represents the length and direction of an imaginary, shortest, straight path between two locations. To illustrate this as well as to emphasize the difference between ordered pairs that represent positions and vectors, we ask you to map a trip we made.

In 2012, we went on a journey to share our research in linear algebra. Below are some displacement vectors to describe parts of this journey using longitude and latitude. For example, we began our trip in Austin, TX and landed in San Jose, CA. Austin has coordinates $30^\circ\ 15'$ N(orth), $97^\circ\ 45'$ W(est) and San Jose's are $37^\circ\ 20'$ N, $121^\circ\ 54'$ W. *(Notice that convention is to report first longitude and then latitude.)* If we think of using longitude and latitude as coordinates in a plane where the first coordinate is position E (positive) or W (negative) and the second coordinate is position N (positive) or S (negative), then Austin's location is $(-97^\circ\ 45', 30^\circ\ 15')$ and San Jose's are $(-121^\circ\ 54', 37^\circ\ 20')$. *(Here, notice the switch in the order in which the coordinates are given because we now want to think of E/W as the x coordinate and N/S as the y coordinate.)* For our displacement vector for this, our first component will correspond to the change in the x coordinate, and the second component will be the change in the second coordinate. For convenience, we extend the notion of vectors so that the components include units as well as real numbers. Notice that for convenience, we extend the notion of vectors so that the components include units as well as real numbers (60 minutes ($'$)= 1 degree($^\circ$). Hence our displacement vector for Austin to San Jose is $\begin{pmatrix} -24^\circ\ 09' \\ 7^\circ\ 05' \end{pmatrix}$.

After visiting San Jose, we returned to Austin before embarking on a multi-legged excursion. That is, from Austin we flew to the first city and then from that city to the next, and so forth. In the end, we returned to Austin.

The following is a table of cities and their coordinates:

City	Coordinates	City	Coordinates
London	$00^\circ\ 08'$ W, $51^\circ\ 30'$ N	Austin	$-97^\circ\ 45'$ E, $30^\circ\ 15'$ N
Pisa	$10^\circ\ 21'$ E, $43^\circ\ 43'$ N	Brussels	$04^\circ\ 21'$ E, $50^\circ\ 51'$ N
Valencia	$00^\circ\ 23'$ E, $39^\circ\ 28'$ N	Darmstadt	$08^\circ\ 39'$ E, $49^\circ\ 52'$ N
Zürich	$08^\circ\ 33'$ E, $47^\circ\ 22'$ N	Krakow	$19^\circ\ 56'$ E, $50^\circ\ 4'$ N

Determine the order in which cities were visited, starting in Austin, given that the legs of the trip (given in order) had the following displacement vectors:

$$\begin{pmatrix} 102^\circ\ 06' \\ 20^\circ\ 36' \end{pmatrix} \rightarrow \begin{pmatrix} 04^\circ\ 18' \\ -00^\circ\ 59' \end{pmatrix} \rightarrow \begin{pmatrix} -00^\circ\ 06' \\ -02^\circ\ 30' \end{pmatrix} \rightarrow \begin{pmatrix} 01^\circ\ 48' \\ -03^\circ\ 39' \end{pmatrix} \rightarrow$$

$$\begin{pmatrix} 09^\circ\ 35' \\ 06^\circ\ 21' \end{pmatrix} \rightarrow \begin{pmatrix} -20^\circ\ 04' \\ 01^\circ\ 26' \end{pmatrix} \rightarrow \begin{pmatrix} 00^\circ\ 31' \\ -12^\circ\ 02' \end{pmatrix} \rightarrow \begin{pmatrix} -98^\circ\ 08' \\ -09^\circ\ 13' \end{pmatrix}$$

Answer:

$$\text{Austin} \rightarrow \text{Brussels} \rightarrow \text{Darmstadt} \rightarrow \text{Zurich} \rightarrow \text{Pisa} \rightarrow \text{Krakaw} \rightarrow \text{London} \rightarrow \text{Valencia} \rightarrow \text{Austin.}$$

(Actually, we visited a few more cities...)

☛ BACK TO TEXT

Homework 1.8.1.3 These days, high performance computers are called clusters and consist of many compute nodes, connected via a communication network. Each node of the cluster is basically equipped with a central processing unit (CPU), memory chips, a hard disk, and a network card. The nodes can be monitored for average power consumption (via power sensors) and application activity.

A system administrator monitors the power consumption of a node of such a cluster for an application that executes for two hours. This yields the following data:

Component	Average power (W)	Time in use (in hours)	Fraction of time in use
CPU	90	1.4	0.7
Memory	30	1.2	0.6
Disk	10	0.6	0.3
Network	15	0.2	0.1
Sensors	5	2.0	1.0

The energy, often measured in KWh, is equal to power times time. Notice that the total energy consumption can be found using the dot product of the vector of components' average power and the vector of corresponding time in use. What is the total energy consumed by this node in KWh? (The power is in Watts (W), so you will want to convert to Kilowatts (KW).)

Answer: Let us walk you through this:

- The CPU consumes 90 Watts, is on 1.4 hours so that the energy used in two hours is 90×1.4 Watt-hours.
- If you analyze the energy used by every component and add them together, you get

$$(90 \times 1.4 + 30 \times 1.2 + 10 \times 0.6 + 15 \times 0.2 + 5 \times 2.0) \text{ Wh} = 181 \text{ Wh}.$$

- Convert to KWh by dividing by 1000, leaving us with the answer .181 KWh.

Now, let's set this up as two vectors, x and y. The first records the power consumption for each of the components and the other for the total time that each of the components is in use:

$$x = \begin{pmatrix} 90 \\ 30 \\ 10 \\ 15 \\ 5 \end{pmatrix} \quad \text{and} \quad y = 2 \begin{pmatrix} 0.7 \\ 0.6 \\ 0.3 \\ 0.1 \\ 1.0 \end{pmatrix}.$$

Instead, compute $x^T y$. Think: How do the two ways of computing the answer relate?

Answer: Verify that if you compute $x^T y$ you arrive at the same result as you did via the initial analysis where you added the energy consumed by the different components (before converting from Wh to KWh).

☛ BACK TO TEXT

Homework 1.8.1.4 (Examples from statistics) Linear algebra shows up often when computing with data sets. In this homework, you find out how dot products can be used to define various sums of values that are often encountered in statistics.

Assume you observe a random variable and you let those sampled values be represented by $\chi_i, i = 0, 1, 2, 3, \cdots, n-1$. We can let x be the vector with components χ_i and $\vec{1}$ be a vector of size n with components all ones:

$$x = \begin{pmatrix} \chi_0 \\ \vdots \\ \chi_{n-1} \end{pmatrix}, \quad \text{and} \quad \vec{1} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}.$$

For any x, the sum of the values of x can be computed using the dot product operation as

- $x^T x$
- $\vec{1}^T x$ ✓
- $x^T \vec{1}$ ✓ (remember that the dot product commutes)

The sample mean of a random variable is the sum of the values the random variable takes on divided by the number of values, n. In other words, if the values the random variable takes on are stored in vector x, then $\bar{x} = \frac{1}{n}\sum_{i=0}^{n-1}\chi_i$. Using a dot product operation, for all x this can be computed as

- $\frac{1}{n}x^T x$
- $\frac{1}{n}\vec{1}^T x$ ✓
- $(\vec{1}^T\vec{1})^{-1}(x^T\vec{1})$ ✓ Notice that $\vec{1}^T\vec{1}) = n$ and hence $\vec{1}^T\vec{1})^{-1} = 1/n$!!!

For any x, the sum of the squares of observations stored in (the elements of) a vector, x, can be computed using a dot product operation as

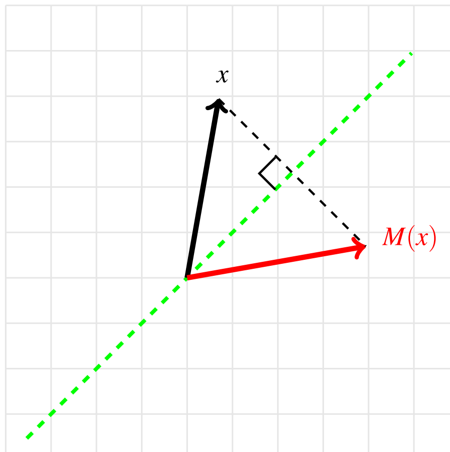
- $x^T x$ ✓
- $\vec{1}^T x$
- $x^T \vec{1}$

☛ BACK TO TEXT

Week 2: Linear Transformations and Matrices (Answers)

Rotating in 2D

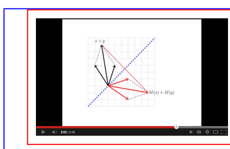
Homework 2.1.1.1 A reflection with respect to a 45 degree line is illustrated by



Think of the dashed green line as a mirror and $M : \mathbb{R}^2 \to \mathbb{R}^2$ as the vector function that maps a vector to its mirror image. If $x, y \in \mathbb{R}^2$ and $\alpha \in \mathbb{R}$, then $M(\alpha x) = \alpha M(x)$ and $M(x+y) = M(x) + M(y)$ (in other words, M is a linear transformation).

True/False

Answer: True



☛ **YouTube**

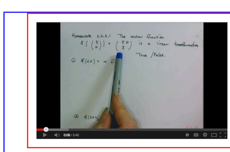
☛ BACK TO TEXT

What is a Linear Transformation?

Homework 2.2.2.1 The vector function $f(\begin{pmatrix} \chi \\ \psi \end{pmatrix}) = \begin{pmatrix} \chi\psi \\ \chi \end{pmatrix}$ is a linear transformation.

TRUE/FALSE

Answer:



☛ **YouTube**

FALSE The first check should be whether $f(0) = 0$. The answer in this case is *yes*. However,

$$f(2\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = f(\begin{pmatrix} 2 \\ 2 \end{pmatrix}) = \begin{pmatrix} 2 \times 2 \\ 2 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

and

$$2f(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = 2\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}.$$

Hence, there is a vector $x \in \mathbb{R}^2$ and $\alpha \in \mathbb{R}$ such that $f(\alpha x) \neq \alpha f(x)$. We conclude that this function is *not* a linear transformation. (Obviously, you may come up with other examples that show the function is not a linear transformation.)

☛ BACK TO TEXT

Homework 2.2.2.2 The vector function $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + 1 \\ \chi_1 + 2 \\ \chi_2 + 3 \end{pmatrix}$ is a linear transformation. (This is the same function as in Homework 1.4.6.1.)

TRUE/FALSE

Answer: FALSE

In Homework 1.4.6.1 you saw a number of examples where $f(\alpha x) \neq \alpha f(x)$.

☛ BACK TO TEXT

Homework 2.2.2.3 The vector function $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 \\ \chi_0 + \chi_1 \\ \chi_0 + \chi_1 + \chi_2 \end{pmatrix}$ is a linear transformation. (This is the same function as in Homework 1.4.6.2.)

TRUE/FALSE

Answer: TRUE

Pick arbitrary $\alpha \in \mathbb{R}$, $x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$, and $y = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}$. Then

- Show $f(\alpha x) = \alpha f(x)$:

$$f(\alpha x) = f(\alpha \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = f(\begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \\ \alpha\chi_2 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_0 + \alpha\chi_1 \\ \alpha\chi_0 + \alpha\chi_1 + \alpha\chi_2 \end{pmatrix}$$

  and

$$\alpha f(x) = \alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \alpha \begin{pmatrix} \chi_0 \\ \chi_0 + \chi_1 \\ \chi_0 + \chi_1 + \chi_2 \end{pmatrix} = \begin{pmatrix} \alpha\chi_0 \\ \alpha(\chi_0 + \chi_1) \\ \alpha(\chi_0 + \chi_1 + \chi_2) \end{pmatrix} = \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_0 + \alpha\chi_1 \\ \alpha\chi_0 + \alpha\chi_1 + \alpha\chi_2 \end{pmatrix}.$$

  Thus, $f(\alpha x) = \alpha f(x)$.

- Show $f(x+y) = f(x) + f(y)$:

$$f(x+y) = f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = f(\begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \\ \chi_2 + \psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \psi_0 \\ (\chi_0 + \psi_0) + (\chi_1 + \psi_1) \\ (\chi_0 + \psi_0) + (\chi_1 + \psi_1) + (\chi_2 + \psi_2) \end{pmatrix}$$

  and

$$f(x) + f(y) \quad = \quad f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 \\ \chi_0 + \chi_1 \\ \chi_0 + \chi_1 + \chi_2 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_0 + \psi_1 \\ \psi_0 + \psi_1 + \psi_2 \end{pmatrix}$$

$$= \begin{pmatrix} \chi_0 + \psi_0 \\ (\chi_0 + \chi_1) + (\psi_0 + \psi_1) \\ (\chi_0 + \chi_1 + \chi_2) + (\psi_0 + \psi_1 + \psi_2) \end{pmatrix} = \begin{pmatrix} \chi_0 + \psi_0 \\ (\chi_0 + \psi_0) + (\chi_1 + \psi_1) \\ (\chi_0 + \psi_0) + (\chi_1 + \psi_1) + (\chi_2 + \psi_2) \end{pmatrix}.$$

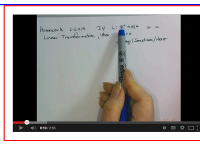
Hence $f(x+y) = f(x) + f(y)$.

☛ BACK TO TEXT

Homework 2.2.2.4 If $L : \mathbb{R}^n \to \mathbb{R}^m$ is a linear transformation, then $L(0) = 0$.
(Recall that 0 equals a vector with zero components of appropriate size.)

Always/Sometimes/Never

Answer: Always.



☛ YouTube

We know that for all scalars α and vector $x \in \mathbb{R}^n$ it is the case that $L(\alpha x) = \alpha L(x)$. Now, pick $\alpha = 0$. We know that for this choice of α it has to be the case that $L(\alpha x) = \alpha L(x)$. We conclude that $L(0x) = 0L(x)$. But $0x = 0$. (Here the first 0 is the scalar 0 and the second is the vector with n components all equal to zero.) Similarly, regardless of what vector $L(x)$ equals, multiplying it by the scalar zero yields the vector 0 (with m zero components). So, $L(0x) = 0L(x)$ implies that $L(0) = 0$.

A typical mathematician would be much more terse, writing down merely: Pick $\alpha = 0$. Then

$$L(0) = L(0x) = L(\alpha x) = \alpha L(x) = 0L(x) = 0.$$

There are actually many ways of proving this:

$$L(0) = L(x - x) = L(x + (-x)) = L(x) + L(-x) = L(x) + (-L(x)) = L(x) - L(x) = 0.$$

Alternatively, $L(x) = L(x + 0) = L(x) + L(0)$, hence $L(0) = L(x) - L(x) = 0$.

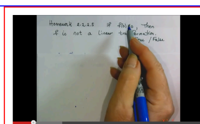
Typically, it is really easy to evaluate $f(0)$. Therefore, if you think a given vector function f is *not* a linear transformation, then you may want to first evaluate $f(0)$. If it does not evaluate to the zero vector, then you know it is not a linear transformation.

☛ BACK TO TEXT

Homework 2.2.2.5 Let $f : \mathbb{R}^n \to \mathbb{R}^m$ and $f(0) \neq 0$. Then f is not a linear transformation.

True/False

Answer: True.



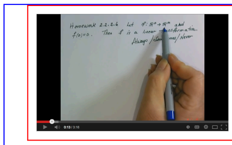
☛ YouTube

☛ BACK TO TEXT

Homework 2.2.2.6 Let $f : \mathbb{R}^n \to \mathbb{R}^m$ and $f(0) = 0$. Then f is a linear transformation.

Always/Sometimes/Never

Answer: Sometimes.



☛ **YouTube**

We have seen examples where the statement is true and examples where f is *not* a linear transformation, yet there $f(0) = 0$. For example, in Homework **??** you have an example where $f(0) = 0$ and f is not a linear transformation.

☛ BACK TO TEXT

Homework 2.2.2.7 Find an example of a function f such that $f(\alpha x) = \alpha f(x)$, but for some x, y it is the case that $f(x+y) \neq f(x) + f(y)$. (This is pretty tricky!)

Answer: $f(\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix}) = f(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = 1$ but $f(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) + f(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = 0 + 0 = 0.$

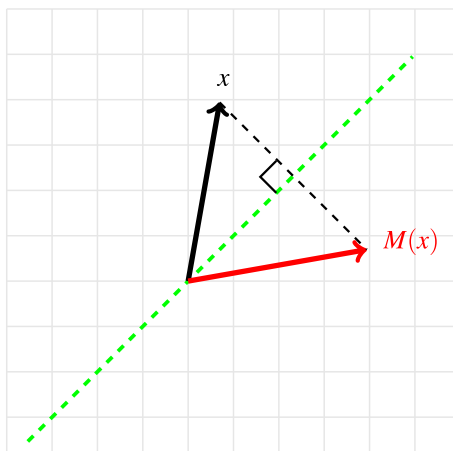
☛ BACK TO TEXT

Homework 2.2.2.8 The vector function $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_1 \\ \chi_0 \end{pmatrix}$ is a linear transformation.

TRUE/FALSE

Answer: TRUE

This is actually the reflection with respect to 45 degrees line that we talked about earlier:



Pick arbitrary $\alpha \in \mathbb{R}$, $x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}$, and $y = \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}$. Then

- Show $f(\alpha x) = \alpha f(x)$:

$$f(\alpha x) = f(\alpha \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = f(\begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \end{pmatrix}) = \begin{pmatrix} \alpha\chi_1 \\ \alpha\chi_0 \end{pmatrix} = \alpha \begin{pmatrix} \chi_1 \\ \chi_0 \end{pmatrix} = \alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}).$$

- Show $f(x+y) = f(x) + f(y)$:

$$f(x+y) = f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}) = f(\begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \end{pmatrix}) = \begin{pmatrix} \chi_1 + \psi_1 \\ \chi_0 + \psi_0 \end{pmatrix}$$

and

$$
\begin{aligned}
f(x)+f(y) &= f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) + f(\begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}) = \begin{pmatrix} \chi_1 \\ \chi_0 \end{pmatrix} + \begin{pmatrix} \psi_1 \\ \psi_0 \end{pmatrix} \\
&= \begin{pmatrix} \chi_1+\psi_1 \\ \chi_0+\psi_0 \end{pmatrix}.
\end{aligned}
$$

Hence $f(x+y) = f(x)+f(y)$.

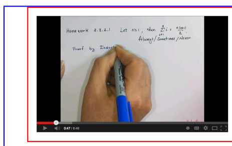
☛ BACK TO TEXT

Examples

Homework 2.3.2.1 Let $n \geq 1$. Then $\sum_{i=1}^{n} i = n(n+1)/2$.

Always/Sometimes/Never

Answer:



☛ YouTube

We can prove this in three different ways:

1. By mathematical induction, carefully mimicing the proof that $\sum_{i=0}^{n-1} i = (n-1)n/2$; or
2. Using a trick similar to the one used in the alternative proof given for $\sum_{i=0}^{n-1} i = (n-1)n/2$; or
3. Using the fact that $\sum_{i=0}^{n-1} i = n(n-1)/2$.

☛ BACK TO TEXT

Homework 2.3.2.2 Let $n \geq 1$. $\sum_{i=0}^{n-1} 1 = n$.

Always/Sometimes/Never

Answer: Always.

Base case: $n = 1$. For this case, we must show that $\sum_{i=0}^{1-1} 1 = 1$.

$\sum_{i=0}^{1-1} 1$

$=$ (Definition of summation)

1

This proves the base case.

Inductive step: Inductive Hypothesis (IH): Assume that the result is true for $n = k$ where $k \geq 1$:

$$\sum_{i=0}^{k-1} 1 = k.$$

We will show that the result is then also true for $n = k+1$:

$$\sum_{i=0}^{(k+1)-1} 1 = (k+1).$$

Assume that $k \geq 1$. Then

$\sum_{i=0}^{(k+1)-1} 1$

$=$ (arithmetic)

$\sum_{i=0}^{k} 1$

$=$ (split off last term)

$\left(\sum_{i=0}^{k-1} 1\right) + 1$

$=$ (I.H.)

$k+1$.

This proves the inductive step.

By the Principle of Mathematical Induction the result holds for all n.

☛ BACK TO TEXT

Homework 2.3.2.3 Let $n \geq 1$ and $x \in \mathbb{R}^m$. Then

$$\sum_{i=0}^{n-1} x = \underbrace{x + x + \cdots + x}_{n \text{ times}} = nx$$

Always/Sometimes/Never

Answer: Always.

$$\sum_{i=0}^{n-1} x = \left(\sum_{i=0}^{n-1} 1\right) x = nx.$$

However, we want you to prove this with mathematical induction:

Base case: $n = 1$. For this case, we must show that $\sum_{i=0}^{1-1} x = x$.

$\sum_{i=0}^{1-1} x$

$=$ $<$ Definition of summation$>$

x

This proves the base case.

Inductive step: Inductive Hypothesis (IH): Assume that the result is true for $n = k$ where $k \geq 1$:

$$\sum_{i=0}^{k-1} x = kx.$$

We will show that the result is then also true for $n = k+1$:

$$\sum_{i=0}^{(k+1)-1} x = (k+1)x.$$

Assume that $k \geq 1$. Then

$\sum_{i=0}^{(k+1)-1} x$

$=$ < arithmetic>

$\sum_{i=0}^{k} x$

$=$ < split off last term>

$\sum_{i=0}^{k-1} x + x$

$=$ < I.H.>

$kx + x$

$=$ < algebra>

$(k+1)x$

This proves the inductive step.

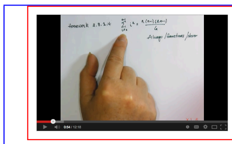
By the Principle of Mathematical Induction the result holds for all n.

☛ BACK TO TEXT

Homework 2.3.2.4 Let $n \geq 1$. $\sum_{i=0}^{n-1} i^2 = (n-1)n(2n-1)/6$.

Always/Sometimes/Never

Answer:



☛ **YouTube**

Always

Base case: $n = 1$. For this case, we must show that $\sum_{i=0}^{1-1} i^2 = (1-1)(1)(2(1)-1)/6$. But $\sum_{i=0}^{1-1} i^2 = 0 = (1-1)(1)(2(1)-1)/6$. This proves the base case.

Inductive step: Inductive Hypothesis (IH): Assume that the result is true for $n = k$ where $k \geq 1$:

$$\sum_{i=0}^{k-1} i^2 = (k-1)k(2k-1)/6.$$

We will show that the result is then also true for $n = k+1$:

$$\sum_{i=0}^{(k+1)-1} i^2 = ((k+1)-1)(k+1)(2(k+1)-1)/6 = (k)(k+1)(2k+1)/6.$$

Assume that $k \geq 1$. Then

$\sum_{i=0}^{(k+1)-1} i^2$

$=$ (arithmetic)

$\sum_{i=0}^{k} i^2$

$=$ (split off last term)

$\sum_{i=0}^{k-1} i^2 + k^2$

$=$ (I.H.)

$(k-1)k(2k-1)/6 + k^2$

$=$ (algebra)

$[(k-1)k(2k-1) + 6k^2]/6.$

Now,

$$(k)(k+1)(2k+1) = (k^2+k)(2k+1) = 2k^3 + 2k^2 + k^2 + k = 2k^3 + 3k^2 + k$$

and

$$(k-1)k(2k-1) + 6k^2 = (k^2-k)(2k-1) + 6k^2 = 2k^3 - 2k^2 - k^2 + k + 6k^2 = 2k^3 + 3k^2 + k.$$

Hence

$$\sum_{i=0}^{(k+1)-1} i^2 = (k)(k+1)(2k+1)/6$$

This proves the inductive step.

By the Principle of Mathematical Induction the result holds for all n.

☛ BACK TO TEXT

From Linear Transformation to Matrix-Vector Multiplication

Homework 2.4.1.1 Give an alternative proof for this theorem that mimics the proof by induction for the lemma that states that $L(v_0 + \cdots + v_{n-1}) = L(v_0) + \cdots + L(v_{n-1})$.

Answer: Proof by induction on k.

Base case: $k = 1$. For this case, we must show that $L(\alpha_0 v_0) = \alpha_0 L(v_0)$. This follows immediately from the definition of a linear transformation.

Inductive step: Inductive Hypothesis (IH): Assume that the result is true for $k = K$ where $K \geq 1$:

$$L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{K-1} v_{K-1}) = \alpha_0 L(v_0) + \alpha_1 L(v_1) + \cdots + \alpha_{K-1} L(v_{K-1}).$$

We will show that the result is **then** also true for $k = K+1$. In other words, that

$$L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{K-1} v_{K-1} + \alpha_K v_K) = \alpha_0 L(v_0) + \alpha_1 L(v_1) + \cdots + \alpha_{K-1} L(v_{K-1}) + \alpha_K L(v_K).$$

Assume that $K \geq 1$ and $k = K+1$. Then

$$\begin{array}{ll}
 & L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{k-1} v_{k-1}) \\
= & \quad < k-1 = (K+1)-1 = K > \\
 & L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_K v_K) \\
= & \quad < \text{expose extra term – We know we can do this, since } K \geq 1 > \\
 & L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{K-1} v_{K-1} + \alpha_K v_K) \\
= & \quad < \text{associativity of vector addition} > \\
 & L((\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{K-1} v_{K-1}) + \alpha_K v_K) \\
= & \quad < L \text{ is a linear transformation)} > \\
 & L(\alpha_0 v_0 + \alpha_1 v_1 + \cdots + \alpha_{K-1} v_{K-1}) + L(\alpha_K v_K) \\
= & \quad < \text{Inductive Hypothesis} > \\
 & \alpha_0 L(v_0) + \alpha_1 L(v_1) + \cdots + \alpha_{K-1} L(v_{K-1}) + L(\alpha_K v_K) \\
= & \quad < \text{Definition of a linear transformation} > \\
 & \alpha_0 L(v_0) + \alpha_1 L(v_1) + \cdots + \alpha_{K-1} L(v_{K-1}) + \alpha_K L(v_K)
\end{array}$$

By the Principle of Mathematical Induction the result holds for all k.

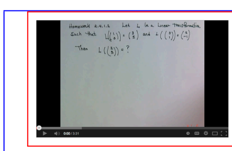
☛ BACK TO TEXT

Homework 2.4.1.2 Let L be a linear transformation such that

$$L\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} 3 \\ 5 \end{pmatrix} \quad \text{and} \quad L\left(\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right) = \begin{pmatrix} 2 \\ -1 \end{pmatrix}.$$

Then $L\left(\begin{pmatrix} 2 \\ 3 \end{pmatrix}\right) =$

Answer:



☛ YouTube

$$\begin{pmatrix} 2 \\ 3 \end{pmatrix} = 2\begin{pmatrix} 1 \\ 0 \end{pmatrix} + 3\begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Hence

$$\begin{aligned}
L\left(\begin{pmatrix} 2 \\ 3 \end{pmatrix}\right) &= L\left(2\begin{pmatrix} 1 \\ 0 \end{pmatrix} + 3\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right) = 2L\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}\right) + 3L\left(\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right) \\
&= 2\begin{pmatrix} 3 \\ 5 \end{pmatrix} + 3\begin{pmatrix} 2 \\ -1 \end{pmatrix} = \begin{pmatrix} 2 \times 3 + 3 \times 2 \\ 2 \times 5 + 3 \times (-1) \end{pmatrix} = \begin{pmatrix} 12 \\ 7 \end{pmatrix}.
\end{aligned}$$

☛ BACK TO TEXT

Homework 2.4.1.3 $L(\begin{pmatrix} 3 \\ 3 \end{pmatrix}) =$

Answer:

$$\begin{pmatrix} 3 \\ 3 \end{pmatrix} = 3 \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Hence

$$L(\begin{pmatrix} 3 \\ 3 \end{pmatrix}) = 3L(\begin{pmatrix} 1 \\ 1 \end{pmatrix}) = 3 \begin{pmatrix} 5 \\ 4 \end{pmatrix} = \begin{pmatrix} 15 \\ 12 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 2.4.1.4 $L(\begin{pmatrix} -1 \\ 0 \end{pmatrix}) =$

Answer:

$$\begin{pmatrix} -1 \\ 0 \end{pmatrix} = (-1) \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Hence

$$L(\begin{pmatrix} -1 \\ 0 \end{pmatrix}) = (-1)L(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = (-1) \begin{pmatrix} 3 \\ 5 \end{pmatrix} = \begin{pmatrix} -3 \\ -5 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 2.4.1.5 $L(\begin{pmatrix} 2 \\ 3 \end{pmatrix}) =$

Answer:

$$\begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \end{pmatrix} + \begin{pmatrix} -1 \\ 0 \end{pmatrix}.$$

Hence

$$\begin{aligned} L(\begin{pmatrix} 2 \\ 3 \end{pmatrix}) &= L(\begin{pmatrix} 3 \\ 3 \end{pmatrix}) + L(\begin{pmatrix} -1 \\ 0 \end{pmatrix}) = \text{(from the previous two exercises)} \\ & \begin{pmatrix} 15 \\ 12 \end{pmatrix} + \begin{pmatrix} -3 \\ -5 \end{pmatrix} = \begin{pmatrix} 12 \\ 7 \end{pmatrix}. \end{aligned}$$

☛ BACK TO TEXT

Homework 2.4.1.6 Let L be a linear transformation such that

$$L\left(\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right) = \begin{pmatrix} 5 \\ 4 \end{pmatrix}.$$

Then $L\left(\begin{pmatrix} 3 \\ 2 \end{pmatrix}\right) =$

Answer: The problem is that you can't write $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ as a linear combination (scalar multiple in this case) of $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$. So, there isn't enough information.

☛ BACK TO TEXT

Homework 2.4.1.7 Let L be a linear transformation such that

$$L\left(\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right) = \begin{pmatrix} 5 \\ 4 \end{pmatrix} \quad \text{and} \quad L\left(\begin{pmatrix} 2 \\ 2 \end{pmatrix}\right) = \begin{pmatrix} 10 \\ 8 \end{pmatrix}.$$

Then $L\left(\begin{pmatrix} 3 \\ 2 \end{pmatrix}\right) =$

Answer: The problem is that you can't write $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ as a linear combination of $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} 2 \\ 2 \end{pmatrix}$. So, there isn't enough information.

☛ BACK TO TEXT

Homework 2.4.1.8 Give the matrix that corresponds to the linear transformation $f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}\right) = \begin{pmatrix} 3\chi_0 - \chi_1 \\ \chi_1 \end{pmatrix}$.

Answer:

- $f\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} 3-0 \\ 0 \end{pmatrix} = \begin{pmatrix} 3 \\ 0 \end{pmatrix}$.
- $f\left(\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right) = \begin{pmatrix} 0-1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$.

Hence $\begin{pmatrix} 3 & -1 \\ 0 & 1 \end{pmatrix}$

☛ BACK TO TEXT

Homework 2.4.1.9 Give the matrix that corresponds to the linear transformation $f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}\right) = \begin{pmatrix} 3\chi_0 - \chi_1 \\ \chi_2 \end{pmatrix}$.

Answer:

- $f(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}) = \begin{pmatrix} 3-0 \\ 0 \end{pmatrix}. = \begin{pmatrix} 3 \\ 0 \end{pmatrix}.$

- $f(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} -1 \\ 0 \end{pmatrix}.$

- $f(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$

Hence $\begin{pmatrix} 3 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

☛ BACK TO TEXT

Practice with Matrix-Vector Multiplication

Homework 2.4.2.1 Compute Ax when $A = \begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix}$ and $x = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$.

Answer: $\begin{pmatrix} -1 \\ -3 \\ -2 \end{pmatrix}$, the first column of the matrix!

☛ BACK TO TEXT

Homework 2.4.2.2 Compute Ax when $A = \begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix}$ and $x = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$.

Answer: $\begin{pmatrix} 2 \\ -1 \\ 2 \end{pmatrix}$, the third column of the matrix!

☛ BACK TO TEXT

Homework 2.4.2.3 If A is a matrix and e_j is a unit basis vector of appropriate length, then $Ae_j = a_j$, where a_j is the jth column of matrix A.

Always/Sometimes/Never

Answer: Always

If e_j is the j unit basis vector then

$$Ae_j = \left(\begin{array}{c|c|c|c|c|c} a_0 & a_1 & \cdots & a_j & \cdots & a_{n-1} \end{array} \right) \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} = 0 \cdot a_0 + 0 \cdot a_1 + \cdots + 1 \cdot a_j + \cdots + 0 \cdot a_{n-1} = a_j.$$

☛ BACK TO TEXT

Homework 2.4.2.4 If x is a vector and e_i is a unit basis vector of appropriate size, then their dot product, $e_i^T x$, equals the ith entry in x, χ_i.

Always/Sometimes/Never

Answer: Always (We saw this already in Week 1.)

$$e_i^T x = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{i-1} \\ \chi_i \\ \chi_{i+1} \\ \vdots \\ \chi_{n-1} \end{pmatrix} = 0 \cdot \chi_0 + 0 \cdot \chi_1 + \cdots + 1 \cdot \chi_i + \cdots + 0 \cdot \chi_{n-1} = \chi_i.$$

☛ BACK TO TEXT

Homework 2.4.2.5 Compute

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}^T \left(\begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right) = \underline{\qquad}$$

Answer:

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}^T \begin{pmatrix} -1 \\ -3 \\ -2 \end{pmatrix} = -2,$$

the $(2,0)$ element of the matrix.

☛ BACK TO TEXT

Homework 2.4.2.6 Compute

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}^T \left(\begin{pmatrix} -1 & 0 & 2 \\ -3 & 1 & -1 \\ -2 & -1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right) = \underline{\qquad}$$

Answer:

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}^T \begin{pmatrix} -1 \\ -3 \\ -2 \end{pmatrix} = -3,$$

the $(1,0)$ element of the matrix.

☛ BACK TO TEXT

Homework 2.4.2.7 Let A be a $m \times n$ matrix and $\alpha_{i,j}$ its (i,j) element. Then $\alpha_{i,j} = e_i^T(Ae_j)$.

Always/Sometimes/Never

Answer: Always

From a previous exercise we know that $Ae_j = a_j$, the jth column of A. From another exercise we know that $e_i^T a_j = \alpha_{i,j}$, the ith component of the jth column of A.

Later, we will see that $e_i^T A$ equals the ith row of matrix A and that $\alpha_{i,j} = e_i^T(Ae_j) = e_i^T Ae_j = (e_i^T A)e_j$ (this kind of multiplication is associative).

☛ BACK TO TEXT

Homework 2.4.2.8 Compute

- $\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\left((-2)\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right) = \begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\begin{pmatrix} 0 \\ -2 \end{pmatrix} = \begin{pmatrix} 0+2 \\ 0+0 \\ 0+-6 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ -6 \end{pmatrix}$

- $(-2)\left(\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right) = (-2)\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = (-2)\begin{pmatrix} -1 \\ 0 \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ -6 \end{pmatrix}$

- $\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\left(\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2+(-1) \\ 1+0 \\ -2+3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

- $\begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 2 & -1 \\ 1 & 0 \\ -2 & 3 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -2 \end{pmatrix} + \begin{pmatrix} -1 \\ 0 \\ 3 \end{pmatrix} = \begin{pmatrix} 2+(-1) \\ 1+0 \\ -2+3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

☛ BACK TO TEXT

Homework 2.4.2.9 Let $A \in \mathbb{R}^{m\times n}$; $x, y \in \mathbb{R}^n$; and $\alpha \in \mathbb{R}$. Then

- $A(\alpha x) = \alpha(Ax)$.

- $A(x+y) = Ax + Ay$.

Always/Sometimes/Never

Answer: Always

$$\begin{aligned}
A(\alpha x) &= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \\ \vdots \\ \alpha\chi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0}(\alpha\chi_0) + \alpha_{0,1}(\alpha\chi_1) + \cdots + \alpha_{0,n-1}(\alpha\chi_{n-1}) \\ \alpha_{1,0}(\alpha\chi_0) + \alpha_{1,1}(\alpha\chi_1) + \cdots + \alpha_{1,n-1}(\alpha\chi_{n-1}) \\ \vdots \\ \alpha_{m-1,0}(\alpha\chi_0) + \alpha_{m-1,1}(\alpha\chi_1) + \cdots + \alpha_{m-1,n-1}(\alpha\chi_{n-1}) \end{pmatrix} \\
&= \begin{pmatrix} \alpha\alpha_{0,0}\chi_0 + \alpha\alpha_{0,1}\chi_1 + \cdots + \alpha\alpha_{0,n-1}\chi_{n-1} \\ \alpha\alpha_{1,0}\chi_0 + \alpha\alpha_{1,1}\chi_1 + \cdots + \alpha\alpha_{1,n-1}\chi_{n-1} \\ \vdots \\ \alpha\alpha_{m-1,0}\chi_0 + \alpha\alpha_{m-1,1}\chi_1 + \cdots + \alpha\alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha(\alpha_{0,0}\chi_0 + \alpha_{0,1}\chi_1 + \cdots + \alpha_{0,n-1}\chi_{n-1}) \\ \alpha(\alpha_{1,0}\chi_0 + \alpha_{1,1}\chi_1 + \cdots + \alpha_{1,n-1}\chi_{n-1}) \\ \vdots \\ \alpha(\alpha_{m-1,0}\chi_0 + \alpha_{m-1,1}\chi_1 + \cdots + \alpha_{m-1,n-1}\chi_{n-1}) \end{pmatrix} \\
&= \alpha \begin{pmatrix} \alpha_{0,0}\chi_0 + \alpha_{0,1}\chi_1 + \cdots + \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0 + \alpha_{1,1}\chi_1 + \cdots + \alpha_{1,n-1}\chi_{n-1} \\ \vdots \\ \alpha_{m-1,0}\chi_0 + \alpha_{m-1,1}\chi_1 + \cdots + \alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} \\
&= \alpha \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \alpha A x
\end{aligned}$$

$$A(x+y) = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \left(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} \right)$$

$$= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \\ \vdots \\ \chi_{n-1} + \psi_{n-1} \end{pmatrix}$$

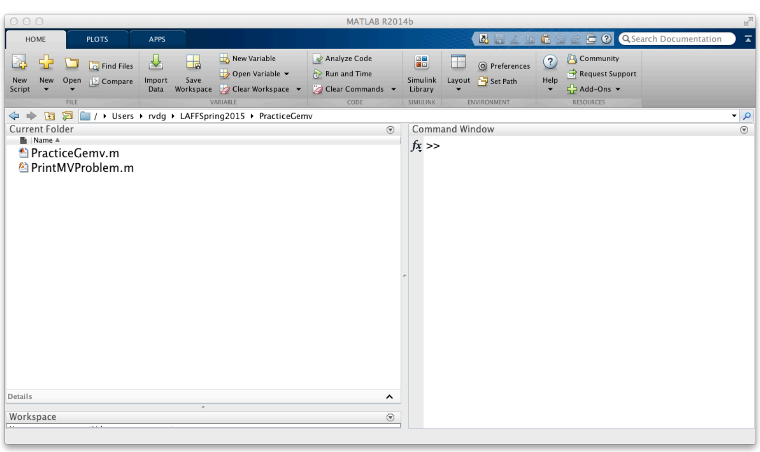
$$
\begin{aligned}
&= \begin{pmatrix} \alpha_{0,0}(\chi_0+\psi_0)+\alpha_{0,1}(\chi_1+\psi_1)+\cdots+\alpha_{0,n-1}(\chi_{n-1}+\psi_{n-1}) \\ \alpha_{1,0}(\chi_0+\psi_0)+\alpha_{1,1}(\chi_1+\psi_1)+\cdots+\alpha_{1,n-1}(\chi_{n-1}+\psi_{n-1}) \\ \vdots \\ \alpha_{m-1,0}(\chi_0+\psi_0)+\alpha_{m-1,1}(\chi_1+\psi_1)+\cdots+\alpha_{m-1,n-1}(\chi_{n-1}+\psi_{n-1}) \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0}\chi_0+\alpha_{0,0}\psi_0+\alpha_{0,1}\chi_1+\alpha_{0,1}\psi_1+\cdots+\alpha_{0,n-1}\chi_{n-1}+\alpha_{0,n-1}\psi_{n-1} \\ \alpha_{1,0}\chi_0+\alpha_{1,0}\psi_0+\alpha_{1,1}\chi_1+\alpha_{1,1}\psi_1+\cdots+\alpha_{1,n-1}\chi_{n-1}+\alpha_{1,n-1}\psi_{n-1} \\ \vdots \\ \alpha_{m-1,0}\chi_0+\alpha_{m-1,0}\psi_0+\alpha_{m-1,1}\chi_1+\alpha_{m-1,1}\psi_1+\cdots+\alpha_{m-1,n-1}\chi_{n-1}+\alpha_{m-1,n-1}\psi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0}\chi_0+\alpha_{0,1}\chi_1++\cdots+\alpha_{0,n-1}\chi_{n-1}+\alpha_{0,0}\psi_0+\alpha_{0,1}\psi_1+\cdots+\alpha_{0,n-1}\psi_{n-1} \\ \alpha_{1,0}\chi_0+\alpha_{1,1}\chi_1+\cdots+\alpha_{1,n-1}\chi_{n-1}+\alpha_{1,0}\psi_0+\alpha_{1,1}\psi_1+\cdots+\alpha_{1,n-1}\psi_{n-1} \\ \vdots \\ \alpha_{m-1,0}\chi_0+\alpha_{m-1,1}\chi_1+\cdots+\alpha_{m-1,n-1}\chi_{n-1}+\alpha_{m-1,0}\psi_0+\alpha_{m-1,1}\psi_1+\cdots+\alpha_{m-1,n-1}\psi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0}\chi_0+\alpha_{0,1}\chi_1++\cdots+\alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0+\alpha_{1,1}\chi_1+\cdots+\alpha_{1,n-1}\chi_{n-1}+ \\ \vdots \\ \alpha_{m-1,0}\chi_0+\alpha_{m-1,1}\chi_1+\cdots+\alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} + \begin{pmatrix} \alpha_{0,0}\psi_0+\alpha_{0,1}\psi_1+\cdots+\alpha_{0,n-1}\psi_{n-1} \\ \alpha_{1,1}\psi_1+\cdots+\alpha_{1,n-1}\psi_{n-1} \\ \vdots \\ \alpha_{m-1,0}\psi_0+\alpha_{m-1,1}\psi_1+\cdots+\alpha_{m-1,n-1}\psi_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} + \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} \\
&= Ax+Ay
\end{aligned}
$$

☛ BACK TO TEXT

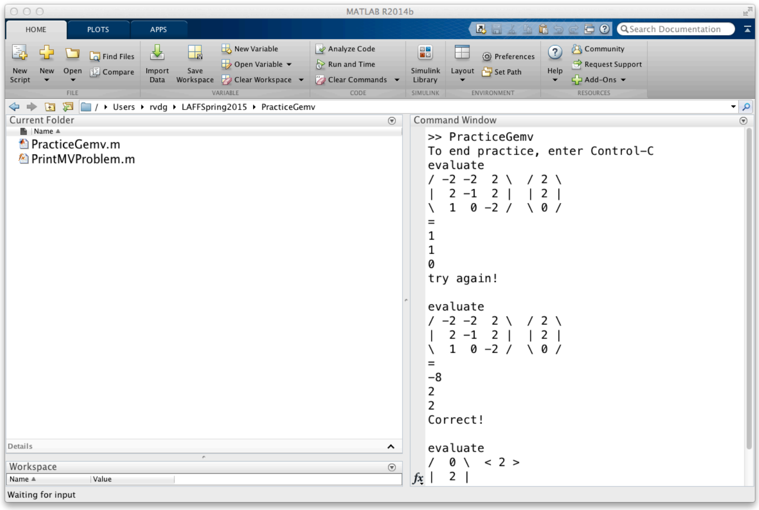
Homework 2.4.2.10 You can practice as little or as much as you want!

Some of the following instructions are for the desktop version of Matlab, but it should be pretty easy to figure out what to do instead with Matlab Online.

Start up Matlab or log on to Matlab Online and change the current directory to Programming/Week02/.



Then type `PracticeGemv` in the command window and you get to practice all the matrix-vector multiplications you want! For example, after a bit of practice my window looks like



Practice all you want!

☛ BACK TO TEXT

It Goes Both Ways

Homework 2.4.3.1 Give the linear transformation that corresponds to the matrix

$$\begin{pmatrix} 2 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

Answer: $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \end{pmatrix}) = \begin{pmatrix} 2 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \end{pmatrix} = \begin{pmatrix} 2\chi_0 + \chi_1 - \chi_3 \\ \chi_2 - \chi_3 \end{pmatrix}.$

☛ BACK TO TEXT

Homework 2.4.3.2 Give the linear transformation that corresponds to the matrix

$$\begin{pmatrix} 2 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

Answer: $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} 2 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 2\chi_0 + \chi_1 \\ \chi_1 \\ \chi_0 \\ \chi_0 + \chi_1 \end{pmatrix}.$

☛ BACK TO TEXT

Homework 2.4.3.3 Let f be a vector function such that $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0^2 \\ \chi_1 \end{pmatrix}$ Then

- (a) f is a linear transformation.
- (b) f is not a linear transformation.
- (c) Not enough information is given to determine whether f is a linear transformation.

How do you know?

Answer: (b): To compute a possible matrix that represents f consider:

$$f(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1^2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{and} \quad f(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0^2 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Thus, *if* f is a linear transformation, then $f(x) = Ax$ where $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Now,

$$Ax = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \neq \begin{pmatrix} \chi_0^2 \\ \chi_1 \end{pmatrix} = f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}\right) = f(x).$$

Hence f is *not* a linear transformation since $f(x) \neq Ax$.

☛ BACK TO TEXT

Homework 2.4.3.4 For each of the following, determine whether it is a linear transformation or not:

- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 \\ 0 \\ \chi_2 \end{pmatrix}$.

  Answer: True To compute a possible matrix that represents f consider:

$$f(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad f(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad f(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

  Thus, *if* f is a linear transformation, then $f(x) = Ax$ where $A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. Now,

$$Ax = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \chi_0 \\ 0 \\ \chi_2 \end{pmatrix} = f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}\right) = f(x).$$

  Hence f is a linear transformation since $f(x) = Ax$.

- $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0^2 \\ 0 \end{pmatrix}$.

  Answer: False To compute a possible matrix that represents f consider:

$$f(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1^2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{and} \quad f(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0^2 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

  Thus, *if* f is a linear transformation, then $f(x) = Ax$ where $A = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$. Now,

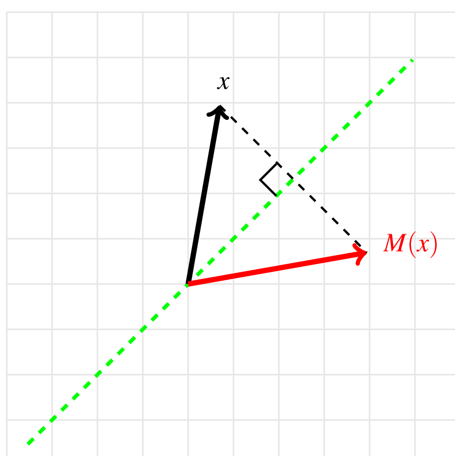
$$Ax = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \chi_0 \\ 0 \end{pmatrix} \neq \begin{pmatrix} \chi_0^2 \\ 0 \end{pmatrix} = f\left(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}\right) = f(x).$$

  Hence f is *not* a linear transformation since $f(x) \neq Ax$.

☛ BACK TO TEXT

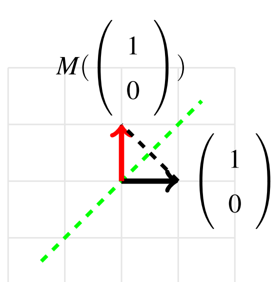
Rotations and Reflections, Revisited

Homework 2.4.4.1 A reflection with respect to a 45 degree line is illustrated by

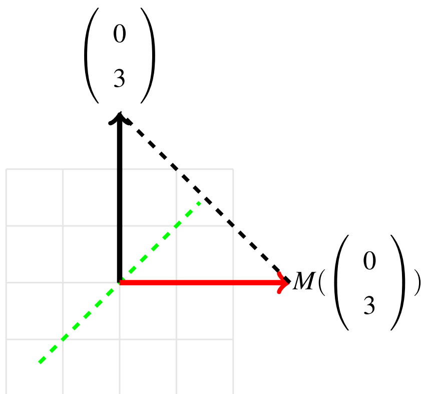


Again, think of the dashed green line as a mirror and let $M : \mathbb{R}^2 \to \mathbb{R}^2$ be the vector function that maps a vector to its mirror image. Evaluate (by examining the picture)

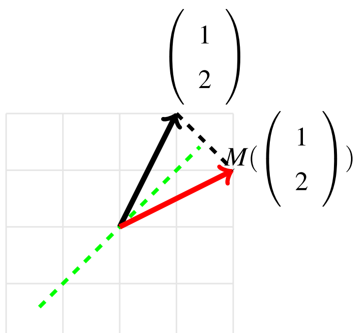
- $M(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) =$. **Answer:**



- $M\left(\begin{pmatrix} 0 \\ 3 \end{pmatrix}\right) =$. **Answer:**

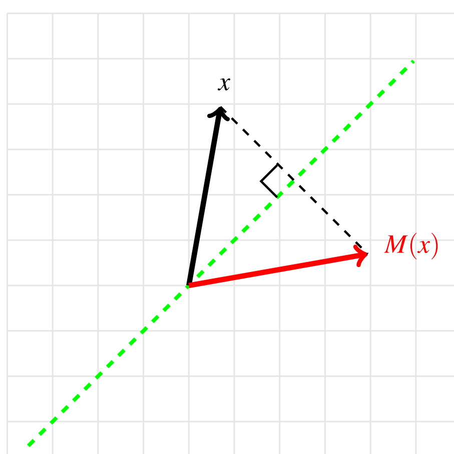


- $M\left(\begin{pmatrix} 1 \\ 2 \end{pmatrix}\right) =$. **Answer:**



☛ BACK TO TEXT

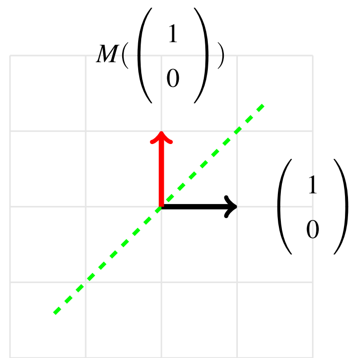
Homework 2.4.4.2 A reflection with respect to a 45 degree line is illustrated by



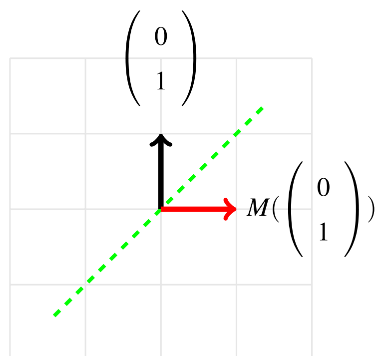
Again, think of the dashed green line as a mirror and let $M : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ be the vector function that maps a vector to its mirror image. Compute the matrix that represents M (by examining the picture).

Answer:

- $M(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$:



- $M(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$:



Hence the matrix is $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework

Homework 2.6.1.1 Suppose a professor decides to assign grades based on two exams and a final. Either all three exams (worth 100 points each) are equally weighted or the final is double weighted to replace one of the exams to benefit the student. The records indicate each score on the first exam as χ_0, the score on the second as χ_1, and the score on the final as χ_2. The professor transforms these scores and looks for the maximum entry. The following describes the linear transformation:

$$l(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \chi_1 + \chi_2 \\ \chi_0 + 2\chi_2 \\ \chi_1 + 2\chi_2 \end{pmatrix}$$

What is the matrix that corresponds to this linear transformation?

Answer: $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 2 \\ 0 & 1 & 2 \end{pmatrix}$

If a student's scores are $\begin{pmatrix} 68 \\ 80 \\ 95 \end{pmatrix}$, what is the transformed score?

Answer: $\begin{pmatrix} 243 \\ 258 \\ 270 \end{pmatrix}$

☛ BACK TO TEXT

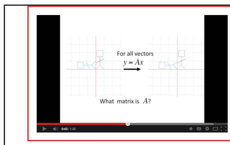
Week 3: Matrix-Vector Operations (Answers)

Opening Remarks

Timmy Two Space

Homework 3.1.1.1 Click on the below link to open a browser window with the "Timmy Two Space" exercise. This exercise was suggested to us by our colleague Prof. Alan Cline. It was first implemented using an IPython Notebook by Ben Holder. During the Spring 2014 offering of LAFF on the edX platform, one of the participants, Ed McCardell, rewrote the activity as ☛ Local copy. (If this link does not work, open `LAFF-2.0xM/Timmy/index.html`).

If you get really frustrated, here is a hint:



☛ **YouTube**

☛ BACK TO TEXT

Special Matrices

The Zero Matrix

Homework 3.2.1.1 Let $L_0 : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be the function defined for every $x \in \mathbb{R}^n$ as $L_0(x) = 0$, where 0 denotes the zero vector "of appropriate size". L_0 is a linear transformation.

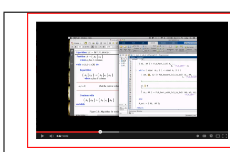
True/False

Answer: True

- $L_0(\alpha x) = 0 = \alpha 0 = \alpha L_0(0)$.
- $L_0(x+y) = 0 = 0+0 = L_0(x) + L_0(y)$.

☛ BACK TO TEXT

Homework 3.2.1.2 With the FLAME API for MATLAB (FLAME@lab) implement the algorithm in Figure 3.1. You will use the function `laff_zerov( x )`, which returns a zero vector of the same size and shape (column or row) as input vector `x`. Since you are still getting used to programming with M-script and FLAME@lab, you may want to follow the instructions in this video:



☛ **YouTube**

Some links that will come in handy:

- ☛ Spark
  (alternatively, open the file ☛ `LAFF-2.0xM/Spark/index.html`)
- ☛ PictureFLAME
  (alternatively, open the file ☛ `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

You will need these in many future exercises. Bookmark them!

Answer:

```
function [ A_out ] = ZeroMatrix_unb( A )

  [ AL, AR ] = FLA_Part_1x2( A, ...
                               0, 'FLA_LEFT' );

  while ( size( AL, 2 ) < size( A, 2 ) )

    [ A0, a1, A2 ]= FLA_Repart_1x2_to_1x3( AL, AR, ...
                                         1, 'FLA_RIGHT' );

    %------------------------------------------------------------%

    a1 = laff_zerov( a1 );

    %------------------------------------------------------------%

    [ AL, AR ] = FLA_Cont_with_1x3_to_1x2( A0, a1, A2, ...
                                           'FLA_LEFT' );

  end

  A_out = [ AL, AR ];

return
```

☛ BACK TO TEXT

Homework 3.2.1.3 In the MATLAB Command Window, type

```
A = zeros( 5,4 )
```

What is the result?

Answer:

☛ BACK TO TEXT

Homework 3.2.1.5 Apply the zero matrix to Timmy Two Space. What happens?

1. Timmy shifts off the grid.
2. Timmy disappears into the origin.
3. Timmy becomes a line on the x-axis.
4. Timmy becomes a line on the y-axis.
5. Timmy doesn't change at all.

Answer: Notice that Timmy disappears... He has been sucked into the origin...

☛ BACK TO TEXT

The Identity Matrix

Homework 3.2.2.1 Let $L_I : \mathbb{R}^n \to \mathbb{R}^n$ be the function defined for every $x \in \mathbb{R}^n$ as $L_I(x) = x$. L_I is a linear transformation.

True/False

Answer: True

- $L_I(\alpha x) = \alpha x = \alpha L(x)$.
- $L_I(x+y) = x+y = L_I(x)+L_I(y)$.

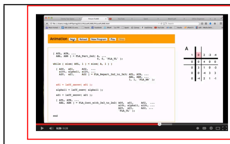
☛ BACK TO TEXT

Homework 3.2.2.2 With the FLAME API for MATLAB (FLAME@lab) implement the algorithm in Figure 3.2. You will use the functions `laff_zerov( x )` and `laff_onev( x )`, which return a zero vector and vector of all ones of the same size and shape (column or row) as input vector `x`, respectively. Try it yourself! (Hint: in Spark, you will want to pick Direction `TL->BR`.) Feel free to look at the below video if you get stuck.

Some links that will come in handy:

- ☛ Spark
  (alternatively, open the file ☛ `LAFF-2.0xM/Spark/index.html`)
- ☛ PictureFLAME
  (alternatively, open the file ☛ `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

You will need these in many future exercises. Bookmark them!



☛ YouTube

Answer:

```
function [ A_out ] = Set_to_identity_unb_var1( A )

  [ AL, AR ] = FLA_Part_1x2( A, ...
                               0, 'FLA_LEFT' );

  while ( size( AL, 2 ) < size( A, 2 ) )

    [ A0, a1, A2 ]= FLA_Repart_1x2_to_1x3( AL, AR, ...
                                          1, 'FLA_RIGHT' );

    %------------------------------------------------------------%

    a01 = laff_zerov( a01 );
    alpha11 = laff_onev( alpha11 );
    a21 = laff_zerov( a21 );

    %------------------------------------------------------------%

    [ AL, AR ] = FLA_Cont_with_1x3_to_1x2( A0, a1, A2, ...
                                            'FLA_LEFT' );

  end

  A_out = [ AL, AR ];

return
```

Here are three more implementations (you may want to view these in PictureFLAME):

```
function [ A_out ] = Set_to_identity_unb_var2( A )
```

```
  [ AL, AR ] = FLA_Part_1x2( A, ...
                               0, 'FLA_LEFT' );

  while ( size( AL, 2 ) < size( A, 2 ) )

    [ A0, a1, A2 ]= FLA_Repart_1x2_to_1x3( AL, AR, ...
                                         1, 'FLA_RIGHT' );

    %------------------------------------------------------------%

    a10t = laff_zerov( a10t );
    alpha11 = laff_onev( alpha11 );
    a12t = laff_zerov( a12t );

    %------------------------------------------------------------%

    [ AL, AR ] = FLA_Cont_with_1x3_to_1x2( A0, a1, A2, ...
                                           'FLA_LEFT' );

  end

  A_out = [ AL, AR ];

return

function [ A_out ] = Set_to_identity_unb_var3( A )

  [ AL, AR ] = FLA_Part_1x2( A, ...
                               0, 'FLA_LEFT' );

  while ( size( AL, 2 ) < size( A, 2 ) )

    [ A0, a1, A2 ]= FLA_Repart_1x2_to_1x3( AL, AR, ...
                                         1, 'FLA_RIGHT' );

    %------------------------------------------------------------%

    a01 = laff_zerov( a01 );
    alpha11 = laff_onev( alpha11 );
    a10t = laff_zerov( a10t );

    %------------------------------------------------------------%

    [ AL, AR ] = FLA_Cont_with_1x3_to_1x2( A0, a1, A2, ...
                                           'FLA_LEFT' );

  end

  A_out = [ AL, AR ];

return

function [ A_out ] = Set_to_identity_unb_var4( A )

  [ AL, AR ] = FLA_Part_1x2( A, ...
                               0, 'FLA_LEFT' );
```

```
  while ( size( AL, 2 ) < size( A, 2 ) )

    [ A0, a1, A2 ]= FLA_Repart_1x2_to_1x3( AL, AR, ...
                                         1, 'FLA_RIGHT' );

    %------------------------------------------------------------%

    a12t = laff_zerov( a12t );
    alpha11 = laff_onev( alpha11 );
    a21 = laff_zerov( a21 );

    %------------------------------------------------------------%

    [ AL, AR ] = FLA_Cont_with_1x3_to_1x2( A0, a1, A2, ...
                                          'FLA_LEFT' );

  end

  A_out = [ AL, AR ];

return
```

☛ BACK TO TEXT

Homework 3.2.2.3 In the MATLAB Command Window, type

```
A = eye( 4,4 )
```

What is the result?

Answer: The result is

```
>> eye( 4, 4)

ans =

     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

☛ BACK TO TEXT

Homework 3.2.2.4 Apply the identity matrix to Timmy Two Space. What happens?

1. Timmy shifts off the grid.
2. Timmy disappears into the origin.
3. Timmy becomes a line on the x-axis.
4. Timmy becomes a line on the y-axis.
5. Timmy doesn't change at all.

Answer: Notice that Timmy doesn't change.

☛ BACK TO TEXT

Homework 3.2.2.5 The trace of a matrix equals the sum of the diagonal elements. What is the trace of the identity $I \in \mathbb{R}^{n\times n}$?

Answer: n

☛ BACK TO TEXT

Diagonal Matrices

Homework 3.2.3.1 Let $A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix}$ and $x = \begin{pmatrix} 2 \\ 1 \\ -2 \end{pmatrix}$. Evaluate Ax.

Answer:

$$Ax = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \\ -2 \end{pmatrix} = \begin{pmatrix} (3)(2) \\ (-1)(1) \\ (2)(-2) \end{pmatrix} = \begin{pmatrix} 6 \\ -1 \\ (-4) \end{pmatrix}.$$

Notice that a diagonal matrix scales individual components of a vector by the corresponding diagonal element of the matrix.

☛ BACK TO TEXT

Homework 3.2.3.2 Let $D = \begin{pmatrix} 2 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & -1 \end{pmatrix}$. What linear transformation, L, does this matrix represent? In particular, answer the following questions:

- $L : \mathbb{R}^n \to \mathbb{R}^m$. What are m and n? $m = n = 3$
- A linear transformation can be described by how it transforms the unit basis vectors:

$$L(e_0) = \begin{pmatrix} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix}; L(e_1) = \begin{pmatrix} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} 0 \\ -3 \\ 0 \end{pmatrix}; L(e_2) = \begin{pmatrix} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$$

- $L(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} 2\chi_0 \\ -3\chi_1 \\ -1\chi_2 \end{pmatrix}$

☛ BACK TO TEXT

Homework 3.2.3.3 Implement a function

```
[ A_out ] = DiagonalMatrix_unb( A, x )
```

based on Figure 3.3.

Answer:

```
function [ A_out ] = DiagonalMatrix_unb( A, x )

  [ ATL, ATR, ...
    ABL, ABR ] = FLA_Part_2x2( A, ...
                               0, 0, 'FLA_TL' );

  [ xT, ...
    xB ] = FLA_Part_2x1( x, ...
                         0, 'FLA_TOP' );

  while ( size( ATL, 1 ) < size( A, 1 ) )

    [ A00,  a01,     A02,  ...
      a10t, alpha11, a12t, ...
      A20,  a21,     A22 ] = FLA_Repart_2x2_to_3x3( ATL, ATR, ...
                                                    ABL, ABR, ...
                                                    1, 1, 'FLA_BR' );

    [ x0, ...
      chi1, ...
      x2 ] = FLA_Repart_2x1_to_3x1( xT, ...
                                    xB, ...
                                    1, 'FLA_BOTTOM' );

    %------------------------------------------------------------%

    a01     = laff_zerov( a01 );
    alpha11 = laff_copy( chi1, alpha11 );
    a21     = laff_zerov( a21 );
    %                            :                               %
    %                      update line n                         %

    %------------------------------------------------------------%

    [ ATL, ATR, ...
      ABL, ABR ] = FLA_Cont_with_3x3_to_2x2( A00,  a01,     A02,  ...
                                             a10t, alpha11, a12t, ...
                                             A20,  a21,     A22,  ...
                                             'FLA_TL' );

    [ xT, ...
      xB ] = FLA_Cont_with_3x1_to_2x1( x0, ...
                                       chi1, ...
                                       x2, ...
                                       'FLA_TOP' );

  end

  A_out = [ ATL, ATR
            ABL, ABR ];

return
```

☛ BACK TO TEXT

Homework 3.2.3.4 Apply the diagonal matrix $\begin{pmatrix} -1 & 0 \\ 0 & 2 \end{pmatrix}$ to Timmy Two Space. What happens?

1. Timmy shifts off the grid.
2. Timmy is rotated.
3. Timmy doesn't change at all.
4. Timmy is flipped with respect to the vertical axis.
5. Timmy is stretched by a factor two in the vertical direction.

Answer: Timmy is flipped with respect to the vertical axis **and** is stretched by a factor two in the vertical direction.

☛ BACK TO TEXT

Homework 3.2.3.5 Compute the trace of $\begin{pmatrix} -1 & 0 \\ 0 & 2 \end{pmatrix}$.

Answer: 1

☛ BACK TO TEXT

Triangular Matrices

Homework 3.2.4.1 Let $L_U : \mathbb{R}^3 \to \mathbb{R}^3$ be defined as $L_U(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = \begin{pmatrix} 2\chi_0 - \chi_1 + \chi_2 \\ 3\chi_1 - \chi_2 \\ -2\chi_2 \end{pmatrix}$. We have proven for similar functions that they are linear transformations, so we will skip that part. What matrix, U, represents this linear transformation?

Answer: $U = \begin{pmatrix} 2 & -1 & 1 \\ 0 & 3 & -1 \\ 0 & 0 & -2 \end{pmatrix}$. (You can either evaluate $L(e_0)$, $L(e_1)$, and $L(e_2)$, or figure this out by examination.)

☛ BACK TO TEXT

Homework 3.2.4.2 A matrix that is both lower and upper triangular is, in fact, a diagonal matrix.

Always/Sometimes/Never

Answer: Always
Let A be both lower and upper triangular. Then $\alpha_{i,j} = 0$ if $i < j$ and $\alpha_{i,j} = 0$ if $i > j$ so that

$$\alpha_{i,j} = \begin{cases} 0 & \text{if } i < j \\ 0 & \text{if } i > j. \end{cases}$$

But this means that $\alpha_{i,j} = 0$ if $i \neq j$, which means A is a diagonal matrix.

☛ BACK TO TEXT

Homework 3.2.4.3 A matrix that is both strictly lower and strictly upper triangular is, in fact, a zero matrix.

Always/Sometimes/Never

Answer: Always

Let A be both strictly lower and strictly upper triangular. Then

$$\alpha_{i,j} = \begin{cases} 0 & \text{if } i \leq j \\ 0 & \text{if } i \geq j \end{cases}.$$

But this means that $\alpha_{i,j} = 0$ for all i and j, which means A is a zero matrix.

☛ BACK TO TEXT

Homework 3.2.4.4 In the above algorithm you could have replaced $a_{01} := 0$ with $a_{12}^T := 0$.

Always/Sometimes/Never

Answer: Always

- $a_{01} = 0$ sets the elements above the diagonal to zero, one column at a time.
- $a_{12}^T = 0$ sets the elements to the right of the diagonal to zero, one row at a time.

☛ BACK TO TEXT

Homework 3.2.4.5 Consider the following algorithm.

Algorithm: $[A] := \text{SET\_TO\_???\_TRIANGULAR\_MATRIX}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

  where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

  where α_{11} is 1×1

?????

  Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Change the ????? in the above algorithm so that it sets A to its

- Upper triangular part. (`Set_to_upper_triangular_matrix_unb`)

  Answer: $a_{21} := 0$ or $a_{10}^T := 0$.

- Strictly upper triangular part. (`Set_to_strictly_upper_triangular_matrix_unb`)

  Answer: $a_{21} := 0; \alpha_{11} := 0$ or $\alpha_{11} := 0; a_{10}^T := 0$.

- Unit upper triangular part. (`Set_to_unit_upper_triangular_matrix_unb`)

  Answer: $a_{21} := 0; \alpha_{11} := 1$ or $\alpha_{11} := 1; a_{10}^T := 0$.

- Strictly lower triangular part. (`Set_to_strictly_lower_triangular_matrix_unb`)

  Answer: $a_{01} := 0; \alpha_{11} := 0$ or $\alpha_{11} := 0; a_{12}^T := 0$.

- Unit lower triangular part. (`Set_to_unit_lower_triangular_matrix_unb`)

  Answer: $a_{01} := 0; \alpha_{11} := 1$ or $\alpha_{11} := 1; a_{12}^T := 0$.

☛ BACK TO TEXT

Homework 3.2.4.6 Implement functions for each of the algorithms from the last homework. In other words, implement functions that, given a matrix A, return a matrix equal to

- the upper triangular part. (`Set_to_upper_triangular_matrix`) **Answer:**

- the strictly upper triangular part. (`Set_to_strictly_upper_triangular_matrix`)

  Answer:

- the unit upper triangular part. (`Set_to_unit_upper_triangular_matrix`)

  Answer:

- strictly lower triangular part. (`Set_to_strictly_lower_triangular_matrix`)

  Answer:

- unit lower triangular part. (`Set_to_unit_lower_triangular_matrix`)

  Answer:

(Implement as many as you enjoy implementing. Then move on.)

☛ BACK TO TEXT

Homework 3.2.4.8 Apply $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ to Timmy Two Space. What happens to Timmy?

1. Timmy shifts off the grid.
2. Timmy becomes a line on the x-axis.
3. Timmy becomes a line on the y-axis.
4. Timmy is skewed to the right.
5. Timmy doesn't change at all.

Answer: Timmy is skewed to the right.

☛ BACK TO TEXT

Transpose Matrix

Homework 3.2.5.1 Let $A = \begin{pmatrix} -1 & 0 & 2 & 1 \\ 2 & -1 & 1 & 2 \\ 3 & 1 & -1 & 3 \end{pmatrix}$ and $x = \begin{pmatrix} -1 \\ 2 \\ 4 \end{pmatrix}$. What are A^T and x^T?

Answer:

$$A^T = \begin{pmatrix} -1 & 0 & 2 & 1 \\ 2 & -1 & 1 & 2 \\ 3 & 1 & -1 & 3 \end{pmatrix} = \begin{pmatrix} -1 & 2 & 3 \\ 0 & -1 & 1 \\ 2 & 1 & -1 \\ 1 & 2 & 3 \end{pmatrix} \text{ and } x^T = \begin{pmatrix} -1 \\ 2 \\ 4 \end{pmatrix}^T = \begin{pmatrix} -1 & 2 & 4 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 3.2.5.2 Consider the following algorithm.

Algorithm: $[B] := \text{TRANSPOSE\_ALTERNATIVE}(A, B)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $B \rightarrow \left(\begin{array}{c|c} B_L & B_R \end{array}\right)$
  where A_T has 0 rows, B_L has 0 columns

while $m(A_T) < m(A)$ **do**

  Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ A_2 \end{array}\right), \left(\begin{array}{c|c} B_L & B_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} B_0 & b_1 & B_2 \end{array}\right)$$

  where a_1 has 1 row, b_1 has 1 column

  Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c|c} B_L & B_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} B_0 & b_1 & B_2 \end{array}\right)$$

endwhile

Modify the above algorithm so that it copies rows of A into columns of B.

Answer: $b_1 := (a_1^T)^T$.

☛ BACK TO TEXT

Homework 3.2.5.3 Implement functions

- `Transpose_unb( A, B )`
- `Transpose_alternative_unb( A, B )`

☛ BACK TO TEXT

Homework 3.2.5.4 The transpose of a lower triangular matrix is an upper triangular matrix.

Always/Sometimes/Never

Answer: Always

Let $L \in \mathbb{R}^{n \times n}$ be lower trangular matrix.

$$L^T = \begin{pmatrix} \lambda_{0,0} & 0 & 0 & \cdots & 0 \\ \lambda_{1,0} & \lambda_{1,1} & 0 & \cdots & 0 \\ \lambda_{2,0} & \lambda_{2,1} & \lambda_{2,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \lambda_{n-1,0} & \lambda_{n-1,1} & \lambda_{n-1,2} & \cdots & \lambda_{n-1,n-1} \end{pmatrix}^T = \begin{pmatrix} \lambda_{0,0} & \lambda_{1,0} & \lambda_{2,0} & \cdots & \lambda_{n-1,0} \\ 0 & \lambda_{1,1} & \lambda_{2,1} & \cdots & \lambda_{n-1,1} \\ 0 & 0 & \lambda_{2,2} & \cdots & \lambda_{n-1,2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_{n-1,n-1} \end{pmatrix}$$

which is upper triangular.

☛ BACK TO TEXT

Homework 3.2.5.5 The transpose of a strictly upper triangular matrix is a strictly lower triangular matrix.

Always/Sometimes/Never

Answer: Always Let $U \in \mathbb{R}^{n \times n}$ be a strictly upper trangular matrix.

$$U^T = \begin{pmatrix} 1 & \upsilon_{1,0} & \upsilon_{2,0} & \cdots & \upsilon_{n-1,0} \\ 0 & 1 & \upsilon_{2,1} & \cdots & \upsilon_{n-1,1} \\ 0 & 0 & 1 & \cdots & \upsilon_{n-1,2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}^T = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \upsilon_{1,0} & 1 & 0 & \cdots & 0 \\ \upsilon_{2,0} & \upsilon_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \upsilon_{n-1,0} & \upsilon_{n-1,1} & \upsilon_{n-1,2} & \cdots & 1 \end{pmatrix}$$

which is strictly lower triangular.

☛ BACK TO TEXT

Homework 3.2.5.6 The transpose of the identity is the identity.

Always/Sometimes/Never

Answer: Always

$$I^T = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}^T = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 3.2.5.7 Evaluate

- $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}^T = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
- $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}^T = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 3.2.5.8 If $A = A^T$ then $A = I$ (the identity).

True/False

Answer: False $A = A^T$ for many matrices.

☛ BACK TO TEXT

Symmetric Matrices

Homework 3.2.6.1 Assume the below matrices are symmetric. Fill in the remaining elements.

$$\begin{pmatrix} 2 & \square & -1 \\ -2 & 1 & -3 \\ \square & \square & -1 \end{pmatrix}; \quad \begin{pmatrix} 2 & \square & \square \\ -2 & 1 & \square \\ -1 & 3 & -1 \end{pmatrix}; \quad \begin{pmatrix} 2 & 1 & -1 \\ \square & 1 & -3 \\ \square & \square & -1 \end{pmatrix}.$$

Answer:

$$\begin{pmatrix} 2 & -2 & -1 \\ -2 & 1 & -3 \\ -1 & -3 & -1 \end{pmatrix}; \quad \begin{pmatrix} 2 & -2 & -1 \\ -2 & 1 & 3 \\ -1 & 3 & -1 \end{pmatrix}; \quad \begin{pmatrix} 2 & 1 & -1 \\ 1 & 1 & -3 \\ -1 & -3 & -1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 3.2.6.2 A triangular matrix that is also symmetric is, in fact, a diagonal matrix. Always/Sometimes/Never

Answer: Always

Without loss of generality is a common expression in mathematical proofs. It is used to argue a specific case that is easier to prove but all other cases can be argued using the same strategy. Thus, given a proof of the conclusion in the special case, it is easy to adapt it to prove the conclusion in all cases. It is often abbreviated as "W.l.o.g.".

W.l.o.g., let A be both symmetric and lower triangular. Then

$$\alpha_{i,j} = 0 \text{ if } i < j$$

since A is lower triangular. But $\alpha_{i,j} = \alpha_{j,i}$ since A is symmetric. We conclude that

$$\alpha_{i,j} = \alpha_{j,i} = 0 \text{ if } i < j.$$

But this means that $\alpha_{i,j} = 0$ if $i \neq j$, which means A is a diagonal matrix.

☛ BACK TO TEXT

Homework 3.2.6.3 In the above algorithm one can replace $a_{01} := a_{10}^T$ by $a_{12}^T = a_{21}$.

Always/Sometimes/Never

Answer: Always

- $a_{01} = (a_{10}^T)^T$ sets the elements above the diagonal to their symmetric counterparts, one column at a time.
- $a_{12}^T = a_{21}^T$ sets the elements to the right of the diagonal to their symmetric counterparts, one row at a time.

☛ BACK TO TEXT

Homework 3.2.6.4 Consider the following algorithm.

Algorithm: $[A] :=$ SYMMETRIZE_FROM_UPPER_TRIANGLE(A)

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

?????

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

What commands need to be introduced between the lines in order to "symmetrize" A assuming that only its upper triangular part is stored initially.

Answer: $a_{21} := (a_{12}^T)^T$ or $a_{21}^T := a_{21}^T$.

☛ BACK TO TEXT

Homework 3.2.6.5 Implement functions

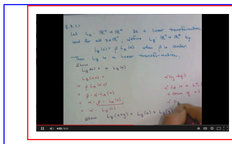
- `Symmetrize_from_lower_triangle_unb( A, B )`
- `Symmetrize_from_upper_triangle_unb( A, B )`

☛ BACK TO TEXT

Operations with Matrices

Scaling a Matrix

Homework 3.3.1.1 Prove the above theorem.



☛ YouTube

Answer:

Let $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$. Then

- $L_B(\alpha x) = \alpha L_B(x)$:
$$L_B(\alpha x) = \beta L_A(\alpha x) = \beta \alpha L_A(x) = \alpha \beta L_A(x) = \alpha L_B(x).$$

- $L_B(x+y) = L_B(x) + L_B(y)$:
$$L_B(x+y) = \beta L_A(x+y) = \beta(L_A(x) + L_A(y)) = \beta L_A(x) + \beta L_A(y) = L_B(x) + L_B(y).$$

Hence L_B is a linear transformation.

☛ BACK TO TEXT

Homework 3.3.1.2 Consider the following algorithm.

Algorithm: $[A] := \text{SCALE\_MATRIX\_ALTERNATIVE}(\beta, A)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right)$
where A_T has 0 rows

while $m(A_T) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ A_2 \end{array} \right)$$
where a_1 has 1 row

?????

Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array} \right) \leftarrow \left(\begin{array}{c} A_0 \\ a_1^T \\ \hline A_2 \end{array} \right)$$

endwhile

What update will scale A one row at a time?

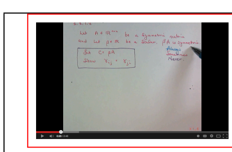
Answer: $a_1^T := \beta a_1^T$

☛ BACK TO TEXT

Homework 3.3.1.3 Implement function `Scale_matrix_unb( beta, A )`.

☛ BACK TO TEXT

Homework 3.3.1.4



☛ YouTube

Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix and $\beta \in \mathbb{R}$ a scalar, βA is symmetric.

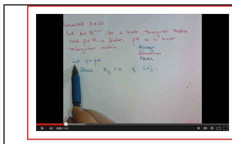
Always/Sometimes/Never

Answer: Always

Let $C = \beta A$. We need to show that $\gamma_{i,j} = \gamma_{j,i}$. But $\gamma_{i,j} = \beta\alpha_{i,j} = \beta\alpha_{j,i} = \gamma_{j,i}$, since A is symmetric. Hence C is symmetric.

☛ BACK TO TEXT

Homework 3.3.1.5



☛ YouTube

Let $A \in \mathbb{R}^{n \times n}$ be a lower triangular matrix and $\beta \in \mathbb{R}$ a scalar, βA is a lower triangular matrix.

Always/Sometimes/Never

Answer: Always

Assume A is a lower triangular matrix. Then $\alpha_{i,j} = 0$ if $i < j$.

Let $C = \beta A$. We need to show that $\gamma_{i,j} = 0$ if $i < j$. But if $i < j$, then $\gamma_{i,j} = \beta\alpha_{i,j} = \beta \times 0 = 0$ since A is lower triangular. Hence C is lower triangular.

☛ BACK TO TEXT

Homework 3.3.1.6 Let $A \in \mathbb{R}^{n \times n}$ be a diagonal matrix and $\beta \in \mathbb{R}$ a scalar, βA is a diagonal matrix.

Always/Sometimes/Never

Answer: Always

Assume A is a diagonal matrix. Then $\alpha_{i,j} = 0$ if $i \neq j$.

Let $C = \beta A$. We need to show that $\gamma_{i,j} = 0$ if $i \neq j$. But if $i \neq j$, then $\gamma_{i,j} = \beta\alpha_{i,j} = \beta \times 0 = 0$ since A is a diagonal matrix. Hence C is a diagonal matrix.

☛ BACK TO TEXT

Homework 3.3.1.7 Let $A \in \mathbb{R}^{m \times n}$ be a matrix and $\beta \in \mathbb{R}$ a scalar, $(\beta A)^T = \beta A^T$.

Always/Sometimes/Never

Answer: Always

$$
\begin{aligned}
(\beta A)^T &= \left(\beta \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \right)^T \\
&= \begin{pmatrix} \beta\alpha_{0,0} & \beta\alpha_{0,1} & \cdots & \beta\alpha_{0,n-1} \\ \beta\alpha_{1,0} & \beta\alpha_{1,1} & \cdots & \beta\alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta\alpha_{m-1,0} & \beta\alpha_{m-1,1} & \cdots & \beta\alpha_{m-1,n-1} \end{pmatrix}^T \\
&= \begin{pmatrix} \beta\alpha_{0,0} & \beta\alpha_{1,0} & \cdots & \beta\alpha_{m-1,0} \\ \beta\alpha_{0,1} & \beta\alpha_{1,1} & \cdots & \beta\alpha_{m-1,1} \\ \vdots & \vdots & & \vdots \\ \beta\alpha_{0,n-1} & \beta\alpha_{1,n-1} & \cdots & \beta\alpha_{m-1,n-1} \end{pmatrix}
\end{aligned}
$$

$$
= \beta \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{m-1,0} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{m-1,1} \\ \vdots & \vdots & & \vdots \\ \alpha_{0,n-1} & \alpha_{1,n-1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix}
$$

$$
= \beta \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix}^T = \beta A^T
$$

☛ BACK TO TEXT

Adding Matrices

Homework 3.3.2.1 The sum of two linear transformations is a linear transformation. More formally: Let $L_A : \mathbb{R}^n \to \mathbb{R}^m$ and $L_B : \mathbb{R}^n \to \mathbb{R}^m$ be two linear transformations. Let $L_C : \mathbb{R}^n \to \mathbb{R}^m$ be defined by $L_C(x) = L_A(x) + L_B(x)$. L_C is a linear transformation.

Always/Sometimes/Never

Answer: Always.

To show that L_C is a linear transformation, we must show that $L_C(\alpha x) = \alpha L_C(x)$ and $L_C(x+y) = L_C(x) + L_C(y)$.

- $L_C(\alpha x) = \alpha L_C(x)$**:**

$$
L_C(\alpha x) = L_A(\alpha x) + L_B(\alpha x) = \alpha L_A(x) + \alpha L_B(x) = \alpha \left(L_A(x) + L_B(x) \right).
$$

- $L_C(x+y) = L_C(x) + L_C(y)$**:**

$$
\begin{aligned}
L_C(x+y) &= L_A(x+y) + L_B(x+y) = L_A(x) + L_A(y) + L_B(x) + L_B(y) \\
&= L_A(x) + L_B(x) + L_A(y) + L_B(y) = L_C(x) + L_C(y).
\end{aligned}
$$

☛ BACK TO TEXT

Homework 3.3.2.2 Consider the following algorithm.

Algorithm: $[A] := \text{ADD\_MATRICES\_ALTERNATIVE}(A, B)$

Partition $A \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right)$

where A_T has 0 rows, B_T has 0 rows

while $m(A_T) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \rightarrow \left(\begin{array}{c} A_0 \\ \hline a_1^T \\ A_2 \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline b_1^T \\ B_2 \end{array}\right)$$

where a_1 has 1 row, b_1 has 1 row

Continue with

$$\left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right) \leftarrow \left(\begin{array}{c} A_0 \\ a_1^T \\ \hline A_2 \end{array}\right), \left(\begin{array}{c} B_T \\ \hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ b_1^T \\ \hline B_2 \end{array}\right)$$

endwhile

What update will add B to A one row at a time, overwriting A with the result?

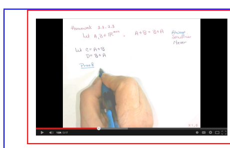
Answer: $a_1^T := a_1^T + b_1^T$

☛ BACK TO TEXT

Homework 3.3.2.3 Let $A, B \in \mathbb{R}^{m \times n}$. $A + B = B + A$.

Always/Sometimes/Never

Answer: Always



☛ YouTube

Proof 1: Let $C = A + B$ and $D = B + A$. We need to show that $C = D$. But

$$\gamma_{i,j} = \alpha_{i,j} + \beta_{i,j} = \beta_{i,j} + \alpha_{i,j} = \delta_{i,j}.$$

Hence $C = D$.

Proof 2:

$$\begin{aligned} A + B &= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} + \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0} & \beta_{m-1,1} & \cdots & \beta_{m-1,n-1} \end{pmatrix} \\ &= \begin{pmatrix} \alpha_{0,0} + \beta_{0,0} & \alpha_{0,1} + \beta_{0,1} & \cdots & \alpha_{0,n-1} + \beta_{0,n-1} \\ \alpha_{1,0} + \beta_{1,0} & \alpha_{1,1} + \beta_{1,1} & \cdots & \alpha_{1,n-1} + \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} + \beta_{m-1,0} & \alpha_{m-1,1} + \beta_{m-1,1} & \cdots & \alpha_{m-1,n-1} + \beta_{m-1,n-1} \end{pmatrix} \end{aligned}$$

$$
\begin{aligned}
&= \begin{pmatrix} \beta_{0,0}+\alpha_{0,0} & \beta_{0,1}+\alpha_{0,1} & \cdots & \beta_{0,n-1}+\alpha_{0,n-1} \\ \beta_{1,0}+\alpha_{1,0} & \beta_{1,1}+\alpha_{1,1} & \cdots & \beta_{1,n-1}+\alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0}+\alpha_{m-1,0} & \beta_{m-1,1}+\alpha_{m-1,1} & \cdots & \beta_{m-1,n-1}+\alpha_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0} & \beta_{m-1,1} & \cdots & \beta_{m-1,n-1} \end{pmatrix} + \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \\
&= B+A
\end{aligned}
$$

☛ BACK TO TEXT

Homework 3.3.2.4 Let $A,B,C \in \mathbb{R}^{m\times n}$. $(A+B)+C = A+(B+C)$.

Always/Sometimes/Never

Answer: Always

Let's introduce the notation $(A)_{i,j}$ for the i,j element of A. Then

$$
\begin{aligned}
((A+B)+C)_{i,j} &= (A+B)_{i,j}+(C)_{i,j} = ((A)_{i,j}+(B)_{i,j})+(C)_{i,j} = (A)_{i,j}+((B)_{i,j}+(C)_{i,j}) \\
&= (A)_{i,j}+(B+C)_{i,j} = (A+(B+C))_{i,j}.
\end{aligned}
$$

Hence $(A+B)+C = A+(B+C)$.

☛ BACK TO TEXT

Homework 3.3.2.5 Let $A,B \in \mathbb{R}^{m\times n}$ and $\gamma \in \mathbb{R}$. $\gamma(A+B) = \gamma A+\gamma B$.

Always/Sometimes/Never

Answer: Always

(Using the notation from the last proof.)

$$
(\gamma(A+B))_{i,j} = \gamma(A+B)_{i,j} = \gamma((A)_{i,j}+(B)_{i,j}) = \gamma(A)_{i,j}+\gamma(B)_{i,j} = (\gamma A+\gamma B)_{i,j}
$$

Hence, the i,j element of $\gamma(A+B)$ equals the i,j element of $\gamma A+\gamma B$, establishing the desired result.

☛ BACK TO TEXT

Homework 3.3.2.6 Let $A \in \mathbb{R}^{m\times n}$ and $\beta,\gamma \in \mathbb{R}$. $(\beta+\gamma)A = \beta A+\gamma A$.

Always/Sometimes/Never

Answer: Always

(Using the notation from the last proof.)

$$
((\beta+\gamma)A)_{i,j} = (\beta+\gamma)(A)_{i,j} = \beta(A)_{i,j}+\gamma(A)_{i,j} = (\beta A)_{i,j}+(\gamma A)_{i,j}.
$$

Hence, the i,j element of $(\beta+\gamma)A$ equals the i,j element of $\beta A+\gamma A$, establishing the desired result.

☛ BACK TO TEXT

Homework 3.3.2.7 Let $A, B \in \mathbb{R}^{n \times n}$. $(A+B)^T = A^T + B^T$.

Always/Sometimes/Never

Answer: Always

$$
\begin{aligned}
(A+B)^T &= \left(\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} + \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0} & \beta_{m-1,1} & \cdots & \beta_{m-1,n-1} \end{pmatrix} \right)^T \\
&= \begin{pmatrix} \alpha_{0,0}+\beta_{0,0} & \alpha_{0,1}+\beta_{0,1} & \cdots & \alpha_{0,n-1}+\beta_{0,n-1} \\ \alpha_{1,0}+\beta_{1,0} & \alpha_{1,1}+\beta_{1,1} & \cdots & \alpha_{1,n-1}+\beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0}+\beta_{m-1,0} & \alpha_{m-1,1}+\beta_{m-1,1} & \cdots & \alpha_{m-1,n-1}+\beta_{m-1,n-1} \end{pmatrix}^T \\
&= \begin{pmatrix} \alpha_{0,0}+\beta_{0,0} & \alpha_{1,0}+\beta_{1,0} & \cdots & \alpha_{m-1,0}+\beta_{m-1,0} \\ \alpha_{0,1}+\beta_{0,1} & \alpha_{1,1}+\beta_{1,1} & \cdots & \alpha_{m-1,1}+\beta_{m-1,1} \\ \vdots & \vdots & & \vdots \\ \alpha_{0,n-1}+\beta_{0,n-1} & \alpha_{1,n-1}+\beta_{1,n-1} & \cdots & \alpha_{m-1,n-1}+\beta_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{m-1,0} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{m-1,1} \\ \vdots & \vdots & & \vdots \\ \alpha_{0,n-1} & \alpha_{1,n-1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} + \begin{pmatrix} \beta_{0,0} & \beta_{1,0} & \cdots & \beta_{m-1,0} \\ \beta_{0,1} & \beta_{1,1} & \cdots & \beta_{m-1,1} \\ \vdots & \vdots & & \vdots \\ \beta_{0,n-1} & \beta_{1,n-1} & \cdots & \beta_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix}^T + \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0} & \beta_{m-1,1} & \cdots & \beta_{m-1,n-1} \end{pmatrix}^T \\
&= A^T + B^T
\end{aligned}
$$

☛ BACK TO TEXT

Homework 3.3.2.8 Let $A, B \in \mathbb{R}^{n \times n}$ be symmetric matrices. $A+B$ is symmetric.

Always/Sometimes/Never

Answer: Always

Let $C = A+B$. We need to show that $\gamma_{j,i} = \gamma_{i,j}$.

$$
\begin{aligned}
& \gamma_{j,i} \\
= & \quad < \text{Definition of matrix addition} > \\
& \alpha_{j,i} + \beta_{j,i} \\
= & \quad < A \text{ and } B \text{ are symmetric} > \\
& \alpha_{i,j} + \beta_{i,j} \\
= & \quad < \text{Definition of matrix addition} > \\
& \gamma_{i,j}
\end{aligned}
$$

☛ BACK TO TEXT

Homework 3.3.2.9 Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric matrices. $A - B$ is symmetric.

Always/Sometimes/Never

Answer: Always

Let $C = A - B$. We need to show that $\gamma_{j,i} = \gamma_{i,j}$.

$$
\begin{array}{ll}
 & \gamma_{j,i} \\
= & \quad < \text{Definition of matrix addition} > \\
 & \alpha_{j,i} - \beta_{j,i} \\
= & \quad < A \text{ and } B \text{ are symmetric} > \\
 & \alpha_{i,j} - \beta_{i,j} \\
= & \quad < \text{Definition of matrix addition} > \\
 & \gamma_{i,j}
\end{array}
$$

☛ BACK TO TEXT

Homework 3.3.2.10 Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric matrices and $\alpha, \beta \in \mathbb{R}$. $\alpha A + \beta B$ is symmetric.

Always/Sometimes/Never

Answer: Always

Let $C = \alpha A + \beta B$. We need to show that $\gamma_{j,i} = \gamma_{i,j}$. The proof is similar to many proofs we have seen.

☛ BACK TO TEXT

Homework 3.3.2.11 Let $A, B \in \mathbb{R}^{n\times n}$.

If A and B are lower triangular matrices then $A + B$ is lower triangular.

True/False

Answer: True

Let $C = A + B$. We know that $\alpha_{i,j} = \beta_{i,j} = 0$ if $i < j$. We need to show that $\gamma_{i,j} = 0$ if $i < j$. But if $i < j$ then $\gamma_{i,j} = \alpha_{i,j} + \beta_{i,j} = 0$.

If A and B are strictly lower triangular matrices then $A + B$ is strictly lower triangular.

True/False

Answer: True

If A and B are unit lower triangular matrices then $A + B$ is unit lower triangular.

True/False

Answer: False! The diagonal of $A + B$ has 2's on it!

If A and B are upper triangular matrices then $A + B$ is upper triangular.

True/False

Answer: True

If A and B are strictly upper triangular matrices then $A + B$ is strictly upper triangular.

True/False

Answer: True

If A and B are unit upper triangular matrices then $A + B$ is unit upper triangular.

True/False

Answer: False! The diagonal of $A+B$ has 2's on it!

☛ BACK TO TEXT

Homework 3.3.2.12 Let $A, B \in \mathbb{R}^{n \times n}$.

If A and B are lower triangular matrices then $A-B$ is lower triangular.

True/False

Answer: True

Let $C = A - B$. We know that $\alpha_{i,j} = \beta_{i,j} = 0$ if $i < j$. We need to show that $\gamma_{i,j} = 0$ if $i < j$. But if $i < j$ then $\gamma_{i,j} = \alpha_{i,j} - \beta_{i,j} = 0 - 0 = 0$.

If A and B are strictly lower triangular matrices then $A-B$ is strictly lower triangular. True/False

Answer: True

If A and B are unit lower triangular matrices then $A-B$ is *strictly* lower triangular. True/False

Answer: True! The diagonal of $A+B$ has 0's on it!

If A and B are upper triangular matrices then $A-B$ is upper triangular. True/False

Answer: True

If A and B are strictly upper triangular matrices then $A-B$ is strictly upper triangular. True/False

Answer: True

If A and B are unit upper triangular matrices then $A-B$ is unit upper triangular. True/False

Answer: False! The diagonal of $A+B$ has 0's on it! (Sorry, you need to read carefully.)

☛ BACK TO TEXT

Via Dot Products

Homework 3.5.1.1 Implement function `Mvmult_n_unb_var1( A, x, y )`.

☛ BACK TO TEXT

Via AXPY Operations

Homework 3.5.2.1 Implement function `Mvmult_n_unb_var2( A, x, y )`.
(Hint: use the function `laff_dots( x, y, alpha )` that updates $\alpha := x^T y + \alpha$.)

☛ BACK TO TEXT

Week 4: From Matrix-Vector Multiplication to Matrix-Matrix Multiplication (Answers)

4.1 Opening Remarks

4.1.1 Predicting the Weather

Homework 4.1.1.1 If today is cloudy, what is the probability that tomorrow is

- sunny?
- cloudy?
- rainy?

Answer: To answer this, we simply consult the table:

- sunny? 0.3
- cloudy? 0.3
- rainy? 0.4

☛ BACK TO TEXT

Homework 4.1.1.2 If today is sunny, what is the probability that the day after tomorrow is sunny? cloudy? rainy?
Answer:

- The probability that it will be sunny the day after tomorrow and sunny tomorrow is 0.4×0.4.
- The probability that it will sunny the day after tomorrow and cloudy tomorrow is 0.3×0.4.
- The probability that it will sunny the day after tomorrow and rainy tomorrow is 0.1×0.2.

Thus, the probability that it will be sunny the day after tomorrow, if it is sunny today, is $0.4 \times 0.4 + 0.3 \times 0.4 + 0.1 \times 0.2 = 0.30$.

Notice that this is the inner product of the vector that is the row for "Tomorrow is sunny" and the column for "Today is cloudy".

By similar arguments, the probability that it is cloudy the day after tomorrow is 0.40 and the probability that it is rainy is 0.3.

☛ BACK TO TEXT

Homework 4.1.1.3 Follow the instructions in the above video

☛ BACK TO TEXT

Homework 4.1.1.4 Given

		Today		
		sunny	cloudy	rainy
Tomorrow	sunny	0.4	0.3	0.1
	cloudy	0.4	0.3	0.6
	rainy	0.2	0.4	0.3

fill in the following table, which predicts the weather the day after tomorrow given the weather today:

		Today		
		sunny	cloudy	rainy
Day after Tomorrow	sunny			
	cloudy			
	rainy			

Now here is the hard part: Do so without using your knowledge about how to perform a matrix-matrix multiplication, since you won't learn about that until later this week... May we suggest that you instead use MATLAB to perform the necessary calculations.

Answer: By now surely you have noticed that the jth column of a matrix A, a_j, equals Ae_j. So, the jth column of Q equals Qe_j. Now, using e_0 as an example,

$$\begin{aligned} q_0 = Qe_0 &= P(Pe_o) = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \left(\begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right) \\ &= \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 0.4 \\ 0.4 \\ 0.2 \end{pmatrix} = \begin{pmatrix} 0.3 \\ 0.4 \\ 0.3 \end{pmatrix} \end{aligned}$$

The other columns of Q can be computed similarly:

		Today		
		sunny	cloudy	rainy
Day after Tomorrow	sunny	0.30	0.25	0.25
	cloudy	0.40	0.45	0.40
	rainy	0.30	0.30	0.35

☛ BACK TO TEXT

4.2 Preparation

4.2.1 Partitioned Matrix-Vector Multiplication

Homework 4.2.1.1 Consider

$$A = \begin{pmatrix} -1 & 2 & 4 & 1 & 0 \\ 1 & 0 & -1 & -2 & 1 \\ 2 & -1 & 3 & 1 & 2 \\ 1 & 2 & 3 & 4 & 3 \\ -1 & -2 & 0 & 1 & 2 \end{pmatrix} \quad \text{and} \quad x = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix},$$

and partition these into submatrices (regions) as follows:

$$\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \quad \text{and} \quad \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right),$$

where $A_{00} \in \mathbb{R}^{3x3}$, $x_0 \in \mathbb{R}^3$, α_{11} is a scalar, and χ_1 is a scalar. Show with lines how A and x are partitioned:

$$\begin{pmatrix} -1 & 2 & 4 & 1 & 0 \\ 1 & 0 & -1 & -2 & 1 \\ 2 & -1 & 3 & 1 & 2 \\ 1 & 2 & 3 & 4 & 3 \\ -1 & -2 & 0 & 1 & 2 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}.$$

Answer:

$$\left(\begin{array}{ccc|c|c} -1 & 2 & 4 & 1 & 0 \\ 1 & 0 & -1 & -2 & 1 \\ 2 & -1 & 3 & 1 & 2 \\ \hline 1 & 2 & 3 & 4 & 3 \\ \hline -1 & -2 & 0 & 1 & 2 \end{array}\right) \quad \left(\begin{array}{c} 1 \\ 2 \\ 3 \\ \hline 4 \\ \hline 5 \end{array}\right).$$

☛ BACK TO TEXT

Homework 4.2.1.2 With the partitioning of matrices A and x in the above exercise, repeat the partitioned matrix-vector multiplication, similar to how this unit started.

Answer:

$$y = \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right) = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right) = \left(\begin{array}{ccccc} A_{00}x_0 & + & a_{01}\chi_1 & + & A_{02}x_2 \\ \hline a_{10}^T x_0 & + & \alpha_{11}\chi_1 & + & a_{12}^T x_2 \\ \hline A_{20}x_0 & + & a_{21}\chi_1 & + & A_{22}x_2 \end{array}\right)$$

$$
= \left(\begin{array}{c}
\begin{pmatrix} -1 & 2 & 4 \\ 1 & 0 & -1 \\ 2 & -1 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} 4 + \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} 5 \\ \hline
\begin{pmatrix} 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + \begin{pmatrix} 4 \end{pmatrix} 4 + \begin{pmatrix} 3 \end{pmatrix} \begin{pmatrix} 5 \end{pmatrix} \\ \hline
\begin{pmatrix} -1 & -2 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + \begin{pmatrix} 1 \end{pmatrix} 4 + \begin{pmatrix} 2 \end{pmatrix} 5
\end{array} \right) =
$$

$$
\left(\begin{array}{ccc}
\begin{pmatrix} (-1)\times(1)+(2)\times(2)+(4)\times(3) \\ (1)\times(1)+(0)\times(2)+(-1)\times(3) \\ (2)\times(1)+(-1)\times(2)+(3)\times(3) \end{pmatrix} + & \begin{pmatrix} (1)\times(4) \\ (-2)\times(4) \\ (1)\times(4) \end{pmatrix} + & \begin{pmatrix} (0)\times(5) \\ (1)\times(5) \\ (2)\times(5) \end{pmatrix} \\ \hline
(1)\times(1)+(2)\times(2)+(3)\times(3)+ & (4)\times(4)+ & (3)\times(5) \\ \hline
(-1)\times(1)+(-2)\times(2)+(0)\times(3)+ & (1)\times(4)+ & (2)\times(5)
\end{array} \right) =
$$

$$
\left(\begin{array}{c}
(-1)\times(1)+(2)\times(2)+(4)\times(3)+(1)\times(4)+(0)\times(5) \\
(1)\times(1)+(0)\times(2)+(-1)\times(3)+(-2)\times(4)+(1)\times(5) \\
(2)\times(1)+(-1)\times(2)+(3)\times(3)+(1)\times(4)+(2)\times(5) \\ \hline
(1)\times(1)+(2)\times(2)+(3)\times(3)+(4)\times(4)+(3)\times(5) \\ \hline
(-1)\times(1)+(-2)\times(2)+(0)\times(3)+(1)\times(4)+(2)\times(5)
\end{array} \right) = \left(\begin{array}{c} 19 \\ -5 \\ 23 \\ \hline 45 \\ \hline 9 \end{array} \right)
$$

☛ BACK TO TEXT

4.2.2 Transposing a Partitioned Matrix

Homework 4.2.2.1 Show, step-by-step, how to transpose

$$
\left(\begin{array}{cc|cc} 1 & -1 & 3 & 2 \\ 2 & -2 & 1 & 0 \\ \hline 0 & -4 & 3 & 2 \end{array} \right)
$$

Answer:

$$
\left(\begin{array}{cc|cc} 1 & -1 & 3 & 2 \\ 2 & -2 & 1 & 0 \\ \hline 0 & -4 & 3 & 2 \end{array} \right)^T = \left(\begin{array}{c|c} \begin{pmatrix} 1 & -1 \\ 2 & -2 \end{pmatrix} & \begin{pmatrix} 3 & 2 \\ 1 & 0 \end{pmatrix} \\ \hline \begin{pmatrix} 0 & -4 \end{pmatrix} & \begin{pmatrix} 3 & 2 \end{pmatrix} \end{array} \right)^T = \left(\begin{array}{c|c} \begin{pmatrix} 1 & -1 \\ 2 & -2 \end{pmatrix}^T & \begin{pmatrix} 0 & -4 \end{pmatrix}^T \\ \hline \begin{pmatrix} 3 & 2 \\ 1 & 0 \end{pmatrix}^T & \begin{pmatrix} 3 & 2 \end{pmatrix}^T \end{array} \right)
$$

$$
= \left(\begin{array}{c|c} \begin{pmatrix} 1 & 2 \\ -1 & -2 \end{pmatrix} & \begin{pmatrix} 0 \\ -4 \end{pmatrix} \\ \hline \begin{pmatrix} 3 & 1 \\ 2 & 0 \end{pmatrix} & \begin{pmatrix} 3 \\ 2 \end{pmatrix} \end{array} \right) = \left(\begin{array}{cc|c} 1 & 2 & 0 \\ -1 & -2 & -4 \\ \hline 3 & 1 & 3 \\ 2 & 0 & 2 \end{array} \right)
$$

☛ BACK TO TEXT

Homework 4.2.2.2 Transpose the following matrices:

1. $\left(\begin{array}{c} 3 \end{array}\right)^T = \left(\begin{array}{c} 3^T \end{array}\right) = \left(\begin{array}{c} 3 \end{array}\right)$

2. $\left(\begin{array}{c} 3 \\ 1 \\ \hline 1 \\ \hline 8 \end{array}\right)^T = \left(\begin{array}{c|c|c} \left(\begin{array}{c} 3 \\ 1 \end{array}\right)^T & \left(\begin{array}{c} 1 \end{array}\right)^T & \left(\begin{array}{c} 8 \end{array}\right)^T \end{array}\right) = \left(\begin{array}{cc|c|c} 3 & 1 & 1 & 8 \end{array}\right)$

3. $\left(\begin{array}{cc|c|c} 3 & 1 & 1 & 8 \end{array}\right)^T = \left(\begin{array}{c} \left(\begin{array}{cc} 3 & 1 \end{array}\right)^T \\ \hline \left(\begin{array}{c} 1^T \end{array}\right) \\ \hline \left(\begin{array}{c} 8^T \end{array}\right) \end{array}\right) = \left(\begin{array}{c} 3 \\ 1 \\ \hline 1 \\ \hline 8 \end{array}\right)$

4. $\left(\begin{array}{c|c|c|c} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{array}\right)^T = \left(\begin{array}{ccc} 1 & 5 & 9 \\ \hline 2 & 6 & 10 \\ \hline 3 & 7 & 11 \\ \hline 4 & 8 & 12 \end{array}\right)$

5. $\left(\begin{array}{c|c|c} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{array}\right)^T = \left(\begin{array}{c|c|c|c} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{array}\right)$

6. $\left(\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \end{array}\right)^T = \left(\begin{array}{c|c} \left(\begin{array}{cc} 1 & 2 \\ 5 & 6 \end{array}\right)^T & \left(\begin{array}{cc} 3 & 4 \\ 7 & 8 \end{array}\right)^T \\ \hline \left(\begin{array}{cc} 9 & 10 \end{array}\right)^T & \left(\begin{array}{cc} 11 & 12 \end{array}\right)^T \end{array}\right) = \left(\begin{array}{cc|c} 1 & 5 & 9 \\ 2 & 6 & 10 \\ \hline 3 & 7 & 11 \\ 4 & 8 & 12 \end{array}\right)$

7. $\left(\left(\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \end{array}\right)^T\right)^T = \left(\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ \hline 9 & 10 & 11 & 12 \end{array}\right)$ (You are transposing twice...)

☛ BACK TO TEXT

4.3 Matrix-Vector Multiplication with Special Matrices

4.3.1 Transpose Matrix-Vector Multiplication

Homework 4.3.1.1 Implement the routines

- `[ y_out ] = Mvmult_t_unb_var1( A, x, y )`; and
- `[ y_out ] = Mvmult_t_unb_var2( A, x, y )`

that compute $y := A^T x + y$ via the algorithms in Figure 4.3.

☛ BACK TO TEXT

Homework 4.3.1.2 Implementations achieve better performance (finish faster) if one accesses data consecutively in memory. Now, most scientific computing codes store matrices in "column-major order" which means that the first column of a matrix is stored consecutively in memory, then the second column, and so forth. Now, this means that an algorithm that accesses a matrix by columns tends to be faster than an algorithm that accesses a matrix by rows. That, in turn, means that when one is presented with more than one algorithm, one should pick the algorithm that accesses the matrix by columns.

Our FLAME notation makes it easy to recognize algorithms that access the matrix by columns.

- For the matrix-vector multiplication $y := Ax + y$, would you recommend the algorithm that uses dot products or the algorithm that uses `axpy` operations?
- For the matrix-vector multiplication $y := A^T x + y$, would you recommend the algorithm that uses dot products or the algorithm that uses `axpy` operations?

Answer: When computing $Ax + y$, it is when you view Ax as taking linear combinations of the *columns* of A that you end up accessing the matrix by columns. Hence, the `axpy`-based algorithm will access the matrix by columns.

When computing $A^T x + y$, it is when you view $A^T x$ as taking dot products of *columns* of A with vector x that you end up accessing the matrix by columns. Hence, the `dot`-based algorithm will access the matrix by columns.

The important thing is: one algorithm doesn't fit all situations. So, it is important to be aware of all algorithms for computing an operation.

☛ BACK TO TEXT

4.3.2 Triangular Matrix-Vector Multiplication

Homework 4.3.2.1 Write routines

- `[ y_out ] = Trmvp_un_unb_var1 ( U, x, y )`; and
- `[ y_out ] = Trmvp_un_unb_var2( U, x, y )`

that implement the algorithms in Figure 4.4 that compute $y := Ux + y$.

☛ BACK TO TEXT

Homework 4.3.2.2 Modify the algorithms in Figure 4.5 so that they compute $y := Lx + y$, where L is a lower triangular matrix: (Just strike out the parts that evaluate to zero. We suggest you do this homework in conjunction with the next one.)

Answer:

☛ BACK TO TEXT

Homework 4.3.2.3 Write the functions

- `[ y_out ] = Trmvp_ln_unb_var1 ( L, x, y )`; and
- `[ y_out ] = Trmvp_ln_unb_var2( L, x, y )`

that implement thenalgorithms for computing $y := Lx + y$ from Homework 4.3.2.2.

☛ BACK TO TEXT

Homework 4.3.2.4 Modify the algorithms in Figure 4.6 to compute $x := Ux$, where U is an upper triangular matrix. **You may not use y. You have to overwrite x without using work space.** Hint: Think carefully about the order in which elements of x are computed and overwritten. You may want to do this exercise hand-in-hand with the implementation in the next homework.

Answer:

☛ BACK TO TEXT

Homework 4.3.2.5 Write routines

- `[ x_out ] = Trmv_un_unb_var1 ( U, x )`; and
- `[ x_out ] = Trmv_un_unb_var2( U, x )`

that implement the algorithms for computing $x := Ux$ from Homework 4.3.2.4.

☛ BACK TO TEXT

Homework 4.3.2.6 Modify the algorithms in Figure 4.7 to compute $x := Lx$, where L is a lower triangular matrix. **You may not use y. You have to overwrite x without using work space.** Hint: Think carefully about the order in which elements of x are computed and overwritten. This question is VERY tricky... You may want to do this exercise hand-in-hand with the implementation in the next homework.

Answer: The key is that you have to march through the matrix and vector from the "bottom-right" to the "top-left". In other words, in the opposite direction!

☛ BACK TO TEXT

Homework 4.3.2.7 Write routines

- `[ y_out ] = Trmv_ln_unb_var1 ( L, x )`; and
- `[ y_out ] = Trmv_ln_unb_var2( L, x )`

that implement the algorithms from Homework 4.3.2.6 for computing $x := Lx$.

☛ BACK TO TEXT

Homework 4.3.2.8 Develop algorithms for computing $y := U^T x + y$ and $y := L^T x + y$, where U and L are respectively upper triangular and lower triangular. Do not explicitly transpose matrices U and L. Write routines

- `[ y_out ] = Trmvp_ut_unb_var1 ( U, x, y )`; and
- `[ y_out ] = Trmvp_ut_unb_var2( U, x, y )`
- `[ y_out ] = Trmvp_lt_unb_var1 ( L, x, y )`; and
- `[ y_out ] = Trmvp_ln_unb_var2( L, x, y )`

that implement these algorithms.

☛ BACK TO TEXT

Homework 4.3.2.9 Develop algorithms for computing $x := U^T x$ and $x := L^T x$, where U and L are respectively upper triangular and lower triangular. Do not explicitly transpose matrices U and L. Write routines

- `[ y_out ] = Trmv_ut_unb_var1 ( U, x )`; and
- `[ y_out ] = Trmv_ut_unb_var2( U, x )`
- `[ y_out ] = Trmv_lt_unb_var1 ( L, x )`; and

- `[ y_out ] = Trmv_ln_unb_var2( L, x )`

that implement these algorithms.

☛ BACK TO TEXT

Homework 4.3.2.10 Compute the cost, in flops, of the algorithm for computing $y := Lx + y$ that uses AXPY s.

Answer: For the `axpy` based algorithm, the cost is in the updates

- $\psi_1 := \lambda_{11}\chi_1 + \psi_1$ (which requires two flops) ; followed by
- $y_2 := \chi_1 l_{21} + y_2$.

Now, during the first iteration, y_2 and l_{21} and x_2 are of length $n-1$, so that that iteration requires $2(n-1)+2 = 2n$ flops. During the kth iteration (starting with $k = 0$), y_2 and l_{21} are of length $(n-k-1)$ so that the cost of that iteration is $2(n-k-1)+2 = 2(n-k)$ flops. Thus, if L is an $n \times n$ matrix, then the total cost is given by

$$\sum_{k=0}^{n-1}[2(n-k)] = 2\sum_{k=0}^{n-1}(n-k) = 2(n+(n-1)+\cdots+1) = 2\sum_{k=1}^{n} k = 2(n+1)n/2.$$

flops. (Recall that we proved in the second week that $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.)

☛ BACK TO TEXT

Homework 4.3.2.11 As hinted at before: Implementations achieve better performance (finish faster) if one accesses data consecutively in memory. Now, most scientific computing codes store matrices in “column-major order” which means that the first column of a matrix is stored consecutively in memory, then the second column, and so forth. Now, this means that an algorithm that accesses a matrix by columns tends to be faster than an algorithm that accesses a matrix by rows. That, in turn, means that when one is presented with more than one algorithm, one should pick the algorithm that accesses the matrix by columns.

Our FLAME notation makes it easy to recognize algorithms that access the matrix by columns. For example, in this unit, if the algorithm accesses submatrix a_{01} or a_{21} then it accesses columns. If it accesses submatrix a_{10}^T or a_{12}^T, then it accesses the matrix by rows.

For each of these, which algorithm accesses the matrix by columns:

- For $y := Ux + y$, TRSVP_UN_UNB_VAR1 or TRSVP_UN_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?
- For $y := Lx + y$, TRSVP_LN_UNB_VAR1 or TRSVP_LN_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?
- For $y := U^T x + y$, TRSVP_UT_UNB_VAR1 or TRSVP_UT_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?
- For $y := L^T x + y$, TRSVP_LT_UNB_VAR1 or TRSVP_LT_UNB_VAR2?
  Does the better algorithm use a `dot` or an `axpy`?

☛ BACK TO TEXT

4.3.3 Symmetric Matrix-Vector Multiplication

Homework 4.3.3.1 Write routines

- `[ y_out ] = Symv_u_unb_var1 ( A, x, y )`; and
- `[ y_out ] = Symv_u_unb_var2( A, x, y )`

that implement the algorithms in Figure 4.8.

☛ BACK TO TEXT

Homework 4.3.3.2 Modify the algorithms in Figure 4.9 to compute $y := Ax + y$, where A is symmetric and stored in the lower triangular part of matrix. You may want to do this in conjunction with the next exercise.

Answer: In the algorithm on the left, the update becomes $\psi_1 := a_{10}^T x_0 + \alpha_{11}\chi_1 + a_{21}^T x_2 + \psi_1$. In the algorithm on the right, the first update becomes $y_0 := \chi_1 (a_{10}^T)^T + y_0$.

☛ BACK TO TEXT

Homework 4.3.3.3 Write routines

- `[ y_out ] = Symv_l_unb_var1 ( A, x, y )`; and
- `[ y_out ] = Symv_l_unb_var2( A, x, y )`

that implement the algorithms from the previous homework.

☛ BACK TO TEXT

Homework 4.3.3.4 **Challenge question!** As hinted at before: Implementations achieve better performance (finish faster) if one accesses data consecutively in memory. Now, most scientific computing codes store matrices in "column-major order" which means that the first column of a matrix is stored consecutively in memory, then the second column, and so forth. Now, this means that an algorithm that accesses a matrix by columns tends to be faster than an algorithm that accesses a matrix by rows. That, in turn, means that when one is presented with more than one algorithm, one should pick the algorithm that accesses the matrix by columns. Our FLAME notation makes it easy to recognize algorithms that access the matrix by columns.

The problem with the algorithms in this unit is that all of them access both part of a row AND part of a column. So, your challenge is to devise an algorithm for computing $y := Ax + y$ where A is symmetric and only stored in one half of the matrix that only accesses parts of columns. We will call these "variant 3". Then, write routines

- `[ y_out ] = Symv_u_unb_var3 ( A, x, y )`; and
- `[ y_out ] = Symv_l_unb_var3( A, x, y )`

Hint: (Let's focus on the case where only the lower triangular part of A is stored.)

- If A is symmetric, then $A = L + \widehat{L}^T$ where L is the lower triangular part of A and $\widehat{L}$ is the strictly lower triangular part of A.
- Identify an algorithm for $y := Lx + y$ that accesses matrix A by columns.
- Identify an algorithm for $y := \widehat{L}^T x + y$ that accesses matrix A by columns.
- You now have two loops that together compute $y := Ax + y = (L + \widehat{L}^T)x + y = Lx + \widehat{L}^T x + y$.
- Can you "merge" the loops into one loop?

☛ BACK TO TEXT

4.4 Matrix-Matrix Multiplication (Product)

4.4.2 From Composing Linear Transformations to Matrix-Matrix Multiplication

Homework 4.4.2.1 Let $L_A : \mathbb{R}^k \to \mathbb{R}^m$ and $L_B : \mathbb{R}^n \to \mathbb{R}^k$ both be linear transformations and, for all $x \in \mathbb{R}^n$, define the function $L_C : \mathbb{R}^n \to \mathbb{R}^m$ by $L_C(x) = L_A(L_B(x))$. $L_C(x)$ is a linear transformations.

Always/Sometimes/Never

Answer: Always

Let $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.

- $L_C(\alpha x) = L_A(L_B(\alpha x)) = L_A(\alpha L_B(x)) = \alpha L_A(L_B(x)) = \alpha L_C(x)$.
- $$\begin{aligned} L_C(x+y) &= L_A(L_B(x+y)) = L_A(L_B(x) + L_B(y)) \\ &= L_A(L_B(x)) + L_A(L_B(y)) = L_C(x) + L_C(y). \end{aligned}$$

This homework confirms that the composition of two linear transformations is itself a linear transformation.

☛ BACK TO TEXT

Homework 4.4.2.2 Let $A \in \mathbb{R}^{m\times n}$. $A^T A$ is well-defined. (By well-defined we mean that $A^T A$ makes sense. In this particular case this means that the dimensions of A^T and A are such that $A^T A$ can be computed.)

Always/Sometimes/Never

Answer: Always A^T is $n \times m$ and A is $m \times n$, and hence the column size of A^T matches the row size of A.

☛ BACK TO TEXT

Homework 4.4.2.3 Let $A \in \mathbb{R}^{m\times n}$. AA^T is well-defined.

Always/Sometimes/Never

Answer: Always

Apply the result in the last exercise, with A replaced by A^T.

☛ BACK TO TEXT

4.4.3 Computing the Matrix-Matrix Product

Homework 4.4.3.1 Compute

$$Q = P \times P = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} = \begin{pmatrix} 0.30 & 0.25 & 0.25 \\ 0.40 & 0.45 & 0.40 \\ 0.30 & 0.30 & 0.35 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 4.4.3.2 Let $A = \begin{pmatrix} 2 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 3 & 1 \\ -1 & 1 & 1 \end{pmatrix}$ and $B = \begin{pmatrix} 2 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$. Compute

- $AB = \begin{pmatrix} 5 & 2 & 5 & 2 \\ -2 & 0 & -2 & 0 \\ 3 & 4 & 3 & 4 \\ -1 & 0 & -1 & 0 \end{pmatrix}$

- $BA = \begin{pmatrix} 4 & 8 & 5 \\ -2 & 2 & 1 \\ 3 & 3 & 2 \end{pmatrix}$

☛ BACK TO TEXT

Homework 4.4.3.3 Let $A \in \mathbb{R}^{m\times k}$ and $B \in \mathbb{R}^{k\times n}$ and $AB = BA$. A and B are square matrices.

Always/Sometimes/Never

Answer: Always

The result of AB is a $m \times n$ matrix. The result of BA is a $k \times k$ matrix. Hence $m = k$ and $n = k$. In other words, $m = n = k$.

☛ BACK TO TEXT

Homework 4.4.3.4 Let $A \in \mathbb{R}^{m\times k}$ and $B \in \mathbb{R}^{k\times n}$.

$$AB = BA.$$

Always/Sometimes/Never

Answer: Sometimes

If $m \neq n$ then BA is not even defined because the sizes of the matrices don't match up. But if A is square and $A = B$, then clearly $AB = AA = BA$.

So, there are examples where the statement is true and examples where the statement is false.

☛ BACK TO TEXT

Homework 4.4.3.5 Let $A, B \in \mathbb{R}^{n\times n}$. $AB = BA$.

Always/Sometimes/Never

Answer: Sometimes

Almost any random matrices A and B will have the property that $AB \neq BA$. But if you pick, for example, $n = 1$ or $A = I$ or $A = 0$ or $A = B$, then $AB = BA$. There are many other examples.

The bottom line: Matrix multiplication, unlike scalar multiplication, does not necessarily commute.

☛ BACK TO TEXT

Homework 4.4.3.6 A^2 is defined as AA. Similarly $A^k = \underbrace{AA\cdots A}_{k \text{ occurrences of } A}$. Consistent with this, $A^0 = I$ so that $A^k = A^{k-1}A$ for $k > 0$.

A^k is well-defined only if A is a square matrix.

True/False

Answer: True

Just check the sizes of the matrices.

☛ BACK TO TEXT

Homework 4.4.3.7 Let A, B, C be matrix "of appropriate size" so that $(AB)C$ is well defined. $A(BC)$ is well defined.

Always/Sometimes/Never

Answer: Always

For $(AB)C$ to be well defined, $A \in \mathbb{R}^{m_A\times n_A}$, $B \in \mathbb{R}^{m_B\times n_B}$, $C \in \mathbb{R}^{m_C\times n_C}$, where $n_A = m_B$ and $n_B = m_C$. But then BC is well defined because $n_B = m_C$ and results in a $m_B \times n_C$ matrix. But then $A(BC)$ is well defined because $n_A = m_B$.

☛ BACK TO TEXT

4.4.4 Special Shapes

Homework 4.4.4.1 Let $A = \left(\begin{array}{c} 4 \end{array} \right)$ and $B = \left(\begin{array}{c} 3 \end{array} \right)$. Then $AB =$ ___.

Answer: $\left(\begin{array}{c} 12 \end{array} \right)$ or 12.

☛ BACK TO TEXT

Homework 4.4.4.2 Let $A = \left(\begin{array}{r} 1 \\ -3 \\ 2 \end{array} \right)$ and $B = \left(\begin{array}{c} 4 \end{array} \right)$. Then $AB =$.

Answer:

$$AB = \left(\begin{array}{r} 1 \\ -3 \\ 2 \end{array} \right) \left(\begin{array}{c} 4 \end{array} \right) = 4 \left(\begin{array}{r} 1 \\ -3 \\ 2 \end{array} \right) = \left(\begin{array}{c} 4 \times 1 \\ 4 \times (-3) \\ 4 \times 2 \end{array} \right) = \left(\begin{array}{r} 4 \\ -12 \\ 8 \end{array} \right).$$

☛ BACK TO TEXT

Homework 4.4.4.3 Let $A = \left(\begin{array}{c} 4 \end{array} \right)$ and $B = \left(\begin{array}{ccc} 1 & -3 & 2 \end{array} \right)$. Then $AB =$.

Answer:

$$AB = \left(\begin{array}{c} 4 \end{array} \right) \left(\begin{array}{ccc} 1 & -3 & 2 \end{array} \right) = \left(\begin{array}{ccc} 4 \cdot 1 & 4 \cdot (-3) & 4 \cdot 2 \end{array} \right) = \left(\begin{array}{ccc} 4 & -12 & 8 \end{array} \right).$$

☛ BACK TO TEXT

Homework 4.4.4.4 Let $A = \left(\begin{array}{ccc} 1 & -3 & 2 \end{array} \right)$ and $B = \left(\begin{array}{r} 2 \\ -1 \\ 0 \end{array} \right)$. Then $AB =$

Answer:

$$AB = \left(\begin{array}{ccc} 1 & -3 & 2 \end{array} \right) \left(\begin{array}{r} 2 \\ -1 \\ 0 \end{array} \right) = 1 \cdot 2 + (-3) \cdot (-1) + 2 \cdot 0 = 2 + 3 + 0 = 5.$$

or

$$AB = \left(\begin{array}{ccc} 1 & -3 & 2 \end{array} \right) \left(\begin{array}{r} 2 \\ -1 \\ 0 \end{array} \right) = (1 \cdot 2 + (-3) \cdot (-1) + 2 \cdot 0) = (2 + 3 + 0 = 5).$$

☛ BACK TO TEXT

Homework 4.4.4.5 Let $A = \left(\begin{array}{r} 1 \\ -3 \\ 2 \end{array} \right)$ and $B = \left(\begin{array}{cc} -1 & -2 \end{array} \right)$. Then $AB =$

Answer:

$$AB = \begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix} \begin{pmatrix} -1 & -2 \end{pmatrix} = \begin{pmatrix} 1\cdot(-1) & 1\cdot(-2) \\ (-3)\cdot(-1) & (-3)\cdot(-2) \\ 2\cdot(-1) & 2\cdot(-2) \end{pmatrix} = \begin{pmatrix} -1 & -2 \\ 3 & 6 \\ -2 & -4 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 4.4.4.6 Let $a = \begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix}$ and $b^T = \begin{pmatrix} -1 & -2 \end{pmatrix}$ and $C = ab^T$. Partition C by columns and by rows:

$$C = \left(c_0 \,\middle|\, c_1 \right) \quad \text{and} \quad C = \begin{pmatrix} \tilde{c}_0^T \\ \tilde{c}_1^T \\ \tilde{c}_2^T \end{pmatrix}$$

Then

- $c_0 = (-1)\begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix} = \begin{pmatrix} (-1)\times & (1) \\ (-1)\times & (-3) \\ (-1)\times & (2) \end{pmatrix}$ True/False
- $c_1 = (-2)\begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix} = \begin{pmatrix} (-2)\times & (1) \\ (-2)\times & (-3) \\ (-2)\times & (2) \end{pmatrix}$ True/False
- $C = \left(\begin{array}{cc|cc} (-1)\times & (1) & (-2)\times & (1) \\ (-1)\times & (-3) & (-2)\times & (-3) \\ (-1)\times & (2) & (-2)\times & (2) \end{array}\right)$ True/False
- $\tilde{c}_0^T = (1)\begin{pmatrix} -1 & -2 \end{pmatrix} = \begin{pmatrix} (1)\times(-1) & (1)\times(-2) \end{pmatrix}$ True/False
- $\tilde{c}_1^T = (-3)\begin{pmatrix} -1 & -2 \end{pmatrix} = \begin{pmatrix} (-3)\times(-1) & (-3)\times(-2) \end{pmatrix}$ True/False
- $\tilde{c}_2^T = (2)\begin{pmatrix} -1 & -2 \end{pmatrix} = \begin{pmatrix} (2)\times(-1) & (2)\times(-2) \end{pmatrix}$ True/False
- $C = \left(\begin{array}{cccc} (-1)\times & (1) & (-2)\times & (1) \\ \hline (-1)\times & (-3) & (-2)\times & (-3) \\ \hline (-1)\times & (2) & (-2)\times & (2) \end{array}\right)$ True/False

Answer: The important thing here is to recognize that if you compute the first two results, then the third result comes for free. If you compute results 4-6, then the last result comes for free.

Also, notice that the columns C are just multiples of a while the rows of C are just multiples of b^T.

☛ BACK TO TEXT

Homework 4.4.4.7 Fill in the boxes:

$$\begin{pmatrix} \square \\ \square \\ \square \\ \square \end{pmatrix} \begin{pmatrix} 2 & -1 & 3 \end{pmatrix} = \begin{pmatrix} 4 & \square & \square \\ -2 & \square & \square \\ 2 & \square & \square \\ 6 & \square & \square \end{pmatrix}$$

Answer:

$$\begin{pmatrix} 2 \\ -1 \\ 1 \\ 3 \end{pmatrix} \begin{pmatrix} 2 & -1 & 3 \end{pmatrix} = \begin{pmatrix} 4 & -2 & 6 \\ -2 & 1 & -3 \\ 2 & -1 & 3 \\ 6 & -3 & 9 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 4.4.4.8 Fill in the boxes:

$$\begin{pmatrix} 2 \\ -1 \\ 1 \\ 3 \end{pmatrix} \begin{pmatrix} \square & \square & \square \end{pmatrix} = \begin{pmatrix} 4 & -2 & 6 \\ \square & \square & \square \\ \square & \square & \square \\ \square & \square & \square \end{pmatrix}$$

Answer:

$$\begin{pmatrix} 2 \\ -1 \\ 1 \\ 3 \end{pmatrix} \begin{pmatrix} 2 & -1 & 3 \end{pmatrix} = \begin{pmatrix} 4 & -2 & 6 \\ -2 & 1 & -3 \\ 2 & -1 & 3 \\ 6 & -3 & 9 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 4.4.4.9 Let $A = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$ and $B = \begin{pmatrix} 1 & -2 & 2 \\ 4 & 2 & 0 \\ 1 & 2 & 3 \end{pmatrix}$. Then $AB =$

Answer: $\begin{pmatrix} 4 & 2 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 4.4.4.10 Let $e_i \in \mathbb{R}^m$ equal the ith unit basis vector and $A \in \mathbb{R}^{m \times n}$. Then $e_i^T A = \check{a}_i^T$, the ith row of A.

Always/Sometimes/Never

Answer: Always

$$\left(\begin{array}{ccccccc} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{array}\right)\left(\begin{array}{cccc} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{i-1,0} & \alpha_{i-1,1} & \cdots & \alpha_{i-1,n-1} \\ \alpha_{i,0} & \alpha_{i,1} & \cdots & \alpha_{i,n-1} \\ \alpha_{i+1,0} & \alpha_{i+1,1} & \cdots & \alpha_{i+1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{array}\right)$$
$$= \left(\begin{array}{cccc} \alpha_{i,0} & \alpha_{i,1} & \cdots & \alpha_{i,n-1} \end{array}\right).$$

☛ BACK TO TEXT

Homework 4.4.4.11 Get as much practice as you want with the MATLAB script in

`LAFF-2.0xM/Programming/Week04/PracticeGemm.m`

☛ BACK TO TEXT

4.6 Wrap Up

4.6.1 Homework

Homework 4.6.1.1 Let $A \in \mathbb{R}^{m\times n}$ and $x \in \mathbb{R}^n$. Then $(Ax)^T = x^T A^T$.

Always/Sometimes/Never

Answer: Always

$$
\begin{array}{l}
(Ax)^T \\
\quad = \quad < \text{Partition } A \text{ into rows } > \\
(\left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a})_1^T \\ \hline \vdots \\ \hline \tilde{a})_{m-1}^T \end{array}\right) x)^T \\
\quad = \quad < \text{Matrix-vector multiplication } > \\
\left(\begin{array}{c} \tilde{a}_0^T x \\ \hline \tilde{a}_1^T x \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T x \end{array}\right)^T \\
\quad = \quad < \text{transpose the column vector } > \\
\left(\begin{array}{c|c|c|c} \tilde{a}_0^T x & \tilde{a}_1^T x & \cdots & \tilde{a}_{m-1}^T x \end{array}\right) \\
\quad = \quad < \text{dot product commutes } > \\
\left(\begin{array}{c|c|c|c} x^T \tilde{a}_0 & x^T \tilde{a}_1 & \cdots & x^T \tilde{a}_{m-1} \end{array}\right) \\
\quad = \quad < \text{special case of matrix-matrix multiplication } > \\
x^T \left(\begin{array}{c|c|c|c} \tilde{a}_0 & \tilde{a}_1 & \cdots & \tilde{a}_{m-1} \end{array}\right) \\
\quad = \quad < \text{transpose the matrix } > \\
x^T \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a})_1^T \\ \hline \vdots \\ \hline \tilde{a})_{m-1}^T \end{array}\right)^T \\
\quad = \quad < \text{unpartition the matrix } > \\
x^T A^T
\end{array}
$$

☛ BACK TO TEXT

Homework 4.6.1.2 Our `laff` library has a routine

```
laff_gemv( trans, alpha, A, x, beta, y )
```

that has the following property

- `laff_gemv( 'No transpose', alpha, A, x, beta, y )` computes $y := \alpha Ax + \beta y$.
- `laff_gemv( 'Transpose', alpha, A, x, beta, y )` computes $y := \alpha A^T x + \beta y$.

The routine works regardless of whether x and/or y are column and/or row vectors.

Our library does NOT include a routine to compute $y^T := x^T A$. What call could you use to compute $y^T := x^T A$ if y^T is stored in `yt` and x^T in `xt`?

- `laff_gemv( 'No transpose', 1.0, A, xt, 0.0, yt ).`
- `laff_gemv( 'No transpose', 1.0, A, xt, 1.0, yt ).`

- `laff_gemv( 'Transpose', 1.0, A, xt, 1.0, yt ).`
- `laff_gemv( 'Transpose', 1.0, A, xt, 0.0, yt ).`

Answer: `laff_gemv( 'Transpose', 1.0, A, xt, 0.0, yt )` computes $y := A^T x$, where y is stored in `yt` and x is stored in `xt`.

To understand this, transpose both sides: $y^T = (A^T x)^T = x^T A^{T^T} = x^T A$.

For this reason, our `laff` library does not include a routine to compute $y^T := \alpha x^T A + \beta y^T$.

You will need this next week!!!

☛ BACK TO TEXT

Homework 4.6.1.3 Let $A = \begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix}$. Compute

- $A^2 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$
- $A^3 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$
- For $k > 1$, $A^k = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 4.6.1.4 Let $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

- $A^2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
- $A^3 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
- For $n \geq 0$, $A^{2n} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
- For $n \geq 0$, $A^{2n+1} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 4.6.1.5 Let $A = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$.

- $A^2 = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$
- $A^3 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$
- For $n \geq 0$, $A^{4n} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
- For $n \geq 0$, $A^{4n+1} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 4.6.1.6 Let A be a square matrix. If $AA = 0$ (the zero matrix) then A is a zero matrix. (AA is often written as A^2.)

True/False

Answer: False!

$$\begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

This may be counter intuitive since if α is a scalar, then $\alpha^2 = 0$ only if $\alpha = 0$.

So, one of the points of this exercise is to make you skeptical about "facts" about scalar multiplications that you may try to transfer to matrix-matrix multiplication.

☛ BACK TO TEXT

Homework 4.6.1.7 There exists a real valued matrix A such that $A^2 = -I$. (Recall: I is the identity)

True/False

Answer: True! Example: $A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$.

This may be counter intuitive since if α is a real scalar, then $\alpha^2 \neq -1$.

☛ BACK TO TEXT

Homework 4.6.1.8 There exists a matrix A that is not diagonal such that $A^2 = I$.

True/False

Answer: True! An examples of a matrices A that is not diagonal yet $A^2 = I$: $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

This may be counter intuitive since if α is a real scalar, then $\alpha^2 = 1$ only if $\alpha = 1$ or $\alpha = -1$. Also, if a matrix is 1×1, then it is automatically diagonal, so you cannot look at 1×1 matrices for inspiration for this problem.

☛ BACK TO TEXT

Week 5: Matrix-Matrix Multiplication (Answers)

5.1 Opening Remarks

5.1.1 Composing Rotations

Homework 5.1.1.1 Which of the following statements are *true*:

- $\begin{pmatrix} \cos(\rho+\sigma+\tau) \\ \sin(\rho+\sigma+\tau) \end{pmatrix} = \left(\begin{array}{c|c} \cos(\tau) & -\sin(\tau) \\ \hline \sin(\tau) & \cos(\tau) \end{array}\right) \begin{pmatrix} \cos(\rho+\sigma) \\ \sin(\rho+\sigma) \end{pmatrix}$

True/False

Answer: True Extending the observations in the video, we know that

$$R_{\rho+\sigma+\tau}(e_0) = R_\tau(R_{\rho+\sigma}(e_0)).$$

But

$$R_{\rho+\sigma+\tau}(e_0) = \begin{pmatrix} \cos(\rho+\sigma+\tau) \\ \sin(\rho+\sigma+\tau) \end{pmatrix}$$

and

$$R_\tau(R_{\rho+\sigma}(e_0)) = \left(\begin{array}{c|c} \cos(\tau) & -\sin(\tau) \\ \hline \sin(\tau) & \cos(\tau) \end{array}\right) \begin{pmatrix} \cos(\rho+\sigma) \\ \sin(\rho+\sigma) \end{pmatrix}.$$

establishing the result.

- $\begin{pmatrix} \cos(\rho+\sigma+\tau) \\ \sin(\rho+\sigma+\tau) \end{pmatrix} = \left(\begin{array}{c|c} \cos(\tau) & -\sin(\tau) \\ \hline \sin(\tau) & \cos(\tau) \end{array}\right) \begin{pmatrix} \cos\rho\cos\sigma - \sin\rho\sin\sigma \\ \sin\rho\cos\sigma + \cos\rho\sin\sigma. \end{pmatrix}$

True/False

Answer: True From the video we know that

$$R_{\rho+\sigma}(e_0) = \left(\begin{array}{c|c} \cos(\sigma) & -\sin(\sigma) \\ \hline \sin(\sigma) & \cos(\sigma) \end{array}\right) \left(\begin{array}{c|c} \cos(\rho) & -\sin(\rho) \\ \hline \sin(\rho) & \cos(\rho) \end{array}\right) \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos\rho\cos\sigma - \sin\rho\sin\sigma \\ \sin\rho\cos\sigma + \cos\rho\sin\sigma. \end{pmatrix}$$

establishing the result.

- $\begin{array}{rcl} \cos(\rho+\sigma+\tau) & = & \cos(\tau)(\cos\rho\cos\sigma - \sin\rho\sin\sigma) - \sin(\tau)(\sin\rho\cos\sigma + \cos\rho\sin\sigma) \\ \sin(\rho+\sigma+\tau) & = & \sin(\tau)(\cos\rho\cos\sigma - \sin\rho\sin\sigma) + \cos(\tau)(\sin\rho\cos\sigma + \cos\rho\sin\sigma) \end{array}$

True/False

Answer: True This is a matter of multiplying the last result.

☛ BACK TO TEXT

5.2 Observations

5.2.2 Properties

Homework 5.2.2.1 Let $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $B = \begin{pmatrix} 0 & 2 & -1 \\ 1 & 1 & 0 \end{pmatrix}$, and $C = \begin{pmatrix} 0 & 1 \\ 1 & 2 \\ 1 & -1 \end{pmatrix}$. Compute

- $AB = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 2 & -1 \end{pmatrix}$

- $(AB)C = \begin{pmatrix} 1 & 3 \\ 1 & 5 \end{pmatrix}$

- $BC = \begin{pmatrix} 1 & 5 \\ 1 & 3 \end{pmatrix}$

- $A(BC) = \begin{pmatrix} 1 & 3 \\ 1 & 5 \end{pmatrix}$

☛ BACK TO TEXT

Homework 5.2.2.2 Let $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times k}$, and $C \in \mathbb{R}^{k \times l}$. $(AB)C = A(BC)$.

Always/Sometimes/Never

Answer:

Proof 1:

Two matrices are equal if corresponding columns are equal. We will show that $(AB)C = A(BC)$ by showing that, for arbitrary j, the jth column of $(AB)C$ equals the jth column of $A(BC)$. In other words, that $((AB)C)e_j = (A(BC))e_j$.

$$
\begin{array}{l}
((AB)C)e_j \\
\quad = \quad < \text{Definition of matrix-matrix multiplication} > \\
(AB)Ce_j \\
\quad = \quad < \text{Definition of matrix-matrix multiplication} > \\
A(B(Ce_j)) \\
\quad = \quad < \text{Definition of matrix-matrix multiplication} > \\
A((BC)e_j) \\
\quad = \quad < \text{Definition of matrix-matrix multiplication} > \\
(A(BC))e_j.
\end{array}
$$

Proof 2 (using partitioned matrix-matrix multiplication):

$(AB)C$

$= \quad <$ Partition by columns $>$

$(AB)\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{l-1}\end{array}\right)$

$= \quad <$ Partitioned matrix-matrix multiplication $>$

$\left(\begin{array}{c|c|c|c} (AB)c_0 & (AB)c_1 & \cdots & (AB)c_{l-1}\end{array}\right)$

$= \quad <$ Definition of matrix-matrix multiplication $>$

$\left(\begin{array}{c|c|c|c} A(Bc_0) & A(Bc_1) & \cdots & A(Bc_{l-1})\end{array}\right)$

$= \quad <$ Partitioned matrix-matrix multiplication $>$

$A\left(\begin{array}{c|c|c|c} Bc_0 & Bc_1 & \cdots & Bc_{l-1}\end{array}\right)$

$= \quad <$ Partitioned matrix-matrix multiplication $>$

$A(B\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{l-1}\end{array}\right))$

$= \quad <$ Partition by columns $>$

$A(BC)$

☛ BACK TO TEXT

Homework 5.2.2.3 Let $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $B = \begin{pmatrix} 2 & -1 \\ 1 & 0 \end{pmatrix}$, and $C = \begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix}$. Compute

- $A(B+C) = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.
- $AB+AC = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.
- $(A+B)C = \begin{pmatrix} -2 & 2 \\ -2 & 2 \end{pmatrix}$.
- $AC+BC = \begin{pmatrix} -2 & 2 \\ -2 & 2 \end{pmatrix}$.

☛ BACK TO TEXT

Homework 5.2.2.4 Let $A \in \mathbb{R}^{m\times k}$, $B \in \mathbb{R}^{k\times n}$, and $C \in \mathbb{R}^{k\times n}$. $A(B+C) = AB+AC$.

Always/Sometimes/Never

Answer: Always

$$
\begin{array}{l}
A(B+C) \\
\quad = \quad < \text{Partition } B \text{ and } C \text{ by columns } > \\
A\left(\left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) + \left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array} \right)\right) \\
\quad = \quad < \text{Definition of matrix addition } > \\
A\left(\begin{array}{c|c|c|c} b_0+c_0 & b_1+c_1 & \cdots & b_{n-1}+c_{n-1} \end{array} \right) \\
\quad = \quad < \text{Partitioned matrix-matrix multiplication } > \\
\left(\begin{array}{c|c|c|c} A(b_0+c_0) & A(b_1+c_1) & \cdots & A(b_{n-1}+c_{n-1}) \end{array} \right) \\
\quad = \quad < \text{Matrix-vector multiplication distributes } > \\
\left(\begin{array}{c|c|c|c} Ab_0+Ac_0 & Ab_1+Ac_1 & \cdots & Ab_{n-1}+Ac_{n-1} \end{array} \right) \\
\quad = \quad < \text{Defintion of matrix addition } > \\
\left(\begin{array}{c|c|c|c} Ab_0 & Ab_1 & \cdots & Ab_{n-1} \end{array} \right) + \left(\begin{array}{c|c|c|c} Ac_0 & Ac_1 & \cdots & Ac_{n-1} \end{array} \right) \\
\quad = \quad < \text{Partitioned matrix-matrix multiplication } > \\
A\left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) + \left(\begin{array}{c|c|c|c} Ac_0 & Ac_1 & \cdots & Ac_{n-1} \end{array} \right) \\
\quad = \quad < \text{Partition by columns } > \\
AB+AC.
\end{array}
$$

☛ BACK TO TEXT

Homework 5.2.2.5 If $A \in \mathbb{R}^{m\times k}$, $B \in \mathbb{R}^{m\times k}$, and $C \in \mathbb{R}^{k\times n}$, then $(A+B)C = AC+BC$.

True/False

Answer: True

$$
\begin{array}{l}
(A+B)C \\
\quad = \quad < \text{Partition } C \text{ by columns.} > \\
(A+B)\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array} \right) \\
\quad = \quad < DE \text{ means } D \text{ multiplies each of the columns of } E > \\
\left(\begin{array}{c|c|c|c} (A+B)c_0 & (A+B)c_1 & \cdots & (A+B)c_{n-1} \end{array} \right) \\
\quad = \quad < \text{Definition of matrix addition} > \\
\left(\begin{array}{c|c|c|c} Ac_0+Bc_0 & Ac_1+Bc_1 & \cdots & Ac_{n-1}+Bc_{n-1} \end{array} \right) \\
\quad = \quad < D+E \text{ means adding corresponding columns} > \\
\left(\begin{array}{c|c|c|c} Ac_0 & Ac_1 & \cdots & Ac_{n-1} \end{array} \right) + \left(\begin{array}{c|c|c|c} Bc_0 & Bc_1 & \cdots & Bc_{n-1} \end{array} \right) \\
\quad = \quad < DE \text{ means } D \text{ multiplies each of the columns of } E > \\
A\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array} \right) + B\left(\begin{array}{c|c|c|c} c_0 & c_1 & \cdots & c_{n-1} \end{array} \right) \\
\quad = \quad < \text{Partition } C \text{ by columns} > \\
AC+BC.
\end{array}
$$

☛ BACK TO TEXT

5.2.3 Transposing a Product of Matrices

Homework 5.2.3.1 Let $A = \begin{pmatrix} 2 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 3 & 1 \\ -1 & 1 & 1 \end{pmatrix}$ and $B = \begin{pmatrix} 2 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$. Compute

- $A^T A = \begin{pmatrix} 7 & 1 & 2 \\ 1 & 11 & 4 \\ 2 & 4 & 3 \end{pmatrix}$

- $AA^T = \begin{pmatrix} 5 & -2 & 3 & -1 \\ -2 & 2 & 2 & 2 \\ 3 & 2 & 11 & 3 \\ -1 & 2 & 3 & 3 \end{pmatrix}$

- $(AB)^T = \begin{pmatrix} 5 & -2 & 3 & -1 \\ 2 & 0 & 4 & 0 \\ 5 & -2 & 3 & -1 \\ 2 & 0 & 4 & 0 \end{pmatrix}$

- $A^T B^T = \begin{pmatrix} 4 & -2 & 3 \\ 8 & 2 & 3 \\ 5 & 1 & 2 \end{pmatrix}$

- $B^T A^T = \begin{pmatrix} 5 & -2 & 3 & -1 \\ 2 & 0 & 4 & 0 \\ 5 & -2 & 3 & -1 \\ 2 & 0 & 4 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 5.2.3.2 Let $A \in \mathbb{R}^{m \times k}$ and $B \in \mathbb{R}^{k \times n}$. $(AB)^T = B^T A^T$.

Always/Sometimes/Never

Answer:

Proof 1:

In an example in the previous unit, we partitioned C into elements (scalars) and A and B by rows and columns, respectively, before performing the partitioned matrix-matrix multiplication $C = AB$. This insight forms the basis for the following proof:

$$
\begin{array}{l}
(AB)^T \quad = \quad < \text{Partition } A \text{ by rows and } B \text{ by columns} > \\
\left(\left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right) \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) \right)^T \\
\quad = \quad < \text{Partitioned matrix-matrix multiplication} > \\
\left(\begin{array}{c|c|c|c} \tilde{a}_0^T b_0 & \tilde{a}_0^T b_1 & \cdots & \tilde{a}_0^T b_{n-1} \\ \hline \tilde{a}_1^T b_0 & \tilde{a}_1^T b_1 & \cdots & \tilde{a}_1^T b_{n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \tilde{a}_{m-1}^T b_0 & \tilde{a}_{m-1}^T b_1 & \cdots & \tilde{a}_{m-1}^T b_{n-1} \end{array} \right)^T \\
\quad = \quad < \text{Transpose the matrix} > \\
\left(\begin{array}{c|c|c|c} \tilde{a}_0^T b_0 & \tilde{a}_1^T b_0 & \cdots & \tilde{a}_{m-1}^T b_0 \\ \hline \tilde{a}_0^T b_1 & \tilde{a}_1^T b_1 & \cdots & \tilde{a}_{m-1}^T b_1 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \tilde{a}_0^T b_{n-1} & \tilde{a}_1^T b_{n-1} & \cdots & \tilde{a}_{m-1}^T b_{n-1} \end{array} \right) \\
\quad = \quad < \text{dot product commutes} > \\
\left(\begin{array}{c|c|c|c} b_0^T \tilde{a}_0 & b_0^T \tilde{a}_1 & \cdots & b_0^T \tilde{a}_{m-1} \\ \hline b_1^T \tilde{a}_0 & b_1^T \tilde{a}_1 & \cdots & b_1^T \tilde{a}_{m-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline b_{n-1}^T \tilde{a}_0 & b_{n-1}^T \tilde{a}_1 & \cdots & b_{m-1}^T \tilde{a}_{m-1} \end{array} \right) \\
\quad = \quad < \text{Partitioned matrix-matrix multiplication} > \\
\left(\begin{array}{c} b_0^T \\ \hline b_1^T \\ \hline \vdots \\ \hline b_{n-1}^T \end{array} \right) \left(\begin{array}{c|c|c|c} \tilde{a}_0 & \tilde{a}_1 & \cdots & \tilde{a}_{m-1} \end{array} \right) \\
\quad = \quad < \text{Partitioned matrix transposition} > \\
\left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right)^T \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right)^T = B^T A^T.
\end{array}
$$

Proof 2:

Let $C = AB$ and $D = B^T A^T$. We need to show that $\gamma_{i,j} = \delta_{j,i}$.

But

$\gamma_{i,j}$

$= \quad <$ Earlier observation $>$

$e_i^T C e_j$

$= \quad < C = AB >$

$e_i^T (AB) e_j$

$= \quad <$ Associativity of multiplication; e_i^T and e_j are matrices $>$

$(e_i^T A)(B e_j)$

$= \quad <$ Property of multiplication; $\widetilde{a}_i^T$ is ith row of A, b_j is jth column of B $>$

$\widetilde{a}_i^T b_j$

$= \quad <$ Dot product commutes $>$

$b_j^T \widetilde{a}_i$

$= \quad <$ Property of multiplication $>$

$(e_j^T B^T)(A^T e_i)$

$= \quad <$ Associativity of multiplication; e_i^T and e_j are matrices $>$

$e_j^T (B^T A^T) e_i$

$= \quad < C = AB >$

$e_j^T D e_i$

$= \quad <$ earlier observation $>$

$\delta_{j,i}$

Proof 3:

(I vaguely recall that somewhere we proved that $(Ax)^T = x^T A^T$... If not, one should prove that first...)

$$
\begin{array}{l}
(AB)^T \\
\quad = \quad < \text{ Partition } B \text{ by columns } > \\
(A \begin{pmatrix} b_0 & b_1 & \cdots & b_{n-1} \end{pmatrix})^T \\
\quad = \quad < \text{ Partitioned matrix-matrix multiplication } > \\
\begin{pmatrix} Ab_0 & Ab_1 & \cdots & Ab_{n-1} \end{pmatrix}^T \\
\quad = \quad < \text{ Transposing a partitioned matrix } > \\
\begin{pmatrix} (Ab_0)^T \\ (Ab_1)^T \\ \vdots \\ (Ab_{n-1})^T \end{pmatrix} \\
\quad = \quad < (Ax)^T = x^T A^T \; > \\
\begin{pmatrix} b_0^T A^T \\ b_1^T A^T \\ \vdots \\ b_{n-1}^T A^T \end{pmatrix} \quad = \quad < \text{ Partitioned matrix-matrix multiplication } > \\
\begin{pmatrix} b_0^T \\ b_1^T \\ \vdots \\ b_{n-1}^T \end{pmatrix} A^T \\
\quad = \quad < \text{ Partitioned matrix transposition } > \\
\begin{pmatrix} b_0 & b_1 & \cdots & b_{n-1} \end{pmatrix}^T A^T \\
\quad = \quad < \text{ Partition } B \text{ by columns } > \\
B^T A^T
\end{array}
$$

Proof 4: (For those who don't like the $\cdots$ in arguments...)

Proof by induction on n, the number of columns of B.

(I vaguely recall that somewhere we proved that $(Ax)^T = x^T A^T$... If not, one should prove that first...)

Base case: $n = 1$. Then $B = (b_0)$. But then $(AB)^T = (Ab_0)^T = b_0^T A^T = B^T A^T$.

Inductive Step: The inductive hypothesis is: Assume that $(AB)^T = B^T A^T$ for all matrices B with $n = N$ columns. We now need to show that, assuming this, $(AB)^T = B^T A^T$ for all matrices B with $n = N + 1$ columns.

Assume that B has $N+1$ columns. Then

$$
\begin{array}{l}
(AB)^T \\
\quad = \quad < \text{Partition } B > \\
(A \left(\begin{array}{cc} B_0 & b_1 \end{array} \right))^T \\
\quad = \quad < \text{Partitioned matrix-matrix multiplication} > \\
(\left(\begin{array}{cc} AB_0 & Ab_1 \end{array} \right))^T \\
\quad = \quad < \text{Partitioned matrix transposition} > \\
(\left(\begin{array}{c} (AB_0)^T \\ (Ab_1)^T \end{array} \right)) \\
\quad = \quad < \text{I.H. and } (Ax)^T = x^T A^T > \\
\left(\begin{array}{c} B_0^T A^T \\ b_1^T A^T \end{array} \right) \\
\quad = \quad < \text{Partitioned matrix-matrix multiplication} > \\
\left(\begin{array}{c} B_0^T \\ b_1^T \end{array} \right) A^T \\
\quad = \quad < \text{Transposing a partitioned matrix} > \\
\left(\begin{array}{cc} B_0 & b_1 \end{array} \right)^T A^T \\
\quad = \quad < \text{Partitioning of } B > \\
B^T A^T
\end{array}
$$

☛ BACK TO TEXT

Homework 5.2.3.3 Let A, B, and C be conformal matrices so that ABC is well-defined. Then $(ABC)^T = C^T B^T A^T$.
Always/Sometimes/Never

Answer: Always

$$(ABC)^T = (A(BC))^T = (BC)^T A^T = (C^T B^T)A^T = C^T B^T A^T$$

☛ BACK TO TEXT

5.2.4 Matrix-Matrix Multiplication with Special Matrices

Homework 5.2.4.1 Compute

- $\left(\begin{array}{rrr} 1 & -2 & -1 \\ 2 & 0 & 2 \end{array} \right) \left(\begin{array}{c} 1 \\ 0 \\ 0 \end{array} \right) = \left(\begin{array}{c} 1 \\ 2 \end{array} \right)$
- $\left(\begin{array}{rrr} 1 & -2 & -1 \\ 2 & 0 & 2 \end{array} \right) \left(\begin{array}{c} 0 \\ 1 \\ 0 \end{array} \right) = \left(\begin{array}{r} -2 \\ 0 \end{array} \right)$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$
- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix}$
- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix}$

Answer: There are at least two things to notice:

1. The first three results provide the columns for the fourth result. The fourth result provides the first two rows of the fifth result.
2. Multiplying the matrix from the right with the identity matrix does not change the matrix.

☛ BACK TO TEXT

Homework 5.2.4.2 Compute

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}$
- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 0 \\ 3 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \\ 3 \end{pmatrix}$
- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ -1 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ -1 \end{pmatrix}$
- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix} = \begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix}$

Answer: There are at least three things to notice:

1. The first three results provide the columns for the fourth result.
2. Multiplying the matrix from the left with the identity matrix does not change the matrix.
3. This homework and the last homework yield the same result.

☛ BACK TO TEXT

Homework 5.2.4.3 Let $A \in \mathbb{R}^{m \times n}$ and let I denote the identity matrix of appropriate size. $AI = IA = A$.

Always/Sometimes/Never

Answer: Always

Partition A and I by columns:

$$A = \left(a_0 \,\middle|\, a_1 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right) \text{ and } I = \left(e_0 \,\middle|\, e_1 \,\middle|\, \cdots \,\middle|\, e_{n-1} \right)$$

and recall that e_j equals the jth unit basis vector.

$AI = A$:

$$\begin{array}{l} AI \\ \quad = \quad < \text{Partition } I \text{ by columns } > \\ A\left(e_0 \,\middle|\, e_1 \,\middle|\, \cdots \,\middle|\, e_{n-1} \right) \\ \quad = \quad < \text{Partitioned matrix-matrix multiplication } > \\ \left(Ae_0 \,\middle|\, Ae_1 \,\middle|\, \cdots \,\middle|\, Ae_{n-1} \right) \\ \quad = \quad < a_j = Ae_j > \\ \left(a_0 \,\middle|\, a_1 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right) \\ \quad = \quad < \text{Partition } A \text{ by columns } > \\ A \end{array}$$

$IA = A$:

$$\begin{array}{l} IA \\ \quad = \quad < \text{Partition } A \text{ by columns } > \\ I\left(a_0 \,\middle|\, a_1 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right) \\ \quad = \quad < \text{Partitioned matrix-matrix multiplication } > \\ \left(Ia_0 \,\middle|\, Ia_1 \,\middle|\, \cdots \,\middle|\, Ia_{n-1} \right) \\ \quad = \quad < Ix = x > \\ \left(a_0 \,\middle|\, a_1 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right) \\ \quad = \quad < \text{Partition } A \text{ by columns } > \\ A \end{array}$$

☛ BACK TO TEXT

Homework 5.2.4.4 Compute

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix}$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ -3 \end{pmatrix} = \begin{pmatrix} 3 \\ -6 \end{pmatrix}$

- $\begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} = \begin{pmatrix} 2 & 2 & 3 \\ 4 & 0 & -6 \end{pmatrix}$

Answer: Notice the relation between the above problems.

☛ BACK TO TEXT

Homework 5.2.4.5 Compute

- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \\ 3 \end{pmatrix}$
- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} -2 \\ 0 \\ 3 \end{pmatrix} = \begin{pmatrix} -4 \\ 0 \\ -9 \end{pmatrix}$
- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ -1 \end{pmatrix} = \begin{pmatrix} -2 \\ -2 \\ 3 \end{pmatrix}$
- $\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{pmatrix} \begin{pmatrix} 1 & -2 & -1 \\ 2 & 0 & 2 \\ -1 & 3 & -1 \end{pmatrix} = \begin{pmatrix} 2 & -4 & -2 \\ -2 & 0 & -2 \\ 3 & -9 & 3 \end{pmatrix}$

Answer: Notice the relation between the above problems.

☛ BACK TO TEXT

Homework 5.2.4.6 Let $A \in \mathbb{R}^{m\times n}$ and let D denote the diagonal matrix with diagonal elements $\delta_0, \delta_1, \cdots, \delta_{n-1}$. Partition A by columns: $A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$.

$$AD = \left(\begin{array}{c|c|c|c} \delta_0 a_0 & \delta_1 a_1 & \cdots & \delta_{n-1} a_{n-1} \end{array} \right).$$

Always/Sometimes/Never

Answer: Always

$$\begin{aligned}
& AD \\
= & \quad < \text{Partition } A \text{ by columns, } D \text{ by elements} > \\
& \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) \left(\begin{array}{c|c|c|c} \delta_0 & 0 & \cdots & 0 \\ \hline 0 & \delta_1 & \cdots & 0 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \vdots & \delta_{n-1} \end{array} \right) \\
= & \quad < \text{Partitioned matrix-matrix multiplication} > \\
& \left(\begin{array}{c|c|c|c} a_0\delta_0 & a_1\delta_1 & \cdots & a_{n-1}\delta_{n-1} \end{array} \right) \\
= & \quad < x\beta = \beta x > \\
& \left(\begin{array}{c|c|c|c} \delta_0 a_0 & \delta_1 a_1 & \cdots & \delta_{n-1} a_{n-1} \end{array} \right).
\end{aligned}$$

☛ BACK TO TEXT

Homework 5.2.4.7 Let $A \in \mathbb{R}^{m\times n}$ and let D denote the diagonal matrix with diagonal elements $\delta_0, \delta_1, \cdots, \delta_{m-1}$. Partition A by rows: $A = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array}\right)$.

$$DA = \left(\begin{array}{c} \delta_0 \widetilde{a}_0^T \\ \hline \delta_1 \widetilde{a}_1^T \\ \hline \vdots \\ \hline \delta_{m-1} \widetilde{a}_{m-1}^T \end{array}\right).$$

Always/Sometimes/Never

Answer: Always

$$DA = \left(\begin{array}{c|c|c|c} \delta_0 & 0 & \cdots & 0 \\ \hline 0 & \delta_1 & \cdots & 0 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \vdots & \delta_{m-1} \end{array}\right) \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array}\right) = \left(\begin{array}{c} \delta_0 \widetilde{a}_0^T \\ \hline \delta_1 \widetilde{a}_1^T \\ \hline \vdots \\ \hline \delta_{m-1} \widetilde{a}_{m-1}^T \end{array}\right)$$

by simple application of partitioned matrix-matrix multiplication.

☛ BACK TO TEXT

Homework 5.2.4.8 Compute $\begin{pmatrix} 1 & -1 & -2 \\ 0 & 2 & 3 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -2 & 1 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} -2 & 0 & -5 \\ 0 & 2 & 7 \\ 0 & 0 & 1 \end{pmatrix}$

☛ BACK TO TEXT

Homework 5.2.4.9 Compute the following, using what you know about partitioned matrix-matrix multiplication:

$$\left(\begin{array}{cc|c} 1 & -1 & -2 \\ 0 & 2 & 3 \\ \hline 0 & 0 & 1 \end{array}\right) \left(\begin{array}{cc|c} -2 & 1 & -1 \\ 0 & 1 & 2 \\ \hline 0 & 0 & 1 \end{array}\right) =$$

Answer:

$$\left(\begin{array}{cc|c} 1 & -1 & -2 \\ 0 & 2 & 3 \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{cc|c} -2 & 1 & -1 \\ 0 & 1 & 2 \\ \hline 0 & 0 & 1 \end{array}\right)$$

$$= \left(\begin{array}{c|c} \begin{pmatrix} 1 & -1 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} -2 & 1 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} -2 \\ 3 \end{pmatrix}\begin{pmatrix} 0 & 0 \end{pmatrix} & \begin{pmatrix} 1 & -1 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} -1 \\ 2 \end{pmatrix} + \begin{pmatrix} -2 \\ 3 \end{pmatrix}(1) \\ \hline \begin{pmatrix} 0 & 0 \end{pmatrix}\begin{pmatrix} -2 & 1 \\ 0 & 1 \end{pmatrix} + (1)\begin{pmatrix} 0 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 0 \end{pmatrix}\begin{pmatrix} -1 \\ 2 \end{pmatrix} + (1)(1) \end{array}\right)$$

$$= \left(\begin{array}{c|c} \begin{pmatrix} -2 & 0 \\ 0 & 2 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} & \begin{pmatrix} -3 \\ 4 \end{pmatrix} + \begin{pmatrix} -2 \\ 3 \end{pmatrix} \\ \hline \begin{pmatrix} 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \end{pmatrix} & (0+0)+(1) \end{array}\right) = \left(\begin{array}{cc|c} -2 & 0 & -5 \\ 0 & 2 & 7 \\ \hline 0 & 0 & 1 \end{array}\right)$$

☛ BACK TO TEXT

Homework 5.2.4.10 Let $U, R \in \mathbb{R}^{n \times n}$ be upper triangular matrices. UR is an upper triangular matrix.

Always/Sometimes/Never

Answer: Always We will prove this by induction on n, the size of the square matrices.

Base case: $n = 1$. If $U, R \in \mathbb{R}^{1 \times 1}$ then they are scalars. (Scalars are inherently upper triangular since they have no elements below the diagonal!). But then UR is also a scalar, which is an upper triangular matrix. Thus the result is true for $n = 1$.

Inductive Step: Induction Hypothesis (I.H.): Assume the result is true for $n = N$, where $N \geq 1$.

We will show the result is true for $n = N + 1$.

Let U and R be $n \times n$ upper triangular matrices with $n = N + 1$. We can partition

$$U = \left(\begin{array}{c|c} U_{00} & u_{01} \\ \hline 0 & \upsilon_{11} \end{array}\right) \quad \text{and} \quad R = \left(\begin{array}{c|c} R_{00} & r_{01} \\ \hline 0 & \rho_{11} \end{array}\right),$$

where U_{00} and R_{00} are $N \times N$ matrices and are upper triangular themselves. Now,

$$\begin{aligned} UR &= \left(\begin{array}{c|c} U_{00} & u_{01} \\ \hline 0 & \upsilon_{11} \end{array}\right)\left(\begin{array}{c|c} R_{00} & r_{01} \\ \hline 0 & \rho_{11} \end{array}\right) \\ &= \left(\begin{array}{c|c} U_{00}R_{00} + u_{01}0 & U_{00}r_{01} + u_{01}\rho_{11} \\ \hline 0R_{00} + \upsilon_{11}0 & 0r_{01} + \upsilon_{11}\rho_{11} \end{array}\right) = \left(\begin{array}{c|c} U_{00}R_{00} & U_{00}r_{01} + u_{01}\rho_{11} \\ \hline 0 & \upsilon_{11}\rho_{11} \end{array}\right). \end{aligned}$$

By the I.H., $U_{00}R_{00}$ is upper triangular. Hence,

$$UR = \left(\begin{array}{c|c} U_{00}R_{00} & U_{00}r_{01} + u_{01}\rho_{11} \\ \hline 0 & \upsilon_{11}\rho_{11} \end{array}\right)$$

is upper triangular.

By the Principle of Mathematical Induction (PMI), the result holds for all n.

☛ BACK TO TEXT

Homework 5.2.4.11 The product of an $n \times n$ lower triangular matrix times an $n \times n$ lower triangular matrix is a lower triangular matrix.

Always/Sometimes/Never

Answer:

Always!

We prove this by induction on n, the size of the square matrices.

Let A and B be lower triangular matrices.

Base case: $n = 1$. If $A, B \in \mathbb{R}^{1\times 1}$ then they are scalars. (Scalars are inherently lower triangular since they have no elements below the diagonal!). But then AB is also a scalar, which is an lower triangular matrix. Thus the result is true for $n = 1$.

Inductive Step: Induction Hypothesis (I.H.): Assume the result is true for $n = N$, where $N \geq 1$.

We will show the result is true for $n = N+1$.

Let A and B be $n \times n$ lower triangular matrices with $n = N+1$. We can partition

$$A = \left(\begin{array}{c|c} A_{00} & 0 \\ \hline a_{10}^T & \alpha_{11} \end{array}\right) \quad \text{and} \quad B = \left(\begin{array}{c|c} B_{00} & 0 \\ \hline b_{10}^T & \beta_{11} \end{array}\right),$$

where A_{00} and B_{00} are $N \times N$ matrices and are lower triangular themselves. Now,

$$\begin{aligned} AB &= \left(\begin{array}{c|c} A_{00} & 0 \\ \hline a_{10}^T & \alpha_{11} \end{array}\right)\left(\begin{array}{c|c} B_{00} & 0 \\ \hline b_{10}^T & \beta_{11} \end{array}\right) \\ &= \left(\begin{array}{c|c} A_{00}B_{00} + 0b_{10}^T & A_{00}0 + 0\beta_{11} \\ \hline a_{10}^T B_{00} + \alpha_{11} b_{10}^T & a_{10}^T 0 + \alpha_{11}\beta_{11} \end{array}\right) = \left(\begin{array}{c|c} A_{00}B_{00} & 0 \\ \hline a_{10}^T B_{00} + \alpha_{11} b_{10}^T & \alpha_{11}\beta_{11} \end{array}\right). \end{aligned}$$

By the I.H., $A_{00}B_{00}$ is lower triangular. Hence,

$$AB = \left(\begin{array}{c|c} A_{00}B_{00} & 0 \\ \hline a_{10}^T B_{00} + \alpha_{11} b_{10}^T & \alpha_{11}\beta_{11} \end{array}\right).$$

is lower triangular.

By the Principle of Mathematical Induction (PMI), the result holds for all n.

☛ BACK TO TEXT

Homework 5.2.4.12 The product of an $n \times n$ lower triangular matrix times an $n \times n$ upper triangular matrix is a diagonal matrix.

Always/Sometimes/Never

Answer: Sometimes An example when the result *is* diagonal: when the two matrices are both diagonal. (A diagonal matrix is a triangular matrix.)

An example when the result is *not* diagonal: Just pick a random example where one of the matrices is not diagonal.

☛ BACK TO TEXT

Homework 5.2.4.13 Let $A \in \mathbb{R}^{m\times n}$. $A^T A$ is symmetric.

Always/Sometimes/Never

Answer: Always

$$(A^T A)^T = A^T (A^T)^T = A^T A.$$

Hence, $A^T A$ is symmetric.

☛ BACK TO TEXT

Homework 5.2.4.14 Evaluate

- $\begin{pmatrix} -1 \\ 1 \\ 2 \end{pmatrix} \begin{pmatrix} -1 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & -1 & -2 \\ -1 & 1 & 2 \\ -2 & 2 & 4 \end{pmatrix}$.
- $\begin{pmatrix} 2 \\ 0 \\ -1 \end{pmatrix} \begin{pmatrix} 2 & 0 & -1 \end{pmatrix} = \begin{pmatrix} 4 & 0 & -2 \\ 0 & 0 & 0 \\ -2 & 0 & 1 \end{pmatrix}$.
- $\left(\begin{array}{r|r} -1 & 2 \\ 1 & 0 \\ 2 & -1 \end{array}\right) \left(\begin{array}{rrr} -1 & 1 & 2 \\ \hline 2 & 0 & -1 \end{array}\right) = \begin{pmatrix} 1 & -1 & -2 \\ -1 & 1 & 2 \\ -2 & 2 & 4 \end{pmatrix} + \begin{pmatrix} 4 & 0 & -2 \\ 0 & 0 & 0 \\ -2 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 5 & -1 & -4 \\ -1 & 1 & 2 \\ -4 & 2 & 5 \end{pmatrix}$.
- $\begin{pmatrix} 1 \\ -2 \\ 2 \end{pmatrix} \begin{pmatrix} 1 & -2 & 2 \end{pmatrix} = \begin{pmatrix} 1 & -2 & 2 \\ -2 & 4 & -4 \\ 2 & -4 & 4 \end{pmatrix}$.
- $\left(\begin{array}{rr|r} -1 & 2 & 1 \\ 1 & 0 & -2 \\ 2 & -1 & 2 \end{array}\right) \left(\begin{array}{rrr} -1 & 1 & 2 \\ 2 & 0 & -1 \\ \hline 1 & -2 & 2 \end{array}\right) =$

  Answer:

  $$\begin{pmatrix} -1 & 2 \\ 1 & 0 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} -1 & 1 & 2 \\ 2 & 0 & -1 \end{pmatrix} + \begin{pmatrix} 1 \\ -2 \\ 2 \end{pmatrix} \begin{pmatrix} 1 & -2 & 2 \end{pmatrix}$$
  $$= \begin{pmatrix} 5 & -1 & -4 \\ -1 & 1 & 2 \\ -4 & 2 & 5 \end{pmatrix} + \begin{pmatrix} 1 & -2 & 2 \\ -2 & 4 & -4 \\ 2 & -4 & 4 \end{pmatrix} = \begin{pmatrix} 6 & -3 & -2 \\ -3 & 5 & -2 \\ -2 & -2 & 9 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 5.2.4.15 Let $x \in \mathbb{R}^n$. The outer product xx^T is symmetric.

Always/Sometimes/Never

Answer: Always

Proof 1: Since $A^T A$ is symmetric for any matrix $A \in \mathbb{R}^{m \times n}$ and vector $A = x^T \in \mathbb{R}^n$ is just the special case where the matrix is a vector.

Proof 2:

$$xx^T = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}^T = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} \begin{pmatrix} \chi_0 & \chi_1 & \cdots & \chi_{n-1} \end{pmatrix}$$
$$= \begin{pmatrix} \chi_0\chi_0 & \chi_0\chi_1 & \cdots & \chi_0\chi_{n-1} \\ \chi_1\chi_0 & \chi_1\chi_1 & \cdots & \chi_1\chi_{n-1} \\ \vdots & \vdots & & \vdots \\ \chi_{n-1}\chi_0 & \chi_{n-1}\chi_1 & \cdots & \chi_{n-1}\chi_{n-1} \end{pmatrix}.$$

Since $\chi_i\chi_j = \chi_j\chi_i$, the (i,j) element of xx^T equals the (j,i) element of xx^T. This means xx^T is symmetric.

Proof 3: $(xx^T)^T = (x^T)^T x^T = xx^T$.

☛ BACK TO TEXT

Homework 5.2.4.16 Let $A \in \mathbb{R}^{n\times n}$ be symmetric and $x \in \mathbb{R}^n$. $A + xx^T$ is symmetric.

Always/Sometimes/Never

Answer: Always If matrices $A, B \in \mathbb{R}^{n\times n}$ are symmetric, then $A+B$ is symmetric since $(A+B)^T = A^T + B^T = A + B$ (which we saw in Homework 3.3.2.8. In this case, $B = xx^T$, which in a previous exercise we saw is symmetric.

☛ BACK TO TEXT

Homework 5.2.4.17 Let $A \in \mathbb{R}^{m\times n}$. Then AA^T is symmetric. (In your reasoning, we want you to use insights from previous homeworks.)

Always/Sometimes/Never

Answer: Always

Proof 1: $(AA^T)^T = (A^T)^T A^T = AA^T$.

Proof 2: We know that $A^T A$ is symmetric. Take $B = A^T$. Then $AA^T = B^T B$ and hence AA^T is symmetric.

Proof 3:

$$\begin{aligned} AA^T &= \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)^T \\ &= \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) \left(\begin{array}{c} a_0^T \\ \hline a_1^T \\ \hline \vdots \\ \hline a_{n-1}^T \end{array} \right) \\ &= a_0 a_0^T + a_1 a_1^T + \cdots + a_{n-1} a_{n-1}^T. \end{aligned}$$

But each $a_j a_j^T$ is symmetric (by a previous exercise) and adding symmetric matrices yields a symmetric matrix. Hence, AA^T is symmetric.

Proof 4:

Proof by induction on n.

Base case: $A = \left(a_0 \right)$, where a_0 is a vector. Then $AA^T = a_0 a^T$. But we saw in an earlier homework that if x is a vector, then xx^T is symmetric.

Induction Step: Assume that AA^T is symmetric for matrices with $n = N$ columns, where $N \geq 1$. We will show that AA^T is symmetric for matrices with $n = N+1$ columns.

Let A have $N+1$ columns.

$$\begin{aligned} & AA^T \\ = & \quad < \text{Partition } A > \\ & \left(\begin{array}{c|c} A_0 & a_1 \end{array} \right) \left(\begin{array}{c|c} A_0 & a_1 \end{array} \right)^T \\ = & \quad < \text{Transpose partitioned matrix } > \\ & \left(\begin{array}{c|c} A_0 & a_1 \end{array} \right) \left(\begin{array}{c} A_0^T \\ \hline a_1^T \end{array} \right) \\ = & \quad < \text{Partitioned matrix-matrix multiplication } > \\ & A_0 A_0^T + a_1 a_1^T \end{aligned}$$

Now, by the I.H. $A_0 A_0^T$ is symmetric. From a previous exercise we know that xx^T is symmetric and hence $a_1 a_1^T$ is. From another exercise we know that adding symmetric matrices yields a symmetric matrix.

By the Principle of Mathematical Induction (PMI), the result holds for all n.

☛ BACK TO TEXT

Homework 5.2.4.18 Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric matrices. AB is symmetric.

Always/Sometimes/Never

Answer: Sometimes Examples of when this is *true*:

- $B = A$ so that $AB = AA$. Then $(AA)^T = A^T A^T = AA$.
- $A = I$ or $B = I$. $IB = B$ and hence IB is symmetric. Similarly, $AI = A$ and hence AI is symmetric.

An examples of when this is *false*: $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ $B = \begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix}$. Then $AB = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix} = \begin{pmatrix} -1 & 2 \\ 1 & -1 \end{pmatrix}$, which is not a symmetric matrices.

☛ BACK TO TEXT

5.3 Algorithms for Computing Matrix-Matrix Multiplication

5.3.1 Lots of Loops

Homework 5.3.1.1 Consider the MATLAB function

```
function [ C_out ] = MatMatMult( A, B, C )

[ m, n ] = size( C );
[ m_A, k ] = size( A );
[ m_B, n_B ] = size( B );

for j = 1:n
   for i = 1:m
      for p = 1:k
         C( i,j ) = A( i, p ) * B( p, j ) + C( i, j );
      end
   end
end
```

- Download the files `MatMatMult.m` and `test_MatMatMult.m` into, for example,

  `LAFF-2.0xM -> Programming -> Week5`

  (creating the directory if necessary).
- Examine the script `test_MatMatMult.m` and then execute it in the MATLAB Command Window: `test_MatMatMult`.
- Now, exchange the order of the loops:

```
for j = 1:n
   for p = 1:k
      for i = 1:m
         C( i,j ) = A( i, p ) * B( p, j ) + C( i, j );
      end
   end
end
```

save the result, and execute `test_MatMatMult` again. What do you notice?

- How may different ways can you order the "triple-nested loop"?

  Answer: There are six different ways of ordering the "triple-nested loop":

  - Consider the loop indices i, j, and p.
  - For the outer-most loop you can choose any of the three indices.
  - For the next loop you are left with two indices from which to choose.
  - For the inner-most loop, you are left with only one choice.

  Thus there are $3 \times 2 \times 1$ (3 factorial) ways to order the loops.

- Try them all and observe how the result of executing `test_MatMatMult` does or does not change.

☛ BACK TO TEXT

5.3.2 Matrix-Matrix Multiplication by Columns

Homework 5.3.2.1 Let A and B be matrices and AB be well-defined and let B have at least four columns. If the first and fourth columns of B are the same, then the first and fourth columns of AB are the same.

Always/Sometimes/Never

Answer: Always

Partition

$$B = \begin{pmatrix} b_0 & b_1 & b_2 & b_3 & B_4 \end{pmatrix},$$

where B_4 represents the part of the matrix to the right of the first four columns. Then

$$AB = A \begin{pmatrix} b_0 & b_1 & b_2 & b_3 & B_4 \end{pmatrix} = \begin{pmatrix} Ab_0 & Ab_1 & Ab_2 & Ab_3 & AB_4 \end{pmatrix}.$$

Now, if $b_0 = b_3$ then $Ab_0 = Ab_3$ and hence the first and fourth columns of AB are equal.

☛ BACK TO TEXT

Homework 5.3.2.2 Let A and B be matrices and AB be well-defined and let A have at least four columns. If the first and fourth columns of A are the same, then the first and fourth columns of AB are the same.

Always/Sometimes/Never

Answer: Sometimes To find an example where the statement is *true*, we first need to make sure that the result has at least four columns, which means that B must have at least four columns. Then an example when the statement is *true*: $A = 0$ (the zero matrix) or $B = I$ (the identity matrix of size at least 4×4).

An example when it is *false*: Almost any matrices A and B. For example:

$$A = \begin{pmatrix} 0 & 1 & 2 & 0 \\ 0 & 1 & 2 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

so that

$$AB = \begin{pmatrix} 0 & 1 & 2 & 0 \\ 0 & 1 & 2 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 2 & 2 \\ 0 & 1 & 2 & 2 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 5.3.2.3

- $$\begin{pmatrix} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \left(\begin{array}{|r|cc} -1 & & \\ 2 & & \\ 1 & & \end{array}\right) = \left(\begin{array}{|r|cc} -3 & & \\ 6 & & \\ 4 & & \end{array}\right)$$
- $$\begin{pmatrix} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \left(\begin{array}{r|r|c} -1 & 0 & \\ 2 & 1 & \\ 1 & -1 & \end{array}\right) = \left(\begin{array}{r|r|c} -3 & -4 & \\ 6 & 1 & \\ 4 & -1 & \end{array}\right)$$
- $$\begin{pmatrix} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \left(\begin{array}{rr|r|} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) = \left(\begin{array}{rr|r|} -3 & -4 & 7 \\ 6 & 1 & -1 \\ 4 & -1 & 3 \end{array}\right)$$

☛ BACK TO TEXT

Homework 5.3.2.4 Implement the routine

```
[ C_out ] = Gemm_unb_var1( A, B, C )
```

based on the algorithm in Figure 5.1.

☛ BACK TO TEXT

5.3.3 Matrix-Matrix Multiplication by Rows

Homework 5.3.3.1 Let A and B be matrices and AB be well-defined and let A have at least four rows. If the first and fourth rows of A are the same, then the first and fourth rows of AB are the same.

Always/Sometimes/Never

Answer: Always

Partition

$$A = \begin{pmatrix} \widetilde{a}_0^T \\ \widetilde{a}_1^T \\ \widetilde{a}_2^T \\ \widetilde{a}_3^T \\ A_4 \end{pmatrix}$$

where A_4 represents the part of the matrix below the first four rows. Then

$$AB = \begin{pmatrix} \widetilde{a}_0^T \\ \widetilde{a}_1^T \\ \widetilde{a}_2^T \\ \widetilde{a}_3^T \\ A_4 \end{pmatrix} B = \begin{pmatrix} \widetilde{a}_0^T B \\ \widetilde{a}_1^T B \\ \widetilde{a}_2^T B \\ \widetilde{a}_3^T B \\ A_4 B \end{pmatrix}.$$

Now, if $\widetilde{a}_0^T = \widetilde{a}_3^T$ then $\widetilde{a}_0^T B = \widetilde{a}_3^T B$ and hence the first and fourth rows of AB are equal.

☛ BACK TO TEXT

Homework 5.3.3.2

- $$\left(\begin{array}{ccc} \hline 1 & -2 & 2 \\ \hline & & \\ & & \end{array}\right) \left(\begin{array}{ccc} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) = \left(\begin{array}{ccc} \hline -3 & -4 & 7 \\ \hline & & \\ & & \end{array}\right)$$

- $$\left(\begin{array}{ccc} 1 & -2 & 2 \\ \hline -1 & 2 & 1 \\ \hline & & \end{array}\right) \left(\begin{array}{ccc} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) = \left(\begin{array}{ccc} -3 & -4 & 7 \\ \hline 6 & 1 & -1 \\ \hline & & \end{array}\right)$$

- $$\left(\begin{array}{ccc} 1 & -2 & 2 \\ -1 & 2 & 1 \\ \hline 0 & 1 & 2 \\ \hline \end{array}\right) \left(\begin{array}{ccc} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) = \left(\begin{array}{ccc} -3 & -4 & 7 \\ 6 & 1 & -1 \\ \hline 4 & -1 & 3 \\ \hline \end{array}\right)$$

☛ BACK TO TEXT

Homework 5.3.3.3 Implement the routine

```
[ C_out ] = Gemm_unb_var2( A, B, C )
```

based on the algorithm in Figure 5.2.

☛ BACK TO TEXT

5.3.4 Matrix-Matrix Multiplication with Rank-1 Updates

Homework 5.3.4.1

- $$\left(\begin{array}{|c|cc} 1 & & \\ -1 & & \\ 0 & & \end{array}\right) \left(\begin{array}{ccc} \hline -1 & 0 & 1 \\ \hline & & \\ & & \end{array}\right) = \left(\begin{array}{ccc} \hline -1 & 0 & 1 \\ \hline 1 & 0 & -1 \\ 0 & 0 & 0 \end{array}\right)$$

- $$\left(\begin{array}{c|c|c} & -2 & \\ & 2 & \\ & 1 & \end{array}\right) \left(\begin{array}{ccc} & & \\ \hline 2 & 1 & -1 \\ \hline & & \end{array}\right) = \left(\begin{array}{ccc} -4 & -2 & 2 \\ \hline 4 & 2 & -2 \\ \hline 2 & 1 & -1 \end{array}\right)$$

- $$\left(\begin{array}{cc|c|} & & 2 \\ & & 1 \\ & & 2 \end{array}\right) \left(\begin{array}{ccc} & & \\ & & \\ \hline 1 & -1 & 2 \\ \hline \end{array}\right) = \left(\begin{array}{ccc} 2 & -2 & 4 \\ 1 & -1 & 2 \\ \hline 2 & -2 & 4 \\ \hline \end{array}\right)$$

- $$\left(\begin{array}{ccc} 1 & -2 & 2 \\ -1 & 2 & 1 \\ 0 & 1 & 2 \end{array}\right) \left(\begin{array}{ccc} -1 & 0 & 1 \\ 2 & 1 & -1 \\ 1 & -1 & 2 \end{array}\right) = \left(\begin{array}{ccc} -3 & -4 & 7 \\ 6 & 1 & -1 \\ 4 & -1 & 3 \end{array}\right)$$

Answer: The important thing to notice is that the last result equals the first three results added together.

☛ BACK TO TEXT

Homework 5.3.4.2 Implement the routine

```
[ C_out ] = Gemm_unb_var2( A, B, C )
```

based on the algorithm in Figure 5.3.

☛ BACK TO TEXT

5.5 Wrap Up

5.5.1 Homework

Homework 5.5.1.1 Let A and B be matrices and AB be well-defined. $(AB)^2 = A^2B^2$.

Always/Sometimes/Never

Answer: Sometimes

The result is obviously true if $A = B$. (There are other examples. E.g., if A or B is a zero matrix, or if A or B is an identity matrix.)

If $A \neq B$, then the result is not well defined unless A and B are both square. (Why?). Let's assume A and B are both square. Even then, generally $(AB)^2 \neq A^2B^2$. Let

$$A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Then

$$(AB)^2 = ABAB = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix}$$

and

$$A^2B^2 = AABB = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}.$$

(I used MATLAB to check some possible matrices. There was nothing special about my choice of using triangular matrices.)

This may be counter intuitive since if α and β are scalars, then $(\alpha\beta)^2 = \alpha^2\beta^2$.

☛ BACK TO TEXT

Homework 5.5.1.2 Let A be symmetric. A^2 is symmetric.

Always/Sometimes/Never

Answer: Always

$$(AA)^T = A^TA^T = AA.$$

☛ BACK TO TEXT

Homework 5.5.1.3 Let $A, B \in \mathbb{R}^{n\times n}$ both be symmetric. AB is symmetric.

Always/Sometimes/Never

Answer: Sometimes Simple examples of when it is *true*: $A = I$ and/or $B = I$. $A = 0$ and/or $B = 0$. All cases where $n = 1$.

Simple example of where it is NOT true:

$$A = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 3 & 0 \\ 0 & -2 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 5.5.1.4 Let $A, B \in \mathbb{R}^{n \times n}$ both be symmetric. $A^2 - B^2$ is symmetric.

Always/Sometimes/Never

Answer: Always We just saw that AA is always symmetric. Hence AA and BB are symmetric. But adding two symmetric matrices yields a symmetric matrix, so the resulting matrix is symmetric.

Or:

$$(A^2 - B^2)^T = (A^2)^T - (B^2)^T = A^2 - B^2.$$

☛ BACK TO TEXT

Homework 5.5.1.5 Let $A, B \in \mathbb{R}^{n \times n}$ both be symmetric. $(A+B)(A-B)$ is symmetric.

Always/Sometimes/Never

Answer: Sometimes

Examples of when it IS symmetric: $A = B$ or $A = 0$ or $A = I$.

Examples of when it is NOT symmetric: Create random 2x2 matrices A and B in MATLAB. Then set $A := A^T A$ and $B = B^T B$ to make them symmetric. With probability 1 you will see that $(A+B)(A-B)$ is not symmetric. Here is an example:

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

BUT, what we really want you to notice is that if you multiply out

$$(A+B)(A-B) = A^2 + BA - AB - B^2$$

the middle terms do NOT cancel. Compare this to the case where you work with real scalars:

$$(\alpha + \beta)(\alpha - \beta) = \alpha^2 + \beta\alpha - \alpha\beta - \beta^2 = \alpha^2 - \beta^2.$$

☛ BACK TO TEXT

Homework 5.5.1.6 Let $A, B \in \mathbb{R}^{n \times n}$ both be symmetric. ABA is symmetric.

Always/Sometimes/Never

Answer: Always

$$(ABA)^T = A^T B^T A^T = ABA.$$

☛ BACK TO TEXT

Homework 5.5.1.7 Let $A, B \in \mathbb{R}^{n \times n}$ both be symmetric. $ABAB$ is symmetric.

Always/Sometimes/Never

Answer: Sometimes It is *true* for, for example, $A = B$. But is is, for example, *false* for

$$A = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 3 & 0 \\ 0 & -2 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 5.5.1.8 Let A be symmetric. $A^T A = AA^T$.

Always/Sometimes/Never

Answer: Always Trivial, since $A = A^T$.

☛ BACK TO TEXT

Homework 5.5.1.9 If $A = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ then $A^T A = AA^T$.

True/False

Answer: False

$$\begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}^T \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = 2 \text{ and } \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}^T = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 5.5.1.10 Propose an algorithm for computing $C := UR$ where C, U, and R are all upper triangular matrices by completing the below algorithm.

Algorithm: $[C] := \text{TRTRMM\_UU\_UNB\_VAR1}(U, R, C)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right)$, $C \rightarrow \left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right)$

where U_{TL} is 0×0, R_{TL} is 0×0, C_{TL} is 0×0

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$$

where υ_{11} is 1×1, ρ_{11} is 1×1, γ_{11} is 1×1

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$$

endwhile

Hint: consider Homework 5.2.4.10. Then implement and test it.

Answer: You will want to implement the following algorithm:

Algorithm: $[C] := \text{TRTRMM\_UU\_UNB\_VAR1}(U, R, C)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right)$, $C \rightarrow \left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right)$

where U_{TL} is 0×0, R_{TL} is 0×0, C_{TL} is 0×0

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$$

where υ_{11} is 1×1, ρ_{11} is 1×1, γ_{11} is 1×1

$c_{01} := U_{00} r_{01}$

$c_{01} := \rho_{11} u_{01} + c_{01}$

$\gamma_{11} := \rho_{11} \upsilon_{11}$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} C_{00} & c_{01} & C_{02} \\ \hline c_{10}^T & \gamma_{11} & c_{12}^T \\ \hline C_{20} & c_{21} & C_{22} \end{array}\right)$$

endwhile

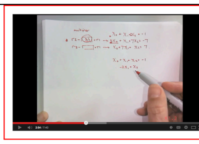
☛ BACK TO TEXT

Week 6: Gaussian Elimination (Answers)

6.2 Gaussian Elimination

6.2.1 Reducing a System of Linear Equations to an Upper Triangular System

Homework 6.2.1.1



☛ YouTube

Practice reducing a system of linear equations to an upper triangular system of linear equations by visiting the Practice with Gaussian Elimination webpage we created for you. For now, only work with the top part of that webpage.

☛ BACK TO TEXT

Homework 6.2.1.2 Compute the solution of the linear system of equations given by

$$\begin{array}{rcrcrcr} -2\chi_0 & + & \chi_1 & + & 2\chi_2 & = & 0 \\ 4\chi_0 & - & \chi_1 & - & 5\chi_2 & = & 4 \\ 2\chi_0 & - & 3\chi_1 & - & \chi_2 & = & -6 \end{array}$$

- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \Box \\ \Box \\ \Box \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ -2 \end{pmatrix}$

Answer:

$$\begin{array}{rcrcrcr} -2\chi_0 & + & \chi_1 & + & 2\chi_2 & = & 0 \\ 4\chi_0 & - & \chi_1 & - & 5\chi_2 & = & 4 \\ 2\chi_0 & - & 3\chi_1 & - & \chi_2 & = & -6 \end{array} \longrightarrow \begin{array}{rcrcrcr} -2\chi_0 & + & \chi_1 & + & 2\chi_2 & = & 0 \\ & & \chi_1 & - & \chi_2 & = & 4 \\ & & -2\chi_1 & + & \chi_2 & = & -6 \end{array} \longrightarrow$$

$$\begin{array}{rcrcrcr} -2\chi_0 & + & \chi_1 & + & 2\chi_2 & = & 0 \\ & & \chi_1 & - & \chi_2 & = & 4 \\ & & & & -\chi_2 & = & 2 \end{array} \longrightarrow \left\{ \begin{array}{ccc} -\chi_2 = 2 & \Rightarrow & \chi_2 = -2 \\ \chi_1 - (-2) = 4 & \Rightarrow & \chi_1 = 2 \\ -2\chi_0 + (2) + 2(-2) = 0 & \Rightarrow & \chi_0 = -1 \end{array} \right.$$

☛ BACK TO TEXT

Homework 6.2.1.3 Compute the coefficients γ_0, γ_1, and γ_2 so that

$$\sum_{i=0}^{n-1} i = \gamma_0 + \gamma_1 n + \gamma_2 n^2$$

(by setting up a system of linear equations).

Answer: Earlier in this course, as an example when discussing proof by induction and then again later when discussing the cost of a matrix-vector multiplication with a triangular matrix and the solution of a triangular system of equations, we encountered

$$\sum_{i=0}^{n-1} i.$$

Now, you may remember that the summation was equivalent to some quadratic (second degree) polynomial in n, but not what the coefficients of that polynomial were:

$$\sum_{i=0}^{n-1} i = \gamma_0 + \gamma_1 n + \gamma_2 n^2,$$

for some constant scalars γ_0, γ_1, and γ_2. What if you wanted to determine what these coefficients are? Well, you now know how to solve linear systems, and we now see that determining the coefficients is a matter of solving a linear system.

Starting with

$$p_2(n) = \sum_{i=0}^{n-1} i = \gamma_0 + \gamma_1 n + \gamma_2 n^2,$$

we compute the value of $p_2(n)$ for $n = 0, 1, 2$:

$$\begin{array}{lcccccccccc}
p_2(0) = \sum_{i=0}^{(0)-1} i & = & \gamma_0 (0)^0 & + & \gamma_1 (0) & + & \gamma_2 (0)^2 & = & 0 & = & 0 \\
p_2(1) = \sum_{i=0}^{(1)-1} i & = & \gamma_0 (1)^0 & + & \gamma_1 (1) & + & \gamma_2 (1)^2 & = & 0 & = & 0 \\
\sum_{i=0}^{(2)-1} i & = & \gamma_0 (2)^0 & + & \gamma_1 (2) & + & \gamma_2 (2)^2 & = & 0+1 & = & 1
\end{array}$$

or, in matrix notation,

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

One can then solve this system to find that

$$\begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 0 \\ -\frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$$

so that

$$\sum_{i=0}^{n-1} i = \frac{1}{2} n^2 - \frac{1}{2} n$$

which equals the $n(n-1)/2$ that we encountered before.

☛ BACK TO TEXT

Homework 6.2.1.4 Compute γ_0, γ_1, γ_2, and γ_3 so that

$$\sum_{i=0}^{n-1} i^2 = \gamma_0 + \gamma_1 n + \gamma_2 n^2 + \gamma_3 n^3.$$

Answer: (Note: $\sum_{i=0}^{-1}$ anything $= 0$.)

$$\begin{array}{lcccccccccccc}
\sum_{i=0}^{(0)-1} i^2 & = & \gamma_0 & + & \gamma_1 (0) & + & \gamma_2 (0)^2 & + & \gamma_3 (0)^3 & = & 0 & = & 0 \\
\sum_{i=0}^{(1)-1} i^2 & = & \gamma_0 & + & \gamma_1 (1) & + & \gamma_2 (1)^2 & + & \gamma_3 (1)^3 & = & 0^2 & = & 0 \\
\sum_{i=0}^{(2)-1} i^2 & = & \gamma_0 & + & \gamma_1 (2) & + & \gamma_2 (2)^2 & + & \gamma_3 (2)^3 & = & 0^2 + 1^2 & = & 1 \\
\sum_{i=0}^{(3)-1} i^2 & = & \gamma_0 & + & \gamma_1 (3) & + & \gamma_2 (3)^2 & + & \gamma_3 (3)^3 & = & 0^2 + 1^2 + 2^2 & = & 5
\end{array}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 5 \end{pmatrix}$$

Notice that $\gamma_0 = 0$. So

$$\begin{pmatrix} 1 & 1 & 1 \\ 2 & 4 & 8 \\ 3 & 9 & 27 \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 5 \end{pmatrix}$$

Solution:

$$\begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{pmatrix} = \frac{1}{6} \begin{pmatrix} 1 \\ -3 \\ 2 \end{pmatrix}$$

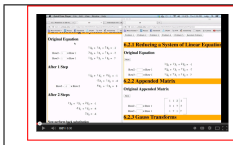
so that

$$\sum_{i=0}^{n-1} i^2 = \frac{1}{6}n - \frac{1}{2}n^2 + \frac{1}{3}n^3.$$

☛ BACK TO TEXT

6.2.2 Appended Matrices

Homework 6.2.2.1



☛ YouTube

Practice reducing a system of linear equations to an upper triangular system of linear equations by visiting the Practice with Gaussian Elimination webpage we created for you. For now, only work with the top two parts of that webpage.

☛ BACK TO TEXT

Homework 6.2.2.2 Compute the solution of the linear system of equations expressed as an appended matrix given by

$$\left(\begin{array}{rrr|r} -1 & 2 & -3 & 2 \\ -2 & 2 & -8 & 10 \\ 2 & -6 & 6 & -2 \end{array}\right)$$

- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \square \\ \square \\ \square \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \\ -2 \end{pmatrix}$

Answer:

$$\left(\begin{array}{rrr|r} -1 & 2 & -3 & 2 \\ -2 & 2 & -8 & 10 \\ 2 & -6 & 6 & -2 \end{array}\right) \longrightarrow \left(\begin{array}{rrr|r} -1 & 2 & -3 & 2 \\ 0 & -2 & -2 & 6 \\ 0 & -2 & 0 & 2 \end{array}\right) \longrightarrow$$

$$\left(\begin{array}{rrr|r} -1 & 2 & -3 & 2 \\ 0 & -2 & -2 & 6 \\ 0 & 0 & 2 & -4 \end{array}\right) \longrightarrow \left\{ \begin{array}{rcl} 2\chi_2 = -4 & \Rightarrow & \chi_2 = -2 \\ -2\chi_1 - (2)(-2) = 6 & \Rightarrow & \chi_1 = -1 \\ -\chi_0 + (2)(-1) + (-3)(-2) = 2 & \Rightarrow & \chi_0 = 2 \end{array} \right.$$

☛ BACK TO TEXT

6.2.3 Gauss Transforms

Homework 6.2.3.1

Compute ONLY the values in the boxes. A $\star$ means a value that we don't care about.

- $\begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 345 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \star & \star & \star \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ \Box & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 2 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ 0 & \Box & \Box \\ \Box & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ \Box & 1 & 0 \\ \Box & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 2 & -2 & 6 \\ -4 & -4 & 2 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ 0 & \Box & \Box \\ 0 & \Box & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \Box & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & -16 & 8 \end{pmatrix} = \begin{pmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & 0 & \Box \end{pmatrix}.$

- $\begin{pmatrix} 1 & 0 & \Box \\ 0 & 1 & \Box \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 4 & -8 \\ 1 & 1 & -4 \\ -1 & -2 & 4 \end{pmatrix} = \begin{pmatrix} \Box & \Box & 0 \\ \Box & \Box & 0 \\ \Box & \Box & \Box \end{pmatrix}.$

Answer:

- $\begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \left(\begin{array}{ccc} 2 & 4 & -2 \\ \hline 4 & -2 & 6 \\ \hline 6 & -4 & 2 \end{array}\right) = \left(\begin{array}{ccc} 2 & 4 & -2 \\ \hline 0 & -10 & 10 \\ \hline 6 & -4 & 2 \end{array}\right).$

- $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 345 & 0 & 1 \end{pmatrix} \left(\begin{array}{ccc} 2 & 4 & -2 \\ \hline 4 & -2 & 6 \\ \hline 6 & -4 & 2 \end{array}\right) = \left(\begin{array}{ccc} 2 & 4 & -2 \\ \hline 4 & -2 & 6 \\ \hline \star & \star & \star \end{array}\right).$

- $\left(\begin{array}{rrr} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & 0 & 1 \end{array}\right)\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 4 & -2 & 6 \\ \hline 6 & -4 & 2 \end{array}\right)=\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 4 & -2 & 6 \\ \hline 0 & -16 & 8 \end{array}\right).$

- $\left(\begin{array}{rrr} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right)\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 2 & -2 & 6 \\ \hline 6 & -4 & 2 \end{array}\right)=\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 0 & -6 & 8 \\ \hline 6 & -4 & 2 \end{array}\right).$

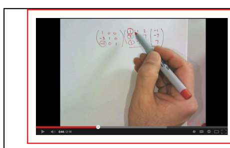
- $\left(\begin{array}{rrr} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 0 & 1 \end{array}\right)\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 2 & -2 & 6 \\ \hline -4 & -4 & 2 \end{array}\right)=\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 0 & -6 & 8 \\ \hline 0 & 4 & -2 \end{array}\right).$

- $\left(\begin{array}{rrr} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1.6 & 1 \end{array}\right)\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 0 & -10 & 10 \\ \hline 0 & -16 & 8 \end{array}\right)=\left(\begin{array}{rrr} 2 & 4 & -2 \\ \hline 0 & -10 & 10 \\ \hline 0 & 0 & -8 \end{array}\right).$

- $\left(\begin{array}{rrr} 1 & 0 & 2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{array}\right)\left(\begin{array}{rrr} 2 & 4 & -8 \\ \hline 1 & 1 & -4 \\ \hline -1 & -2 & 4 \end{array}\right)=\left(\begin{array}{rrr} 0 & 0 & 0 \\ \hline 0 & -1 & 0 \\ \hline -1 & -2 & 4 \end{array}\right).$

☛ BACK TO TEXT

Homework 6.2.3.2



☛ YouTube

Practice reducing an appended sytem to an upper triangular form with Gauss transforms by visiting the Practice with Gaussian Elimination webpage we created for you. For now, only work with the top three parts of that webpage.

☛ BACK TO TEXT

6.2.4 Computing Separately with the Matrix and Right-Hand Side (Forward Substitution)

Homework 6.2.4.1 No video this time! We trust that you have probably caught on to how to use the webpage.

Practice reducing a matrix to an upper triangular matrix with Gauss transforms and then applying the Gauss transforms to a right-hand side by visiting the Practice with Gaussian Elimination webpage we created for you. Now you can work with all parts of the webpage. Be sure to compare and contrast!

☛ BACK TO TEXT

6.2.5 Towards an Algorithm

Homework 6.2.5.2 Implement the algorithms in Figures 6.1 and 6.2

- `[ A_out ] = GaussianElimination( A )`
- `[ b_out ] = ForwardSubstitution( A, b )`

You can check that they compute the right answers with the following script:

- test_GausianElimination.m

This script exercises the functions by factoring the matrix

```
A = [
     2     0     1     2
    -2    -1     1    -1
     4    -1     5     4
    -4     1    -3    -8
]
```

by calling

```
LU = GaussianElimination( A )
```

Next, solve $Ax = b$ where

```
b = [
     2
     2
    11
    -3
]
```

by first apply forward substitution to b, using the output matrix `LU`:

```
bhat = ForwardSubstitution( LU, b )
```

extracting the upper triangular matrix U from `LU`:

```
U = triu( LU )
```

and then solving $Ux = \widehat{b}$ (which is equivalent to backward substitution) with the MATLAB intrinsic function:

```
x = U \ bhat
```

Finally, check that you got the right answer:

```
b - A * x
```

(the result should be a zero vector with four elements).

Answer:

Here are our implementations of the functions:

- GaussianElimination.m
- ForwardSubstitution.m

You can check that they compute the right answers with the following script:

- test_GausianElimination.m

☛ BACK TO TEXT

6.3 Solving $Ax = b$ via LU Factorization

6.3.1 LU factorization (Gaussian elimination)

Homework 6.3.1.1 Implement the algorithm in Figures 6.4.

- `[ A_out ] = LU_unb_var5( A )`

You can check that they compute the right answers with the following script:

- test_LU_unb_var5.m

This script exercises the functions by factoring the matrix

```
A = [
     2      0      1      2
    -2     -1      1     -1
     4     -1      5      4
    -4      1     -3     -8
]
```

by calling

```
LU = LU_unb_var5( A )
```

Next, it extracts the unit lower triangular matrix and upper triangular matrix:

```
L = tril( LU, -1 ) + eye( size( A ) )

U = triu( LU )
```

and checks if the correct factors were computed:

```
A - L * U
```

which should yield a 4×4 zero matrix.

Answer:

Here is our implementations of the function:

- LU_unb_var5.m

☛ BACK TO TEXT

Homework 6.3.1.2 Compute the LU factorization of

$$\begin{pmatrix} 1 & -2 & 2 \\ 5 & -15 & 8 \\ -2 & -11 & -11 \end{pmatrix}.$$

Answer:

$$\begin{pmatrix} 1 & -2 & 2 \\ 5 & -15 & 8 \\ -2 & -11 & -11 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 5 & 1 & 0 \\ -2 & 3 & 1 \end{pmatrix} \begin{pmatrix} 1 & -2 & 2 \\ 0 & -5 & -2 \\ 0 & 0 & -1 \end{pmatrix}.$$

Here are the details when executing the algorithm:

Iteration	Before	After
1	$\left(\begin{array}{\|c\|cc} \hline 1 & -2 & 2 \\ \hline 5 & -15 & 8 \\ -2 & -11 & -11 \end{array}\right)$	$\left(\begin{array}{\|c\|cc} \hline 1 & -2 & 2 \\ \hline 5 & -5 & -2 \\ -2 & -15 & -7 \end{array}\right)$
2	$\left(\begin{array}{c\|c\|c} 1 & -2 & 2 \\ \hline 5 & -5 & -2 \\ \hline -2 & -15 & -7 \end{array}\right)$	$\left(\begin{array}{c\|cc} 1 & -2 & 2 \\ \hline 5 & -5 & -2 \\ \hline -2 & 3 & -1 \end{array}\right)$
3	$\left(\begin{array}{cc\|c} 1 & -2 & 2 \\ 5 & -5 & -2 \\ \hline -2 & 3 & -1 \end{array}\right)$	$\left(\begin{array}{cc\|c} 1 & -2 & 2 \\ 5 & -5 & -2 \\ \hline -2 & 3 & -1 \end{array}\right)$

The unit lower triangular matrix L and upper triangular matrix U can then be read off as:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 5 & 1 & 0 \\ -2 & 3 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & -2 & 2 \\ 0 & -5 & -2 \\ 0 & 0 & -1 \end{pmatrix}$$

☛ BACK TO TEXT

6.3.2 Solving $Lz = b$ (Forward substitution)

Homework 6.3.2.1 Implement the algorithm in Figure 6.15.

- `[ b_out ] = Ltrsv_unb_var1( L, b )`

You can check that they compute the right answers with the following script:

- `test_Ltrsv_unb_var1.m`

This script exercises the function by setting the matrix

```
L = [
     1     0     0     0
    -1     1     0     0
     2     1     1     0
    -2    -1     1     1
]
```

and solving $Lx = b$ with the right-hand size vector

```
b = [
     2
     2
    11
    -3
]
```

by calling

```
x = Ltrsv_unb_var1( L, b )
```

Finally, it checks if `x` is indeed the answer by checking if

```
b - L * x
```

equals the zero vector.

```
x = U \ z
```

We can the check if this solves $Ax = b$ by computing

```
b - A * x
```

which should yield a zero vector.

Answer:

Here is our implementations of the function:

- `Ltrsv_unb_var1.m`

☛ BACK TO TEXT

6.3.3 Solving $Ux = b$ (Back substitution)

Homework 6.3.2.1 Side-by-side, solve the upper triangular linear system

$$\begin{array}{rcrcrcr} -2\chi_0 & - & \chi_1 & + & \chi_2 & = & 6 \\ & & -3\chi_1 & - & 2\chi_2 & = & 9 \\ & & & & \chi_2 & = & 3 \end{array}$$

via back substitution and by executing the above algorithm with

$$U = \begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 6 \\ 9 \\ 3 \end{pmatrix}.$$

Compare and contrast!

☛ BACK TO TEXT

6.3.4 Putting it all together to solve $Ax = b$

Homework 6.3.2.1 Implement the function

- `[ A_out, b_out ] = Solve( A, b )`

that

- Computes the LU factorization of matrix A, $A = LU$, overwriting the upper triangular part of A with U and the strictly lower triangular part of A with the strictly lower triangular part of L. The result is then returned in variable `A_out`.
- Uses the factored matrix to solve $Ax = b$.

Use the routines you wrote in the previous subsections (6.3.1-6.3.3).

You can check that it computes the right answer with the following script:

- `test_Solve.m`

This script exercises the function by starting with matrix

```
A = [
     2     0     1     2
    -2    -1     1    -1
     4    -1     5     4
    -4     1    -3    -8
]
```

Next, it solves $Ax = b$ with

```
b = [
     2
     2
    11
    -3
]
```

by calling

```
x = Solve( A, b )
```

Finally, it checks if x indeed solves $Ax = b$ by computing

```
b - A * x
```

which should yield a zero vector of size four.

Answer:

Here is our implementations of the function:

- Solve.m

☛ BACK TO TEXT

6.3.5 Cost

Homework 6.3.3.1 Assume that during the kth iteration U_{BR} is $k \times k$. (Notice we are purposely saying that U_{BR} is $k \times k$ because this algorithm moves in the opposite direction!)

Then answer the following questions:

- U_{22} is a ?×? matrix.
- u_{12}^T is a column/row vector of size ????.
- b_2 is a column vector of size ???.

Now,

- The axpy/dot operation $\beta_1 := \beta_1 - u_{12}^T b_2$ requires ??? flops since the vectors are of size ????.

We need to sum this over all iterations $k = 0, \ldots, (n-1)$ (You may ignore the divisions):

????? flops.

Compute how many floating point operations this equal. Then, approximate the result.

Answer: Then answer the following questions:

- U_{22} is a $k \times k$ matrix.
- u_{12}^T is a column/row vector of size k.
- b_2 is a column vector of size k.

Now,

- The dot operation $\beta_1 := \beta_1 - u_{12}^T b_2$ requires $2k$ flops since the vectors are of size k.

We need to sum this over all iterations $k = 0, \ldots, (n-1)$:

$$\sum_{k=0}^{n-1} 2k \text{ floating point operations.}$$

Compute how many floating point operations this equal. Then, approximate the result.

Let us compute how many floating point this equals:

$$\begin{array}{l} \sum_{k=0}^{n-1} 2k \\ \quad = \quad < \text{Factor out 2} > \\ 2\sum_{k=0}^{n-1} k \\ \quad = \quad < \text{Results from Week 2!} > \\ 2\frac{(n-1)n}{2} \\ \quad = \quad < \text{Algebra} > \\ (n-1)n. \end{array}$$

Now, when n is large $n-1$ equals, approximately, n so that the cost for the forward substitution equals, approximately,

$$n^2 \text{ flops.}$$

☛ BACK TO TEXT

Week7: More Gaussian Elimination and Matrix Inversion (Answers)

7.2 When Gaussian Elimination Breaks Down

7.2.1 When Gaussian Elimination Works

Homework 7.2.1.1 Let $L \in \mathbb{R}^{1\times 1}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer: Always

Since L is 1×1, it is a scalar:

$$(1)(\chi_0) = (\beta_0).$$

From basic algebra we know that then $\chi_0 = \beta_0$ is the unique solution.

☛ BACK TO TEXT

Homework 7.2.1.2 Give the solution of $\begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$.

Answer: The above translates to the system of linear equations

$$\begin{array}{rcrcl} \chi_0 & & & = & 1 \\ 2\chi_0 & + & \chi_1 & = & 2 \end{array}$$

which has the solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2-(2)(1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.3 Give the solution of $\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$.

(Hint: look carefully at the last problem, and you will be able to save yourself some work.)

Answer: A clever way of solving the above is to slice and dice:

$$\underbrace{\left(\begin{array}{rr|r} 1 & 0 & 0 \\ 2 & 1 & 0 \\ \hline -1 & 2 & 1 \end{array}\right) \left(\begin{array}{c} \chi_0 \\ \chi_1 \\ \hline \chi_2 \end{array}\right)}_{\left(\begin{array}{c} \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \\ \hline \begin{pmatrix} -1 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} + \begin{pmatrix} \chi_2 \end{pmatrix} \end{array}\right)} = \underbrace{\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}}_{\left(\begin{array}{c} \begin{pmatrix} 1 \\ 2 \end{pmatrix} \\ \hline 3 \end{array}\right)}$$

Hence, from the last exercise, we conclude that

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

We can then compute χ_2 by substituting in:

$$\begin{pmatrix} -1 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} + \begin{pmatrix} \chi_2 \end{pmatrix} = 3$$

So that

$$\chi_2 = 3 - \begin{pmatrix} -1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 3 - (-1) = 4.$$

Thus, the solution is the vector

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 4 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.4 Let $L \in \mathbb{R}^{2\times 2}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer: Always

Since L is 2×2, the linear system has the form

$$\begin{pmatrix} 1 & 0 \\ \lambda_{1,0} & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}.$$

But that translates to the system of linear equations

$$\begin{array}{rcrcl} \chi_0 & & & = & \beta_0 \\ \lambda_{1,0}\chi_0 & + & \chi_1 & = & \beta_1 \end{array}$$

which has the unique solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 - \lambda_{1,0}\chi_0 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.5 Let $L \in \mathbb{R}^{3\times 3}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer: Always

Notice

$$\underbrace{\left(\begin{array}{cc|c} 1 & 0 & 0 \\ \lambda_{1,0} & 1 & 0 \\ \hline \lambda_{2,0} & \lambda_{2,1} & 1 \end{array}\right) \left(\begin{array}{c} \chi_0 \\ \chi_1 \\ \hline \chi_2 \end{array}\right)}_{\left(\begin{array}{c} \begin{pmatrix} 1 & 0 \\ \lambda_{1,0} & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \\ \hline \begin{pmatrix} \lambda_{2,0} & \lambda_{2,1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} + \begin{pmatrix} \chi_2 \end{pmatrix} \end{array}\right)} = \underbrace{\begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix}}_{\left(\begin{array}{c} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} \\ \hline \beta_2 \end{array}\right)}$$

Hence, from the last exercise, we conclude that the unique solutions for χ_0 and χ_1 are

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 - \lambda_{1,0}\beta_0 \end{pmatrix}.$$

We can then compute χ_2 by substituting in:

$$\begin{pmatrix} \lambda_{2,0} & \lambda_{2,1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} + \begin{pmatrix} \chi_2 \end{pmatrix} = \beta_2$$

So that

$$\chi_2 = \beta_2 - \begin{pmatrix} \lambda_{2,0} & \lambda_{2,1} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 - \lambda_{1,0}\beta_0 \end{pmatrix}$$

Since there is no ambiguity about what χ_2 must equal, the solution is unique:

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 - \lambda_{1,0}\beta_0 \\ \beta_2 - \begin{pmatrix} \lambda_{2,0} & \lambda_{2,1} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 - \lambda_{1,0}\beta_0 \end{pmatrix} \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.6 Let $L \in \mathbb{R}^{n\times n}$ be a unit lower triangular matrix. $Lx = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer:

Always

The last exercises were meant to make you notice that this can be proved with a proof by induction on the size, n, of L.

Base case: $n = 1$. In this case $L = (1)$, $x = (\chi_1)$ and $b = (\beta_1)$. The result follows from the fact that $(1)(\chi_1) = (\beta_1)$ has the unique solution $\chi_1 = \beta_1$.

Inductive step: Inductive Hypothesis (I.H.): Assume that $Lx = b$ has a unique solution for all $L \in \mathbb{R}^{n\times n}$ and right-hand side vectors b.

We now want to show that then $Lx = b$ has a unique solution for all $L \in \mathbb{R}^{(n+1)\times(n+1)}$ and right-hand side vectors b.

Partition

$$L \rightarrow \left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right), \quad x \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \end{array}\right) \quad \text{and} \quad b \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \end{array}\right),$$

where, importantly, $L_{00} \in \mathbb{R}^{n\times n}$. Then $Lx = b$ becomes

$$\underbrace{\left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)\left(\begin{array}{c} x_0 \\ \hline \chi_1 \end{array}\right)}_{\left(\begin{array}{c} L_{00}x_0 \\ \hline l_{10}^T x_0 + \lambda_{11}\chi_1 \end{array}\right)} = \left(\begin{array}{c} b_0 \\ \hline \beta_1 \end{array}\right)$$

or

$$\begin{array}{c} L_{00}x_0 = b_0 \\ \hline l_{10}^T x_0 + \lambda_{11}\chi_1 = \beta_1 \end{array}.$$

By the Inductive Hypothesis, we know that $L_{00}x_0 = b_0$ has a unique solution. But once x_0 is set, $\lambda_{11}\chi_1 = \beta_1 - l_{10}^T x_0$ uniquely determines χ_1.

By the **Principle of Mathematical Induction**, the result holds.

☛ BACK TO TEXT

Homework 7.2.1.7 The proof for the last exercise suggests an alternative algorithm (Variant 2) for solving $Lx = b$ when L is unit lower triangular. Use Figure 7.3 to state this alternative algorithm and then implement it, yielding

- `[ b_out ] = Ltrsv_unb_var2( L, b )`

You can check that they compute the right answers with the script in

- `test_Ltrsv_unb_var2.m`

Answer:

Here is our implementations of the function:

- `Ltrsv_unb_var2.m`

☛ BACK TO TEXT

Homework 7.2.1.8 Let $L \in \mathbb{R}^{n\times n}$ be a unit lower triangular matrix. $Lx = 0$, where 0 is the zero vector of size n, has the unique solution $x = 0$.

Always/Sometimes/Never

Answer: Always

Obviously $x = 0$ is *a* solution. But a previous exercise showed that when L is a unit lower triangular matrix, $Lx = b$ has a unique solution for *all* b. Hence, it has a unique solution for $b = 0$.

☛ BACK TO TEXT

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR2}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

where λ_{11} is 1×1, β_1 has 1 row

$\beta_1 := \beta_1 - l_{10}^T b_0$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Figure 7.1: Alternative algorithm (Variant 2) for solving $Lx = b$, overwriting b with the result vector x for use in Homework 7.2.1.7. Here L is a lower triangular matrix.

Homework 7.2.1.9 Let $U \in \mathbb{R}^{1\times 1}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer: Always

Since U is 1×1, it is a nonzero scalar

$$(\upsilon_{0,0})(\chi_0) = (\beta_0).$$

From basic algebra we know that then $\chi_0 = \beta_0/\upsilon_{0,0}$ is the unique solution.

☛ BACK TO TEXT

Homework 7.2.1.10 Give the solution of $\begin{pmatrix} -1 & 1 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$.

Answer: The above translates to the system of linear equations

$$\begin{array}{rrcl} -1\chi_0 & +\chi_1 & = & 1 \\ & 2\chi_1 & = & 2 \end{array}$$

which has the solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} (1-\chi_1)/(-1) \\ 2/2 \end{pmatrix} = \begin{pmatrix} (1-(1))/(-1) \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.11 Give the solution of $\begin{pmatrix} -2 & 1 & -2 \\ 0 & -1 & 1 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix}$.

Answer: A clever way of solving the above is to slice and dice:

$$\underbrace{\left(\begin{array}{c|cc} -2 & 1 & -2 \\ \hline 0 & -1 & 1 \\ 0 & 0 & 2 \end{array}\right) \left(\begin{array}{c} \chi_0 \\ \hline \chi_1 \\ \chi_2 \end{array}\right)}_{\left(\begin{array}{c} -2\chi_0 + \begin{pmatrix} 1 & -2 \end{pmatrix} \begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} \\ \hline \begin{pmatrix} -1 & 1 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} \end{array}\right)} = \underbrace{\begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix}}_{\left(\begin{array}{c} 0 \\ \hline \begin{pmatrix} 1 \\ 2 \end{pmatrix} \end{array}\right)}$$

Hence, from the last exercise, we conclude that

$$\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

We can then compute χ_0 by substituting in:

$$-2\chi_0 + \begin{pmatrix} 1 & -2 \end{pmatrix} \begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} = 0$$

So that

$$-2\chi_2 = 0 - \begin{pmatrix} 1 & -2 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 0 - (-2) = 2.$$

Thus, the solution is the vector

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.12 Let $U \in \mathbb{R}^{2\times 2}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer: Always

Since U is 2×2, the linear system has the form

$$\begin{pmatrix} \upsilon_{0,0} & \upsilon_{0,1} \\ 0 & \upsilon_{1,1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}.$$

But that translates rto the system of linear equations

$$\begin{array}{rcrcl} \upsilon_{0,0}\chi_0 & + & \upsilon_{0,1}\chi_1 & = & \beta_0 \\ & & \upsilon_{1,1}\chi_1 & = & \beta_1 \end{array}$$

which has the unique solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} (\beta_0 - \upsilon_{0,1}\chi_1)/\upsilon_{0,0} \\ \beta_1/\upsilon_{1,1} \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.13 Let $U \in \mathbb{R}^{3\times 3}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer: Always

Notice

$$\underbrace{\left(\begin{array}{c|cc} \upsilon_{0,0} & \upsilon_{0,1} & \upsilon_{0,2} \\ \hline 0 & \upsilon_{1,1} & \upsilon_{1,2} \\ 0 & 0 & \upsilon_{2,2} \end{array}\right)\left(\begin{array}{c} \chi_0 \\ \hline \chi_1 \\ \chi_2 \end{array}\right)}_{\left(\begin{array}{c} \upsilon_{0,0}\chi_0 + \begin{pmatrix} \upsilon_{0,1} & \upsilon_{0,2} \end{pmatrix}\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} \\ \hline \begin{pmatrix} \upsilon_{1,1} & \upsilon_{1,2} \\ 0 & \upsilon_{2,2} \end{pmatrix}\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} \end{array}\right)} = \underbrace{\left(\begin{array}{c} \beta_0 \\ \hline \beta_1 \\ \beta_2 \end{array}\right)}_{\left(\begin{array}{c} \beta_0 \\ \hline \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} \end{array}\right)}$$

Hence, from the last exercise, we conclude that the unique solutions for χ_0 and χ_1 are

$$\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} (\beta_1 - \upsilon_{1,2}\chi_2)/\upsilon_{1,1} \\ \beta_2/\upsilon_{2,2} \end{pmatrix}.$$

We can then compute χ_0 by substituting in:

$$\upsilon_{0,0}\chi_0 + \begin{pmatrix} \upsilon_{0,1} & \upsilon_{0,2} \end{pmatrix}\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} = \beta_0$$

So that

$$\chi_0 = \left(\beta_0 - \begin{pmatrix} \upsilon_{0,1} & \upsilon_{0,2} \end{pmatrix}\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix}\right)/\upsilon_{0,0}$$

Since there is no ambiguity about what χ_2, χ_1, and χ_0 (computed in that order) must equal, the solution is unique:

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \left(\beta_0 - \begin{pmatrix} \upsilon_{0,1} & \upsilon_{0,2} \end{pmatrix}\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix}\right)/\upsilon_{0,0} \\ (\beta_1 - \upsilon_{1,2}\chi_2)/\upsilon_{1,1} \\ \beta_2/\upsilon_{2,2} \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.1.14 Let $U \in \mathbb{R}^{n\times n}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = b$, where x is the unknown and b is given, has a unique solution.

Always/Sometimes/Never

Answer: Always

Again, the last exercises were meant to make you notice that this can be proved with a proof by induction on the size, n, of U.

Base case: $n = 1$. In this case $U = (\upsilon_{11})$, $x = (\chi_1)$ and $b = (\beta_1)$. The result follows from the fact that $(\upsilon_{11})(\chi_1) = (\beta_1)$ has the unique solution $\chi_1 = \beta_1/\upsilon_{11}$.

Inductive step: Inductive Hypothesis (I.H.): Assume that $Ux = b$ has a unique solution for all upper triangular $U \in \mathbb{R}^{n\times n}$ that do not have zeroes on their diagonal, and right-hand side vectors b.

We now want to show that then $Ux = b$ has a unique solution for all upper triangular $U \in \mathbb{R}^{(n+1)\times(n+1)}$ that do not have zeroes on their diagonal, and right-hand side vectors b.

Partition

$$U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right), \quad x \rightarrow \left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right) \quad \text{and} \quad b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right),$$

where, importantly, $U_{22} \in \mathbb{R}^{n\times n}$. Then $Ux = b$ becomes

$$\underbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)\left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right)}_{\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 \\ \hline U_{22}x_2 \end{array}\right)} = \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

or

$$\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 = \beta_1 \\ \hline U_{22}x_2 = b_2 \end{array}.$$

By the Inductive Hypothesis, we know that $U_{22}x_2 = b_2$ has a unique solution. But once x_2 is set, $\chi_1 = (\beta_1 - u_{12}^T x_2)/\upsilon_{11}$ uniquely determines χ_1.

By the **Principle of Mathematical Induction**, the result holds.

☛ BACK TO TEXT

Homework 7.2.1.15 Let $U \in \mathbb{R}^{n\times n}$ be an upper triangular matrix with no zeroes on its diagonal. $Ux = 0$, where 0 is the zero vector of size n, has the unique solution $x = 0$.

Always/Sometimes/Never

Answer: Always

Obviously $x = 0$ is *a* solution. But a previous exercise showed that when U is an upper triangular matrix with no zeroes on its diagonal, $Ux = b$ has a unique solution for *all* b. Hence, it has a unique solution for $b = 0$.

☛ BACK TO TEXT

Homework 7.2.1.16 Let $A \in \mathbb{R}^{n\times n}$. *If* Gaussian elimination completes and the resulting upper triangular system has no zero coefficients on the diagonal (U has no zeroes on its diagonal), *then* there is a unique solution x to $Ax = b$ for all $b \in \mathbb{R}$.

Always/Sometimes/Never

Answer: Always

We already argued that under the stated conditions, the LU factorization algorithm computes $A \rightarrow LU$ as do the algorithms for solving $Lz = b$ and $Ux = z$. Hence, **a** solution is computed. Now we address the question of whether this is a unique solution.

To show uniqueness, we assume there there are two solutions, r and s, and then show that $r = s$.
Assume that $Ar = b$ and $As = b$. Then

$$A \underbrace{(r-s)}_{w} = Ar - As = b - b = 0.$$

Let's let $w = r - s$ so that we know that $Aw = 0$. Now, IF we can show that the assumptions imply that $w = 0$, then we know that $r = s$.

Now, $Aw = 0$ means $(LU)w = 0$ which them means that $L(Uw) = 0$. But, since L is unit lower triangular, by one of the previous exercises we know that $Lz = 0$ has the unique solution $z = 0$. Hence $Uw = 0$. But, since U is an upper triangular matrix with no zeroes on its diagonal, we know from a previous exercise that $w = 0$. Hence $r = s$ and the solution to $Ax = b$ is unique.

☛ BACK TO TEXT

7.2.2 The Problem

Homework 7.2.2.1 Solve the following linear system, via the steps in Gaussian elimination that you have learned so far.

$$\begin{aligned} 2\chi_0 + 4\chi_1 + (-2)\chi_2 &= -10 \\ 4\chi_0 + 8\chi_1 + 6\chi_2 &= 20 \\ 6\chi_0 + (-4)\chi_1 + 2\chi_2 &= 18 \end{aligned}$$

Mark all that are correct:

(a) The process breaks down.

(b) There is no solution.

(c) $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 4 \end{pmatrix}$

Answer: (a) and (c)

Solving this linear system via Gaussian elimination relies on the fact that its solution does not change if equations are reordered.

Now,

- By subtracting $(4/2) = 2$ times the first row from the second row and $(6/2) = 3$ times the first row from the third row, we get

$$\begin{aligned} 2\chi_0 + 4\chi_1 + (-2)\chi_2 &= -10 \\ 0\chi_0 + 0\chi_1 + 10\chi_2 &= 40 \\ 0\chi_0 + (-16)\chi_1 + 8\chi_2 &= 48 \end{aligned}$$

- Now we've got a problem. The algorithm we discussed so far would want to subtract $((-16)/0)$ times the second row from the third row, which causes a divide-by-zero error. Instead, we have to use the fact that reordering the equations does not change the answer, swapping the second row with the third:

$$\begin{aligned} 2\chi_0 + 4\chi_1 + (-2)\chi_2 &= -10 \\ 0\chi_0 + (-16)\chi_1 + 8\chi_2 &= 48 \\ 0\chi_0 + 0\chi_1 + 10\chi_2 &= 40 \end{aligned}$$

at which point we are done transforming our system into an upper triangular system, and the backward substition can commence to solve the problem.

☛ BACK TO TEXT

Homework 7.2.2.2 Perform Gaussian elimination with

$$\begin{array}{rcl} 0\chi_0 + & 4\chi_1 + (-2)\chi_2 = & -10 \\ 4\chi_0 + & 8\chi_1 + \quad 6\chi_2 = & 20 \\ 6\chi_0 + (-4)\chi_1 + & 2\chi_2 = & 18 \end{array}$$

Answer:

- We start by trying to subtract $(4/0)$ times the first row from the second row and $(6/0)$ times the first row from the third row. This causes a "divide by zero" error.
- Instead, we begin by swaping the first row with any of the other two rows:

$$\begin{array}{rcl} 4\chi_0 + & 8\chi_1 + \quad 6\chi_2 = & 20 \\ 0\chi_0 + & 4\chi_1 + (-2)\chi_2 = & -10 \\ 6\chi_0 + (-4)\chi_1 + & 2\chi_2 = & 18 \end{array}$$

- By subtracting $(0/4) = 0$ times the first row from the second row and $(6/4) = 3/2$ times the first row from the third row, we get

$$\begin{array}{rcl} 4\chi_0 + & 8\chi_1 + \quad 6\chi_2 = & 20 \\ 0\chi_0 + & 4\chi_1 + (-2)\chi_2 = & -10 \\ 0\chi_0 + (-16)\chi_1 + & (-7)\chi_2 = & -12 \end{array}$$

- Next, we subtract $(-16)/4 = -4$ times the second row from the third to obtain

$$\begin{array}{rcl} 4\chi_0 + 8\chi_1 + & 6\chi_2 = & 20 \\ 0\chi_0 + 4\chi_1 + & (-2)\chi_2 = & -10 \\ 0\chi_0 + 0\chi_1 + & (-15)\chi_2 = & -52 \end{array}$$

  at which point we are done transforming our system into an upper triangular system, and the backward substition can commence to solve the problem.

☛ BACK TO TEXT

7.2.3 Permutations

Homework 7.2.3.1 Compute

$$\underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}}_{P} \underbrace{\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}}_{A} =$$

Answer:

$$\left(\begin{array}{c|c|c} 0 & 1 & 0 \\ \hline 0 & 0 & 1 \\ \hline 1 & 0 & 0 \end{array}\right)\left(\begin{array}{ccc} -2 & 1 & 2 \\ \hline 3 & 2 & 1 \\ \hline -1 & 0 & -3 \end{array}\right) = \left(\begin{array}{c} 0\times\left(\begin{array}{ccc} -2 & 1 & 2 \end{array}\right)+1\times\left(\begin{array}{ccc} 3 & 2 & 1 \end{array}\right)+0\times\left(\begin{array}{ccc} -1 & 0 & -3 \end{array}\right) \\ \hline 0\times\left(\begin{array}{ccc} -2 & 1 & 2 \end{array}\right)+0\times\left(\begin{array}{ccc} 3 & 2 & 1 \end{array}\right)+1\times\left(\begin{array}{ccc} -1 & 0 & -3 \end{array}\right) \\ \hline 1\times\left(\begin{array}{ccc} -2 & 1 & 2 \end{array}\right)+0\times\left(\begin{array}{ccc} 3 & 2 & 1 \end{array}\right)+0\times\left(\begin{array}{ccc} -1 & 0 & -3 \end{array}\right) \end{array}\right)$$

$$= \left(\begin{array}{ccc} 3 & 2 & 1 \\ \hline -1 & 0 & -3 \\ \hline -2 & 1 & 2 \end{array}\right).$$

Notice that multiplying the matrix by P from the left permuted the order of the rows in the matrix. Here is another way of looking at the same thing:

$$\left(\begin{array}{c} e_1^T \\ \hline e_2^T \\ \hline e_0^T \end{array}\right)\left(\begin{array}{ccc} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{array}\right) = \left(\begin{array}{c} e_1^T\left(\begin{array}{ccc} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{array}\right) \\ \hline e_2^T\left(\begin{array}{ccc} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{array}\right) \\ \hline e_0^T\left(\begin{array}{ccc} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{array}\right) \end{array}\right) = \left(\begin{array}{ccc} 3 & 2 & 1 \\ \hline -1 & 0 & -3 \\ \hline -2 & 1 & 2 \end{array}\right).$$

Here we use the fact that $e_i^T A$ equals the ith row of A.

☛ BACK TO TEXT

Homework 7.2.3.2 For each of the following, give the permutation matrix $P(p)$:

- If $p = \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \end{pmatrix}$ then $P(p) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$,
- If $p = \begin{pmatrix} 3 \\ 2 \\ 1 \\ 0 \end{pmatrix}$ then $P(p) = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$
- If $p = \begin{pmatrix} 1 \\ 0 \\ 2 \\ 3 \end{pmatrix}$ then $P(p) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
- If $p = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 0 \end{pmatrix}$ then $P(p) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$

☛ BACK TO TEXT

Homework 7.2.3.3 Let $p = (2,0,1)^T$. Compute

- $P(p)\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} =$
- $P(p)\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} =$

Answer:

$$P(p)\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} = \begin{pmatrix} -1 \\ -2 \\ 3 \end{pmatrix} \quad \text{and} \quad P(p)\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} = \begin{pmatrix} -1 & 0 & -3 \\ -2 & 1 & 2 \\ 3 & 2 & 1 \end{pmatrix}.$$

Hint: it is not necessary to write out $P(p)$: the vector p indicates the order in which the elements and rows need to appear.

☛ BACK TO TEXT

Homework 7.2.3.4 Let $p = (2,0,1)^T$ and $P = P(p)$. Compute

$$\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} P^T =$$

Answer:

$$\left(\begin{array}{r|r|r} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{array}\right)\left(\begin{array}{c|c|c} 0 & 0 & 1 \\ \hline 1 & 0 & 0 \\ \hline 0 & 1 & 0 \end{array}\right)^T = \left(\begin{array}{r|r|r} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{array}\right)\left(\begin{array}{c|c|c} 0 & 1 & 0 \\ \hline 0 & 0 & 1 \\ \hline 1 & 0 & 0 \end{array}\right)$$

$$= \left((0)\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} + (0)\begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix} + (1)\begin{pmatrix} 2 \\ 1 \\ -3 \end{pmatrix} \middle| (1)\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} + (0)\begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix} + (0)\begin{pmatrix} 2 \\ 1 \\ -3 \end{pmatrix} \middle| (0)\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} + (1)\begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix} + (0)\begin{pmatrix} 2 \\ 1 \\ -3 \end{pmatrix} \right)$$

$$= \left(\begin{array}{r|r|r} 2 & -2 & 1 \\ 1 & 3 & 2 \\ -3 & -1 & 0 \end{array}\right)$$

Alternatively:

$$\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}\left(\begin{array}{c} e_2^T \\ \hline e_0^T \\ \hline e_1 \end{array}\right)^T = \begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}\left(\begin{array}{c|c|c} e_2 & e_0 & e_1 \end{array}\right)$$

$$= \left(\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} e_2 \middle| \begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} e_0 \middle| \begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} e_1 \right)$$

$$= \left(\begin{pmatrix} 2 \\ 1 \\ -3 \end{pmatrix} \middle| \begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} \middle| \begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix} \right) = \left(\begin{array}{r|r|r} 2 & -2 & 1 \\ 1 & 3 & 2 \\ -3 & -1 & 0 \end{array}\right)$$

Hint: it is not necessary to write out $P(p)$: the vector p indicates the order in which the columns need to appear. In this case, you can go directly to the answer

$$\left(\begin{array}{r|r|r} 2 & -2 & 1 \\ 1 & 3 & 2 \\ -3 & -1 & 0 \end{array}\right).$$

☛ BACK TO TEXT

Homework 7.2.3.5 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation vector. Consider

$$x = \left(\begin{array}{c} \chi_0 \\ \hline \chi_1 \\ \hline \vdots \\ \hline \chi_{n-1} \end{array}\right).$$

Applying permuation matrix $P = P(p)$ to x yields

$$Px = \left(\begin{array}{c} \chi_{k_0} \\ \hline \chi_{k_1} \\ \hline \vdots \\ \hline \chi_{k_{n-1}} \end{array}\right).$$

Always/Sometimes/Never

Answer: Always

$$Px = P(p)x = \left(\begin{array}{c} e_{k_0}^T \\ \hline e_{k_1}^T \\ \hline \vdots \\ \hline e_{k_{n-1}}^T \end{array}\right) x = \left(\begin{array}{c} e_{k_0}^T x \\ \hline e_{k_1}^T x \\ \hline \vdots \\ \hline e_{k_{n-1}}^T x \end{array}\right) = \left(\begin{array}{c} \chi_{k_0} \\ \hline \chi_{k_1} \\ \hline \vdots \\ \hline \chi_{k_{n-1}} \end{array}\right).$$

(Recall that $e_i^T x = \chi_i$.)

☛ BACK TO TEXT

Homework 7.2.3.6 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation. Consider

$$A = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{n-1}^T \end{array}\right).$$

Applying $P = P(p)$ to A yields

$$PA = \left(\begin{array}{c} \widetilde{a}_{k_0}^T \\ \hline \widetilde{a}_{k_1}^T \\ \hline \vdots \\ \hline \widetilde{a}_{k_{n-1}}^T \end{array}\right).$$

Always/Sometimes/Never

Answer: Always

$$PA = P(p)A = \left(\begin{array}{c} e_{k_0}^T \\ \hline e_{k_1}^T \\ \hline \vdots \\ \hline e_{k_{n-1}}^T \end{array}\right) A = \left(\begin{array}{c} e_{k_0}^T A \\ \hline e_{k_1}^T A \\ \hline \vdots \\ \hline e_{k_{n-1}}^T A \end{array}\right) = \left(\begin{array}{c} \widetilde{a}_{k_0}^T \\ \hline \widetilde{a}_{k_1}^T \\ \hline \vdots \\ \hline \widetilde{a}_{k_{n-1}}^T \end{array}\right).$$

(Recall that $e_i^T A$ equals the ith row of A.)

☛ BACK TO TEXT

Homework 7.2.3.7 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation, $P = P(p)$, and $A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$.

$AP^T = \left(\begin{array}{c|c|c|c} a_{k_0} & a_{k_1} & \cdots & a_{k_{n-1}} \end{array} \right)$.

Aways/Sometimes/Never

Answer: Always

Recall that unit basis vectors have the property that $Ae_k = a_k$.

$$AP^T = A \left(\begin{array}{c} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{array}\right)^T = A \left(\begin{array}{c|c|c|c} e_{k_0} & e_{k_1} & \cdots & e_{k_{n-1}} \end{array} \right)$$

$$= \left(Ae_{k_0} \,\middle|\, Ae_{k_1} \,\middle|\, \cdots \,\middle|\, Ae_{k_{n-1}} \right) = \left(a_{k_0} \,\middle|\, a_{k_1} \,\middle|\, \cdots \,\middle|\, a_{k_{n-1}} \right).$$

☛ BACK TO TEXT

Homework 7.2.3.8 If P is a permutation matrix, then so is P^T.

True/False

Answer: This follows from the observation that if P can be viewed either as a rearrangement of the rows of the identity or as a (usually different) rearrangement of the columns of the identity.

☛ BACK TO TEXT

Homework 7.2.3.9 Compute

$$\widetilde{P}(1)\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} = \begin{pmatrix} 3 \\ -2 \\ -1 \end{pmatrix} \quad \text{and} \quad \widetilde{P}(1)\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix} = \begin{pmatrix} 3 & 2 & 1 \\ -2 & 1 & 2 \\ -1 & 0 & -3 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.3.10 Compute

$$\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}\widetilde{P}(1) = \begin{pmatrix} 1 & -2 & 2 \\ 2 & 3 & 1 \\ 0 & -1 & -3 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.2.3.11 When $\widetilde{P}(\pi)$ (of appropriate size) multiplies a matrix from the left, it swaps row 0 and row π, leaving all other rows unchanged.

Always/Sometimes/Never

Answer: Always

$$\widetilde{P}(\pi)A = \begin{pmatrix} \boxed{e_\pi^T} \\ e_1^T \\ \vdots \\ e_{\pi-1}^T \\ \boxed{e_0^T} \\ e_{\pi+1}^T \\ \vdots \\ e_{n-1}^T \end{pmatrix} A = \begin{pmatrix} \boxed{e_\pi^T A} \\ e_1^T A \\ \vdots \\ e_{\pi-1}^T A \\ \boxed{e_0^T A} \\ e_{\pi+1}^T A \\ \vdots \\ e_{n-1}^T A \end{pmatrix} = \begin{pmatrix} \boxed{\hat{a}_\pi^T} \\ \hat{a}_1^T \\ \vdots \\ \hat{a}_{\pi-1}^T \\ \boxed{\hat{a}_0^T} \\ \hat{a}_{\pi+1}^T \\ \vdots \\ \hat{a}_{n-1}^T \end{pmatrix}.$$

(Here $\hat{a}_i^T$ equals the ith row of A.)

☛ BACK TO TEXT

Homework 7.2.3.12 When $\widetilde{P}(\pi)$ (of appropriate size) multiplies a matrix from the right, it swaps column 0 and column π, leaving all other columns unchanged.

Always/Sometimes/Never

Answer: Always

$$
\begin{array}{rcl}
A\widetilde{P}(\pi)^T & = & A \left(\begin{array}{c} \boxed{e_\pi^T} \\ e_1^T \\ \vdots \\ e_{\pi-1}^T \\ \boxed{e_0^T} \\ e_{\pi+1}^T \\ \vdots \\ e_{n-1}^T \end{array} \right)^T = A \left(\begin{array}{c|c|c|c|c|c|c|c} \boxed{e_\pi} & e_1 & \cdots & e_{\pi-1} & \boxed{e_0} & e_{\pi+1} & \cdots & e_{n-1} \end{array} \right) \\
& = & \left(\begin{array}{c|c|c|c|c|c|c|c} \boxed{Ae_\pi} & Ae_1 & \cdots & Ae_{\pi-1} & \boxed{Ae_0} & Ae_{\pi+1} & \cdots & Ae_{n-1} \end{array} \right) \\
& = & \left(\begin{array}{c|c|c|c|c|c|c|c} \boxed{a_\pi} & a_1 & \cdots & a_{\pi-1} & \boxed{a_0} & a_{\pi+1} & \cdots & a_{n-1} \end{array} \right).
\end{array}
$$

(Here a_j equals the jth column of A.)

☛ BACK TO TEXT

7.2.4 Gaussian Elimination with Row Swapping (LU Factorization with Partial Pivoting)

Homework 7.2.4.1 Compute

- $\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \end{array} \right) \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array} \right) \left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array} \right) = \left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 6 & -4 & 2 \\ 4 & 8 & 6 \end{array} \right)$

- $\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 3 & 1 & 0 \\ 2 & 0 & 1 \end{array} \right) \left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 0 & -16 & 8 \\ 0 & 0 & 10 \end{array} \right) = \left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 6 & -4 & 2 \\ 4 & 8 & 6 \end{array} \right)$

- What do you notice?

  Answer:

$$
\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \end{array} \right) \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array} \right) \left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array} \right) = \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 3 & 1 & 0 \\ 2 & 0 & 1 \end{array} \right) \left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 0 & -16 & 8 \\ 0 & 0 & 10 \end{array} \right),
$$

  which is meant to illustrate that LU factorization with row swapping (partial pivoting) produces the LU factorization of the original matrix, but with rearranged rows.

☛ BACK TO TEXT

Homework 7.2.4.2
(You may want to print the blank worksheet at the end of this week so you can follow along.)
Perform Gaussian elimination with row swapping (row pivoting):

i	L_i	$\tilde{P}$	A	p
0		$\begin{array}{\|c\|cc} \hline 0 & 1 & 0 \\ \hline 1 & 0 & 0 \\ 0 & 0 & 1 \end{array}$	$\begin{array}{\|c\|cc} \hline 0 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}$	$\begin{array}{c} \hline 1 \\ \hline \cdot \\ \cdot \end{array}$
	$\begin{array}{\|c\|cc} \hline 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ -\frac{3}{2} & 0 & 1 \end{array}$		$\begin{array}{\|c\|cc} \hline 4 & 8 & 6 \\ \hline 0 & 4 & -2 \\ 6 & -4 & 2 \end{array}$	$\begin{array}{c} \hline 1 \\ \hline \cdot \\ \cdot \end{array}$
1		$\begin{array}{c\|c\|c} & & \\ \hline & 1 & 0 \\ \hline & 0 & 1 \end{array}$	$\begin{array}{c\|c\|c} 4 & 8 & 6 \\ \hline 0 & 4 & -2 \\ \hline \frac{3}{2} & -16 & -7 \end{array}$	$\begin{array}{c} 1 \\ \hline 0 \\ \hline \cdot \end{array}$
	$\begin{array}{c\|c\|c} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 4 & 1 \end{array}$		$\begin{array}{c\|c\|c} 4 & 8 & 6 \\ \hline 0 & 4 & -2 \\ \hline \frac{3}{2} & -16 & -7 \end{array}$	$\begin{array}{c} 1 \\ \hline 0 \\ \hline \cdot \end{array}$
2			$\begin{array}{cc\|c\|} 4 & 8 & 6 \\ 0 & 4 & -2 \\ \hline \frac{3}{2} & -4 & -15 \\ \hline \end{array}$	$\begin{array}{c} 1 \\ 0 \\ \hline 0 \\ \hline \end{array}$

☛ BACK TO TEXT

7.3 The Inverse Matrix

7.3.2 Back to Linear Transformations

Homework 7.3.2.1 Let $L : \mathbb{R}^n \to \mathbb{R}^n$ be a linear transformation that is a bijection and let L^{-1} denote its inverse.

L^{-1} is a linear transformation.

Always/Sometimes/Never

Answer: Always
Let $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.

- $L^{-1}(\alpha x) = \alpha L^{-1}(x)$. Let $u = L^{-1}(x)$. Then $x = L(u)$. Now,
$$L^{-1}(\alpha x) = L^{-1}(\alpha L(u)) = L^{-1}(L(\alpha u)) = \alpha u = \alpha L^{-1}(x).$$
- $L^{-1}(x+y) = L^{-1}(x) + L^{-1}(y)$. Let $u = L^{-1}(x)$ and $v = L^{-1}(y)$ so that $L(u) = x$ and $L(v) = y$. Now,
$$L^{-1}(x+y) = L^{-1}(L(u)+L(v)) = L^{-1}(L(u+v)) = u+v = L^{-1}(x)+L^{-1}(y).$$

Hence L^{-1} is a linear transformation.

☛ BACK TO TEXT

Homework 7.3.2.2 Let A, B, and C all be $n \times n$ matrices. If $AB = I$ and $CA = I$ then $B = C$.

True/False

Answer: Multiplying both sides of $AB = I$ on the left by C implies that $CAB = C$. Multiplying both sides of $CA = I$ on the right by B implies that $CAB = B$. Hence $C = CAB = B$.

☛ BACK TO TEXT

7.3.3 Simple Examples

Homework 7.3.3.1 If I is the identity matrix, then $I^{-1} = I$. True/False

Answer: True

What is the matrix that undoes Ix? Well, $Ix = x$, so to undo it, you do nothing. But the matrix that does nothing is the identity matrix.

Check: $II = II = I$. Hence I equals the inverse of I.

☛ BACK TO TEXT

Homework 7.3.3.2 Find
$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix}^{-1} =$$

Answer: Question: What effect does applying $\begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix}$ to a vector have? Answer: The first component is multiplied by -1, the second by 2 and the third by $1/3$. To undo this, one needs to take the result first resulting component and multiply it by -1, the second resulting component by $1/2$ and the third resulting component by 3. This motivates that
$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix}^{-1} = \begin{pmatrix} -1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & 3 \end{pmatrix}.$$

Now we check:
$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & 3 \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.3.3.3 Assume $\delta_j \neq 0$ for $0 \leq j < n$.

$$\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{\delta_0} & 0 & \cdots & 0 \\ 0 & \frac{1}{\delta_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\delta_{n-1}} \end{pmatrix}.$$

Always/Sometimes/Never

Answer: Always by the insights of the previous exercise.

Check:

$$\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix} \begin{pmatrix} \frac{1}{\delta_0} & 0 & \cdots & 0 \\ 0 & \frac{1}{\delta_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\delta_{n-1}} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 7.3.3.4 Find

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix}^{-1} =$$

Important: read the answer!

Answer: What effect does applying $\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix}$ to a vector have? Answer:

- It takes first component and adds it to the second component.
- It takes -2 times the first component and adds it to the third component.

How do you undo this?

- You take first component and *subtract* it from the second component.
- You take 2 times first component and add it from the second component.

The Gauss transform that does this is given by

$$\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 0 & 1 \end{pmatrix}.$$

Notice that the elements below the diagonal are negated: 1 becomes -1 and -2 becomes 2.

Check:

$$\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 7.3.3.5

$$\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right)^{-1} = \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right).$$

True/False

Answer: True by the argument for the previous exercise.

Check:

$$\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) = \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & I \end{array}\right).$$

☛ BACK TO TEXT

Homework 7.3.3.6 Assume the matrices below are partitioned conformally so that the multiplications and comparison are legal.

$$\left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & 0 & I \end{array}\right)\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) = \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

Answer: Always Just multiply it out.

☛ BACK TO TEXT

Homework 7.3.3.7 Find

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} =$$

Answer: The thing to ask is “What does this matrix do when applied to a vector?” It swaps the top two elements. How do you “undo” that? You swap the top two elements. So, the inverse is

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Check:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 7.3.3.8 Find

$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{-1} =$$

Answer: The thing to ask is "What does this matrix do when applied to a vector?" This permutation rotates the rows down one row. How do you "undo" that? You need to rotate the rows up one row. So, the inverse is

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$

Check:

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 7.3.3.9 Let P be a permutation matrix. Then $P^{-1} = P$.

Always/Sometimes/Never

Answer: Sometimes The last two exercises provide and example and a counter example.

☛ BACK TO TEXT

Homework 7.3.3.10 Let P be a permutation matrix. Then $P^{-1} = P^T$.

Always/Sometimes/Never

Answer: Always (Note: it took me a good hour to think of the below explanation. The simple thing would have been to simply verify the result.)

What "action" does applying the matrix have when applied to a vector? It permutes the components. Now, it would seem that to "un"permute the vector, one probably has to apply another permutation. So we are looking for a permutation.

Let's let $p = \begin{pmatrix} k_0 & k_1 & \cdots & k_{n-1} \end{pmatrix}^T$ be the permutation vector such that $P = P(p)$. Then

$$P = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \quad \text{and} \quad Px = \begin{pmatrix} \chi_{k_0} \\ \hline \chi_{k_1} \\ \hline \vdots \\ \hline \chi_{k_{n-1}} \end{pmatrix}.$$

Let's say that B is the inverse of P. Then we want to choose B so that

$$\underbrace{\underbrace{\begin{pmatrix} \chi_0 \\ \hline \chi_1 \\ \hline \vdots \\ \hline \chi_{n-1} \end{pmatrix}}_{\chi_0 e_0 + \chi_1 e_1 + \cdots + \chi_{n-1} e_{n-1}}}_{\chi_{k_0} e_{k_0} + \chi_{k_1} e_{k-1} + \cdots + \chi_{k_{n-1}} e_{k_{n-1}}} = BPx = \underbrace{\left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right)\begin{pmatrix} \chi_{k_0} \\ \hline \chi_{k_1} \\ \hline \vdots \\ \hline \chi_{k_{n-1}} \end{pmatrix}}_{\chi_{k_0} b_0 + \chi_{k_1} b_1 + \cdots + \chi_{k_{n-1}} b_{n-1}}.$$

Hmmm, but if you pick $b_j = e_{k_j}$, the left- and the right-hand sides are equal. Thus,

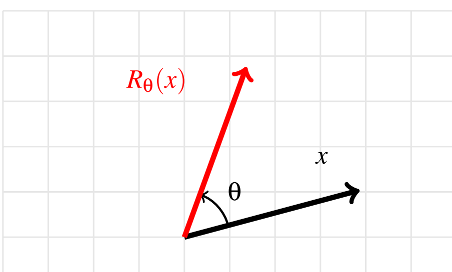
$$P^{-1} = B = \left(\begin{array}{c|c|c|c} e_{k_0} & e_{k_1} & \cdots & e_{k_{n-1}} \end{array} \right) = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}^T = P^T.$$

It is easy to check this by remembering that $P^{-1}P = PP^{-1} = I$ has to be true:

$$\begin{aligned} PP^T &= \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}^T = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \left(\begin{array}{c|c|c|c} e_{k_0} & e_{k_1} & \cdots & e_{k_{n-1}} \end{array} \right) \\ &= \begin{pmatrix} e_{k_0}^T e_{k_0} & e_{k_0}^T e_{k_1} & \cdots & e_{k_0}^T e_{k_{n-1}} \\ e_{k_1}^T e_{k_0} & e_{k_1}^T e_{k_1} & \cdots & e_{k_1}^T e_{k_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ e_{k_{n-1}}^T e_{k_0} & e_{k_{n-1}}^T e_{k_1} & \cdots & e_{k_{n-1}}^T e_{k_{n-1}} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} = I. \end{aligned}$$

☛ BACK TO TEXT

Homework 7.3.3.11 Recall from Week 2 how $R_\theta(x)$ rotates a vector x through angle θ:



R_θ is represented by the matrix

$$R = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}.$$

What transformation will "undo" this rotation through angle θ? (Mark all correct answers)

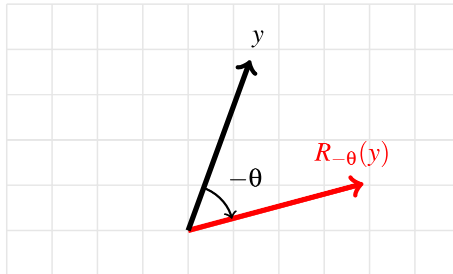
(a) $R_{-\theta}(x)$

(b) Ax, where $A = \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{pmatrix}$

(c) Ax, where $A = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}$

Answer: (a), (b), (c)

Well, if $y = R_\theta(x)$, then y must be rotated through angle $-\theta$ to transform it back into x:



So, the inverse function for $R_\theta(x)$ is $R_{-\theta}(x)$. The matrix that represents R_θ is given by

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}.$$

Thus, the matrix that represents $R_{-\theta}$ is given by

$$\begin{pmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{pmatrix}.$$

But we remember from trigonometry that $\cos(-\theta) = \cos(\theta)$ and $\sin(-\theta) = -\sin(\theta)$. This would mean that

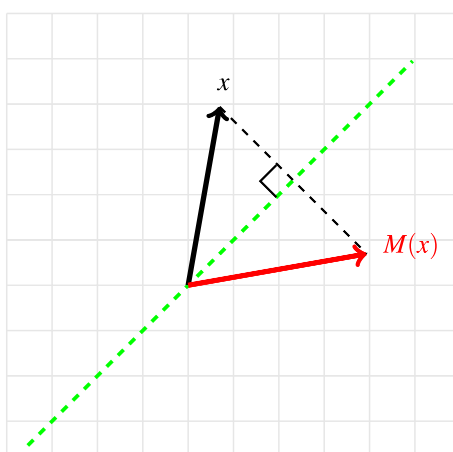
$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}^{-1} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}.$$

Let us check!

$$\begin{aligned} &\begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \\ &\quad = \begin{pmatrix} \cos(\theta)\cos(\theta) + \sin(\theta)\sin(\theta) & -\cos(\theta)\sin(\theta) + \sin(\theta)\cos(\theta) \\ -\sin(\theta)\cos(\theta) + \cos(\theta)\sin(\theta) & \sin(\theta)\sin(\theta) + \cos(\theta)\cos(\theta) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \end{aligned}$$

☛ BACK TO TEXT

Homework 7.3.3.12 Consider a reflection with respect to the 45 degree line:



If A represents the linear transformation M, then

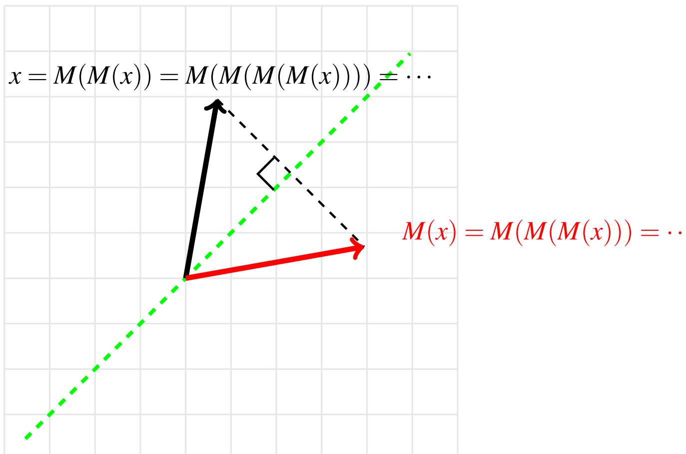
(a) $A^{-1} = -A$

(b) $A^{-1} = A$

(c) $A^{-1} = I$

(d) All of the above.

Answer: (b) $A^{-1} = A$: Reflect anything twice, and you should get the original answer back. (The angle doesn't need to be 45 degrees. This is true for any reflection.)



☛ BACK TO TEXT

7.3.4 More Advanced (but Still Simple) Examples

Homework 7.3.4.1 Compute $\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix}^{-1} = \begin{pmatrix} -\frac{1}{2} & 0 \\ 1 & \frac{1}{2} \end{pmatrix}$.

Answer: Here is how you can find the answer First, solve

$$\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

If you do forward substitution, you see that the solution is $\begin{pmatrix} -\frac{1}{2} \\ 1 \end{pmatrix}$, which becomes the first column of A^{-1}.

Next, solve

$$\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

If you do forward substitution, you see that the solution is $\begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix}$, which becomes the second column of A^{-1}.

Check:

$$\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix} \begin{pmatrix} -\frac{1}{2} & 0 \\ 1 & \frac{1}{2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 7.3.4.2 Find

$$\begin{pmatrix} 1 & -2 \\ 0 & 2 \end{pmatrix}^{-1} =$$

Answer:

$$\begin{pmatrix} 1 & 1 \\ 0 & \frac{1}{2} \end{pmatrix}.$$

Here is how you can find this matrix: First, solve

$$\begin{pmatrix} 1 & -2 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

If you do backward substitution, you see that the solution is $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$, which becomes the first column of A^{-1}.

Next, solve

$$\begin{pmatrix} 1 & -2 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

If you do backward substitution, you see that the solution is $\begin{pmatrix} 1 \\ \frac{1}{2} \end{pmatrix}$, which becomes the second column of A^{-1}.

Check:

$$\begin{pmatrix} 1 & -2 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & \frac{1}{2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 7.3.4.3 Let $\alpha_{0,0} \neq 0$ and $\alpha_{1,1} \neq 0$. Then

$$\begin{pmatrix} \alpha_{0,0} & 0 \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{\alpha_{0,0}} & 0 \\ -\frac{\alpha_{1,0}}{\alpha_{0,0}\alpha_{1,1}} & \frac{1}{\alpha_{1,1}} \end{pmatrix}$$

True/False

Answer: True

Here is how you can find the matrix: First, solve

$$\begin{pmatrix} \alpha_{0,0} & 0 \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

If you solve the lower triangular system, you see that the solution is $\begin{pmatrix} \frac{1}{\alpha_{0,0}} \\ -\frac{\alpha_{1,0}}{\alpha_{0,0}\alpha_{1,1}} \end{pmatrix}$, which becomes the first column of A^{-1}.

Next, solve

$$\begin{pmatrix} \alpha_{0,0} & 0 \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

If you solve this system, you find the solution $\begin{pmatrix} 0 \\ \frac{1}{\alpha_{1,1}} \end{pmatrix}$, which becomes the second column of A^{-1}.

Check:

$$\begin{pmatrix} \alpha_{0,0} & 0 \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \begin{pmatrix} \frac{1}{\alpha_{0,0}} & 0 \\ -\frac{\alpha_{1,0}}{\alpha_{0,0}\alpha_{1,1}} & \frac{1}{\alpha_{1,1}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 7.3.4.4 Partition lower triangular matrix L as

$$L = \left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)$$

Assume that L has no zeroes on its diagonal. Then

$$L^{-1} = \left(\begin{array}{c|c} L_{00}^{-1} & 0 \\ \hline -\frac{1}{\lambda_{11}} l_{10}^T L_{00}^{-1} & \frac{1}{\lambda_{11}} \end{array}\right)$$

True/False

Answer: True Stictly speaking, one needs to show that L_{00} has an inverse... This would require a proof by induction. We'll skip that part. Instead, we'll just multiply out:

$$\begin{aligned} \left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right) \left(\begin{array}{c|c} L_{00}^{-1} & 0 \\ \hline -\frac{l_{10}^T L_{00}^{-1}}{\lambda_{11}} & \frac{1}{\lambda_{11}} \end{array}\right) &= \left(\begin{array}{c|c} L_{00} L_{00}^{-1} & L_{00} 0 + 0 \frac{1}{\lambda_{11}} \\ \hline l_{10}^T L_{00}^{-1} - \lambda_{11} \frac{l_{10}^T L_{00}^{-1}}{\lambda_{11}} & l_{10}^T \times 0 + \lambda_{11} \frac{1}{\lambda_{11}} \end{array}\right) \\ &= \left(\begin{array}{c|c} I & 0 \\ \hline 0 & 1 \end{array}\right). \end{aligned}$$

☛ BACK TO TEXT

Homework 7.3.4.5 The inverse of a lower triangular matrix with no zeroes on its diagonal is a lower triangular matrix.

True/False

Answer: True

Proof by induction on n, the size of the square matrix.
Let L be the lower triangular matrix.

Base case: $n = 1$. Then $L = (\lambda_{11})$, with $\lambda_{11} \neq 0$. Clearly, $L^{-1} = (1/\lambda_{11})$.

Inductive step: Inductive Hypothesis: Assume that the inverse of any $n \times n$ lower triangular matrix with no zeroes on its diagonal is a lower triangular matrix.

We need to show that the inverse of any $(n+1) \times (n+1)$ lower triangular matrix, L, with no zeroes on its diagonal is a lower triangular matrix.

Partition

$$L = \left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)$$

We know then that L_{00} has no zeroes on its diagonal and $\lambda_{11} \neq 0$. We also saw that then

$$L^{-1} = \left(\begin{array}{c|c} L_{00}^{-1} & 0 \\ \hline -\frac{1}{\lambda_{11}} l_{10}^T L_{00}^{-1} & \frac{1}{\lambda_{11}} \end{array}\right)$$

Hence, the matrix has an inverse, and it is lower triangular.

By the **Principle of Mathematical Induction**, the result holds.

☛ BACK TO TEXT

Homework 7.3.4.7 Find

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}^{-1} =$$

Answer:

$$\begin{pmatrix} -1 & 2 \\ 1 & -1 \end{pmatrix}.$$

Here is how you can find this matrix: First, you compute the LU factorization. Since there is only one step, this is easy:

$$\begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix}.$$

Thus,

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix}.$$

Next, you solve

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

by solving $\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \zeta_0 \\ \zeta_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ followed by $\begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \zeta_0 \\ \zeta_1 \end{pmatrix}$. If you do this right, you get $\begin{pmatrix} -1 \\ 1 \end{pmatrix}$, which becomes the first column of the inverse.

You solve

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

in a similar manner, yielding the second column of the inverse, $\begin{pmatrix} 2 \\ -1 \end{pmatrix}$.

Check:

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} -1 & 2 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 7.3.4.8 If $\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1} \neq 0$ then

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}^{-1} = \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix}$$

(Just check by multiplying... Deriving the formula is time consuming.)

True/False

Answer: True

Check:

$$\begin{aligned}
&\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix} \\
&= \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix} \\
&= \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{0,0}\alpha_{1,1} - \alpha_{0,1}\alpha_{1,0} & -\alpha_{0,0}\alpha_{0,1} + \alpha_{0,1}\alpha_{0,0} \\ \alpha_{1,0}\alpha_{1,1} - \alpha_{1,1}\alpha_{1,0} & -\alpha_{1,0}\alpha_{0,1} + \alpha_{1,1}\alpha_{0,0} \end{pmatrix} \\
&= \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{0,0}\alpha_{1,1} - \alpha_{0,1}\alpha_{1,0} & 0 \\ 0 & \alpha_{1,1}\alpha_{0,0} - \alpha_{1,0}\alpha_{0,1} \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}
\end{aligned}$$

☛ BACK TO TEXT

Homework 7.3.4.9 The 2×2 matrix $A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}$ has an inverse if and only if $\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1} \neq 0$.

True/False

Answer: True

This is an immediate consequence of the last exercise.

☛ BACK TO TEXT

7.3.5 Properties

Homework 7.3.5.1 Let $\alpha \neq 0$ and B have an inverse. Then

$$(\alpha B)^{-1} = \frac{1}{\alpha} B^{-1}.$$

True/False

Answer: True

$$(\alpha B)(\frac{1}{\alpha} B^{-1}) = \alpha B \frac{1}{\alpha} B^{-1} = \underbrace{\alpha \frac{1}{\alpha}}_{1} \underbrace{BB^{-1}}_{I} = I.$$

☛ BACK TO TEXT

Homework 7.3.5.2 Which of the following is true regardless of matrices A and B (as long as they have an inverse and are of the same size)?

(a) $(AB)^{-1} = A^{-1}B^{-1}$

(b) $(AB)^{-1} = B^{-1}A^{-1}$

(c) $(AB)^{-1} = B^{-1}A$

(d) $(AB)^{-1} = B^{-1}$

Answer: (b)

$$(AB)(B^{-1}A^{-1}) = A \underbrace{(BB^{-1})}_{I} A^{-1} = \underbrace{AA^{-1}}_{I} = I.$$

☛ BACK TO TEXT

Homework 7.3.5.3 Let square matrices $A, B, C \in \mathbb{R}^{n\times n}$ have inverses A^{-1}, B^{-1}, and C^{-1}, respectively. Then $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$.

Always/Sometimes/Never

Answer: Always

$$(ABC)(C^{-1}B^{-1}A^{-1}) = AB \underbrace{(CC^{-1})}_{I} B^{-1}A^{-1} = A \underbrace{(BB^{-1})}_{I} A^{-1} = \underbrace{AA^{-1}}_{I} = I.$$

☛ BACK TO TEXT

Homework 7.3.5.4 Let square matrix A have inverse A^{-1}. Then $(A^T)^{-1} = (A^{-1})^T$.

Always/Sometimes/Never

Answer: Always

$$A^T(A^{-1})^T = (\underbrace{A^{-1}A}_{I})^T = I^T = I$$

Since $(A^{-1})^T = (A^T)^{-1}$, we will often write A^{-T}.

☛ BACK TO TEXT

Homework 7.3.5.5

$$(A^{-1})^{-1} = A$$

Always/Sometimes/Never

Answer: Always

$$(A^{-1})^{-1}A^{-1} = (AA^{-1})^{-1} = I^{-1} = I$$

☛ BACK TO TEXT

Week 8: More on Matrix Inversion (Answers)

8.1.1 When LU Factorization with Row Pivoting Fails

Homework 8.1.1.1 Assume that $A, B, C \in \mathbb{R}^{n \times n}$, let $BA = C$, and B be nonsingular.

A is nonsingular if and only if C is nonsingular.

True/False

Answer: True

We will prove this by first proving that if A is nonsingular then C is nonsingular and then proving that if C is nonsingular then A is nonsingular.

($\Rightarrow$) Assume A is nonsingular. If we can show that C has an inverse, then C is nonsingular. But

$$(A^{-1}B^{-1})C = (A^{-1}B^{-1})BA = A^{-1}(B^{-1}B)A = A^{-1}A = I$$

and hence $C^{-1} = A^{-1}B^{-1}$ exists. Since C^{-1} exists, C is nonsingular.

($\Leftarrow$) Assume C is nonsingular. Then $A = B^{-1}C$ since B is nonsingular. But then $(C^{-1}B)A = (C^{-1}B)B^{-1}C = I$ and hence $A^{-1} = C^{-1}B$ exists. Since A^{-1} exists, A is nonsingular.

☛ BACK TO TEXT

8.2 Gauss-Jordan Elimination

8.2.1 Solving $Ax = b$ via Gauss-Jordan Elimination

Homework 8.2.1.1 Perform the following steps

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} \hline -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ \hline 2\chi_0 & - & 3\chi_1 & + & 7\chi_2 & = & 11 \\ -4\chi_0 & + & 3\chi_1 & - & 7\chi_2 & = & -9 \end{array} \longrightarrow \begin{array}{rcrcrcr} \hline -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ \hline & & -\chi_1 & + & 2\chi_2 & = & 4 \\ & & -\chi_1 & + & 3\chi_2 & = & 5 \end{array}$$

  one must subtract $\lambda_{1,0} = \boxed{-1}$ times the first row from the second row and subtract $\lambda_{2,0} = \boxed{2}$ times the first row from the third row.

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ \hline & & -\chi_1 & + & 2\chi_2 & = & 4 \\ \hline & & -\chi_1 & + & 3\chi_2 & = & 5 \end{array} \longrightarrow \begin{array}{rcrcrcr} -2\chi_0 & & & - & \chi_2 & = & 1 \\ \hline & & -\chi_1 & + & 2\chi_2 & = & 4 \\ \hline & & & & \chi_2 & = & 1 \end{array}$$

  one must subtract $\upsilon_{0,1} = \boxed{-2}$ times the second row from the first row and subtract $\lambda_{2,1} = \boxed{1}$ times the second row from the third row.

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} -2\chi_0 & & & - & \chi_2 & = & 1 \\ & & -\chi_1 & + & 2\chi_2 & = & 4 \\ \hline & & & & \chi_2 & = & 1 \\ \hline \end{array} \longrightarrow \begin{array}{rcrcrcr} -2\chi_0 & & & & & = & 2 \\ & & -\chi_1 & & & = & 2 \\ \hline & & & & \chi_2 & = & 1 \\ \hline \end{array}$$

  one must subtract $\upsilon_{0,2} = \boxed{-1}$ times the third row from the first row and subtract $\upsilon_{1,2} = \boxed{2}$ times the third row from the first row.

- To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} -2\chi_0 & & & & & = & 2 \\ & & -\chi_1 & & & = & 2 \\ & & & & \chi_2 & = & 1 \end{array} \longrightarrow \begin{array}{rcrcrcr} \chi_0 & & & & & = & -1 \\ & & \chi_1 & & & = & -2 \\ & & & & \chi_2 & = & 1 \end{array}$$

one must multiply the first row by $\delta_{0,0} = \boxed{-1/2}$, the second row by $\delta_{1,1} = \boxed{-1}$, and the third row by $\delta_{2,2} = \boxed{1}$.

- Use the above exercises to compute the vector x that solves

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ 2\chi_0 & - & 3\chi_1 & + & 7\chi_2 & = & 11 \\ -4\chi_0 & + & 3\chi_1 & - & 7\chi_2 & = & -9 \end{array}$$

Answer: The answer can be read off from the last result:

$$\begin{array}{rcrcrcr} \chi_0 & & & & & = & -1 \\ & & \chi_1 & & & = & -2 \\ & & & & \chi_2 & = & 1 \end{array}$$

☛ BACK TO TEXT

Homework 8.2.1.2 Perform the process illustrated in the last exercise to solve the systems of linear equations

- $\begin{pmatrix} 3 & 2 & 10 \\ -3 & -3 & -14 \\ 3 & 1 & 3 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -7 \\ 9 \\ -5 \end{pmatrix}$ **Answer:**

  – To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} \hline 3\chi_0 & + & 2\chi_1 & + & 10\chi_2 & = & -7 \\ \hline -3\chi_0 & - & 3\chi_1 & - & 14\chi_2 & = & 9 \\ 3\chi_0 & + & \chi_1 & + & 3\chi_2 & = & -5 \end{array} \longrightarrow \begin{array}{rcrcrcr} \hline 3\chi_0 & + & 2\chi_1 & + & 10\chi_2 & = & -7 \\ \hline & - & \chi_1 & - & 4\chi_2 & = & 2 \\ & - & \chi_1 & - & 7\chi_2 & = & 2 \end{array}$$

  one must subtract $\lambda_{1,0} = -1$ times the first row from the second row and subtract $\lambda_{2,0} = 1$ times the first row from the third row.

  – To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} 3\chi_0 & + & 2\chi_1 & + & 10\chi_2 & = & -7 \\ \hline & - & \chi_1 & - & 4\chi_2 & = & 2 \\ & - & \chi_1 & - & 7\chi_2 & = & 2 \end{array} \longrightarrow \begin{array}{rcrcrcr} 3\chi_0 & & & + & 2\chi_2 & = & -3 \\ \hline & - & \chi_1 & - & 4\chi_2 & = & 2 \\ & & & - & 3\chi_2 & = & 0 \end{array}$$

  one must subtract $\upsilon_{0,1} = -2$ times the second row from the first row and subtract $\lambda_{2,1} = 1$ times the second row from the third row.

  – To transform the system on the left to the one on the right:

$$\begin{array}{rcrcrcr} 3\chi_0 & & & + & 2\chi_2 & = & -3 \\ \hline & - & \chi_1 & - & 4\chi_2 & = & 2 \\ & & & - & 3\chi_2 & = & 0 \end{array} \longrightarrow \begin{array}{rcrcrcr} 3\chi_0 & & & & & = & -3 \\ \hline & - & \chi_1 & & & = & 2 \\ & & & - & 3\chi_2 & = & 0 \end{array}$$

  one must subtract $\upsilon_{0,2} = -2/3$ times the third row from the first row and subtract $\upsilon_{1,2} = 4/3$ times the third row from the first row.

- To transform the system on the left to the one on the right:

$$
\begin{array}{rcrcrcr}
3\chi_0 & & & & & = & -3 \\
\hline
 & - & \chi_1 & & & = & 2 \\
 & & & - & 3\chi_2 & = & 0
\end{array}
\longrightarrow
\begin{array}{rcrcrcr}
\chi_0 & & & & & = & -1 \\
 & & \chi_1 & & & = & -2 \\
 & & & & \chi_2 & = & 1
\end{array}
$$

  one must multiply the first row by $\delta_{0,0} = 1/3$, the second row by $\delta_{1,1} = -1$, and the third row by $\delta_{2,2} = -1/3$.

-

$$
\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -1 \\ -2 \\ 0 \end{pmatrix}
$$

- $\begin{pmatrix} 2 & -3 & 4 \\ 2 & -2 & 3 \\ 6 & -7 & 9 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -8 \\ -5 \\ -17 \end{pmatrix}$

Answer:

- To transform the system on the left to the one on the right:

$$
\begin{array}{rcrcrcr}
\hline
2\chi_0 & - & 3\chi_1 & + & 4\chi_2 & = & -8 \\
\hline
2\chi_0 & - & 2\chi_1 & + & 3\chi_2 & = & -5 \\
6\chi_0 & - & 7\chi_1 & + & 9\chi_2 & = & -17
\end{array}
\longrightarrow
\begin{array}{rcrcrcr}
\hline
2\chi_0 & - & 3\chi_1 & + & 4\chi_2 & = & -8 \\
\hline
 & & \chi_1 & - & 1\chi_2 & = & 3 \\
 & & 2\chi_1 & - & 3\chi_2 & = & 7
\end{array}
$$

  one must subtract $\lambda_{1,0} = 1$ times the first row from the second row and subtract $\lambda_{2,0} = 3$ times the first row from the third row.

- To transform the system on the left to the one on the right:

$$
\begin{array}{rcrcrcr}
2\chi_0 & - & 3\chi_1 & + & 4\chi_2 & = & -8 \\
\hline
 & & \chi_1 & - & 1\chi_2 & = & 3 \\
 & & 2\chi_1 & - & 3\chi_2 & = & 7
\end{array}
\longrightarrow
\begin{array}{rcrcrcr}
2\chi_0 & & & + & \chi_2 & = & 1 \\
\hline
 & & \chi_1 & - & 1\chi_2 & = & 3 \\
 & & 0 & & -\chi_2 & = & 1
\end{array}
$$

  one must subtract $\upsilon_{0,1} = -3$ times the second row from the first row and subtract $\lambda_{2,1} = -2$ times the second row from the third row.

- To transform the system on the left to the one on the right:

$$
\begin{array}{rcrcrcr}
2\chi_0 & & & + & \chi_2 & = & 1 \\
\hline
 & & \chi_1 & - & 1\chi_2 & = & 3 \\
 & & 0 & & -\chi_2 & = & 1
\end{array}
\longrightarrow
\begin{array}{rcrcrcr}
2\chi_0 & & & & & = & 2 \\
\hline
 & & \chi_1 & & & = & 2 \\
 & & 0 & & -\chi_2 & = & 1
\end{array}
$$

  one must subtract $\upsilon_{0,2} = -1$ times the third row from the first row and subtract $\upsilon_{1,2} = 1$ times the third row from the first row.

- To transform the system on the left to the one on the right:

$$
\begin{array}{rcrcrcr}
2\chi_0 & & & & & = & 2 \\
\hline
 & & \chi_1 & & & = & 2 \\
 & & 0 & & -\chi_2 & = & 1
\end{array}
\longrightarrow
\begin{array}{rcrcrcr}
\chi_0 & & & & & = & 1 \\
\hline
 & & \chi_1 & & & = & 2 \\
 & & 0 & & \chi_2 & = & -1
\end{array}
$$

  one must multiply the first row by $\delta_{0,0} = 1/2$, the second row by $\delta_{1,1} = 1$, and the third row by $\delta_{2,2} = -1$.

$$\begin{pmatrix}\chi_0\\ \chi_1\\ \chi_2\end{pmatrix}=\begin{pmatrix}1\\ 2\\ -1\end{pmatrix}$$

☛ BACK TO TEXT

8.2.2 Solving $Ax = b$ via Gauss-Jordan Elimination: Gauss Transforms

Homework 8.2.2.1 Evaluate

- $$\left(\begin{array}{|r|rr}\hline 1&0&0\\\hline 1&1&0\\ -2&0&1\end{array}\right)\left(\begin{array}{|r|rr||r}\hline -2&2&-5&-7\\\hline 2&-3&7&11\\ -4&3&-7&-9\end{array}\right)=\left(\begin{array}{|r|rr||r}\hline -2&2&-5&-7\\\hline 0&-1&2&4\\ 0&-1&3&5\end{array}\right)$$

- $$\left(\begin{array}{r|r|r}1&2&0\\\hline 0&1&0\\\hline 0&-1&1\end{array}\right)\left(\begin{array}{r|r|r||r}-2&2&-5&-7\\\hline 0&-1&2&4\\\hline 0&-1&3&5\end{array}\right)=\left(\begin{array}{r|r|r||r}-2&0&-1&1\\\hline 0&-1&2&4\\\hline 0&0&1&1\end{array}\right)$$

- $$\left(\begin{array}{rr|r}1&0&1\\ 0&1&-2\\\hline 0&0&1\end{array}\right)\left(\begin{array}{rr|r||r}-2&0&-1&1\\ 0&-1&2&4\\\hline 0&0&1&1\end{array}\right)=\left(\begin{array}{rr|r||r}-2&0&0&2\\ 0&-1&0&2\\\hline 0&0&1&1\end{array}\right)$$

- $$\begin{pmatrix}-\frac{1}{2}&0&0\\ 0&-1&0\\ 0&0&1\end{pmatrix}\left(\begin{array}{rrr||r}-2&0&0&2\\ 0&-1&0&2\\ 0&0&1&1\end{array}\right)=\left(\begin{array}{rrr||r}1&0&0&-1\\ 0&1&0&-2\\ 0&0&1&1\end{array}\right)$$

- Use the above exercises to compute $x=\begin{pmatrix}\chi_0\\ \chi_1\\ \chi_2\end{pmatrix}$ that solves

$$\begin{array}{rcrcrcr}-2\chi_0&+&2\chi_1&-&5\chi_2&=&-7\\ 2\chi_0&-&3\chi_1&+&7\chi_2&=&11\\ -4\chi_0&+&3\chi_1&-&7\chi_2&=&-9\end{array}$$

Answer: You can read the answer off in the last column:

$$x=\begin{pmatrix}\chi_0\\ \chi_1\\ \chi_2\end{pmatrix}=\begin{pmatrix}-1\\ -2\\ 1\end{pmatrix}$$

☛ BACK TO TEXT

Homework 8.2.2.2 This exercise shows you how to use MATLAB to do the heavy lifting for Homework 8.2.2.1. Again solve

$$\begin{array}{rcrcrcr} -2\chi_0 & + & 2\chi_1 & - & 5\chi_2 & = & -7 \\ 2\chi_0 & - & 3\chi_1 & + & 7\chi_2 & = & 11 \\ -4\chi_0 & + & 3\chi_1 & - & 7\chi_2 & = & -9 \end{array}$$

via Gauss-Jordan elimination. This time we set this up as an appended matrix:

$$\left(\begin{array}{rrr|r} -2 & 2 & -5 & -7 \\ 2 & -3 & 7 & 11 \\ -4 & 3 & -7 & -9 \end{array}\right).$$

We can enter this into MATLAB as

```
A = [
-2  2 -5  ??
 2 -3  7  ??
-4  3 -7  ??
]
```

(You enter `??`.) Create the Gauss transform, G_0, that zeroes the entries in the first column below the diagonal:

```
G0 = [
 1  0  0
??  1  0
??  0  1
]
```

(You fill in the `??`). Now apply the Gauss transform to the appended system:

```
A0 = G0 * A
```

Similarly create G_1,

```
G1 = [
 1 ??  0
 0  1  0
 0 ??  1
]
```

A_1, G_2, and A_2, where A_2 equals the appended system that has been transformed into a diagonal system. Finally, let D equal to a diagonal matrix so that $A_3 = D * A2$ has the identity for the first three columns.

You can then find the solution to the linear system in the last column.

Answer:

```
>> A = [
-2  2  -5  -7
 2 -3   7  11
-4  3  -7  -9
]

A =

    -2     2    -5    -7
     2    -3     7    11
    -4     3    -7    -9
```

```
>> G0 = [
 1 0 0
 1 1 0
-2 0 1
]

G0 =

     1     0     0
     1     1     0
    -2     0     1

>> A0 = G0 * A

A0 =

    -2     2    -5    -7
     0    -1     2     4
     0    -1     3     5

>> G1 = [
1  2 0
0  1 0
0 -1 1
]

G1 =

     1     2     0
     0     1     0
     0    -1     1

>> A1 = G1 * A0

A1 =

    -2     0    -1     1
     0    -1     2     4
     0     0     1     1

>> G2 = [
1 0  1
0 1 -2
0 0  1
]

G2 =

     1     0     1
     0     1    -2
     0     0     1

>> A2 = G2 * A1

A2 =
```

```
    -2      0      0      2
     0     -1      0      2
     0      0      1      1

>> D = [
-1/2   0 0
0    -1 0
0     0 1
]

D =

   -0.5000         0         0
         0   -1.0000         0
         0         0    1.0000

>> A3 = D * A2

A3 =

     1      0      0     -1
     0      1      0     -2
     0      0      1      1
```

The answer can now be read off in the last column.

☛ BACK TO TEXT

Homework 8.2.2.3 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & b_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21} & A_{22} & b_2 \end{array}\right) = \left(\begin{array}{c|c|c||c} D_{00} & a_{01}-\alpha_{11}u_{01} & A_{02}-u_{01}a_{12}^T & b_0-\beta_1 u_{01} \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21}-\alpha_{11}l_{21} & A_{22}-l_{21}a_{12}^T & b_2-\beta_1 l_{21} \end{array}\right)$$

Always/Sometimes/Never **Answer:** Always

Just multiply it out.

☛ BACK TO TEXT

Homework 8.2.2.4 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$; and
- $l_{21} := a_{21}/\alpha_{11}$.

Consider the following expression:

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & b_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21} & A_{22} & b_2 \end{array}\right) = \left(\begin{array}{c|c|c||c} D_{00} & 0 & A_{02}-u_{01}a_{12}^T & b_0-\beta_1 u_{01} \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & 0 & A_{22}-l_{21}a_{12}^T & b_2-\beta_1 l_{21} \end{array}\right)$$

Always/Sometimes/Never

Answer: Always

Just multiply it out.

☛ BACK TO TEXT

8.2.3 Solving $Ax = b$ via Gauss-Jordan Elimination: Multiple Right-Hand Sides

Homework 8.2.3.1 Evaluate

- $$\left(\begin{array}{r|rr} 1 & 0 & 0 \\ \hline 1 & 1 & 0 \\ -2 & 0 & 1 \end{array}\right)\left(\begin{array}{r|rr||rr} -2 & 2 & -5 & -7 & 8 \\ \hline 2 & -3 & 7 & 11 & -13 \\ -4 & 3 & -7 & -9 & 9 \end{array}\right) = \left(\begin{array}{r|rr||rr} -2 & 2 & -5 & -7 & \Box \\ \hline 0 & -1 & 2 & 4 & \Box \\ 0 & -1 & 3 & 5 & \Box \end{array}\right) \quad \begin{array}{rcr} \Box & = & 8 \\ \hline \Box & = & -5 \\ \Box & = & -7 \end{array}$$

- $$\left(\begin{array}{r|r|r} 1 & 2 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1 & 1 \end{array}\right)\left(\begin{array}{r|r|r||rr} -2 & 2 & -5 & -7 & 8 \\ \hline 0 & -1 & 2 & 4 & -5 \\ \hline 0 & -1 & 3 & 5 & -7 \end{array}\right) = \left(\begin{array}{r|r|r||rr} -2 & 0 & -1 & 1 & \Box \\ \hline 0 & -1 & 2 & 4 & \Box \\ \hline 0 & 0 & 1 & 1 & \Box \end{array}\right) \quad \begin{array}{rcr} \Box & = & -2 \\ \hline \Box & = & -5 \\ \hline \Box & = & -2 \end{array}$$

- $$\left(\begin{array}{rr|r} 1 & 0 & 1 \\ 0 & 1 & -2 \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{rr|r||rr} -2 & 0 & -1 & 1 & -2 \\ 0 & -1 & 2 & 4 & -5 \\ \hline 0 & 0 & 1 & 1 & -2 \end{array}\right) = \left(\begin{array}{rr|r||rr} -2 & 0 & 0 & 2 & \Box \\ 0 & -1 & 0 & 2 & \Box \\ \hline 0 & 0 & 1 & 1 & \Box \end{array}\right) \quad \begin{array}{rcr} \Box & = & -4 \\ \hline \Box & = & -1 \\ \Box & = & -2 \end{array}$$

- $$\left(\begin{array}{rrr} -\frac{1}{2} & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{array}\right)\left(\begin{array}{rrr||rr} -2 & 0 & 0 & 2 & -4 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & 1 & 1 & -2 \end{array}\right) = \left(\begin{array}{rrr||rr} 1 & 0 & 0 & -1 & \Box \\ 0 & 1 & 0 & -2 & \Box \\ 0 & 0 & 1 & 1 & \Box \end{array}\right) \quad \begin{array}{rcr} \Box & = & 2 \\ \hline \Box & = & 1 \\ \Box & = & -2 \end{array}$$

- Use the above exercises to compute $x_0 = \begin{pmatrix} \chi_{00} \\ \chi_{10} \\ \chi_{20} \end{pmatrix}$ and $x_1 = \begin{pmatrix} \chi_{01} \\ \chi_{11} \\ \chi_{21} \end{pmatrix}$ that solve

$$\begin{array}{rcrcrcr} -2\chi_{00} & + & 2\chi_{10} & - & 5\chi_{20} & = & -7 \\ 2\chi_{00} & - & 3\chi_{10} & + & 7\chi_{20} & = & 11 \\ -4\chi_{00} & + & 3\chi_{10} & - & 7\chi_{20} & = & -9 \end{array} \quad \text{and} \quad \begin{array}{rcrcrcr} -2\chi_{01} & + & 2\chi_{11} & - & 5\chi_{21} & = & 8 \\ 2\chi_{01} & - & 3\chi_{11} & + & 7\chi_{21} & = & -13 \\ -4\chi_{01} & + & 3\chi_{11} & - & 7\chi_{21} & = & 9 \end{array}$$

Answer: You can now read off the answers

$$x_0 = \begin{pmatrix} \chi_{00} \\ \chi_{10} \\ \chi_{20} \end{pmatrix} = \begin{pmatrix} -1 \\ -2 \\ 1 \end{pmatrix} \text{ and } x_1 = \begin{pmatrix} \chi_{01} \\ \chi_{11} \\ \chi_{21} \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -2 \end{pmatrix}$$

You will want to plug these results into the original equations, to double check they work.

☛ BACK TO TEXT

Homework 8.2.3.2 This exercise shows you how to use MATLAB to do the heavy lifting for Homework 8.2.3.1. Start with the appended system:

$$\left(\begin{array}{rrr|rr} -2 & 2 & -5 & -7 & 8 \\ 2 & -3 & 7 & 11 & -13 \\ -4 & 3 & -7 & -9 & 9 \end{array}\right)$$

Enter this into MATLAB as

```
A = [
-2  2 -5  ?? ??
 2 -3  7  ?? ??
-4  3 -7  ?? ??
]
```

(You enter `??`.) Create the Gauss transform, G_0, that zeroes the entries in the first column below the diagonal:

```
G0 = [
 1  0  0
??  1  0
??  0  1
]
```

(You fill in the `??`). Now apply the Gauss transform to the appended system:

```
A0 = G0 * A
```

Similarly create G_1,

```
G1 = [
 1 ??  0
 0  1  0
 0 ??  1
]
```

A_1, G_2, and A_2, where A_2 equals the appended system that has been transformed into a diagonal system. Finally, let D equal to a diagonal matrix so that $A_3 = D * A2$ has the identity for the first three columns.

You can then find the solutions to the linear systems in the last column.

Answer:

```
>> A = [
-2  2  -5  -7   8
 2 -3   7  11 -13
-4  3  -7  -9   9
]

A =
```

```
    -2     2    -5    -7     8
     2    -3     7    11   -13
    -4     3    -7    -9     9

>> G0 = [
 1 0 0
 1 1 0
-2 0 1
]

G0 =

     1     0     0
     1     1     0
    -2     0     1

>> A0 = G0 * A

A0 =

    -2     2    -5    -7     8
     0    -1     2     4    -5
     0    -1     3     5    -7

>> G1 = [
1  2 0
0  1 0
0 -1 1
]

G1 =

     1     2     0
     0     1     0
     0    -1     1

>> A1 = G1 * A0

A1 =

    -2     0    -1     1    -2
     0    -1     2     4    -5
     0     0     1     1    -2

>> G2 = [
1 0  1
0 1 -2
0 0  1
]

G2 =

     1     0     1
     0     1    -2
     0     0     1
```

```
>> A2 = G2 * A1

A2 =

    -2     0     0     2    -4
     0    -1     0     2    -1
     0     0     1     1    -2

>> D = [
-1/2  0 0
0    -1 0
0     0 1
]

D =

   -0.5000         0         0
         0   -1.0000         0
         0         0    1.0000

>> A3 = D * A2

A3 =

     1     0     0    -1     2
     0     1     0    -2     1
     0     0     1     1    -2
```

The answers can now be read off in the last columns.

☛ BACK TO TEXT

Homework 8.2.3.3 Evaluate

- $$\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline \Box & 1 & 0 \\ \Box & 0 & 1 \end{array}\right)\left(\begin{array}{c|cc||cc} 3 & 2 & 10 & -7 & 16 \\ \hline -3 & -3 & -14 & 9 & -25 \\ 3 & 1 & 4 & -5 & 3 \end{array}\right) = \left(\begin{array}{c|cc||cc} 3 & 2 & 10 & \Box & \Box \\ \hline 0 & -1 & -4 & \Box & \Box \\ 0 & -1 & -6 & \Box & \Box \end{array}\right)$$

- $$\left(\begin{array}{c|c|c} 1 & \Box & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & \Box & 1 \end{array}\right)\left(\begin{array}{c|c|c||cc} 3 & 2 & 10 & -7 & 16 \\ \hline 0 & -1 & -4 & 2 & -9 \\ \hline 0 & -1 & -6 & 2 & -13 \end{array}\right) = \left(\begin{array}{c|c|c||cc} 3 & 0 & 2 & \Box & \Box \\ \hline 0 & -1 & -4 & \Box & \Box \\ \hline 0 & 0 & -2 & \Box & \Box \end{array}\right)$$

- $$\left(\begin{array}{cc|c} 1 & 0 & \Box \\ 0 & 1 & \Box \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{cc|c||cc} 3 & 0 & 2 & -3 & -2 \\ 0 & -1 & -4 & 2 & -9 \\ \hline 0 & 0 & -2 & 0 & -4 \end{array}\right) = \left(\begin{array}{cc|c||cc} 3 & 0 & 0 & \Box & \Box \\ 0 & -1 & 0 & \Box & \Box \\ \hline 0 & 0 & -2 & \Box & \Box \end{array}\right)$$

- $$\left(\begin{array}{ccc} \Box & 0 & 0 \\ 0 & \Box & 0 \\ 0 & 0 & \Box \end{array}\right)\left(\begin{array}{ccc||cc} 3 & 0 & 0 & -3 & -6 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & -2 & 0 & -4 \end{array}\right) = \left(\begin{array}{ccc||cc} 1 & 0 & 0 & \Box & \Box \\ 0 & 1 & 0 & \Box & \Box \\ 0 & 0 & 1 & \Box & \Box \end{array}\right)$$

Use the above exercises to compute $x_0 = \begin{pmatrix} \chi_{0,0} \\ \chi_{1,0} \\ \chi_{2,0} \end{pmatrix}$ and $x_1 = \begin{pmatrix} \chi_{0,1} \\ \chi_{1,1} \\ \chi_{2,1} \end{pmatrix}$ that solve

$$\begin{array}{rcrcrcr} 3\chi_{0,0} & + & 2\chi_{1,0} & + & 10\chi_{2,0} & = & -7 \\ -3\chi_{0,0} & - & 3\chi_{1,0} & - & 14\chi_{2,0} & = & 9 \\ 3\chi_{0,0} & + & 1\chi_{1,0} & + & 4\chi_{2,0} & = & -5 \end{array} \quad \text{and} \quad \begin{array}{rcrcrcr} 3\chi_{0,0} & + & 2\chi_{1,0} & + & 10\chi_{2,0} & = & 16 \\ -3\chi_{0,0} & - & 3\chi_{1,0} & - & 14\chi_{2,0} & = & -25 \\ 3\chi_{0,0} & + & 1\chi_{1,0} & + & 4\chi_{2,0} & = & 3 \end{array}$$

(You could use MATLAB to do the heavy lifting, like in the last homework...)

Answer:

- $\left(\begin{array}{|r|rr} \hline 1 & 0 & 0 \\ \hline 1 & 1 & 0 \\ -1 & 0 & 1 \end{array}\right) \left(\begin{array}{|r|rr||rr} \hline 3 & 2 & 10 & -7 & 16 \\ \hline -3 & -3 & -14 & 9 & -25 \\ 3 & 1 & 4 & -5 & 3 \end{array}\right) = \left(\begin{array}{|r|rr||rr} \hline 3 & 2 & 10 & -7 & 16 \\ \hline 0 & -1 & -4 & 2 & -9 \\ 0 & -1 & -6 & 2 & -13 \end{array}\right)$
- $\left(\begin{array}{r|r|r} 1 & 2 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1 & 1 \end{array}\right) \left(\begin{array}{r|rr||rr} 3 & 2 & 10 & -7 & 16 \\ \hline 0 & -1 & -4 & 2 & -9 \\ \hline 0 & -1 & -6 & 2 & -13 \end{array}\right) = \left(\begin{array}{r|rr||rr} 3 & 0 & 2 & -3 & -2 \\ \hline 0 & -1 & -4 & 2 & -9 \\ \hline 0 & 0 & -2 & 0 & -4 \end{array}\right)$
- $\left(\begin{array}{rr|r} 1 & 0 & 1 \\ 0 & 1 & -2 \\ \hline 0 & 0 & 1 \end{array}\right) \left(\begin{array}{rr|r||rr} 3 & 0 & 2 & -3 & -2 \\ 0 & -1 & -4 & 2 & -9 \\ \hline 0 & 0 & -2 & 0 & -4 \end{array}\right) = \left(\begin{array}{rr|r||rr} 3 & 0 & 0 & -3 & -6 \\ 0 & -1 & 0 & 2 & -1 \\ \hline 0 & 0 & -2 & 0 & -4 \end{array}\right)$
- $\begin{pmatrix} \frac{1}{3} & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -\frac{1}{2} \end{pmatrix} \left(\begin{array}{rrr||rr} 3 & 0 & 0 & -3 & -6 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & -2 & 0 & -4 \end{array}\right) = \left(\begin{array}{rrr||rr} 1 & 0 & 0 & -1 & -2 \\ 0 & 1 & 0 & -2 & 1 \\ 0 & 0 & 1 & 0 & 2 \end{array}\right)$
- $x_0 = \begin{pmatrix} -1 \\ -2 \\ 0 \end{pmatrix}$ and $x_1 = \begin{pmatrix} -2 \\ 1 \\ 2 \end{pmatrix}$.

☛ BACK TO TEXT

Homework 8.2.3.4 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right) \left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & B_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_1^T \\ \hline 0 & a_{21} & A_{22} & B_2 \end{array}\right) = \left(\begin{array}{c|c|c||c} D_{00} & a_{01} - \alpha_{11} u_{01} & A_{02} - u_{01} a_{12}^T & B_0 - u_{01} b_1^T \\ \hline 0 & \alpha_{11} & a_{12}^T & b_1^T \\ \hline 0 & a_{21} - \alpha_{11} l_{21} & A_{22} - l_{21} a_{12}^T & B_2 - l_{21} b_1^T \end{array}\right)$$

Always/Sometimes/Never

Answer: Always

Just multiply it out.

☛ BACK TO TEXT

Homework 8.2.3.5 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$; and
- $l_{21} := a_{21}/\alpha_{11}$.

The following expression holds:

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c} D_{00} & a_{01} & A_{02} & b_0 \\ \hline 0 & \alpha_{11} & a_{12}^T & \beta_1 \\ \hline 0 & a_{21} & A_{22} & b_2 \end{array}\right) = \left(\begin{array}{c|c|c||c} D_{00} & 0 & A_{02}-u_{01}a_{12}^T & B_0-u_{01}b_1^T \\ \hline 0 & \alpha_{11} & a_{12}^T & b_1^T \\ \hline 0 & 0 & A_{22}-l_{21}a_{12}^T & B_2-l_{21}b_1^T \end{array}\right)$$

Always/Sometimes/Never

Answer: Always

Just multiply it out.

☛ BACK TO TEXT

8.2.4 Computing A^{-1} via Gauss-Jordan Elimination

Homework 8.2.4.1 Evaluate

- $$\left(\begin{array}{|ccc} 1 & 0 & 0 \\ \hline 1 & 1 & 0 \\ -2 & 0 & 1 \end{array}\right)\left(\begin{array}{|ccc||ccc} -2 & 2 & -5 & 1 & 0 & 0 \\ \hline 2 & -3 & 7 & 0 & 1 & 0 \\ -4 & 3 & -7 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{|ccc||ccc} -2 & 2 & -5 & \boxed{-2} & \boxed{2} & \boxed{5} \\ \hline 0 & -1 & 2 & \boxed{0} & \boxed{-1} & \boxed{2} \\ 0 & -1 & 3 & \boxed{0} & \boxed{-1} & \boxed{3} \end{array}\right)$$

- $$\left(\begin{array}{c|c|c} 1 & 2 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -1 & 1 \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} -2 & 2 & -5 & 1 & 0 & 0 \\ \hline 0 & -1 & 2 & 1 & 1 & 0 \\ \hline 0 & -1 & 3 & -2 & 0 & 1 \end{array}\right) = \left(\begin{array}{c|c|c||c|c|c} -2 & 0 & -1 & \boxed{-2} & \boxed{0} & \boxed{-1} \\ \hline 0 & -1 & 2 & \boxed{0} & \boxed{-1} & \boxed{2} \\ \hline 0 & 0 & 1 & \boxed{0} & \boxed{0} & \boxed{1} \end{array}\right)$$

- $$\left(\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & 1 & -2 \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{cc|c||cc|c} -2 & 0 & -1 & 3 & 2 & 0 \\ 0 & -1 & 2 & 1 & 1 & 0 \\ \hline 0 & 0 & 1 & -3 & -1 & 1 \end{array}\right) = \left(\begin{array}{cc|c||cc|c} -2 & 0 & 0 & \boxed{-2} & \boxed{0} & \boxed{0} \\ 0 & -1 & 0 & \boxed{0} & \boxed{-1} & \boxed{0} \\ \hline 0 & 0 & 1 & \boxed{0} & \boxed{0} & \boxed{1} \end{array}\right)$$

- $$\left(\begin{array}{ccc} -\frac{1}{2} & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{array}\right)\left(\begin{array}{ccc||ccc} -2 & 0 & 0 & 0 & 1 & 1 \\ 0 & -1 & 0 & 7 & 3 & -2 \\ 0 & 0 & 1 & -3 & -1 & 1 \end{array}\right) = \left(\begin{array}{ccc||ccc} 1 & 0 & 0 & \boxed{0} & \boxed{-\frac{1}{2}} & \boxed{-\frac{1}{2}} \\ 0 & 1 & 0 & \boxed{-7} & \boxed{-3} & \boxed{2} \\ 0 & 0 & 1 & \boxed{-3} & \boxed{-1} & \boxed{1} \end{array}\right)$$

- $$\left(\begin{array}{ccc} -2 & 2 & -5 \\ 2 & -3 & 7 \\ -4 & 3 & -7 \end{array}\right)\left(\begin{array}{ccc} 0 & -\frac{1}{2} & -\frac{1}{2} \\ -7 & -3 & 2 \\ -3 & -1 & 1 \end{array}\right) = \left(\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right)$$

☛ BACK TO TEXT

Homework 8.2.4.2 In this exercise, you will use MATLAB to compute the inverse of a matrix using the techniques discussed in this unit.

Initialize	`A = [ -2  2 -5`
	`       2 -3  7`
	`      -4  3 -7 ]`

Create an appended matrix by appending the identity	`A_appended = [ A    eye(size(A)) ]`

Create the first Gauss transform to introduce zeros in the first column (fill in the ?s).	`G0 = [ 1  0  0`
	`       ?  1  0`
	`       ?  0  1 ]`

Apply the Gauss transform to the appended system	`A0 = G0 * A_appended`

Create the second Gauss transform to introduce zeros in the second column	`G1 = [ 1  ?  0`
	`       0  1  0`
	`       0  ?  1 ]`

Apply the Gauss transform to the appended system	`A1 = G1 * A0`

Create the third Gauss transform to introduce zeros in the third column	`G2 = [ 1  0  ?`
	`       0  1  ?`
	`       0  0  1 ]`

Apply the Gauss transform to the appended system	`A2 = G2 * A1`

Create a diagonal matrix to set the diagonal elements to one	`D3 = [ -1/2  0  0`
	`         0   -1  0`
	`         0    0  1 ]`

Apply the diagonal matrix to the appended system	`A3 = D3 * A2`
Extract the (updated) appended columns	`Ainv = A3( :, 4:6 )`
Check that the inverse was computed	`A * Ainv`

The result should be a 3×3 identity matrix.

Answer: Homework_8_2_8_2_Answer.m.

☛ BACK TO TEXT

Homework 8.2.4.3 Compute

- $$\begin{pmatrix} 3 & 2 & 9 \\ -3 & -3 & -14 \\ 3 & 1 & 3 \end{pmatrix}^{-1} = \begin{pmatrix} \frac{5}{3} & 1 & -\frac{1}{3} \\ -11 & -6 & 5 \\ 2 & 1 & -1 \end{pmatrix}$$

- $$\begin{pmatrix} 2 & -3 & 4 \\ 2 & -2 & 3 \\ 6 & -7 & 9 \end{pmatrix}^{-1} = \begin{pmatrix} -\frac{3}{2} & \frac{1}{2} & \frac{1}{2} \\ 0 & 3 & -1 \\ 1 & 2 & -1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 8.2.4.4 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} D_{00} & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21} & A_{22} & B_{20} & 0 & I \end{array}\right)$$
$$= \left(\begin{array}{c|c|c||c|c|c} D_{00} & a_{01}-\alpha_{11}u_{01} & A_{02}-u_{01}a_{12}^T & B_{00}-u_{01}b_{10}^T & -u_{01} & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21}-\alpha_{11}l_{21} & A_{22}-l_{21}a_{12}^T & B_{20}-l_{21}b_{10}^T & -l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

Answer: Always
Just multiply it out.

☛ BACK TO TEXT

Homework 8.2.4.5 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$; and
- $l_{21} := a_{21}/\alpha_{11}$.

Consider the following expression:

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} D_{00} & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21} & A_{22} & B_{20} & 0 & I \end{array}\right)$$
$$= \left(\begin{array}{c|c|c||c|c|c} D_{00} & 0 & A_{02}-u_{01}a_{12}^T & B_{00}-u_{01}b_{10}^T & -u_{01} & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & 0 & A_{22}-l_{21}a_{12}^T & B_{20}-l_{21}b_{10}^T & -l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

Answer: Always
Just multiply it out.

☛ BACK TO TEXT

8.2.5 Computing A^{-1} via Gauss-Jordan Elimination, Alternative

Homework 8.2.5.1

- Determine $\delta_{0,0}$, $\lambda_{1,0}$, $\lambda_{2,0}$ so that

$$\left(\begin{array}{|c|cc} \delta_{0,0} & 0 & 0 \\ \hline \lambda_{1,0} & 1 & 0 \\ \lambda_{2,0} & 0 & 1 \end{array}\right)\left(\begin{array}{|c|cc||ccc} -1 & -4 & -2 & 1 & 0 & 0 \\ \hline 2 & 6 & 2 & 0 & 1 & 0 \\ -1 & 0 & 3 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{|c|cc||c|cc} 1 & 4 & 2 & -1 & 0 & 0 \\ \hline 0 & -2 & -2 & 2 & 1 & 0 \\ 0 & 4 & 5 & -1 & 0 & 1 \end{array}\right)$$

Answer: $\delta_{0,0} = 1/\alpha_{0,0} = -1$; $\lambda_{1,0} = -\alpha_{1,0}/\alpha_{0,0} = 2$; $\lambda_{2,0} = -\alpha_{2,0}/\alpha_{0,0} = -1$.

$$\left(\begin{array}{r|r|r} -1 & 0 & 0 \\ \hline 2 & 1 & 0 \\ -1 & 0 & 1 \end{array}\right)\left(\begin{array}{r|rr||r|r|r} -1 & -4 & -2 & 1 & 0 & 0 \\ \hline 2 & 6 & 2 & 0 & 1 & 0 \\ -1 & 0 & 3 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{r|rr||r|r|r} 1 & 4 & 2 & -1 & 0 & 0 \\ \hline 0 & -2 & -2 & 2 & 1 & 0 \\ 0 & 4 & 5 & -1 & 0 & 1 \end{array}\right)$$

- Determine $\upsilon_{0,1}$, $\delta_{1,1}$, and $\lambda_{2,1}$ so that

$$\left(\begin{array}{c|c|c} 1 & \upsilon_{0,1} & 0 \\ \hline 0 & \delta_{1,1} & 0 \\ \hline 0 & \lambda_{2,1} & 1 \end{array}\right)\left(\begin{array}{r|r|r||r|r|r} 1 & 4 & 2 & -1 & 0 & 0 \\ \hline 0 & -2 & -2 & 2 & 1 & 0 \\ \hline 0 & 4 & 5 & -1 & 0 & 1 \end{array}\right) = \left(\begin{array}{r|r|r||r|r|r} 1 & 0 & -2 & 3 & 2 & 0 \\ \hline 0 & 1 & 1 & -1 & -\frac{1}{2} & 0 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right)$$

Answer: $\upsilon_{0,1} = \alpha_{0,1}/\alpha_{1,1} = 2$, $\delta_{1,1} = 1/\alpha_{1,1} = -\frac{1}{2}$, $\lambda_{2,1} = \alpha_{2,1}/\alpha_{1,1} = 2$.

$$\left(\begin{array}{r|rr} 1 & 2 & 0 \\ \hline 0 & -\frac{1}{2} & 0 \\ 0 & 2 & 1 \end{array}\right)\left(\begin{array}{r|r|r||r|r|r} 1 & 4 & 2 & -1 & 0 & 0 \\ \hline 0 & -2 & -2 & 2 & 1 & 0 \\ \hline 0 & 4 & 5 & -1 & 0 & 1 \end{array}\right) = \left(\begin{array}{r|r|r||r|r|r} 1 & 0 & -2 & 3 & 2 & 0 \\ \hline 0 & 1 & 1 & -1 & -\frac{1}{2} & 0 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right)$$

- Determine $\upsilon_{0,2}$, $\upsilon_{0,2}$, and $\delta_{2,2}$ so that

$$\left(\begin{array}{cc|c} 1 & 0 & \upsilon_{0,2} \\ 0 & 1 & \upsilon_{1,2} \\ \hline 0 & 0 & \delta_{2,2} \end{array}\right)\left(\begin{array}{rr|r||rr|r} 1 & 0 & -2 & 3 & 2 & 0 \\ 0 & 1 & 1 & -1 & -\frac{1}{2} & 0 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right) = \left(\begin{array}{rr|r||rr|r} 1 & 0 & 0 & 9 & 6 & 2 \\ 0 & 1 & 0 & -4 & -\frac{5}{2} & -1 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right)$$

Answer: $\upsilon_{0,2} = \alpha_{0,2}/\alpha_{2,2} = 2$, $\upsilon_{1,2} = \alpha_{1,2}/\alpha_{1,1} = -1$, $\delta_{2,2} = 1/\alpha_{1,1} = 1$.

$$\left(\begin{array}{rr|r} 1 & 0 & 2 \\ 0 & 1 & -1 \\ \hline 0 & 0 & 1 \end{array}\right)\left(\begin{array}{rr|r||rr|r} 1 & 0 & -2 & 3 & 2 & 0 \\ 0 & 1 & 1 & -1 & -\frac{1}{2} & 0 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right) = \left(\begin{array}{rr|r||rr|r} 1 & 0 & 0 & 9 & 6 & 2 \\ 0 & 1 & 0 & -4 & -\frac{5}{2} & -1 \\ \hline 0 & 0 & 1 & 3 & 2 & 1 \end{array}\right)$$

- Evaluate

$$\begin{pmatrix} -1 & -4 & -2 \\ 2 & 6 & 2 \\ -1 & 0 & 3 \end{pmatrix}\begin{pmatrix} 9 & 6 & 2 \\ -4 & -\frac{5}{2} & -1 \\ 3 & 2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 8.2.5.2 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense.

$$\left(\begin{array}{c|c|c} I & -u_{01} & 0 \\ \hline 0 & \delta_{11} & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c|c|c||c|c|c} I & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline 0 & a_{21} & A_{22} & B_{20} & 0 & I \end{array}\right)$$

$$= \left(\begin{array}{c|c|c||c|c|c} I & a_{01} - \alpha_{11}u_{01} & A_{02} - u_{01}a_{12}^T & B_{00} - u_{01}b_{10}^T & -u_{01} & 0 \\ \hline 0 & \delta_{11}\alpha_{11} & \delta_{11}a_{12}^T & \delta_{11}b_{10}^T & \delta_{11} & 0 \\ \hline 0 & a_{21} - \alpha_{11}l_{21} & A_{22} - l_{21}a_{12}^T & B_{20} - l_{21}b_{10}^T & -l_{21} & I \end{array}\right)$$

Always/Sometimes/Never

Answer: Always

Just multiply it out.

BACK TO TEXT

Homework 8.2.5.3 Assume below that all matrices and vectors are partitioned "conformally" so that the operations make sense. Choose

- $u_{01} := a_{01}/\alpha_{11}$;
- $l_{21} := a_{21}/\alpha_{11}$; and
- $\delta_{11} := 1/\alpha_{11}$.

$$
\left(\begin{array}{c|c|c}
I & -u_{01} & 0 \\ \hline
0 & \delta_{11} & 0 \\ \hline
0 & -l_{21} & I
\end{array}\right)
\left(\begin{array}{c|c|c||c|c|c}
I & a_{01} & A_{02} & B_{00} & 0 & 0 \\ \hline
0 & \alpha_{11} & a_{12}^T & b_{10}^T & 1 & 0 \\ \hline
0 & a_{21} & A_{22} & B_{20} & 0 & I
\end{array}\right)
$$

$$
= \left(\begin{array}{c|c|c||c|c|c}
I & 0 & A_{02} - u_{01} a_{12}^T & B_{00} - u_{01} b_{10}^T & -u_{01} & 0 \\ \hline
0 & 1 & a_{12}^T/\alpha_{11} & b_{10}^T/\alpha_{11} & 1/\alpha_{11} & 0 \\ \hline
0 & 0 & A_{22} - l_{21} a_{12}^T & B_{20} - l_{21} b_{10}^T & -l_{21} & I
\end{array}\right)
$$

Always/Sometimes/Never

Answer: Always

Just multiply it out.

BACK TO TEXT

Homework 8.2.5.4 Implement the algorithm in Figure 8.7 yielding the function

- `[ A_out ] = GJ_Inverse_alt_unb( A, B )`. Assume that it is called as

  `Ainv = GJ_Inverse_alt_unb( A, B )`

  Matrices A and B must be square and of the same size.

Check that it computes correctly with the script

- `test_GJ_Inverse_alt_unb.m`.

Answer: Our implementation: `GJ_Inverse_alt_unb.m`.

BACK TO TEXT

Homework 8.2.5.5 If you are very careful, you can overwrite matrix A with its inverse without requiring the matrix B.

Modify the algorithm in Figure 8.7 so that it overwrites A with its inverse without the use of matrix B yielding the function

- `[ A_out ] = GJ_Inverse_inplace_unb( A )`.

Check that it computes correctly with the script

- `test_GJ_Inverse_inplace_unb.m`.

Answer: The modified algorithm can be found in Figure 8.2.

Our implementation: `GJ_Inverse_inplace_unb.m`.

BACK TO TEXT

Algorithm: $[A] := \text{GJ\_INVERSE\_INPLACE}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

where α_{11} is 1×1

$$\left(\begin{array}{c|c|c} & a_{01} := a_{01}/\alpha_{11} & A_{02} := A_{02} - a_{01}a_{12}^T \\ \hline & & \\ \hline & a_{21} := a_{21}/\alpha_{11} & A_{22} := A_{22} - a_{21}a_{12}^T \end{array}\right) \quad \left(\begin{array}{c|c|c} A_{00} := A_{00} - a_{01}a_{10}^T & a_{01} := -a_{01} & \\ \hline & & \\ \hline A_{20} := A_{20} - a_{21}a_{10}^T & a_{21} := -a_{21} & \end{array}\right)$$

(Note: above a_{01} and a_{21} must be updated before the operations to their right.)

$$\left(\begin{array}{c|c|c} & & \\ \hline & & a_{12}^T := a_{12}^T/\alpha_{11} \\ \hline & & \end{array}\right) \quad \left(\begin{array}{c|c|c} & & \\ \hline a_{10}^T := a_{10}^T/\alpha_{11} & \alpha_{11} = 1/\alpha_{11} & \\ \hline & & \end{array}\right)$$

(Note: above α_{11} must be updated last.)

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

Figure 8.2: Algorithm that transforms matrix A into A^{-1}.

8.3 (Almost) Never, Ever Invert a Matrix

8.3.1 Solving $Ax = b$

Homework 8.3.1.1 Let $A \in \mathbb{R}^{n \times n}$ and $x, b \in \mathbb{R}^n$. What is the cost of solving $Ax = b$ via LU factorization (assuming there is nothing special about A)? You may ignore the need for pivoting.

Answer: LU factorization requires approximately $\frac{2}{3}n^3$ flops and the two triangular solves require approximately n^2 flops each, for a total cost of

$$\frac{2}{3}n^3 + 2n^2 \text{ flops.}$$

☛ BACK TO TEXT

Homework 8.3.1.2 Let $A \in \mathbb{R}^{n \times n}$ and $x, b \in \mathbb{R}^n$. What is the cost of solving $Ax = b$ if you first invert matrix A and than compute $x = A^{-1}b$? (Assume there is nothing special about A and ignore the need for pivoting.)

Answer: Inverting the matrix requires approximately $2n^3$ flops and the matrix-vector multiplication approximately $2n^2$ flops, for a total cost of, approximately,

$$2n^3 + 2n^2 \text{ flops.}$$

☛ BACK TO TEXT

Homework 8.3.1.3 What is wrong with the above argument?

Answer: One problem is that EVEN if the inversion of A came for free, then solving with the triangular matrices takes approximately $2n^2$ flops while performing the matrix-vector multiplication requires the same number of flops.

Now, there may be cases where solving with the triangular matrices achieves lower performance on some architectures, in which case one could contemplate forming A^{-1}. However, in 30 years in the field, we have never run into a situation where this becomes worthwhile.

☛ BACK TO TEXT

Week 9: Vector Spaces (Answers)

9.2 When Systems Don't Have a Unique Solution

9.2.1 When Solutions Are Not Unique

Homework 9.2.1.1 Evaluate

1. $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}$

2. $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}$

3. $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}$

Does the system $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}$ have multiple solutions? Yes/No

Answer: Yes

Clearly, this system has multiple solutions.

☛ BACK TO TEXT

9.2.2 When Linear Systems Have No Solutions

Homework 9.2.2.1 The system $\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 3 \end{pmatrix}$ has no solution.

True/False

Answer: True

- Set this up as an appended system

$$\left(\begin{array}{rrr|r} 2 & -4 & -2 & 4 \\ -2 & 4 & 1 & -3 \\ 2 & -4 & 0 & 3 \end{array}\right).$$

  Now, start applying Gaussian elimination (with row exchanges if necessary).

- Use the first row to eliminate the coefficients in the first column below the diagonal:

$$\left(\begin{array}{rrr|r} 2 & -4 & -2 & 4 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 2 & -1 \end{array}\right).$$

- There is now a zero on the diagonal and no row below it with which to exchange to put a nonzero there. So, we move on and use the second row to eliminate the coefficients in the third column below the second row:

$$\left(\begin{array}{rrr|r} 2 & -4 & -2 & 4 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{array}\right).$$

The last row translates to $0 \times \chi_0 + 0 \times \chi_1 + 0 \times \chi_2 = 1$ or $0 = 1$. This is a contradiction. It follows that this system does not have a solution since there are no choices for χ_0, χ_1, and χ_2 such that $0 = 1$.

☛ BACK TO TEXT

9.2.4 What is Going On?

Homework 9.2.4.1 Let $Ax_s = b$, $Ax_{n_0} = 0$ and $Ax_{n_1} = 0$. Also, let $\beta_0, \beta_1 \in \mathbb{R}$. Then $A(x_s + \beta_0 x_{n_0} + \beta_1 x_{n_1}) = b$.

Always/Sometimes/Never

Answer: Always

$$\begin{array}{l} A(x_s + \beta_0 x_{n_0} + \beta_1 x_{n_1}) \\ \quad = \qquad < \text{Distribute } A > \\ Ax_s + A(\beta_0 x_{n_0}) + A(\beta_1 x_{n_1}) \\ \quad = \qquad < \text{Constant can be brought out} > \\ Ax_s + \beta_0 A x_{n_0} + \beta_1 A x_{n_1} \\ \quad = \qquad < Ax_s = b \text{ and } Ax_{n_0} = Ax_{n_1} = 0 > \\ b + 0 + 0 \\ \quad = \qquad < \text{algebra} > \\ b. \end{array}$$

☛ BACK TO TEXT

9.2.5 Toward a Systematic Approach to Finding All Solutions

Homework 9.2.5.1 Find the general solution (an expression for all solutions) for

$$\begin{pmatrix} 2 & -2 & -4 \\ -2 & 1 & 4 \\ 2 & 0 & -4 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}.$$

Answer:

- Set this up as an appended system

$$\left(\begin{array}{rrr|r} 2 & -2 & -4 & 4 \\ -2 & 1 & 4 & -3 \\ 2 & 0 & -4 & 2 \end{array}\right).$$

Now, start applying Gaussian elimination (with row exchanges if necessary).

- Use the first row to eliminate the coefficients in the first column below the diagonal:

$$\left(\begin{array}{rrr|r} \boxed{2} & -2 & -4 & 4 \\ 0 & -1 & 0 & 1 \\ 0 & 2 & 0 & -2 \end{array}\right).$$

- Use the second row to eliminate the coefficients in the second column below the second row:

$$\left(\begin{array}{rrr|r} \boxed{2} & -2 & -4 & 4 \\ 0 & \boxed{-1} & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

  This is good news: while we are now left with two equations with three unknowns, the last row translates to $0 = 0$ and hence there is no contradication. There will be an infinite number of solutions.

- Subtract a multiple of the second row from the first row to eliminate the coefficient above the pivot.

$$\left(\begin{array}{rrr|r} \boxed{2} & 0 & -4 & 2 \\ 0 & \boxed{-1} & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

- Divide the first and second row by appropriate values to make the normalize the pivots to one:

$$\left(\begin{array}{rrr|r} \boxed{1} & 0 & -2 & 1 \\ 0 & \boxed{1} & 0 & -1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

- The above really represents two appended systems, the second one corresponding to the case where the right-hand side is zero:

$$\left(\begin{array}{rrr|r} \boxed{1} & 0 & -2 & 1 \\ 0 & \boxed{1} & 0 & -1 \\ 0 & 0 & 0 & 0 \end{array}\right) \quad \left(\begin{array}{rrr|r} \boxed{1} & 0 & -2 & 0 \\ 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

- The free variable is now χ_2 since there is no pivot in the column corresponding to that component of x.

- We translate the appended system on the left back into a linear system but with $\chi_2 = 0$ (since it can be chosen to equal any value and zero is convenient):

$$\begin{array}{rcrcrcr} \chi_0 & & & - & 2(0) & = & 1 \\ & & \chi_1 & & & = & -1 \\ & & & & \chi_2 & = & 0 \end{array}$$

  Solving this yields the specific solution $x_s = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}$.

- We translate the appended system on the right back into a linear system but with $\chi_1 = 1$ (we can choose any value except for zero, and one is is convenient):

$$\begin{array}{rcrcrcr} \chi_0 & & & - & 2(1) & = & 0 \\ & & \chi_1 & & & = & 0 \\ & & & & \chi_2 & = & 1 \end{array}$$

Solving this yields the solution $x_n = \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}$.

- The general solution now is

$$x = x_s + \beta x_n = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}.$$

The above seems like a very long-winded way of answering the question. In practice, here is what you will want to do:

1. Set the linear system up as an appended system $$\left(\begin{array}{rrr\|r} 2 & -2 & -4 & 4 \\ -2 & 1 & 4 & -3 \\ 2 & 0 & -4 & 2 \end{array}\right).$$	**2.** Check if you need to pivot (exchange) rows, identify the pivot, and eliminate the elements below the pivot: $$\left(\begin{array}{rrr\|r} \boxed{2} & -2 & -4 & 4 \\ 0 & -1 & 0 & 1 \\ 0 & 2 & 0 & -2 \end{array}\right).$$
3. Check if you need to pivot (exchange) rows, identify the pivot, and eliminate the elements below the pivot: $$\left(\begin{array}{rrr\|r} \boxed{2} & -2 & -4 & 4 \\ 0 & \boxed{-1} & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$	**4.** Eliminate elements above the pivot: $$\left(\begin{array}{rrr\|r} \boxed{2} & 0 & -4 & 2 \\ 0 & \boxed{-1} & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$
5. Divide to normalize the pivots to one: $$\left(\begin{array}{rrr\|r} \boxed{1} & 0 & -2 & 1 \\ 0 & \boxed{1} & 0 & -1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$	**6.** Identify the free variable(s) as corresponding to the column(s) in which there is no pivot. In this case, that is χ_2.
7. Set the free variable(s) to zero and solve. This gives you the specific solution $$x_s = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}.$$	**8.** Set the right-hand side to zero in the transformed system $$\left(\begin{array}{rrr\|r} \boxed{1} & 0 & -2 & 0 \\ 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$
9. Set the free variables one by one to one (and the others to zero). This gives you vectors that satisfy $Ax = 0$: $$x_n = \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}.$$	**10.** The general solution now is $$x = x_s + \beta x_n = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}.$$ (If there is more than one free variable, you will get more terms with vectors that satisfy $Ax = 0$. More on that later.)

☛ BACK TO TEXT

Homework 9.2.5.2 Find the general solution (an expression for all solutions) for

$$\begin{pmatrix} 2 & -4 & -2 \\ -2 & 4 & 1 \\ 2 & -4 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -3 \\ 2 \end{pmatrix}.$$

Answer:

- Set this up as an appended system

$$\left(\begin{array}{rrr|r} 2 & -4 & -2 & 4 \\ -2 & 4 & 1 & -3 \\ 2 & -4 & 0 & 2 \end{array}\right).$$

  Now, start applying Gaussian elimination (with row exchanges if necessary).

- Use the first row to eliminate the coefficients in the first column below the diagonal:

$$\left(\begin{array}{rrr|r} 2 & -4 & -2 & 4 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 2 & -2 \end{array}\right).$$

- There is now a zero on the diagonal and no row below it with which to exchange to put a nonzero there. So, we move on and use the second row to eliminate the coefficients in the third column below the second row:

$$\left(\begin{array}{rrr|r} 2 & -4 & -2 & 4 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

  This is good news: while we are now left with two equations with three unknowns, the last row translates to $0 = 0$. Hence, there is no contradication. There will be an infinite number of solutions.

- Subtract a multiple of the second row from the first row to eliminate the coefficient of the last term in the first row:

$$\left(\begin{array}{rrr|r} 2 & -4 & 0 & 2 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

- Divide the first and second row by appropriate values to make the first coefficient in the row (the pivot) equal to one:

$$\left(\begin{array}{rrr|r} 1 & -2 & 0 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

- We argued in this unit that the above really represents two appended systems, the second one corresponding to the case where the right-hand side is zero:

$$\left(\begin{array}{rrr|r} 1 & -2 & 0 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 \end{array}\right) \quad \left(\begin{array}{rrr|r} 1 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

- The rule for picking free variables is to look in each row for the first zero to the left of the pivot (as described above) in that row. If that zero is **not** under the pivot for the row above, than that column corresponds to a free variable. This leads us to choose χ_1 as the free variable.

- We translate the appended system on the left back into a linear system but with $\chi_1 = 0$ (since it can be chosen to equal any value and zero is convenient):

$$\begin{array}{rcrcrcr} \chi_0 & - & 2(0) & & & = & 1 \\ & & \chi_1 & & & = & 0 \\ & & & & \chi_2 & = & -1 \end{array}$$

  Solving this yields the specific solution $x_s = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$.

- We translate the appended system on the right back into a linear system but with $\chi_1 = 1$ (we can choose any value except for zero, and one is is convenient):

$$\begin{array}{rcrcrcr} \chi_0 & - & 2(1) & & & = & 0 \\ & & \chi_1 & & & = & 1 \\ & & & & \chi_2 & = & 0 \end{array}$$

  Solving this yields the solution $x_n = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}$.

- The general solution now is

$$x = x_s + \beta x_n = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} + \beta \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}.$$

The above seems like a very long-winded way of answering the question. In practice, here is what you will want to do:

1. Set the linear system up as an appended system

$$\left(\begin{array}{rrr|r} 2 & -4 & -2 & 4 \\ -2 & 4 & 1 & -3 \\ 2 & -4 & 0 & 2 \end{array}\right).$$

2. Check if you need to pivot (exchange) rows, identify the pivot, and eliminate the elements below the pivot:

$$\left(\begin{array}{rrr|r} \boxed{2} & -4 & -2 & 4 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 2 & -2 \end{array}\right).$$

3. Check if you need to pivot (exchange) rows, identify the pivot, and eliminate the elements below the pivot:

$$\left(\begin{array}{rrr|r} \boxed{2} & -4 & -2 & 4 \\ 0 & 0 & \boxed{-1} & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

4. Eliminate elements above the pivot in the second row:

$$\left(\begin{array}{rrr|r} \boxed{2} & -4 & 0 & 2 \\ 0 & 0 & \boxed{-1} & 1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

5. Divide to normalize the pivots to one:

$$\left(\begin{array}{rrr|r} \boxed{1} & -2 & 0 & 1 \\ 0 & 0 & \boxed{1} & -1 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

6. Identify the free variable(s) as corresponding to the column(s) in which there is no pivot. In this case, that is χ_1.

7. Set the free variables to zero and solve. This gives you the specific solution

$$x_s = \left(\begin{array}{r} 1 \\ 0 \\ -1 \end{array}\right).$$

8. Set the right-hand side to zero in the transformed system

$$\left(\begin{array}{rrr|r} \boxed{1} & -2 & 0 & 0 \\ 0 & 0 & \boxed{1} & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

9. Set the free variables one by one to one (and the others to zero). This gives you vectors that satisfy $Ax = 0$:

$$x_n = \left(\begin{array}{r} 2 \\ 1 \\ 0 \end{array}\right).$$

10. The general solution now is

$$x = x_s + \beta x_n = \left(\begin{array}{r} 1 \\ 0 \\ -1 \end{array}\right) + \beta \left(\begin{array}{r} 2 \\ 1 \\ 0 \end{array}\right).$$

(If there is more than one free variable, you will get more terms with vectors that satisfy $Ax = 0$. More on that later.)

☛ BACK TO TEXT

9.3 Review of Sets

9.3.3 Operations with Sets

Homework 9.3.3.1 Let S and T be two sets. Then $S \subset S \cup T$.

Always/Sometimes/Never

Answer: Always

When proving that one set is a subset of another set, you start by saying "Let $x \in S$" by which you mean "Let x be an arbitrary element in S". You then proceed to show that this arbitrary element is also an element of the other set.

Let $x \in S$. We will prove that $x \in S \cup T$.

$x \in S$
$\quad \Rightarrow \qquad < P \Rightarrow P \vee R >$
$x \in S \vee x \in T$
$\quad \Rightarrow \qquad < x \in S \cup T \Leftrightarrow x \in S \vee x \in T >$
$x \in S \cup T$
Hence $x \subset S \cup T$.

☛ BACK TO TEXT

Homework 9.3.3.2 Let S and T be two sets. Then $S \cap T \subset S$.

Always/Sometimes/Never

Answer: Always

Let $x \in S \cap T$. We will prove that $x \in S$.
$x \in S \cap T$
$\quad \Rightarrow \qquad < x \in S \cap T \Leftrightarrow x \in S \wedge x \in T >$
$x \in S \wedge x \in T$
$\quad \Rightarrow \qquad < P \wedge R \Rightarrow P >$
$x \in S$
Hence $S \cap T \subset S$.

☛ BACK TO TEXT

9.4 Vector Spaces

9.4.2 Subspaces

Homework 9.4.2.1 Which of the following subsets of $\mathbb{R}^3$ are subspaces of $\mathbb{R}^3$?

1. The plane of vectors $x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$ such that $\chi_0 = 0$. In other words, the set of all vectors $\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} 0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \right\}$.

   Answer: Yes.

   - 0 is in the set: Let $\chi_1 = 0$ and $\chi_2 = 0$.
   - $x + y$ is in the set: If x and y are in the set, then $x = \begin{pmatrix} 0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$ and $y = \begin{pmatrix} 0 \\ \psi_1 \\ \psi_2 \end{pmatrix}$. But then $x + y = \begin{pmatrix} 0 \\ \chi_1 + \psi_1 \\ \chi_2 + \psi_2 \end{pmatrix}$ is in the set.
   - αx is in the set: If x is in the set and $\alpha \in \mathbb{R}$, then $\alpha x = \begin{pmatrix} 0 \\ \alpha\chi_1 \\ \alpha\chi_2 \end{pmatrix}$ is in the set.

2. Similarly, the plane of vectors x with $\chi_0 = 1$: $\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} 1 \\ \chi_1 \\ \chi_2 \end{pmatrix} \right\}$.

   Answer: No. 0 is not in the set and hence this cannot be a subspace.

3. $\left\{ x \in \mathbb{R}^3 \left| x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \wedge \chi_0\chi_1 = 0 \right. \right\}$. (Recall, $\wedge$ is the logical "and" operator.)

Answer: No. $x+y$ is not in the set if $x = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ and $y = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$.

4. $\left\{ x \in \mathbb{R}^3 \left| x = \beta_0 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} \text{ where } \beta_0, \beta_1 \in \mathbb{R} \right. \right\}$.

Answer: Yes. Again, this is a matter of showing that if x and y are in the set and $\alpha \in \mathbb{R}$ then $x+y$ and αx are in the set.

5. $\left\{ x \in \mathbb{R}^3 \left| x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \wedge \chi_0 - \chi_1 + 3\chi_2 = 0 \right. \right\}$.

Answer: Yes. Once again, this is a matter of showing that if x and y are in the set and $\alpha \in \mathbb{R}$ then $x+y$ and αx are in the set.

☛ BACK TO TEXT

Homework 9.4.2.2 The empty set, $\emptyset$, is a subspace of $\mathbb{R}^n$.

True/False

Answer: False

0 (the zero vector) is not an element of $\emptyset$.

Notice that the other two conditions **are** met: "If $u, w \in \emptyset$ then $u + w \in \emptyset$" is *true* because $\emptyset$ is empty. Similarly "If $\alpha \in \mathbb{R}$ and $v \in \emptyset$ then $\alpha v \in \emptyset$" is *true* because $\emptyset$ is empty. This is kind of subtle.

☛ BACK TO TEXT

Homework 9.4.2.3 The set $\{0\}$ where 0 is a vector of size n is a subspace of $\mathbb{R}^n$.

True/False

Answer: True

- 0 (the zero vector) is an element of $\{0\}$.
- If $u, w \in \{0\}$ then $(u + w) \in \{0\}$: this is *true* because if $u, w \in \{0\}$ then $v = w = 0$ and $v + w = 0 + 0 = 0$ is an element of $\{0\}$.
- If $\alpha \in \mathbb{R}$ and $v \in \{0\}$ then $\alpha v \in \{0\}$: this is *true* because if $v \in \{0\}$ then $v = 0$ and for any $\alpha \in \mathbb{R}$ it is the case that $\alpha v = \alpha 0 = 0$ is an element of $\{0\}$.

☛ BACK TO TEXT

Homework 9.4.2.4 The set $S \subset \mathbb{R}^n$ described by

$$\{x \mid \|x\|_2 < 1\}.$$

is a subspace of $\mathbb{R}^n$. (Recall that $\|x\|_2$ is the Euclidean length of vector x so this describes all elements with length less than or equal to one.)

True/False

Answer: False

Pick any vector $v \in S$ such that $v \neq 0$. Let $\alpha > 1/\|v\|_2$. Then

$$\|\alpha v\|_2 = \alpha\|v\|_2 > (1/\|v\|_2)\|v\|_2 = 1$$

and hence $\alpha v \notin S$.

☛ BACK TO TEXT

Homework 9.4.2.5 The set $S \subset \mathbb{R}^n$ described by

$$\left\{ \left(\begin{array}{c} \nu_0 \\ 0 \\ \vdots \\ 0 \end{array} \right) \middle| \, \nu_0 \in \mathbb{R} \right\}$$

is a subspace of $\mathbb{R}^n$.

True/False

Answer: True

- $0 \in S$: (pick $\nu_0 = 0$).
- If $u, w \in S$ then $(u + w) \in S$: Pick $u, w \in S$. Then for some ν_0 and some ω_0

$$v = \left(\begin{array}{c} \nu_0 \\ 0 \\ \vdots \\ 0 \end{array} \right) \quad \text{and} \quad w = \left(\begin{array}{c} \omega_0 \\ 0 \\ \vdots \\ 0 \end{array} \right).$$

  But then $v + w = \left(\begin{array}{c} \nu_0 \\ 0 \\ \vdots \\ 0 \end{array} \right) + \left(\begin{array}{c} \omega_0 \\ 0 \\ \vdots \\ 0 \end{array} \right) = \left(\begin{array}{c} \nu_0 + \omega_0 \\ 0 \\ \vdots \\ 0 \end{array} \right)$, which is also in S.
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$: Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some ν_0

$$v = \left(\begin{array}{c} \nu_0 \\ 0 \\ \vdots \\ 0 \end{array} \right).$$

  But then $\alpha v = \left(\begin{array}{c} \alpha\nu_0 \\ 0 \\ \vdots \\ 0 \end{array} \right)$, which is also in S.

☛ BACK TO TEXT

Homework 9.4.2.6 The set $S \subset \mathbb{R}^n$ described by
$$\{\nu e_j \mid \nu \in \mathbb{R}\},$$
where e_j is a unit basis vector, is a subspace.

True/False

Answer: True

- $0 \in S$: (pick $\nu = 0$).
- If $u, w \in S$ then $(u+w) \in S$: Pick $u, w \in S$. Then for some ν and some ω, $v = \nu e_j$ and $w = \omega e_j$. But then $v + w = \nu e_j + \omega e_j = (\nu + \omega) e_j$, which is also in S.
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$: Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some ν, $v = \nu e_j$. But then $\alpha v = \alpha(\nu e_j) = (\alpha\nu) e_j$, which is also in S.

☛ BACK TO TEXT

Homework 9.4.2.7 The set $S \subset \mathbb{R}^n$ described by
$$\{\chi a \mid \chi \in \mathbb{R}\},$$
where $a \in \mathbb{R}^n$, is a subspace.

True/False

Answer: True

- $0 \in S$: (pick $\chi = 0$).
- If $u, w \in S$ then $(u+w) \in S$: Pick $u, w \in S$. Then for some ν and some ω, $v = \nu a$ and $w = \omega a$. But then $v + w = \nu a + \omega a = (\nu + \omega) a$, which is also in S.
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$: Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some ν, $v = \nu a$. But then $\alpha v = \alpha(\nu a) = (\alpha\nu) a$, which is also in S.

☛ BACK TO TEXT

Homework 9.4.2.8 The set $S \subset \mathbb{R}^n$ described by
$$\{\chi_0 a_0 + \chi_1 a_1 \mid \chi_0, \chi_1 \in \mathbb{R}\},$$
where $a_0, a_1 \in \mathbb{R}^n$, is a subspace.

True/False

Answer: True

- $0 \in S$: (pick $\chi_0 = \chi_1 = 0$).
- If $u, w \in S$ then $(u+w) \in S$: Pick $u, w \in S$. Then for some $\nu_0, \nu_1, \omega_0, \omega_1 \in \mathbb{R}$, $v = \nu_0 a_0 + \nu_1 a_1$ and $w = \omega_0 a_0 + \omega_1 a_1$. But then $v + w = \nu_0 a_0 + \nu_1 a_1 + \omega_0 a_0 + \omega_1 a_1 = (\nu_0 + \omega_0) a_0 + (\nu_1 + \omega_1) a_1$, which is also in S.
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$: Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some $\nu_0, \nu_1 \in \mathbb{R}$, $v = \nu_0 a_0 + \nu_1 a_1$. But then $\alpha v = \alpha(\nu_0 a_0 + \nu_1 a_1) = (\alpha\nu_0) a_0 + (\alpha\nu_1) a_1$, which is also in S.

What this means is that the set of all linear combinations of two vectors is a subspace.

☛ BACK TO TEXT

Homework 9.4.2.9 The set $S \subset \mathbb{R}^m$ described by

$$\left\{ \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \middle| \chi_0, \chi_1 \in \mathbb{R} \right\},$$

where $a_0, a_1 \in \mathbb{R}^m$, is a subspace.

True/False

Answer: True

- $0 \in S$: (pick $\chi_0 = \chi_1 = 0$).
- If $v, w \in S$ then $(v+w) \in S$: Pick $v, w \in S$. Then for some $\nu_0, \nu_1, \omega_0 \omega_1 \in \mathbb{R}$,

$$v = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \nu_0 \\ \nu_1 \end{pmatrix} \quad \text{and} \quad w = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \omega_0 \\ \omega_1 \end{pmatrix}.$$

But then

$$\begin{aligned} v + w &= \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \nu_0 \\ \nu_1 \end{pmatrix} + \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \omega_0 \\ \omega_1 \end{pmatrix} \\ &= \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \left(\begin{pmatrix} \nu_0 \\ \nu_1 \end{pmatrix} + \begin{pmatrix} \omega_0 \\ \omega_1 \end{pmatrix} \right) \\ &= \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \nu_0 + \omega_0 \\ \nu_1 + \omega_1 \end{pmatrix}, \end{aligned}$$

which is also in S.

- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$: Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some $\nu_0, \upsilon_1 \in \mathbb{R}$, $v = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \upsilon_0 \\ \upsilon_1 \end{pmatrix}$. But then

$$\alpha v = \alpha \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \upsilon_0 \\ \upsilon_1 \end{pmatrix} = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \alpha \begin{pmatrix} \upsilon_0 \\ \upsilon_1 \end{pmatrix} = \left(\begin{array}{c|c} a_0 & a_1 \end{array} \right) \begin{pmatrix} \alpha \upsilon_0 \\ \alpha \upsilon_1 \end{pmatrix},$$

which is also in S.

What this means is that the set of all linear combinations of two vectors is a subspace, expressed as a matrix-vector multiplication (with a matrix consisting of two columns in this case). In other words, this exercise is simply a restatement of the previous exercise. We are going somewhere with this!

☛ BACK TO TEXT

Homework 9.4.2.10 The set $S \subset \mathbb{R}^m$ described by

$$\left\{ Ax \mid x \in \mathbb{R}^2 \right\},$$

where $A \in \mathbb{R}^{m \times 2}$, is a subspace.

True/False

Answer: True

- $0 \in S$: (pick $x = 0$).

- Now here we need to use different letters for x and y, since x is already being used. If $v, w \in S$ then $(v+w) \in S$: Pick $v, w \in S$. Then for some $x, y \in \mathbb{R}^2$, $v = Ax$ and $w = Ay$. But then $v + w = Ax + Ay = A(x + y)$, which is also in S.

- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$: Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some $x \in \mathbb{R}^2$, $v = Ax$. But then $\alpha v = \alpha(Ax) = A(\alpha x)$, which is also in S since $\alpha x \in \mathbb{R}^2$.

What this means is that the set of all linear combinations of two vectors is a subspace, except expressed even more explicitly as a matrix-vector multiplication. In other words, this exercise is simply a restatement of the previous two exercises. Now we are getting somewhere!

☛ BACK TO TEXT

9.4.3 The Column Space

Homework 9.4.3.1 The set $S \subset \mathbb{R}^m$ described by

$$\{Ax \mid x \in \mathbb{R}^n\},$$

where $A \in \mathbb{R}^{m \times n}$, is a subspace.

True/False

Answer: True

- $0 \in S$: (pick $x = 0$).

- Now here we need to use different letters for x and y, since x is already being used. If $v, w \in S$ then $(v+w) \in S$: Pick $v, w \in S$. Then for some $x, y \in \mathbb{R}^n$, $v = Ax$ and $w = Ay$. But then $v + w = Ax + Ay = A(x + y)$, which is also in S.

- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$: Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some $x \in \mathbb{R}^n$, $v = Ax$. But then $\alpha v = \alpha(Ax) = A(\alpha x)$, which is also in S since $\alpha x \in \mathbb{R}^n$.

☛ BACK TO TEXT

Homework 9.4.3.2 Match the matrices on the left to the column space on the right. (You should be able to do this “by examination.”)

1. $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ **Answer:** f.

2. $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ **Answer:** c.

3. $\begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}$ **Answer:** c.

4. $\begin{pmatrix} 0 & 0 \\ 1 & -2 \end{pmatrix}$ **Answer:** d.

5. $\begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}$ **Answer:** a.

6. $\begin{pmatrix} 1 & 0 \\ 2 & 3 \end{pmatrix}$ **Answer:** a.

7. $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$ **Answer:** e.

8. $\begin{pmatrix} 1 & -2 \\ 2 & -4 \end{pmatrix}$ **Answer:** e.

9. $\begin{pmatrix} 1 & -2 & -1 \\ 2 & -4 & -2 \end{pmatrix}$ **Answer:** e.

a. $\mathbb{R}^2$.

b. $\left\{ \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \middle| \chi_0 = 0 \vee \chi_1 = 0 \right\}$

c. $\left\{ \begin{pmatrix} \alpha \\ 0 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

d. $\left\{ \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

e. $\left\{ \alpha \begin{pmatrix} 1 \\ 2 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

f. $\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix} \right\}$

(Recall that $\vee$ is the logical "or" operator.)

☛ BACK TO TEXT

Homework 9.4.3.3 Which of the following matrices have a FINITE number of elements in their column space? (Mark all that apply.)

1. The identity matrix.
2. The zero matrix.
3. All matrices.
4. None of the above.

Answer: Only the zero matrix has a finte number of elements in its column space. Indeed, the number of elements in its column space is one: the zero vector.

☛ BACK TO TEXT

9.4.4 The Null Space

Homework 9.4.4.1 Let $A \in \mathbb{R}^{m\times n}$. The null space of A, $\mathcal{N}(A)$, is a subspace

True/False

Answer: True

- $0 \in \mathcal{N}(A)$: $A0 = 0$.
- If $x, y \in \mathcal{N}(A)$ then $x + y \in \mathcal{N}(A)$: Let $x, y \in \mathcal{N}(A)$ so that $Ax = 0$ and $Ay = 0$. Then $A(x+y) = Ax + Ay = 0 + 0 = 0$ which means that $x + y \in \mathcal{N}(A)$.
- If $\alpha \in \mathbb{R}$ and $x \in \mathcal{N}(A)$ then $\alpha x \in \mathcal{N}(A)$: Let $\alpha \in \mathbb{R}$ and $x \in \mathcal{N}(A)$ so that $Ax = 0$. Then $A(\alpha x) = A\alpha x = \alpha A x = \alpha 0 = 0$ which means that $\alpha x \in \mathcal{N}(A)$.

Hence $\mathcal{N}(A)$ is a subspace.

☛ BACK TO TEXT

Homework 9.4.4.2 For each of the matrices on the left match the set of vectors on the right that describes its null space. (You should be able to do this "by examination.")

1. $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$
2. $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$
3. $\begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}$
4. $\begin{pmatrix} 0 & 0 \\ 1 & -2 \end{pmatrix}$
5. $\begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}$
6. $\begin{pmatrix} 1 & 0 \\ 2 & 3 \end{pmatrix}$
7. $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$
8. $\begin{pmatrix} 1 & -2 \\ 2 & -4 \end{pmatrix}$

a. $\mathbb{R}^2$.

b. $\left\{ \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \middle| \chi_0 = 0 \vee \chi_1 = 0 \right\}$

c. $\left\{ \begin{pmatrix} \alpha \\ 0 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

d. $\emptyset$

e. $\left\{ \begin{pmatrix} 0 \\ \alpha \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

f. $\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix} \right\}$

g. $\left\{ \begin{pmatrix} 0 \end{pmatrix} \right\}$

h. $\left\{ \alpha \begin{pmatrix} 1 \\ 2 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

i. $\left\{ \alpha \begin{pmatrix} 2 \\ 1 \end{pmatrix} \middle| \alpha \in \mathbb{R} \right\}$

(Recall that $\vee$ is the logical "or" operator.)

Answer:

1. $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ **Answer:** (a) Any vector in $\mathbb{R}^2$ maps to the zero vector.

2. $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ **Answer:** (c) $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ means $\chi_1 = 0$ with no restriction on χ_0.

3. $\begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}$ **Answer:** (c) $\begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ means $-2\chi_1 = 0$ with no restriction on χ_0.

4. $\begin{pmatrix} 0 & 0 \\ 1 & -2 \end{pmatrix}$ **Answer:** (i) $\begin{pmatrix} 0 & 0 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ means $\chi_0 + -2\chi_1 = 0$. (Only) vectors of form $\alpha\begin{pmatrix} 2 \\ 1 \end{pmatrix}$ satisfy this.

5. $\begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}$ **Answer:** (f) $\begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ means $\chi_1 = 0$ and $2\chi_0 = 0$. Only the zero vector satisfies this.

6. $\begin{pmatrix} 1 & 0 \\ 2 & 3 \end{pmatrix}$ **Answer:** (f) $\begin{pmatrix} 1 & 0 \\ 2 & 3 \end{pmatrix}\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ means $\chi_0 = 0$ and $2\chi_0 + 3\chi_1 = 0$. Only the zero vector satisfies this.

7. $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$ **Answer:** (g) $\begin{pmatrix} 1 \\ 2 \end{pmatrix}\begin{pmatrix} \chi \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ means $\chi_0 = 0$ and $2\chi_0 = 0$. Only $\chi_0 = 0$ satisfies this.

8. $\begin{pmatrix} 1 & -2 \\ 2 & -4 \end{pmatrix}$ **Answer:** (i) $\begin{pmatrix} 1 & -2 \\ 2 & -4 \end{pmatrix}\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$. By examination, $\begin{pmatrix} 2 \\ 1 \end{pmatrix}$ satisfies this, as does any multiple of that vectors.

☛ BACK TO TEXT

9.5 Span, Linear Independence, and Bases

9.5.2 Linear Independence

Homework 9.5.2.1

$$\text{Span}\left(\left\{\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right\}\right) = \text{Span}\left(\left\{\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 3 \end{pmatrix}\right\}\right)$$

True/False

Answer: True

- $S \subset T$: Let $x \in \text{Span}\left(\left\{\begin{pmatrix}1\\0\\1\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}\right\}\right)$ Then there exist α_0 and α_1 such that

  $x = \alpha_0 \begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\0\\1\end{pmatrix}$. This in turn means that

$$x = \alpha_0 \begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\0\\1\end{pmatrix} + (0) \begin{pmatrix}1\\0\\3\end{pmatrix}.$$

  Hence $x \in \text{Span}\left(\left\{\begin{pmatrix}1\\0\\1\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}1\\0\\3\end{pmatrix}\right\}\right)$.

- $T \subset S$: Let $x \in \text{Span}\left(\left\{\begin{pmatrix}1\\0\\1\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}1\\0\\3\end{pmatrix}\right\}\right)$ Then there exist α_0, α_1, and α_2 such that

  $x = \alpha_0 \begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\0\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}1\\0\\3\end{pmatrix}$. But $\begin{pmatrix}1\\0\\3\end{pmatrix} = \begin{pmatrix}1\\0\\1\end{pmatrix} + 2 \begin{pmatrix}0\\0\\1\end{pmatrix}$. Hence

$$x = \alpha_0 \begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_1 \begin{pmatrix}0\\0\\1\end{pmatrix} + \alpha_2 \left(\begin{pmatrix}1\\0\\1\end{pmatrix} + 2 \begin{pmatrix}0\\0\\1\end{pmatrix}\right) = (\alpha_0 + \alpha_2) \begin{pmatrix}1\\0\\1\end{pmatrix} + (\alpha_1 + 2\alpha_2) \begin{pmatrix}0\\0\\1\end{pmatrix}.$$

  Therefore $x \in \text{Span}\left(\begin{pmatrix}1\\0\\1\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}\right)$.

☛ BACK TO TEXT

Homework 9.5.2.2 Let the set of vectors $\{a_0, a_1, \ldots, a_{n-1}\} \subset \mathbb{R}^m$ be linearly dependent. Then at least one of these vectors can be written as a linear combination of the others.

True/False

Answer: True

Since the vectors are linearly dependent, then there must exist $\chi_0, \chi_1, \ldots, \chi_{n-1} \in \mathbb{R}$ such that $\chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} = 0$ and for at least one j, $0 \leq j < n$, $\chi_j \neq 0$. But then

$$\chi_j a_j = -\chi_0 a_0 + -\chi_1 a_1 - \cdots - \chi_{j-1} a_{j-1} - \chi_{j+1} a_{j+1} - \cdots - \chi_{n-1} a_{n-1}$$

and therefore

$$a_j = -\frac{\chi_0}{\chi_j} a_0 + -\frac{\chi_1}{\chi_j} a_1 - \cdots - \frac{\chi_{j-1}}{\chi_j} a_{j-1} - \frac{\chi_{j+1}}{\chi_j} a_{j+1} - \cdots - \frac{\chi_{n-1}}{\chi_j} a_{n-1}.$$

In other words, a_j can be written as a linear combination of the other $n-1$ vectors.

☛ BACK TO TEXT

Homework 9.5.2.3 Let $U \in \mathbb{R}^{n \times n}$ be an upper triangular matrix with nonzeroes on its diagonal. Then its columns are linearly independent.

Always/Sometimes/Never

Answer: Always

We saw in a previous week that $Ux = b$ has a unique solution if U is upper triangular with nonzeroes on its diagonal. Hence $Ux = 0$ has the unique solution $x = 0$ (the zero vector). This implies that U has linearly independent columns.

☛ BACK TO TEXT

Homework 9.5.2.4 Let $L \in \mathbb{R}^{n \times n}$ be a lower triangular matrix with nonzeroes on its diagonal. Then its rows are linearly independent. (Hint: How do the rows of L relate to the columns of L^T?)

Always/Sometimes/Never

Answer: The rows of L are linearly independent if and only if the columns of L^T are linearly independent. But L^T is an upper triangular matrix. Since now the diagonal elements of L^T are all nonzero, L^T has linearly independent columns and hence L has linearly independent rows.

☛ BACK TO TEXT

9.5.3 Bases for Subspaces

Homework 9.5.3.1 The vectors $\{e_0, e_1, \ldots, e_{n-1}\} \subset \mathbb{R}^n$ are a basis for $\mathbb{R}^n$.

True/False

Answer: True

Clearly, $\text{Span}(e_0, e_1, \ldots, e_{n-1}) = \mathbb{R}^n$. Now, the identity $I = \left(\begin{array}{c|c|c|c} e_0 & e_1 & \cdots & e_{n-1} \end{array} \right)$. Clearly, $Ix = 0$ only has the solution $x = 0$. Hence the columns of I are linearly independent which means the vectors $\{e_0, e_1, \ldots, e_{n-1}\}$ are linearly independent.

☛ BACK TO TEXT

Week 10: Vector Spaces, Orthogonality, and Linear Least Squares (Answers)

10.1 Opening Remarks

10.1.1 Visualizing Planes, Lines, and Solutions

Homework 10.1.1.1 Consider, again, the equation from the last example:

$$\chi_0 \quad - \quad 2\chi_1 \quad + \quad 4\chi_2 \quad = \quad -1$$

Which of the following represent(s) a general solution to this equation? (Mark all)

- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$.
- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta_0 \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$.
- $\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -5 \\ 0 \\ 1 \end{pmatrix} + \beta_0 \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$.

☛ BACK TO TEXT

Homework 10.1.1.2 Now you find the general solution for the **second** equation in the system of linear equations with which we started this unit. Consider

$$\chi_0 \qquad\qquad = \quad 2$$

Which of the following is a true statement about this equation:

- $\begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta_0 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.

- $\begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}$ is a general solution.

Answer: All are correct.

We can write this as an appended system:

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 2 \end{array}\right).$$

Now, we would perform Gaussian or Gauss-Jordan elimination with this, except that there really isn't anything to do, other than to identify the pivot, the free variables, and the dependent variables:

$$\begin{array}{c} \left(\begin{array}{ccc|c} \boxed{1} & 0 & 0 & 2 \end{array}\right). \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ \text{dependent variable} & \text{free variable} & \text{free variable} \end{array} \end{array}$$

Here the pivot is highlighted with the box. There are two free variables, χ_1 and χ_2, and there is one dependent variable, χ_0. To find a specific solution, we can set χ_1 and χ_2 to any value, and solve for χ_0. Setting $\chi_1 = \chi_2 = 0$ is particularly convenient, leaving us with $\chi_0 = 2$, so that the specific solution is given by

$$x_s = \begin{pmatrix} \boxed{2} \\ 0 \\ 0 \end{pmatrix}.$$

To find solutions in the null space, we look for solutions of $\left(\begin{array}{ccc|c} \boxed{1} & 0 & 0 & 0 \end{array}\right)$ in the form

$$x_{n_0} = \begin{pmatrix} \boxed{\chi_0} \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad x_{n_1} = \begin{pmatrix} \boxed{\chi_0} \\ 0 \\ 1 \end{pmatrix}$$

which yields the vectors

$$x_{n_0} = \begin{pmatrix} \boxed{0} \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad x_{n_1} = \begin{pmatrix} \boxed{0} \\ 0 \\ 1 \end{pmatrix}.$$

This then gives us the general solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = x_s + \beta_0 x_{n_0} + \beta_1 x_{n_1} = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 10.1.1.3 Now you find the general solution for the **third** equation in the system of linear equations with which we started this unit. Consider

$$\chi_0 \; + \; 2\chi_1 \; + \; 4\chi_2 \; = \; 3$$

Which of the following is a true statement about this equation:

- $\left(\begin{array}{c} 3 \\ 0 \\ 0 \end{array}\right)$ is a specific solution.

- $\left(\begin{array}{c} 2 \\ 1 \\ -0.25 \end{array}\right)$ is a specific solution.

- $\left(\begin{array}{c} 3 \\ 0 \\ 0 \end{array}\right) + \beta_0 \left(\begin{array}{c} -2 \\ 1 \\ 0 \end{array}\right) + \beta_1 \left(\begin{array}{c} -4 \\ 0 \\ 1 \end{array}\right)$ is a general solution.

- $\left(\begin{array}{c} 2 \\ 1 \\ -0.25 \end{array}\right) + \beta_0 \left(\begin{array}{c} -2 \\ 1 \\ 0 \end{array}\right) + \beta_1 \left(\begin{array}{c} -4 \\ 0 \\ 1 \end{array}\right)$ is a general solution.

- $\left(\begin{array}{c} 3 \\ 0 \\ 0 \end{array}\right) + \beta_0 \left(\begin{array}{c} -4 \\ 2 \\ 0 \end{array}\right) + \beta_1 \left(\begin{array}{c} -4 \\ 0 \\ 1 \end{array}\right)$ is a general solution.

Answer: All are correct.

We can write this as an appended system:

$$\left(\begin{array}{ccc|c} 1 & 2 & 4 & 3 \end{array}\right).$$

Now, we would perform Gaussian or Gauss-Jordan elimination with this, except that there really isn't anything to do, other than to identify the pivot, the free variables, and the dependent variables:

$$\begin{array}{c} \left(\begin{array}{ccc|c} \boxed{1} & 2 & 4 & 3 \end{array}\right). \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ \text{dependent variable} & \text{free variable} & \text{free variable} \end{array} \end{array}$$

Here the pivot is highlighted with the box. There are two free variables, χ_1 and χ_2, and there is one dependent variable, χ_0. To find a specific solution, we can set χ_1 and χ_2 to any value, and solve for χ_0. Setting $\chi_1 = \chi_2 = 0$ is particularly convenient, leaving us with $\chi_0 = 3$, so that the specific solution is given by

$$x_s = \left(\begin{array}{c} \boxed{3} \\ 0 \\ 0 \end{array}\right).$$

To find solutions in the null space, we look for solutions of $\left(\begin{array}{ccc|c} \boxed{1} & 2 & 4 & 0 \end{array}\right)$ in the form

$$x_{n_0} = \left(\begin{array}{c} \boxed{\chi_0} \\ 1 \\ 0 \end{array}\right) \quad \text{and} \quad x_{n_1} = \left(\begin{array}{c} \boxed{\chi_0} \\ 0 \\ 1 \end{array}\right)$$

which yields the vectors

$$x_{n_0} = \begin{pmatrix} \boxed{\text{-2}} \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad x_{n_1} = \begin{pmatrix} \boxed{\text{-4}} \\ 0 \\ 1 \end{pmatrix}.$$

This then gives us the general solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = x_s + \beta_0 x_{n_0} + \beta_1 x_{n_1} = \begin{pmatrix} 3 \\ 0 \\ 0 \end{pmatrix} + \beta_0 \begin{pmatrix} -2 \\ 1 \\ 0 \end{pmatrix} + \beta_1 \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 10.1.1.4 We notice that it would be nice to put lines where planes meet. Now, let's start by focusing on the first two equations: Consider

$$\begin{array}{ccccccc} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \\ \chi_0 & & & & & = & 2 \end{array}$$

Compute the general solution of this system with two equations in three unknowns and indicate which of the following is true about this system?

- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 3/2 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 3/2 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}$ is a general solution.

Answer: All are correct.

We can write this as an appended system:

$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & -1 \\ 1 & & & 2 \end{array}\right).$$

Now, perform Gaussian elimination with this to yield

$$\begin{array}{c} \left(\begin{array}{ccc|c} \boxed{1} & -2 & 4 & -1 \\ & \boxed{2} & -4 & 3 \end{array}\right). \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ \text{dependent variable} & \text{dependent variable} & \text{free variable} \end{array} \end{array}$$

Here the pivots are highlighted with the box. There is one free variable, χ_2, and there are two dependent variables, χ_0 and χ_1. To find a specific solution, we set $\chi_2 = 0$ leaving us with the specific solution is given by

$$x_s = \begin{pmatrix} \boxed{2} \\ \boxed{3/2} \\ 0 \end{pmatrix}.$$

The solution in the null space (setting $\chi_2 = 1$) is

$$x_n = \begin{pmatrix} \boxed{0} \\ \boxed{2} \\ 1 \end{pmatrix}.$$

This then gives us the general solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = x_s + \beta x_n = \begin{pmatrix} 2 \\ 3/2 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 10.1.1.5 Similarly, consider

$$\begin{array}{ccccccc} \chi_0 & & & & & = & 2 \\ \chi_0 & + & 2\chi_1 & + & 4\chi_2 & = & 3 \end{array}$$

Compute the general solution of this system that has two equations with three unknowns and indicate which of the following is true about this system?

- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 1/2 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 2 \\ 1/2 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ -2 \\ 1 \end{pmatrix}$ is a general solution.

- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ -2 \\ 1 \end{pmatrix}$ is a general solution.

Answer: All are correct.

We can write this as an appended system:

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 2 \\ 1 & 2 & 4 & 3 \end{array}\right).$$

Now, perform Gauss-Jordan elimination with this to yield

$$\begin{array}{c} \left(\begin{array}{ccc|c} \boxed{1} & & 0 & 2 \\ & \boxed{2} & 4 & 1 \end{array}\right). \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ \text{dependent variable} & \text{dependent variable} & \text{free variable} \end{array} \end{array}$$

Here the pivots are highlighted by boxes. There is one free variable, χ_2, and there are two dependent variables, χ_0 and χ_1. To find a specific solution, we set $\chi_2 = 0$ leaving us with the specific solution is given by

$$x_s = \begin{pmatrix} \boxed{2} \\ \boxed{1/2} \\ 0 \end{pmatrix}.$$

The solution in the null space (setting $\chi_2 = 1$) is

$$x_n = \begin{pmatrix} \boxed{0} \\ \boxed{-2} \\ 1 \end{pmatrix}.$$

This then gives us the general solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = x_s + \beta x_n = \begin{pmatrix} 2 \\ 1/2 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ -2 \\ 1 \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 10.1.1.6 Finally consider

$$\begin{array}{rcrcrcr} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \\ \chi_0 & + & 2\chi_1 & + & 4\chi_2 & = & 3 \end{array}$$

Compute the general solution of this system with two equations in three unknowns and indicate which of the following is true about this system? UPDATE

- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$ is a specific solution.

- $\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$ is a specific solution.
- $\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.
- $\begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix} + \beta \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$ is a general solution.

Answer: All are correct.

We can write this as an appended system:

$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & -1 \\ 1 & 2 & 4 & 3 \end{array}\right).$$

Now, perform Gauss-Jordan elimination with this to yield

$$\begin{array}{c} \left(\begin{array}{ccc|c} \boxed{1} & -2 & 4 & -1 \\ & \boxed{4} & 0 & 4 \end{array}\right). \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ \text{dependent variable} & \text{dependent variable} & \text{free variable} \end{array} \end{array}$$

Here the pivots are highlighted by boxes. There is one free variable, χ_2, and there are two dependent variables, χ_0 and χ_1. To find a specific solution, we set $\chi_2 = 0$ leaving us with the specific solution is given by

$$x_s = \begin{pmatrix} \boxed{1} \\ \boxed{1} \\ 0 \end{pmatrix}.$$

The solution in the null space (setting $\chi_2 = 1$) is

$$x_n = \begin{pmatrix} \boxed{-4} \\ 0 \\ 1 \end{pmatrix}.$$

This then gives us the general solution

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = x_s + \beta x_n = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}.$$

☛ BACK TO TEXT

10.2 How the Row Echelon Form Answers (Almost) Everything

10.2.1 Example

Homework 10.2.1.1 Consider the linear system of equations

$$\underbrace{\begin{pmatrix} 1 & 3 & 1 & 2 \\ 2 & 6 & 4 & 8 \\ 0 & 0 & 2 & 4 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}}_{b}.$$

Write it as an appended system and reduce it to to row echelon form (but not reduced row echelon form). Identify the pivots, the free variables and the dependent variables. **Answer:**

$$\left(\begin{array}{cccc|c} 1 & 3 & 1 & 2 & 1 \\ 2 & 6 & 4 & 8 & 3 \\ 0 & 0 & 2 & 4 & 1 \end{array}\right) \rightarrow \left(\begin{array}{cccc|c} \boxed{1} & 3 & 1 & 2 & 1 \\ 0 & 0 & \boxed{2} & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

The pivots are highlighted. The free variables are χ_1 and χ_3 and the dependent variables are χ_0 and χ_2.

☛ BACK TO TEXT

10.2.2 The Important Attributes of a Linear System

Homework 10.2.2.1 Consider $\begin{pmatrix} 1 & 2 & 2 \\ 2 & 4 & 5 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \end{pmatrix}$.

- Reduce the system to row echelon form (but not reduced row echelon form).

  Answer:

$$\left(\begin{array}{ccc|c} \boxed{1} & 2 & 2 & 1 \\ 0 & 0 & \boxed{1} & 2 \end{array}\right)$$

- Identify the free variables.

  Answer: χ_1 is the only free variable.

- Identify the dependent variables.

  Answer: χ_0 and χ_2 are the dependent variables.

- What is the dimension of the column space? 2

- What is the dimension of the row space? 2

- What is the dimension of the null space? 1

- Give a set of linearly independent vectors that span the column space

  Answer:

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 2 \\ 5 \end{pmatrix},$$

- Give a set of linearly independent vectors that span the row space.

  Answer:

$$\begin{pmatrix} 1 \\ 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

- What is the rank of the matrix? 2
- Give a general solution.

  Answer:

$$\begin{pmatrix} \boxed{-3} \\ 0 \\ \boxed{2} \end{pmatrix} + \alpha \begin{pmatrix} \boxed{-2} \\ 1 \\ \boxed{0} \end{pmatrix}.$$

☛ BACK TO TEXT

Homework 10.2.2.2 Consider $Ax = b$ where

$$A = \begin{pmatrix} 0 & 1 & 0 & 3 \\ 0 & 2 & 0 & 6 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}.$$

- Compute the row echelon form of this system of linear equations.

  Answer:

$$\left(\begin{array}{cccc|c} 0 & 1 & 0 & 3 & \beta_0 \\ 0 & 0 & 0 & 0 & \beta_1 - 2\beta_0 \end{array}\right)$$

- When does the equation have a solution?

  Answer: When $\beta_1 - 2\beta_0 = 0$.

☛ BACK TO TEXT

Homework 10.2.2.3 Which of these statements is a correct definition of the rank of a given matrix $A \in \mathbb{R}^{m \times n}$?

1. The number of nonzero rows in the reduced row echelon form of A. **True/False** **Answer:** True
2. The number of columns minus the number of rows, $n - m$. **True/False** **Answer:** False
3. The number of columns minus the number of free columns in the row reduced form of A. (Note: a free column is a column that does not contain a pivot.) **Answer:** True **True/False**

4. The number of 1s in the row reduced form of A. **True/False** **Answer:** False

☛ BACK TO TEXT

Homework 10.2.2.4 Compute

$$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} \begin{pmatrix} 3 & -1 & 2 \end{pmatrix}.$$

Reduce it to row echelon form. What is the rank of this matrix?

Answer:

$$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} \begin{pmatrix} 3 & -1 & 2 \end{pmatrix} = \begin{pmatrix} 3 & -1 & 2 \\ 6 & -2 & 4 \\ 9 & -3 & 6 \end{pmatrix}.$$

Row echelon form:

$$\begin{pmatrix} \boxed{3} & -1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Rank: 1.

☛ BACK TO TEXT

10.3 Orthogonal Vectors and Spaces

10.3.1 Orthogonal Vectors

Homework 10.3.1.1 For each of the following, indicate whether the vectors are orthogonal:

$\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$	True/False
$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$	True/False
The unit basis vectors e_i and e_j.	Always/Sometimes/Never
$\begin{pmatrix} c \\ s \end{pmatrix}$ and $\begin{pmatrix} -s \\ c \end{pmatrix}$	Always/Sometimes/Never

Answer:

$\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$	True because $\begin{pmatrix} 1 \\ -1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 0$
$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$	True because $\begin{pmatrix} 1 \\ 0 \end{pmatrix}^T \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 0$
The unit basis vectors e_i and e_j.	Sometimes because $e_i^T e_j = 0$ if $i \neq j$ but $e_i^T e_j = 1$ if $i = j$.
$\begin{pmatrix} c \\ s \end{pmatrix}$ and $\begin{pmatrix} -s \\ c \end{pmatrix}$	Always because $\begin{pmatrix} c \\ s \end{pmatrix}^T \begin{pmatrix} -s \\ c \end{pmatrix} = 0$

☛ BACK TO TEXT

Homework 10.3.1.2 Let $A \in \mathbb{R}^{m \times n}$. Let a_i^T be a row of A and $x \in \mathcal{N}(A)$. Then a_i is orthogonal to x.

Always/Sometimes/Never

Answer: Always Since $x \in \mathcal{N}(A)$, $Ax = 0$. But then, partitioning A by rows,

$$0 = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = Ax = \begin{pmatrix} a_0^T \\ a_1^T \\ \vdots \\ a_{m-1}^T \end{pmatrix} x = \begin{pmatrix} a_0^T x \\ a_1^T x \\ \vdots \\ a_{m-1}^T x \end{pmatrix}.$$

This means that $a_i^T x = 0$ for all $0 \leq i < m$.

☛ BACK TO TEXT

10.3.2 Orthogonal Spaces

Homework 10.3.2.1 Let $\mathbf{V} = \{0\}$ where 0 denotes the zero vector of size n. Then $\mathbf{V} \perp \mathbb{R}^n$.

Always/Sometimes/Never

Answer: Always Let $x \in \mathbf{V}$ and $y \in \mathbb{R}^n$. Then $x = 0$ since that is the only element in set (subspace) $\mathbf{V}$. Hence $x^T y = 0^T y = 0$ and therefore x and y are orthogonal.

☛ BACK TO TEXT

Homework 10.3.2.2 Let

$$\mathbf{V} = \text{Span}\left(\left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \right\} \right) \quad \text{and} \quad \mathbf{W} = \text{Span}\left(\left\{ \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right\} \right)$$

Then $\mathbf{V} \perp \mathbf{W}$.

True/False

Answer: True

Let $x \in \mathbf{V}$ and $y \in \mathbf{W}$. Then

$$x = \chi_0 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \chi_1 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \chi_0 \\ \chi_1 \\ 0 \end{pmatrix} \quad \text{and} \quad y = \psi_2 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \psi_2 \end{pmatrix}.$$

But then $x^T y = \chi_0 \times 0 + \chi_1 \times 0 + 0 \times \psi_2 = 0$. Hence x and y are orthogonal.

☛ BACK TO TEXT

Homework 10.3.2.3 Let $\mathbf{V}, \mathbf{W} \subset \mathbb{R}^n$ be subspaces. If $\mathbf{V} \perp \mathbf{W}$ then $\mathbf{V} \cap \mathbf{W} = \{0\}$, the zero vector.

Always/Sometimes/Never

Answer: Always

- $\mathbf{V} \cap \mathbf{W} \subset \{0\}$: Let $x \in \mathbf{V} \cap \mathbf{W}$. We will show that $x = 0$ and hence $x \in \{0\}$.

$$
\begin{array}{l}
x \in \mathbf{V} \cap \mathbf{W} \\
\quad \Rightarrow \quad < \text{Definition of } S \cap T > \\
x \in \mathbf{V} \wedge x \in \mathbf{W} \\
\quad \Rightarrow \quad < \mathbf{V} \perp \mathbf{W} > \\
x^T x = 0 \\
\quad \Rightarrow \quad < x^T x = 0 \text{ iff } x = 0 > \\
x = 0
\end{array}
$$

- $\{0\} \subset \mathbf{V} \cap \mathbf{W}$: Let $x \in \{0\}$. We will show that then $x \in \mathbf{V} \cap \mathbf{W}$.

$$
\begin{array}{l}
x \in \{0\} \\
\quad \Rightarrow \quad < 0 \text{ is the only element of } \{0\} > \\
x = 0 \\
\quad \Rightarrow \quad < 0 \in \mathbf{V} \text{ and } 0 \in \mathbf{W} > \\
x \in \mathbf{V} \wedge x \in \mathbf{W} \\
\quad \Rightarrow \quad < \text{Definition of } S \cap T > \\
x \in \mathbf{V} \cap \mathbf{W}
\end{array}
$$

☛ BACK TO TEXT

Homework 10.3.2.4 If $\mathbf{V} \subset \mathbb{R}^n$ is a subspace, then $\mathbf{V}^\perp$ is a subspace.

True/False

Answer: True

- $0 \in \mathbf{V}^\perp$: Let $x \in \mathbf{V}$. Then $0^T x = 0$ and hence $0 \in \mathbf{V}^\perp$.

- If $x, y \in \mathbf{V}^\perp$ then $x + y \in \mathbf{V}^\perp$: Let $x, y \in \mathbf{V}^\perp$ and let $z \in \mathbf{V}$. We need to show that $(x+y)^T z = 0$.

$$
\begin{array}{l}
(x+y)^T z \\
\quad = \quad < \text{property of dot} > \\
x^T z + y^T z \\
\quad = \quad < x, y \in \mathbf{V}^\perp \text{ and } z \in \mathbf{V} > \\
0 + 0 \\
\quad = \quad < algebra > \\
0
\end{array}
$$

Hence $x + y \in \mathbf{V}^\perp$.

- If $\alpha \in \mathbb{R}$ and $x \in \mathbf{V}^\perp$ then $\alpha x \in \mathbf{V}^\perp$: Let $\alpha \in \mathbb{R}$, $x \in \mathbf{V}^\perp$ and let $z \in \mathbf{V}$. We need to show that $(\alpha x)^T z = 0$.

$$\begin{array}{l} (\alpha x)^T z \\ \quad = \quad < \text{algebra} > \\ \alpha x^T z \\ \quad = \quad < x \in \mathbf{V}^\perp \text{ and } z \in \mathbf{V} > \\ \alpha 0 \\ \quad = \quad < \text{algebra} > \\ 0 \end{array}$$

  Hence $\alpha x \in \mathbf{V}^\perp$.

Hence $\mathbf{V}^\perp$ is a subspace.

☛ BACK TO TEXT

10.4 Approximating a Solution

10.4.2 Finding the Best Solution

Homework 10.4.2.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$.

1. Is b in the column space of A?

   True/False

   Answer: False

   The augmented system is

   $$\left(\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{array}\right).$$

   Transforming this to row-echelon form yields

   $$\left(\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & -2 \end{array}\right),$$

   which means the last equation is inconsistent with the first two.

2. $A^T b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

3. $A^T A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

4. $(A^T A)^{-1} = \begin{pmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{pmatrix}$

5. $A^\dagger = \begin{pmatrix} 2/3 & -1/3 & 1/3 \\ -1/3 & 2/3 & 1/3 \end{pmatrix}$.

6. $A^\dagger A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

7. Compute the approximate solution, in the least squares sense, of $Ax \approx b$.

   $$x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \end{pmatrix}$$

8. What is the project of b onto the column space of A?

   $$\widehat{b} = \begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_1 \\ \widehat{\beta}_2 \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 2/3 \end{pmatrix}$$

☛ BACK TO TEXT

Homework 10.4.2.2 Consider $A = \begin{pmatrix} 1 & -1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}$ and $b = \begin{pmatrix} 4 \\ 5 \\ 9 \end{pmatrix}$.

1. b is in the column space of A, $\mathcal{C}(A)$.

   True/False

2. Compute the approximate solution, in the least squares sense, of $Ax \approx b$.

   $$x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} =$$

3. What is the project of b onto the column space of A?

   $$\widehat{b} = \begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_1 \\ \widehat{\beta}_2 \end{pmatrix} =$$

4. $A^\dagger =$.

5. $A^\dagger A =$.

☛ BACK TO TEXT

Homework 10.4.2.3 What 2×2 matrix B projects the x-y plane onto the line $x + y = 0$?

Answer: Let's think this through.

- All points on the plane can be given by (β_0, β_1) and are pointed to by the vectors $b = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}$.

- The point $(1,-1)$ is on the line and is pointed to by the vector $a = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$. Thus, all vectors that point to points on that line can be characterized by $a\chi = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} \chi$, where χ is a scalar. Notice that this is equivalent to saying all these vectors are in the column space A where A consists only of the vector a.
- So, the problem can now be translated into "Find the matrix that projects vectors $b \in \mathbb{R}^2$ onto the column space of $A = \begin{pmatrix} a \end{pmatrix}$."
- But we saw that the matrix that projects onto the columns space of A is given by

$$B = A(A^T A)^{-1} A^T$$

which in this case means

$$\begin{aligned} B &= \begin{pmatrix} 1 \\ -1 \end{pmatrix} \left(\begin{pmatrix} 1 \\ -1 \end{pmatrix}^T \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right)^{-1} \begin{pmatrix} 1 \\ -1 \end{pmatrix}^T = \begin{pmatrix} 1 \\ -1 \end{pmatrix} (2)^{-1} \begin{pmatrix} 1 \\ -1 \end{pmatrix}^T \\ &= \frac{1}{2} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \begin{pmatrix} 1 & -1 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{2} & \frac{1}{-2} \\ \frac{1}{-2} & \frac{1}{2} \end{pmatrix}. \end{aligned}$$

☛ BACK TO TEXT

Homework 10.4.2.4 Find the line that best fits the following data:

x	y
-1	2
1	-3
0	0
2	-5

Answer:
Answer Let $y = \gamma_0 + \gamma_1 x$ be the straight line, where γ_0 and γ_1 are to be determined. Then

$$\begin{aligned} \gamma_0 + \gamma_1(-1) &= 2 \\ \gamma_0 + \gamma_1(1) &= -3 \\ \gamma_0 + \gamma_1(0) &= 0 \\ \gamma_0 + \gamma_1(2) &= -5 \end{aligned}$$

which in matrix notation means that we wish to approximately solve $Ac = b$ where

$$A = \begin{pmatrix} 1 & -1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 2 \end{pmatrix}, \quad c = \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix}, \quad \text{and} \quad b = \begin{pmatrix} 2 \\ -3 \\ 0 \\ -5 \end{pmatrix}.$$

The solution to this is given by $c = (A^T A)^{-1} A^T b$.

$$
\begin{aligned}
A^T A &= \begin{pmatrix} 1 & -1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 2 \end{pmatrix}^T \begin{pmatrix} 1 & -1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 4 & 2 \\ 2 & 6 \end{pmatrix} \\
(A^T A)^{-1} &= \frac{1}{(4)(6)-(2)(2)} \begin{pmatrix} 6 & -2 \\ -2 & 4 \end{pmatrix} = \frac{1}{20} \begin{pmatrix} 6 & -2 \\ -2 & 4 \end{pmatrix} \\
A^T b &= \begin{pmatrix} 1 & -1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 2 \end{pmatrix}^T \begin{pmatrix} 2 \\ -3 \\ 0 \\ -5 \end{pmatrix} = \begin{pmatrix} -6 \\ -15 \end{pmatrix} \\
(A^T A)^{-1} A^T b &= \frac{1}{20} \begin{pmatrix} 6 & -2 \\ -2 & 4 \end{pmatrix} \begin{pmatrix} -6 \\ -15 \end{pmatrix} = \frac{1}{20} \begin{pmatrix} -6 \\ -48 \end{pmatrix}
\end{aligned}
$$

which I choose not to simplify.

So, the desired coefficients are given by $\gamma_0 = -3/10$ and $\gamma_1 = -12/5$.

☛ BACK TO TEXT

Homework 10.4.2.5 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 2 \\ 7 \end{pmatrix}$.

1. b is in the column space of A, $\mathcal{C}(A)$.

   True/False

2. Compute the approximate solution, in the least squares sense, of $Ax \approx b$.

   $x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} =$

3. What is the projection of b onto the column space of A?

   $\widehat{b} = \begin{pmatrix} \widehat{\beta}_0 \\ \widehat{\beta}_1 \\ \widehat{\beta}_2 \end{pmatrix} =$

4. $A^\dagger =$.

5. $A^\dagger A =$.

☛ BACK TO TEXT

10.7 Wrap Up

Week 11: Orthogonal Projection, Low Rank Approximation, and Orthogonal Bases (Answers)

11.2 Projecting a Vector onto a Subspace

11.2.1 Component in the Direction of ...

Homework 11.2.1.1 Let $a = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $P_a(x)$ and $P_a^{\perp}(x)$ be the projection of vector x onto $\text{Span}(\{a\})$ and $\text{Span}(\{a\})^{\perp}$, respectively. Compute

Preparation: $(a^T a)^{-1} = (e_0^T e_0)^{-1} = 1^{-1} = 1$. So, $a(a^T a)^{-1} a^T = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}^T$.

1. $P_a(\begin{pmatrix} 2 \\ 0 \end{pmatrix}) = a(a^T a)^{-1} a^T x = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}^T \begin{pmatrix} 2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} 2 = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$.
   Now, you could have figured this out more simply: The vector x, in this case, is clearly just a multiple of vector a, and hence its projection onto the span of a is just the vector x itself.
2. $P_a^{\perp}(\begin{pmatrix} 2 \\ 0 \end{pmatrix}) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.
   Since x is a multiple of a, the component of x perpendicular to a is clearly 0, the zero vector. Alternatively, can compute $x - P_a(x)$ using the result you computed for $P_a(x)$. Alternatively, you can compute $(I - a(a^T a)^{-1} a^T)x$.
3. $P_a(\begin{pmatrix} 4 \\ 2 \end{pmatrix}) = \begin{pmatrix} 4 \\ 0 \end{pmatrix}$.
4. $P_a^{\perp}(\begin{pmatrix} 4 \\ 2 \end{pmatrix}) = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$.
5. Draw a picture for each of the above.

☛ BACK TO TEXT

Homework 11.2.1.2 Let $a = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$ and $P_a(x)$ and $P_a^{\perp}(x)$ be the projection of vector x onto $\text{Span}(\{a\})$ and $\text{Span}(\{a\})^{\perp}$, respectively. Compute

Preparation: $(a^T a)^{-1} = 2^{-1} = 1/2$. So, $a(a^T a)^{-1} a^T = \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}^T$.

1. $P_a(\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}) = a(a^T a)^{-1} a^T x = \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}^T \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} (1) = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{pmatrix}$.

Notice that we did not actually form $a(a^Ta)^{-1}a^T$. Let's see if we had:

$$\frac{1}{2}\begin{pmatrix}1\\1\\0\end{pmatrix}\begin{pmatrix}1\\1\\0\end{pmatrix}^T\begin{pmatrix}0\\1\\1\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1\\1\\0\end{pmatrix}\begin{pmatrix}1 & 1 & 0\end{pmatrix}\begin{pmatrix}0\\1\\1\end{pmatrix}$$
$$= \frac{1}{2}\begin{pmatrix}1 & 1 & 0\\1 & 1 & 0\\0 & 0 & 0\end{pmatrix}\begin{pmatrix}0\\1\\1\end{pmatrix} = \begin{pmatrix}\frac{1}{2} & \frac{1}{2} & 0\\\frac{1}{2} & \frac{1}{2} & 0\\0 & 0 & 0\end{pmatrix}\begin{pmatrix}0\\1\\1\end{pmatrix} = \begin{pmatrix}\frac{1}{2}\\\frac{1}{2}\\0\end{pmatrix}$$

That is a LOT more work!!!

2. $P_a^\perp(\begin{pmatrix}0\\1\\1\end{pmatrix}) = \begin{pmatrix}0\\1\\1\end{pmatrix} - \begin{pmatrix}\frac{1}{2}\\\frac{1}{2}\\0\end{pmatrix} = \begin{pmatrix}-\frac{1}{2}\\\frac{1}{2}\\1\end{pmatrix}$
   This time, had we formed $I - a(a^Ta)^{-1}a^T$, the work would have been even more.

3. $P_a(\begin{pmatrix}0\\0\\1\end{pmatrix}) = \begin{pmatrix}0\\0\\0\end{pmatrix}$.

4. $P_a^\perp(\begin{pmatrix}0\\0\\1\end{pmatrix}) = \begin{pmatrix}0\\0\\1\end{pmatrix}$.

☛ BACK TO TEXT

Homework 11.2.1.3 Let $a, v, b \in \mathbb{R}^m$.
What is the approximate cost of computing $(av^T)b$, obeying the order indicated by the parentheses?

- $m^2 + 2m$.
- $3m^2$.
- $2m^2 + 4m$.

Answer: Forming the outer product av^T requires m^2 flops. Multiplying the resulting matrix times vector b requires another $2m^2$ flops, for a total of $3m^2$ flops.
What is the approximate cost of computing $(v^Tb)a$, obeying the order indicated by the parentheses?

- $m^2 + 2m$.
- $3m$.
- $2m^2 + 4m$.

Answer: Computing the inner product v^Tb requires approximates $2m$ flops. Scaling vector a by the resulting scalar requires m flops, for a total of $3m$ flops.

☛ BACK TO TEXT

Homework 11.2.1.4 Given $a, x \in \mathbb{R}^m$, let $P_a(x)$ and $P_a^\perp(x)$ be the projection of vector x onto $\text{Span}(\{a\})$ and $\text{Span}(\{a\})^\perp$, respectively. Then which of the following are true:

1. $P_a(a) = a$. True/False **Answer:** True

   - **Explanation 1:** a is in Span$(\{a\})$, hence it is its own projection.
   - **Explanation 2:** $(a(a^T a)^{-1} a^T) a = a(a^T a)^{-1}(a^T a) = a$.

2. $P_a(\chi a) = \chi a$. True/False

   Answer: True

   - **Explanation 1:** χa is in Span$(\{a\})$, hence it is its own projection.
   - **Explanation 2:** $(a(a^T a)^{-1} a^T)\chi a = \chi a(a^T a)^{-1}(a^T a) = \chi a$.

3. $P_a^{\perp}(\chi a) = 0$ (the zero vector). True/False

   Answer: True

   - **Explanation 1:** χa is in Span$(\{a\})$, and has no component orthogonal to a.
   - **Explanation 2:**

$$\begin{array}{rcl} P_a^{\perp}(\chi a) & = & (I - a(a^T a)^{-1} a^T)\chi a = \chi a - a(a^T a)^{-1} a^T \chi a \\ & = & \chi a - \chi a(a^T a)^{-1} a^T a = \chi a - \chi a = 0. \end{array}$$

4. $P_a(P_a(x)) = P_a(x)$. True/False

   Answer: True

   - **Explanation 1:** $P_a(x)$ is in Span$(\{a\})$, hence it is its own projection.
   - **Explanation 2:**

$$\begin{array}{rcl} P_a(P_a(x)) & = & (a(a^T a)^{-1} a^T)(a(a^T a)^{-1} a^T) x \\ & = & a(a^T a)^{-1}(a^T a)(a^T a)^{-1} a^T) x = a(a^T a)^{-1} a^T) x = P_a(x). \end{array}$$

5. $P_a^{\perp}(P_a^{\perp}(x)) = P_a^{\perp}(x)$. True/False

   Answer: True

   - **Explanation 1:** $P_a^{\perp}(x)$ is in Span$(\{a\})^{\perp}$, hence it is its own projection.
   - **Explanation 2:**

$$\begin{array}{rcl} P_a^{\perp}(P_a^{\perp}(x)) & = & (I - a(a^T a)^{-1} a^T)(I - a(a^T a)^{-1} a^T) x \\ & = & (I - a(a^T a)^{-1} a^T - (I - a(a^T a)^{-1} a^T) a(a^T a)^{-1} a^T) x \\ & = & (I - a(a^T a)^{-1} a^T - a(a^T a)^{-1} a^T + a(a^T a)^{-1} a^T a(a^T a)^{-1} a^T) x \\ & = & (I - a(a^T a)^{-1} a^T - a(a^T a)^{-1} a^T + a(a^T a)^{-1} a^T) x \\ & = & (I - a(a^T a)^{-1} a^T) x = P_a^{\perp}(x). \end{array}$$

6. $P_a(P_a^{\perp}(x)) = 0$ (the zero vector). True/False

   Answer: True

   - **Explanation 1:** $P_a^{\perp}(x)$ is in Span$(\{a\})^{\perp}$, hence orthogonal to Span$(\{a\})$.
   - **Explanation 2:**

$$\begin{array}{rcl} P_a(P_a^{\perp}(x)) & = & (a(a^T a)^{-1} a^T)(I - a(a^T a)^{-1} a^T) x \\ & = & (a(a^T a)^{-1} a^T - a(a^T a)^{-1} a^T a(a^T a)^{-1} a^T) x \\ & = & (a(a^T a)^{-1} a^T - a(a^T a)^{-1} a^T) x = 0. \end{array}$$

(Hint: Draw yourself a picture.)

☛ BACK TO TEXT

11.2.2 An Application: Rank-1 Approximation

Homework 11.2.2.1 Let $\mathbf{S}$ and $\mathbf{T}$ be subspaces of $\mathbb{R}^m$ and $\mathbf{S} \subset \mathbf{T}$.
$\dim(\mathbf{S}) \leq \dim(\mathbf{T})$.

Always/Sometimes/Never

Answer: Always

Proof by contradiction:

Let $\dim(\mathbf{S}) = k$ and $\dim(\mathbf{T}) = n$, where $k > n$. Then we can find a set of k vectors $\{s_0, \ldots, s_{k-1}\}$ that form a basis for $\mathbf{S}$ and a set of n vectors $\{t_0, \ldots, t_{n-1}\}$ that form a basis for $\mathbf{T}$.

Let

$$S = \left(\begin{array}{c|c|c|c} s_0 & s_1 & \cdots & s_{k-1} \end{array} \right) \quad \text{and} \quad T = \left(\begin{array}{c|c|c|c} t_0 & t_1 & \cdots & t_{n-1} \end{array} \right).$$

$s_j \in \mathbf{T}$ and hence can be written as $s_j = T x_j$. Thus

$$S = T \underbrace{\left(\begin{array}{c|c|c|c} x_0 & x_1 & \cdots & x_{k-1} \end{array} \right)}_{X} = TX$$

But X is $n \times k$ which has more columns that it has rows. Hence, there must exist vector $z \neq 0$ such that $Xz = 0$.

But then

$$Sz = TXz = T0 = 0$$

and hence S does not have linearly independent columns. But, we assumed that the colums of S formed a basis, and hence this is a contradiction. We conclude that $\dim(\mathbf{S}) \leq \dim(\mathbf{T})$.

☛ BACK TO TEXT

Homework 11.2.2.2 Let $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. Then the $m \times n$ matrix uv^T has a rank of at most one.

True/False

Answer: True

Let $y \in \mathcal{C}(uv^T)$. We will show that then $y \in \text{Span}(\{u\})$ and hence $\mathcal{C}(uv^T) \subset \text{Span}(\{u\})$.

$$\begin{array}{l}
y \in \mathcal{C}(uy^T) \\
\quad \Rightarrow \quad < \text{there exists a } x \in \mathbb{R}^n \text{ such that } y = uv^T x > \\
y = uv^T x \\
\quad \Rightarrow \quad < u\alpha = \alpha u \text{ when } \alpha \in \mathbb{R} > \\
y = (v^T x) u \\
\quad \Rightarrow \quad < \text{Definition of span and } v^T x \text{ is a scalar} > \\
y \in \text{Span}(\{u\})
\end{array}$$

Hence $\dim(\mathcal{C}(uv^T)) \leq \dim(\text{Span}(\{u\})) \leq 1$. Since $\text{rank}(uv^T) = \dim(\mathcal{C}(uv^T))$ we conclude that $\text{rank}(uv^T) \leq 1$.

☛ BACK TO TEXT

Homework 11.2.2.3 Let $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. Then uv^T has rank equal to zero if
(Mark all correct answers.)

1. $u = 0$ (the zero vector in $\mathbb{R}^m$).

2. $v = 0$ (the zero vector in $\mathbb{R}^n$).

3. Never.

4. Always.

Answer:

1. $u = 0$ (the zero vector in $\mathbb{R}^m$).
2. $v = 0$ (the zero vector in $\mathbb{R}^n$).

☛ BACK TO TEXT

11.2.3 Projection onto a Subspace

No video this section

Homework 11.2.3.1 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 2 \\ 7 \end{pmatrix}$.

1. Find the projection of b onto the column space of A.

   Answer: The formula for the projection, when A has linearly independent columns, is

   $$A(A^T A)^{-1} A b.$$

   Now

   $$\begin{aligned}
   A^T A &= \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}^T \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix} = \begin{pmatrix} 6 & -8 \\ -8 & 18 \end{pmatrix} \\
   (A^T A)^{-1} &= \frac{1}{(6)(18) - (-8)(-8)} \begin{pmatrix} 18 & 8 \\ 8 & 6 \end{pmatrix} = \frac{1}{44} \begin{pmatrix} 18 & 8 \\ 8 & 6 \end{pmatrix} \\
   A^T b &= \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}^T \begin{pmatrix} 1 \\ 2 \\ 7 \end{pmatrix} = \begin{pmatrix} -11 \\ 27 \end{pmatrix} \\
   (A^T A)^{-1} A^T b &= \frac{1}{44} \begin{pmatrix} 18 & 8 \\ 8 & 6 \end{pmatrix} \begin{pmatrix} -11 \\ 27 \end{pmatrix} = \frac{1}{44} \begin{pmatrix} 18 \\ 74 \end{pmatrix} \\
   A(A^T A)^{-1} A^T b &= \frac{1}{44} \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix} \begin{pmatrix} 18 \\ 74 \end{pmatrix} = \frac{1}{44} \begin{pmatrix} 92 \\ -56 \\ 260 \end{pmatrix}
   \end{aligned}$$

   which I choose not to simplify...

2. Split b into $z + w$ where z is in the column space and w is perpendicular (orthogonal) to that space.

   Answer: Notice that $z = A(A^T A)^{-1} A b$ so that

   $$w = b - A(A^T A)^{-1} A b = \begin{pmatrix} 1 \\ 2 \\ 7 \end{pmatrix} - \frac{1}{44} \begin{pmatrix} 92 \\ -56 \\ 260 \end{pmatrix}.$$

3. Which of the four subspaces ($\mathcal{C}(A)$, $\mathcal{R}(A)$, $\mathcal{N}(A)$, $\mathcal{N}(A^T)$) contains w?

   Answer: This vector is orthogonal to the column space and therefore is in the left null space of A: $w \in \mathcal{N}(A^T)$.

☛ BACK TO TEXT

11.2.5 An Application: Rank-k Approximation

Homework 11.2.5.1 Let $U \in \mathbb{R}^{m\times k}$ and $V \in \mathbb{R}^{n\times k}$. Then the $m\times n$ matrix UV^T has rank at most k.

True/False

Answer: True

Again, we build on the insight that if $S,T \subset \mathbb{R}^m$ are subspaces and $S \subset T$, then $\dim(S) \le \dim(T)$. Here $T = \mathcal{C}(\{U\})$ and $S = \mathcal{C}(UV^T)$.

Now, clearly $\text{rank}(U) = \dim(\mathcal{C}(U)) \le k$ since U is a $m\times k$ matrix. Let $y \in \mathcal{C}(UV^T)$. We will show that then $y \in \mathcal{C}(U)$.

$$
\begin{array}{l}
y \in \mathcal{C}(UV^T) \\
\quad \Rightarrow \quad < \text{there exists a } x \in \mathbb{R}^n \text{ such that } y = UV^T x > \\
y = UV^T x \\
\quad \Rightarrow \quad < z = V^T x > \\
y = Uz \\
\quad \Rightarrow \quad < \text{Definition of column space} > \\
y \in \mathcal{C}(U).
\end{array}
$$

Hence $\text{rank}(UV^T = \dim(\mathcal{C}(UV^T)) \le \dim(\mathcal{C}(U)) = \text{rank}(U) \le k$.

☛ BACK TO TEXT

Homework 11.2.5.2 We discussed in this section that the projection of B onto the column space of A is given by $A(A^TA)^{-1}A^TB$. So, if we compute $V = (A^TA)^{-1}A^TB$, then AV is an approximation to B that requires only $m\times k$ matrix A and $k\times n$ matrix V.

To compute V, we can perform the following steps:

- Form $C = A^TA$.
- Compute the LU factorization of C, overwriting C with the resulting L and U.
- Compute $V = A^TB$.
- Solve $LX = V$, overwriting V with the solution matrix X.
- Solve $UX = V$, overwriting V with the solution matrix X.
- Compute the approximation of B as $A\cdot V$ (A times V). In practice, you would not compute this approximation, but store A and V instead, which typically means less data is stored.

To experiments with this, download Week11.zip, place it in

```
LAFF-2.0xM -> Programming
```

and unzip it. Then examine the file `Week11/CompressPicture.m`, look for the comments on what operations need to be inserted, and insert them. Execute the script in the Command Window and see how the picture in file `building.png` is approximated. Play with the number of columns used to approximate. Find your own picture! (It will have to be a black-and-white picture for what we discussed to work.

Notice that A^TA is a symmetric matrix, and it can be shown to be symmetric positive definite under most circumstances (when A has linearly independent columns). This means that instead of the LU factorization, one can use the Cholesky factorization (see the enrichment in Week 8). In `Week11.zip` you will also find a function for computing the Cholesky factorization. Try to use it to perform the calculations.

Answer: See the file `Week11/CompressPicture_Answer.m`.

☛ BACK TO TEXT

11.3 Orthonormal Bases

11.3.1 The Unit Basis Vectors, Again

Homework 11.3.1.1 Consider the vectors

$$v_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad v_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad v_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

1. Compute
   (a) $v_0^T v_1 = 1$
   (b) $v_0^T v_2 = 1$
   (c) $v_1^T v_2 = 2$
2. These vectors are orthonormal. True/False

   Answer: False

   They are neither orthogonal to each other nor are vectors v_1 and v_2 of length one.

☛ BACK TO TEXT

11.3.2 Orthonormal Vectors

Homework 11.3.2.1

1. $\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
2. $\begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
3. The vectors $\begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$ are orthonormal. True/False

   Answer: True
4. The vectors $\begin{pmatrix} \sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ -\sin(\theta) \end{pmatrix}$ are orthonormal. True/False

   Answer: True

☛ BACK TO TEXT

Homework 11.3.2.2 Let $q_0, q_1, \ldots, q_{k-1} \in \mathbb{R}^m$ be a set of orthonormal vectors. Let

$$Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right).$$

Then $Q^T Q = I$.

TRUE/FALSE

Answer: TRUE

$$Q^T Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)^T \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) = \left(\begin{array}{c} q_0^T \\ \hline q_1^T \\ \hline \vdots \\ \hline q_{k-1}^T \end{array} \right) \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)$$

$$= \left(\begin{array}{c|c|c|c} q_0^T q_0 & q_0^T q_1 & \cdots & q_0^T q_{k-1} \\ \hline q_1^T q_0 & q_1^T q_1 & \cdots & q_1^T q_{k-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline q_{k-1}^T q_0 & q_{k-1}^T q_1 & \cdots & q_{k-1}^T q_{k-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} 1 & 0 & \cdots & 0 \\ \hline 0 & 1 & \cdots & 0 \\ \hline \vdots & \vdots & & \vdots \\ \hline 0 & 0 & \cdots & 1 \end{array} \right)$$

$$\text{since } q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

☛ BACK TO TEXT

Homework 11.3.2.3 Let $Q \in \mathbb{R}^{m \times k}$ (with $k \leq m$) and $Q^T Q = I$. Partition

$$Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right).$$

Then $q_0, q_1, \ldots, q_{k-1}$ are orthonormal vectors.

TRUE/FALSE

Answer: TRUE

$$Q^T Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)^T \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) = \left(\begin{array}{c} q_0^T \\ \hline q_1^T \\ \hline \vdots \\ \hline q_{k-1}^T \end{array} \right) \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)$$

$$= \left(\begin{array}{c|c|c|c} q_0^T q_0 & q_0^T q_1 & \cdots & q_0^T q_{k-1} \\ \hline q_1^T q_0 & q_1^T q_1 & \cdots & q_1^T q_{k-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline q_{k-1}^T q_0 & q_{k-1}^T q_1 & \cdots & q_{k-1}^T q_{k-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} 1 & 0 & \cdots & 0 \\ \hline 0 & 1 & \cdots & 0 \\ \hline \vdots & \vdots & & \vdots \\ \hline 0 & 0 & \cdots & 1 \end{array} \right).$$

Hence $q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$

☛ BACK TO TEXT

Homework 11.3.2.4 Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Then the matrix that projects vectors onto $\text{Span}(\{q\})$ is given by qq^T.

True/False

Answer: True

The matrix that projects onto Span($\{q\}$) is given by $q(q^Tq)^{-1}q^T$. But $q^Tq = 1$ since q is of length one. Thus $q \underbrace{(q^Tq)^{-1}}_{1} q^T = qq^T$.

☛ BACK TO TEXT

Homework 11.3.2.5 Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Let $x \in \mathbb{R}^m$. Then the component of x in the direction of q (in Span($\{q\}$)) is given by q^Txq.

True/False

Answer: True

In the last exercise we saw that the matrix that projects onto Span($\{q\}$) is given by qq^T. Thus, the component of x in the direction of q is given by $qq^Tx = q(q^Tx) = q^Txq$ (since q^Tx is a scalar).

☛ BACK TO TEXT

Homework 11.3.2.6 Let $Q \in \mathbb{R}^{m\times n}$ have orthonormal columns (which means $Q^TQ = I$). Then the matrix that projects vectors onto the column space of Q, $\mathcal{C}(Q)$, is given by QQ^T.

True/False

Answer: True

The matrix that projects onto $\mathcal{C}(Q)$ is given by $Q(Q^TQ)^{-1}Q^T$. But then $Q \underbrace{(Q^TQ)^{-1}}_{I^{-1}=I} Q^T = QQ^T$.

☛ BACK TO TEXT

Homework 11.3.2.7 Let $Q \in \mathbb{R}^{m\times n}$ have orthonormal columns (which means $Q^TQ = I$). Then the matrix that projects vectors onto the space orthogonal to the columns of Q, $\mathcal{C}(Q)^\perp$, is given by $I - QQ^T$.

True/False

Answer: True

In the last problem we saw that the matrix that projects onto $\mathcal{C}(Q)$ is given by QQ^T. Hence, the matrix that projects onto the space orthogonal to $\mathcal{C}(Q)$ is given by $I - QQ^T$.

☛ BACK TO TEXT

11.3.4 Orthogonal Bases (Alternative Explanation)

Homework 11.3.4.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ Compute an orthonormal basis for $\mathcal{C}(A)$.

Answer: Here

$$a_0 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \quad \text{and} \quad a_1 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}.$$

- Compute the length of a_0: $\rho_{0,0} = \|a_0\|_2 = \sqrt{a_0^Ta_0} = \sqrt{2}$.
- Normalize a_0 to length one: $q_0 = a_0/\rho_{0,0} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$ (which can be put in standard form, but let's leave it alone...)

- Compute the length of the component of a_1 in the direction of q_0: $\rho_{0,1} = q_0^T a_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}^T \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}}$.

- Compute the component of a_1 orthogonal to q_0:

$$a_1^\perp = a_1 - \rho_{0,1} q_0 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} - \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} - \frac{1}{2} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} \\ 1 \\ \frac{1}{2} \end{pmatrix}.$$

- Compute the length of $a_1^\perp$: $\rho_{1,1} = \| a_1^\perp \|_2 = \sqrt{a_1^{\perp T} a_1^\perp} = \sqrt{\frac{1}{4} + 1 + \frac{1}{4}} = \frac{\sqrt{6}}{2}$.

- Normalize $a_1^\perp$ to have length one: $q_1 = a_1^\perp / \rho_{1,1} = \frac{2}{\sqrt{6}} \begin{pmatrix} -\frac{1}{2} \\ 1 \\ \frac{1}{2} \end{pmatrix}$. (which can be put in standard form, but let's not!)

- The orthonormal basis is then given by the vectors q_0 and q_1.

☛ BACK TO TEXT

Homework 11.3.4.2 Consider $A = \begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 2 \end{pmatrix}$. Compute an orthonormal basis for $\mathcal{C}(A)$.

Answer: Here

$$a_0 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad a_1 = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}, \quad \text{and} \quad a_2 = \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix}.$$

So,

- Compute the length of a_0: $\rho_{0,0} = \| a_0 \|_2 = \sqrt{a_0^T a_0} = \sqrt{3}$.

- Normalize a_0 to length one: $q_0 = a_0 / \rho_{0,0} = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ (which can be put in standard form, but let's leave it alone...)

- Compute the length of the component of a_1 in the direction of q_0:

$$\rho_{0,1} = q_0^T a_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} = 0.$$

  Ah! This means that a_1 is orthogonal to q_1.

- Compute the component of a_1 orthogonal to q_0:

$$a_1^\perp = a_1 - \rho_{0,1} q_0 = a_1 - 0 q_0 = a_1 = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

- Compute the length of $a_1^\perp$: $\rho_{1,1} = \| a_1^\perp \|_2 = \sqrt{{a_1^\perp}^T a_1^\perp} = \sqrt{2}$.

- Normalize $a_1^\perp$ to have length one: $q_1 = a_1^\perp / \rho_{1,1} = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$. (which can be put in standard form, but let's not!)

- Compute the lengths of the components of a_2 in the directions of q_0 and q_1:

$$\begin{array}{rcl} \rho_{0,2} & = & q_0^T a_2 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} = \frac{3}{\sqrt{3}} = \sqrt{3}. \\ \rho_{1,2} & = & q_1^T a_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}^T \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} = \frac{2}{\sqrt{2}} = \sqrt{2}. \end{array}$$

- Compute the component of a_2 orthogonal to q_0 and q_1:

$$\begin{array}{rcl} a_2^\perp & = & a_2 - \rho_{0,2} q_0 - \rho_{1,2} q_1 = \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} - \sqrt{3} \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \sqrt{2} \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \\ & = & \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \end{array}$$

  This means that a_2 has no component orthogonal to q_0 and q_1. This in turn means that a_2 is in $\text{Span}(\{q_0, q_1\}) = \mathcal{C}(a_0, a_1)$.

- An orthonormal basis for $\mathcal{C}(A)$ is given by $\{q_0, q_1\}$.

☛ BACK TO TEXT

Homework 11.3.4.3 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$. Compute an orthonormal basis for $\mathcal{C}(A)$.

Answer: Here

$$a_0 = \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix} \quad \text{and} \quad a_1 = \begin{pmatrix} 1 \\ -1 \\ 4 \end{pmatrix}.$$

- Compute the length of a_0: $\rho_{0,0} = \| a_0 \|_2 = \sqrt{a_0^T a_0} = \sqrt{6}$.

- Normalize a_0 to length one: $q_0 = a_0 / \rho_{0,0} = \frac{1}{\sqrt{6}} \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix}$ (which can be put in standard form, but let's leave it alone...)

- Compute the length of the component of a_1 in the direction of q_0: $\rho_{0,1} = q_0^T a_1 = \frac{1}{\sqrt{6}} \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix}^T \begin{pmatrix} 1 \\ -1 \\ 4 \end{pmatrix} = \frac{-8}{\sqrt{2}}$.

- Compute the component of a_1 orthogonal to q_0:

$$a_1^\perp = a_1 - \rho_{0,1} q_0 = \begin{pmatrix} 1 \\ -1 \\ 4 \end{pmatrix} - \frac{-8}{\sqrt{6}} \frac{1}{\sqrt{6}} \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 4 \end{pmatrix} + \frac{4}{3} \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix} = \frac{1}{3} \begin{pmatrix} 7 \\ 1 \\ 4 \end{pmatrix}.$$

- Compute the length of $a_1^\perp$: $\rho_{1,1} = \|a_1^\perp\|_2 = \sqrt{a_1^{\perp T} a_1^\perp} = \left\| \frac{1}{3} \begin{pmatrix} 7 \\ 1 \\ 4 \end{pmatrix} \right\|_2 = \left|\frac{1}{3}\right| \left\| \begin{pmatrix} 7 \\ 1 \\ 4 \end{pmatrix} \right\|_2 = \frac{\sqrt{49+1+16}}{3} = \frac{\sqrt{66}}{3}$.

- Normalize $a_1^\perp$ to have length one: $q_1 = a_1^\perp / \rho_{1,1} = \frac{3}{\sqrt{66}} \frac{1}{3} \begin{pmatrix} 7 \\ 1 \\ 4 \end{pmatrix} = \frac{1}{\sqrt{66}} \begin{pmatrix} 7 \\ 1 \\ 4 \end{pmatrix}$. (which can be put in standard form, but let's not!)

- The orthonormal basis is then given by the vectors q_0 and q_1.

☛ BACK TO TEXT

11.3.5 The QR Factorization

Homework 11.3.5.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$.

- Compute the QR factorization of this matrix.

  (Hint: Look at Homework 11.3.4.1)

  Answer: Notice that this is the same matrix as in Homework **??**. Thus, it is a matter of taking the results and plugging them into the matrices Q and R:

$$\begin{pmatrix} a_0 & a_1 \end{pmatrix} = \begin{pmatrix} q_0 & q_1 \end{pmatrix} \begin{pmatrix} \rho_{0,0} & \rho_{0,1} \\ 0 & \rho_{1,1} \end{pmatrix}.$$

  From Homework **??** we then get that

$$\left(\begin{array}{c|c} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{array}\right) = \left(\begin{array}{c|c} \frac{\sqrt{2}}{2} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} & \frac{\sqrt{6}}{3} \begin{pmatrix} -\frac{1}{2} \\ 1 \\ \frac{1}{2} \end{pmatrix} \end{array} \right) \left(\begin{array}{c|c} \sqrt{2} & \frac{\sqrt{2}}{2} \\ \hline 0 & \frac{\sqrt{6}}{2} \end{array} \right)$$

- Check that $QR = A$.

☛ BACK TO TEXT

Homework 11.3.5.2 Considerx !m

$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$. Compute the QR factorization of this matrix.

(Hint: Look at Homework 11.3.4.3)

Answer: Notice that this is the same matrix as in Homework 11.3.4.3. Thus, it is a matter of taking the results and plugging them into the matrices Q and R:

$$\left(a_0 \quad a_1 \right) = \left(q_0 \quad q_1 \right) \begin{pmatrix} \rho_{0,0} & \rho_{0,1} \\ 0 & \rho_{1,1} \end{pmatrix}.$$

From Homework 11.3.4.3 we then get that

$$\left(\begin{array}{c|c} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{array} \right) = \left(\begin{array}{c|c} \frac{1}{\sqrt{6}} \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix} & \frac{1}{\sqrt{66}} \begin{pmatrix} 7 \\ 1 \\ 4 \end{pmatrix} \end{array} \right) \left(\begin{array}{c|c} \sqrt{6} & \frac{-8}{\sqrt{2}} \\ \hline 0 & \frac{\sqrt{66}}{3} \end{array} \right)$$

Check that $A = QR$.

Answer: Just multiply it out.

☛ BACK TO TEXT

11.3.6 Solving the Linear Least-Squares Problem via QR Factorization

Homework 11.3.6.1 In Homework 11.3.4.1 you were asked to consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ and compute an orthonormal basis for $\mathcal{C}(A)$.

In Homework 11.3.5.1 you were then asked to compute the QR factorization of that matrix. Of course, you could/should have used the results from Homework 11.3.4.1 to save yourself calculations. The result was the following factorization $A = QR$:

$$\left(\begin{array}{c|c} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{array} \right) = \left(\begin{array}{c|c} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} & \frac{\sqrt{2}}{\sqrt{3}} \begin{pmatrix} -\frac{1}{2} \\ 1 \\ \frac{1}{2} \end{pmatrix} \end{array} \right) \left(\begin{array}{c|c} \sqrt{2} & \frac{1}{\sqrt{2}} \\ \hline 0 & \frac{\sqrt{6}}{2} \end{array} \right)$$

Now, compute the "best" solution (in the linear least-squares sense), $\hat{x}$, to

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}.$$

(This is the same problem as in Homework 10.4.2.1.)

- $u = Q^T b =$

  Answer:

$$\left(\begin{array}{c|c} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} & \frac{\sqrt{2}}{\sqrt{3}} \begin{pmatrix} -\frac{1}{2} \\ 1 \\ \frac{1}{2} \end{pmatrix} \end{array} \right)^T \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{\sqrt{2}}{2\sqrt{3}} \end{pmatrix}$$

- The solution to $R\hat{x} = u$ is $\hat{x} =$

  Answer:

$$\left(\begin{array}{c|c} \sqrt{2} & \frac{1}{\sqrt{2}} \\ \hline 0 & \frac{\sqrt{6}}{2} \end{array}\right)\left(\begin{array}{c} \hat{\chi}_0 \\ \hat{\chi}_1 \end{array}\right) = \left(\begin{array}{c} \frac{1}{\sqrt{2}} \\ \frac{\sqrt{2}}{2\sqrt{3}} \end{array}\right)$$

  or

$$\left(\begin{array}{c} \hat{\chi}_0 \\ \hat{\chi}_1 \end{array}\right) = \left(\begin{array}{c} 1/3 \\ 1/3 \end{array}\right)$$

☛ BACK TO TEXT

11.3.7 The QR Factorization (Again)

Homework 11.3.7.1 Implement the algorithm for computing the QR factorization of a matrix in Figure 11.1

```
[ Q_out, R_out ] = QR_unb( A, Q, R )
```

where `A` and `Q` are $m \times n$ matrices and `R` is an $n \times n$ matrix. You will want to use the routines `laff_gemv`, `laff_norm`, and `laff_invscal`. (Alternatively, use native MATLAB operations.) Store the routine in

```
LAFF-2.0xM -> Programming -> Week11 -> QR_unb.m
```

Test the routine with

```
A = [  1 -1  2
       2  1 -3
      -1  3  2
       0 -2 -1 ];

Q = zeros( 4, 3 );
R = zeros( 3, 3 );
[ Q_out, R_out ] = QR_unb( A, Q, R );
```

Next, see if $A = QR$:

```
A - Q_out * R_out
```

This should equal, approximately, the zero matrix. Check if Q has mutually orthogonal columns:

```
Q_out' * Q_out
```

This should equal, approximately, the identity matrix.

Finally, repeat the above, but with matrix

```
epsilon = 1e-8

A = [     1      1       1
       epsilon   0       0
          0   epsilon    0
          0      0    epsilon ]

Q = zeros( 4, 3 );
R = zeros( 4, 3 );
[ Q_out, R_out ] = QR_unb( A, Q, R );
```

Again, check if $A = QR$ and if Q has mutually orthogonal columns. To understand what went wrong, you may want to read Robert's notes for his graduate class. For details, see the enrichment for this week.

☛ BACK TO TEXT

11.4 Change of Basis

11.4.2 Change of Basis

Homework 11.4.2.1 The vectors

$$q_0 = \frac{\sqrt{2}}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad q_1 = \frac{\sqrt{2}}{2}\begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.$$

are mutually orthonormal.

True/False

Answer: Let $Q = \left(\begin{array}{c|c} q_0 & q_1 \end{array} \right)$. Then q_0 and q_1 are mutually orthonormal if and only if $Q^T Q = I$. Now,

$$Q^T Q = \begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}^T \begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} \begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Hence, the vectors are mutually orthonormal.

☛ BACK TO TEXT

Homework 11.4.2.2 If $Q \in \mathbb{R}^{n \times n}$ has mutually orthonormal columns then which of the following are true:

1. $Q^T Q = I$ True/False
   Answer: True
2. $QQ^T = I$ True/False
   Answer: True
3. $QQ^{-1} = I$ True/False
   Answer: True
4. $Q^{-1} = Q^T$ True/False
   Answer: True

☛ BACK TO TEXT

11.5 Singular Value Decomposition

11.5.1 The Best Low Rank Approximation

Homework 11.5.1.1 Let $B = U\Sigma V^T$ be the SVD of B, with $U \in \mathbb{R}^{m \times r}$, $\Sigma \in \mathbb{R}^{r \times r}$, and $V \in \mathbb{R}^{n \times r}$. Partition

$$U = \left(\begin{array}{c|c|c|c} u_0 & u_1 & \cdots & u_{r-1} \end{array} \right), \quad \Sigma = \left(\begin{array}{c|c|c|c} \sigma_0 & 0 & \cdots & 0 \\ \hline 0 & \sigma_1 & \cdots & 0 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \sigma_{r-1} \end{array} \right), V = \left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_{r-1} \end{array} \right).$$

$U\Sigma V^T = \sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \cdots + \sigma_{r-1} u_{r-1} v_{r-1}^T.$

Always/Sometimes/Never

Answer: Always

$$
\begin{array}{rcl}
B & = & U\Sigma V^T \\
& = & \underbrace{\underbrace{\left(\begin{array}{c|c|c|c} u_0 & u_1 & \cdots & u_{r-1}\end{array}\right)\left(\begin{array}{c|c|c|c} \sigma_0 & 0 & \cdots & 0 \\ \hline 0 & \sigma_1 & \cdots & 0 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \sigma_{r-1}\end{array}\right)}_{\left(\begin{array}{c|c|c|c} \sigma_0 u_0 & \sigma_1 u_1 & \cdots & \sigma_{r-1} u_{r-1}\end{array}\right)} \underbrace{\left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_{r-1}\end{array}\right)^T}_{\left(\begin{array}{c} v_0^T \\ \hline v_1^T \\ \hline \vdots \\ \hline v_{r-1}^T\end{array}\right)}}_{\sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \cdots + \sigma_{r-1} u_{r-1} v_{r-1}^T.}
\end{array}
$$

☛ BACK TO TEXT

Homework 11.5.1.2 Let $B = U\Sigma V^T$ be the SVD of B with $U \in \mathbb{R}^{m\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$, and $V \in \mathbb{R}^{n\times r}$.

- $\mathcal{C}(B) = \mathcal{C}(U)$

  Always/Sometimes/Never

  Answer: Always

  Recall that if we can show that $\mathcal{C}(B) \subset \mathcal{C}(U)$ and $\mathcal{C}(U) \subset \mathcal{C}(B)$, then $\mathcal{C}(B) = \mathcal{C}(U)$.

  $\mathcal{C}(B) \subset \mathcal{C}(U)$: Let $y \in \mathcal{C}(B)$. Then there exists a vector x such that $y = Bx$. But then $y = U\underbrace{\Sigma V^T x}_{z} = Uz$. Hence $y \in \mathcal{C}(U)$.

  $\mathcal{C}(U) \subset \mathcal{C}(B)$: Let $y \in \mathcal{C}(U)$. Then there exists a vector x such that $y = Ux$. But $y = U\underbrace{\Sigma \overbrace{V^T V}^{I} \Sigma^{-1}}_{I} x = U\Sigma V^T \underbrace{V^T\Sigma^{-1}x}_{w} = Bw$. Hence $y \in \mathcal{C}(B)$.

- $\mathcal{R}(B) = \mathcal{C}(V)$

  Always/Sometimes/Never

  Answer: Always

  The proof is very similar, working with B^T since $\mathcal{R}(B) = \mathcal{C}(B^T)$.

☛ BACK TO TEXT

Homework 11.5.1.3 You will now want to revisit exercise 11.2.5.2 and compare an approximation by projecting onto a few columns of the picture versus using the SVD to approximate. You can do so by executing the script `Week11/CompressPictureWithSVD.m` that you downloaded in `Week11.zip`. That script creates three figures: the first is the original picture. The second is the approximation as we discussed in Section 11.2.5. The third uses the SVD. Play with the script, changing variable `k`.

☛ BACK TO TEXT

Week 12: Eigenvalues, Eigenvectors, and Diagonalization (Answers)

12.2 Getting Started

12.2.2 Simple Examples

Homework 12.2.2.1 Which of the following are eigenpairs (λ, x) of the 2×2 zero matrix:

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} x = \lambda x,$$

where $x \neq 0$.

(Mark all correct answers.)

1. $(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix})$.
2. $(0, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
3. $(0, \begin{pmatrix} 0 \\ 1 \end{pmatrix})$.
4. $(0, \begin{pmatrix} -1 \\ 1 \end{pmatrix})$.
5. $(0, \begin{pmatrix} 1 \\ 1 \end{pmatrix})$.
6. $(0, \begin{pmatrix} 0 \\ 0 \end{pmatrix})$.

Answer:

Answers 2., 3., 4., and 5. are all correct, since for all $Ax = \lambda x$.

Answers 1. and 6. are not correct because the zero vector is not considered an eigenvector.

☛ BACK TO TEXT

Homework 12.2.2.2 Which of the following are eigenpairs (λ, x) of the 2×2 zero matrix:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} x = \lambda x,$$

where $x \neq 0$.

(Mark all correct answers.)

1. $(1, \begin{pmatrix} 0 \\ 0 \end{pmatrix})$.

2. $(1, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.

3. $(1, \begin{pmatrix} 0 \\ 1 \end{pmatrix})$.

4. $(1, \begin{pmatrix} -1 \\ 1 \end{pmatrix})$.

5. $(1, \begin{pmatrix} 1 \\ 1 \end{pmatrix})$.

6. $(-1, \begin{pmatrix} 1 \\ -1 \end{pmatrix})$.

Answer:

Answers 2., 3., 4., and 5. are all correct, since for all $Ax = \lambda x$.

Answer 1. is not correct because the zero vector is not considered an eigenvector. Answer 6. is not correct because it doesn't satisfy $Ax = \lambda x$.

☛ BACK TO TEXT

Homework 12.2.2.3 Let $A = \begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix}$.

- $\begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 3 \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ so that $(3, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$ is an eigenpair.

  True/False

  Answer: True

  Just multiply it out.

- The set of all eigenvectors associated with eigenvalue 3 is characterized by (mark all that apply):
  - All vectors $x \neq 0$ that satisfy $Ax = 3x$.

    Answer: True: this is the definition of an eigenvalue associated with an eigenvalue.
  - All vectors $x \neq 0$ that satisfy $(A - 3I)x = 0$.

    Answer: True: this is an alternate condition.
  - All vectors $x \neq 0$ that satisfy $\begin{pmatrix} 0 & 0 \\ 0 & -4 \end{pmatrix} x = 0$. **Answer:** $(A - 3I) = \begin{pmatrix} 0 & 0 \\ 0 & -4 \end{pmatrix}$ in this case.
  - $\left\{ \begin{pmatrix} \chi_0 \\ 0 \end{pmatrix} \middle| \chi_0 \text{ is a scalar} \right\}$

    Answer: False: the zero vector is in this set.

- $\begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -1 \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ so that $(-1, \begin{pmatrix} 0 \\ 1 \end{pmatrix})$ is an eigenpair.

True/False

Answer: True

Just multiply it out.

☛ BACK TO TEXT

Homework 12.2.2.4 Consider the diagonal matrix $\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}$.

Eigenpairs for this matrix are given by $(\delta_0, e_0), (\delta_1, e_1), \cdots, (\delta_{n-1}, e_{n-1})$, where e_j equals the jth unit basis vector.

Always/Sometimes/Never

Answer: Always

We will simply say "by examination" this time and get back to a more elegant solution in Unit **??**.

☛ BACK TO TEXT

Homework 12.2.2.5 Which of the following are eigenpairs (λ, x) of the 2×2 triangular matrix:

$$\begin{pmatrix} 3 & 1 \\ 0 & -1 \end{pmatrix} x = \lambda x,$$

where $x \neq 0$.

(Mark all correct answers.)

1. $(-1, \begin{pmatrix} -1 \\ 4 \end{pmatrix})$.
2. $(1/3, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
3. $(3, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
4. $(-1, \begin{pmatrix} 1 \\ 0 \end{pmatrix})$.
5. $(3, \begin{pmatrix} -1 \\ 0 \end{pmatrix})$.
6. $(-1, \begin{pmatrix} 3 \\ -1 \end{pmatrix})$.

Answer:

Answers 1., 3., and 5. are all correct, since for all $Ax = \lambda x$.

Answers 2., 4., and 6. are not correct because they don't satisfy $Ax = \lambda x$.

☛ BACK TO TEXT

Homework 12.2.2.6 Consider the upper triangular matrix $U = \left(\begin{array}{cccc} \upsilon_{0,0} & \upsilon_{0,1} & \cdots & \upsilon_{0,n-1} \\ 0 & \upsilon_{1,1} & \cdots & \upsilon_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \upsilon_{n-1,n-1} \end{array}\right)$.

The eigenvalues of this matrix are $\upsilon_{0,0}, \upsilon_{1,1}, \ldots, \upsilon_{n-1,n-1}$.

Always/Sometimes/Never

Answer: Always

The upper triangular matrix $U - \upsilon_{i,i} I$ has a zero on the diagonal. Hence it is singular. Because it is singular, $\upsilon_{i,i}$ is an eigenvalue. This is true for $i = 0, \ldots, n-1$.

☛ BACK TO TEXT

Homework 12.2.2.7 Consider $A = \left(\begin{array}{cc} 1 & 3 \\ 3 & 1 \end{array}\right)$

- The eigenvalue *largest in magnitude* is 4.
- Which of the following are eigenvectors associated with this largest eigenvalue (in magnitude):
  - $\left(\begin{array}{r} 1 \\ -1 \end{array}\right)$ false
  - $\left(\begin{array}{r} 1 \\ 1 \end{array}\right)$ true
  - $\left(\begin{array}{r} 2 \\ 2 \end{array}\right)$ true
  - $\left(\begin{array}{r} -1 \\ 2 \end{array}\right)$ false
- The eigenvalue *smallest in magnitude* is -2.
- Which of the following are eigenvectors associated with this largest eigenvalue (in magnitude):
  - $\left(\begin{array}{r} 1 \\ -1 \end{array}\right)$ true
  - $\left(\begin{array}{r} 1 \\ 1 \end{array}\right)$ false
  - $\left(\begin{array}{r} 2 \\ 2 \end{array}\right)$ false

– $\begin{pmatrix} -1 \\ 2 \end{pmatrix}$ false

☛ BACK TO TEXT

Homework 12.2.2.8 Consider $A = \begin{pmatrix} -3 & -4 \\ 5 & 6 \end{pmatrix}$

- The eigenvalue *largest in magnitude* is 2.
- The eigenvalue *smallest in magnitude* is 1.

☛ BACK TO TEXT

Homework 12.2.2.9 Consider $A = \begin{pmatrix} 2 & 2 \\ -1 & 4 \end{pmatrix}$. Which of the following are the eigenvalues of A:

- 4 and 2.
- $3+i$ and 2.
- $3+i$ and $3-i$.
- $2+i$ and $2-i$.

Answer: We only asked for the eigenvalues, but we will compute both eigenvalues and eigenvectors, for completeness.

To find the eigenvalues and eigenvectors of this matrix, we form $A - \lambda I = \begin{pmatrix} 2-\lambda & 2 \\ -1 & 4-\lambda \end{pmatrix}$ and check when the characteristic polynomial is equal to zero:

$$\det\left(\begin{pmatrix} 2-\lambda & 2 \\ -1 & 4-\lambda \end{pmatrix}\right) = (2-\lambda)(4-\lambda) - (2)(-1) = \lambda^2 - 6\lambda + 8 + 2 = \lambda^2 - 6\lambda + 10.$$

When is this equal to zero? We will use the quadratic formula:

$$\lambda = \frac{-(-6) \pm \sqrt{(-6)^2 - 4(10)}}{2} = \lambda = \frac{6 \pm \sqrt{36-40}}{2} = \lambda = \frac{6}{2} \pm \frac{\sqrt{-4}}{2} = 3 \pm i.$$

Thus, this matrix has complex valued eigenvalues in form of a conjugate pair: $\lambda_0 = 3+i$ and $\lambda_1 = 3-i$. To find the corresponding eigenvectors:

$\lambda_0 = 3+i$:

$$\begin{aligned} A - \lambda_0 I &= \begin{pmatrix} 2-(3+i) & 2 \\ -1 & 4-(3+i) \end{pmatrix} \\ &= \begin{pmatrix} -1-i & 2 \\ -1 & 1-i \end{pmatrix}. \end{aligned}$$

Find a nonzero vector in the null space:

$$\begin{pmatrix} -1-i & 2 \\ -1 & 1-i \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

By examination,

$$\begin{pmatrix} -1-i & 2 \\ -1 & 1-i \end{pmatrix} \begin{pmatrix} 2 \\ 1+i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Eigenpair: $(3+i, \begin{pmatrix} 2 \\ 1+i \end{pmatrix})$.

$\lambda_0 = 3-i$:

$$\begin{aligned} A - \lambda_1 I &= \begin{pmatrix} 2-(3-i) & 2 \\ -1 & 4-(3-i) \end{pmatrix} \\ &= \begin{pmatrix} -1+i & 2 \\ -1 & 1+i \end{pmatrix}. \end{aligned}$$

Find a nonzero vector in the null space:

$$\begin{pmatrix} -1+i & 2 \\ -1 & 1+i \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

By examination,

$$\begin{pmatrix} -1+i & 2 \\ -1 & 1+i \end{pmatrix} \begin{pmatrix} 2 \\ 1-i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Eigenpair: $(3-i, \begin{pmatrix} 2 \\ 1-i \end{pmatrix})$.

☛ BACK TO TEXT

12.2.3 Diagonalizing

Homework 12.2.3.1 The matrix

$$\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

can be diagonalized.

True/False

Answer: False

Since this matrix is upper triangular, we know that only the scalar $\lambda_0 = \lambda_1 = 0$ is an eigenvector. The problem is that the dimension of the null space of this matrix

$$\dim(\mathcal{N}(A - \lambda I)) = \dim(\mathcal{N}(\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix})) = 1.$$

Thus, we cannot find two linearly independent eigenvectors to choose as the columns of matrix X.

☛ BACK TO TEXT

Homework 12.2.3.2 In Homework 12.2.2.7 you considered the matrix

$$A = \begin{pmatrix} 1 & 3 \\ 3 & 1 \end{pmatrix}$$

and computed the eigenpairs

$$(4, \begin{pmatrix} 1 \\ 1 \end{pmatrix}) \quad \text{and} \quad (-2, \begin{pmatrix} 1 \\ -1 \end{pmatrix}).$$

- Matrix A can be diagonalized by matrix $X = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. (Yes, this matrix is not unique, so please use the info from the eigenpairs, in order...)
- $AX = \begin{pmatrix} 4 & -2 \\ 4 & 2 \end{pmatrix}$
- $X^{-1} = \begin{pmatrix} 0.5 & 0.5 \\ 0.5 & -0.5 \end{pmatrix}$
- $X^{-1}AX = \begin{pmatrix} 4 & 0 \\ 0 & -2 \end{pmatrix}$

☛ BACK TO TEXT

12.2.4 Eigenvalues and Eigenvectors of 3×3 Matrices

Homework 12.2.4.1 Let $A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix}$. Then which of the following are true:

- $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue 3.

True/False

Answer: True
Just multiply it out.

- $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue -1.

True/False

Answer: True
Just multiply it out.

- $\begin{pmatrix} 0 \\ \chi_1 \\ 0 \end{pmatrix}$, where $\chi_1 \neq 0$ is a scalar, is an eigenvector associated with eigenvalue -1.

True/False

Answer: True
Just multiply it out.

- $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is an eigenvector associated with eigenvalue 2.

True/False

Answer: True
Just multiply it out.

☛ BACK TO TEXT

Homework 12.2.4.2 Let $A = \begin{pmatrix} \alpha_{0,0} & 0 & 0 \\ 0 & \alpha_{1,1} & 0 \\ 0 & 0 & \alpha_{2,2} \end{pmatrix}$. Then which of the following are true:

- $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue $\alpha_{0,0}$.

True/False

Answer: True
Just multiply it out.

- $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue $\alpha_{1,1}$.

True/False

Answer: True

Just multiply it out.

- $\begin{pmatrix} 0 \\ \chi_1 \\ 0 \end{pmatrix}$ where $\chi_1 \neq 0$ is an eigenvector associated with eigenvalue $\alpha_{1,1}$.

True/False

Answer: True

Just multiply it out.

- $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is an eigenvector associated with eigenvalue $\alpha_{2,2}$.

True/False

Answer: True

Just multiply it out.

☛ BACK TO TEXT

Homework 12.2.4.3 Let $A = \begin{pmatrix} 3 & 1 & -1 \\ 0 & -1 & 2 \\ 0 & 0 & 2 \end{pmatrix}$. Then which of the following are true:

- 3, -1, and 2 are eigenvalues of A.
- $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue 3.

True/False

Answer: True

Just multiply it out.

- $\begin{pmatrix} -1/4 \\ 1 \\ 0 \end{pmatrix}$ is an eigenvector associated with eigenvalue -1.

True/False

Answer: True

Just multiply it out.

- $\begin{pmatrix} -1/4\chi_1 \\ \chi_1 \\ 0 \end{pmatrix}$ where $\chi_1 \neq 0$ is an eigenvector associated with eigenvalue -1.

True/False

Answer: True

Just multiply it out.

- $\begin{pmatrix} 1/3 \\ 2/3 \\ 1 \end{pmatrix}$ is an eigenvector associated with eigenvalue 2.

True/False

Answer: True

Just multiply it out.

☛ BACK TO TEXT

Homework 12.2.4.4 Let $A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} \\ 0 & \alpha_{1,1} & \alpha_{1,2} \\ 0 & 0 & \alpha_{2,2} \end{pmatrix}$. Then the eigenvalues of this matrix are $\alpha_{0,0}$, $\alpha_{1,1}$, and $\alpha_{2,2}$.

True/False

Answer: True

☛ BACK TO TEXT

Homework 12.2.4.5 Consider $A = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$. Which of the following is true about this matrix:

- $(1, \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix})$ is an eigenpair.
- $(0, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix})$ is an eigenpair.
- $(0, \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix})$ is an eigenpair.
- This matrix is defective.

Answer:

$$\det\left(\begin{pmatrix} 1-\lambda & 0 & 0 \\ 2 & -\lambda & 1 \\ 1 & 0 & -\lambda \end{pmatrix}\right) = \underbrace{\underbrace{[\ \underbrace{(1-\lambda)(-\lambda)^2}_{-(1-\lambda)\lambda^2} + 0 + 0]}_{\lambda^3-\lambda^2} - [\ \underbrace{0+0+0}_{0}\ ]}_{-(1-\lambda)\lambda^2}.$$

So, $\lambda_0 = \lambda_1 = 0$ is a double root, while $\lambda_2 = 1$ is the third root.

$\lambda_2 = 1$:	$\lambda_0 = \lambda_1 = 0$:
$$A - \lambda_2 I = \begin{pmatrix} 0 & 0 & 0 \\ 2 & -1 & 1 \\ 1 & 0 & -1 \end{pmatrix}$$ We wish to find a nonzero vector in the null space: $$\begin{pmatrix} 0 & 0 & 0 \\ 2 & -1 & 1 \\ 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$ By examination, I noticed that $$\begin{pmatrix} 0 & 0 & 0 \\ 2 & -1 & 1 \\ 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$ Eigenpair: $$(1, \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}).$$	$$A - 0I = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$ Reducing this to row-echelon form gives us the matrix $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}.$$ Notice that there is only one free variable, and hence this matrix is defective! The sole linearly independent eigenvector associated with $\lambda = 0$ is $$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$ Eigenpair: $$(0, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix})$$

☛ BACK TO TEXT

12.3 The General Case

12.3.1 Eigenvalues and Eigenvectors of $n \times n$ matrices: Special Cases

Homework 12.3.1.1 Let $A \in \mathbb{R}^{n \times n}$ be a diagonal matrix: $A = \begin{pmatrix} \alpha_{0,0} & 0 & 0 & \cdots & 0 \\ 0 & \alpha_{1,1} & 0 & \cdots & 0 \\ 0 & 0 & \alpha_{2,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_{n-1,n-1} \end{pmatrix}$. Then e_i is an eigenvector associated with eigenvalue $\alpha_{i,i}$.

True/False

Answer: True

Just multiply it out. Without loss of generality (which means: take as a typical case), let $i = 1$. Then

$$\begin{pmatrix} \alpha_{0,0} & 0 & 0 & \cdots & 0 \\ 0 & \alpha_{1,1} & 0 & \cdots & 0 \\ 0 & 0 & \alpha_{2,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_{n-1,n-1} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \alpha_{1,1} \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Here is another way of showing this, leveraging our notation: Partition

$$A = \left(\begin{array}{c|c|c} A_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & 0 \\ \hline 0 & 0 & A_{22} \end{array}\right) \quad \text{and} \quad e_j = \left(\begin{array}{c} 0 \\ \hline 1 \\ \hline 0 \end{array}\right),$$

where α_{11} denotes diagonal element $\alpha_{j,j}$. Then

$$\left(\begin{array}{c|c|c} A_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & 0 \\ \hline 0 & 0 & A_{22} \end{array}\right) \left(\begin{array}{c} 0 \\ \hline 1 \\ \hline 0 \end{array}\right) = \left(\begin{array}{c} 0 \\ \hline \alpha_{11} \\ \hline 0 \end{array}\right) = \alpha_{11} \left(\begin{array}{c} 0 \\ \hline 1 \\ \hline 0 \end{array}\right)$$

☛ BACK TO TEXT

Homework 12.3.1.2 Let $A = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & 0 & A_{22} \end{array}\right)$, where A_{00} is square. Then α_{11} is an eigenvalue of A and $\left(\begin{array}{c} -(A_{00} - \alpha_{11} I)^{-1} a_{01} \\ 1 \\ 0 \end{array}\right)$ is a corresponding eigenvalue (provided $A_{00} - \alpha_{11} I$ is nonsingular).

True/False

Answer: True

What we are going to show is that $(A - \alpha_{11} I)x = 0$ for the given vector.

$$\left(\begin{array}{c|c|c} (A_{00} - \alpha_{11} I) & a_{01} & A_{02} \\ \hline 0 & 0 & a_{12}^T \\ \hline 0 & 0 & (A_{22} - \alpha_{11} I) \end{array}\right) \left(\begin{array}{c} -(A_{00} - \alpha_{11} I)^{-1} a_{01} \\ 1 \\ 0 \end{array}\right)$$

$$= \left(\begin{array}{c} (A_{00} - \alpha_{11} I)[-(A_{00} - \alpha_{11} I)^{-1} a_{01}] + a_{01} + 0 \\ 0 + 0 + 0 \\ 0 \end{array}\right) = \left(\begin{array}{c} -a_{01} + a_{01} + 0 \\ 0 \\ 0 \end{array}\right) = \left(\begin{array}{c} 0 \\ \hline 0 \\ \hline 0 \end{array}\right)$$

A more constructive way of verifying the result is to notice that clearly

$$\left(\begin{array}{c|c|c} (A_{00} - \alpha_{11} I) & a_{01} & A_{02} \\ \hline 0 & 0 & a_{12}^T \\ \hline 0 & 0 & (A_{22} - \alpha_{11} I) \end{array}\right)$$

is singular since if one did Gaussian elimination with it, a zero pivot would be encountered exactly where the 0 in the middle appears. Now, consider a vector of form $\left(\begin{array}{c} x_0 \\ \hline 1 \\ \hline 0 \end{array}\right)$. Then

$$\left(\begin{array}{c|c|c} (A_{00} - \alpha_{11} I) & a_{01} & A_{02} \\ \hline 0 & 0 & a_{12}^T \\ \hline 0 & 0 & (A_{22} - \alpha_{11} I) \end{array}\right) \left(\begin{array}{c} x_0 \\ \hline 1 \\ \hline 0 \end{array}\right) = \left(\begin{array}{c} 0 \\ \hline 0 \\ \hline 0 \end{array}\right)$$

means that $(A_{00} - \alpha_{11} I)x_0 + a_{01} = 0$. (First component on both sides of the equation.) Solve this to find that $x_0 = -(A_{00} - \alpha_{11} I)^{-1} a_{01}$.

☛ BACK TO TEXT

Homework 12.3.1.3 The eigenvalues of a triangular matrix can be found on its diagonal.

True/False

Answer: True

☛ BACK TO TEXT

12.3.2 Eigenvalues of $n \times n$ Matrices

Homework 12.3.2.1 If $A \in \mathbb{R}^{n\times n}$, then $\Lambda(A)$ has n distinct elements.

True/False

Answer: False

The characteristic polynomial of A may have roots that have multiplicity greater than one. If $\Lambda(A) = \{\lambda_0, \lambda_1, \ldots, \lambda_{k-1}\}$, where $\lambda_i \neq \lambda_j$ if $i \neq j$, then

$$p_n(\lambda) = (\lambda-\lambda_0)^{n_0}(\lambda-\lambda_1)^{n_1}\cdots(\lambda-\lambda_{k-1})^{n_{k-1}}$$

with $n_0 + n_1 + \cdots + n_{k-1} = n$. Here n_j is the multiplicity of root λ_j.

☛ BACK TO TEXT

Homework 12.3.2.2 Let $A \in \mathbb{R}^{n\times n}$ and $\lambda \in \Lambda(A)$. Let S be the set of all vectors that satisfy $Ax = \lambda x$. (Notice that S is the set of all eigenvectors corresponding to λ *plus* the zero vector.) Then S is a subspace.

True/False

Answer: True

The easiest argument is to note that $Ax = \lambda x$ is the same as $(A-\lambda I)x = 0$ so that S is the null space of $(A-\lambda I)$. But the null space is a subspace, so S is a subspace.

Alternative proof: Let $x, y \in S$ and $\alpha \in \mathbb{R}$. Then

- $x + y \in S$: Since $x, y \in S$ we know that $Ax = \lambda x$ and $Ay = \lambda y$. But then

  $$A(x+y) = Ax + Ay = \lambda x + \lambda y = \lambda(x+y).$$

  Hence $x + y \in S$.

- $\alpha x \in S$: Since $x \in S$ we know that $Ax = \lambda x$. But then

  $$A(\alpha x) = A(\alpha x) = \alpha Ax = \alpha\lambda x = \lambda(\alpha x).$$

  Hence $\alpha x \in S$.

☛ BACK TO TEXT

12.3.3 Diagonalizing, Again

Homework 12.3.3.1 Consider $A = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{pmatrix}$.

- The algebraic multiplicity of $\lambda = 2$ is 4.
- The geometric multiplicity of $\lambda = 2$ is 2.

- The following vectors are linearly independent eigenvectors associated with $\lambda = 2$:

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \text{ and } \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}.$$

True/False

Answer: True

Just multiply out.

☛ BACK TO TEXT

Homework 12.3.3.2 Let $A \in \mathbb{A}^{n\times n}$, $\lambda \in \Lambda(A)$, and S be the set of all vectors x such that $Ax = \lambda x$. Finally, let λ have algebraic multiplicity k (meaning that it is a root of multiplicity k of the characteristic polynomial).

The dimension of S is k ($\dim(S) = k$).

Always/Sometimes/Never

Answer: Sometimes

An example of where it is true:

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

This matrix has 1 as its only eigenvalue, and it has algebraic multiplicity two. But both $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ are corresponding eigenvectors and hence $S = \mathbb{R}^2$, which has dimension 2, which equals the algebraic multiplicity of 1.

An example of where it is false:

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

This matrix has 1 as its only eigenvalue, and it has algebraic multiplicity two. Now, to find the eigenvectors we consider

$$\begin{pmatrix} 1-1 & 1 \\ 0 & 1-1 \end{pmatrix} = \begin{pmatrix} 0 & \boxed{1} \\ 0 & 0 \end{pmatrix}.$$

It is in row echelon form and has one pivot. Hence, the dimension of its null space is $2 - 1 = 1$. Since S is this null space, its dimension equals one.

☛ BACK TO TEXT

12.3.4 Properties of Eigenvalues and Eigenvectors

Homework 12.3.4.1 Let $A \in \mathbb{R}^{n\times n}$ and $A = \begin{pmatrix} A_{0,0} & A_{0,1} \\ 0 & A_{1,1} \end{pmatrix}$, where $A_{0,0}$ and $A_{1,1}$ are square matrices.

$\Lambda(A) = \Lambda(A_{0,0}) \cup \Lambda(A_{1,1})$.

Always/Sometimes/Never

Answer: Always

We will show that $\Lambda(A) \subset \Lambda(A_{0,0}) \cup \Lambda(A_{1,1})$ and $\Lambda(A_{0,0}) \cup \Lambda(A_{1,1}) \subset \Lambda(A)$.

$\underline{\Lambda(A) \subset \Lambda(A_{0,0}) \cup \Lambda(A_{1,1})}$: Let $\lambda \in \Lambda(A)$. Then there exists $x \neq 0$ such that $Ax = \lambda x$. Partition $x = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix}$. Then $\begin{pmatrix} A_{0,0} & A_{0,1} \\ 0 & A_{1,1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix}$ $\lambda \begin{pmatrix} x_0 \\ x_1 \end{pmatrix}$ which implies that $\begin{pmatrix} A_{0,0}x_0 + A_{0,1}x_1 \\ A_{1,1}x_1 \end{pmatrix} = \begin{pmatrix} \lambda x_0 \\ \lambda x_1 \end{pmatrix}$. Now, either $x_1 \neq 0$ (the zero vector), in which case $A_{1,1}x_1 = \lambda x_1$ and hence $\lambda \in \Lambda(A_{1,1})$, or $x_1 = 0$, in which case $A_{0,0}x_0 = \lambda x_0$ and hence $\lambda \in \Lambda(A_{0,0})$ since x_0 and x_1 cannot both equal zero vectors. Hence $\lambda \in \Lambda(A_{0,0})$ or $\lambda \in \Lambda(A_{1,1})$, which means that $\lambda \in \Lambda(A_{0,0}) \cup \Lambda(A_{1,1})$.

$\underline{\Lambda(A_{0,0}) \cup \Lambda(A_{1,1}) \subset \Lambda(A)}$: Let $\lambda \in \Lambda(A_{0,0}) \cup \Lambda(A_{1,1})$.

Case 1: $\lambda \in \Lambda(A_{0,0})$. Then there exists $x_0 \neq 0$ s.t. that $A_{0,0}x_0 = \lambda x_0$. Observe that

$$\begin{pmatrix} A_{0,0} & A_{0,1} \\ 0 & A_{1,1} \end{pmatrix} \underbrace{\begin{pmatrix} x_0 \\ 0 \end{pmatrix}}_{x} = \begin{pmatrix} \lambda x_0 \\ 0 \end{pmatrix} = \lambda \underbrace{\begin{pmatrix} x_0 \\ 0 \end{pmatrix}}_{x}.$$

Hence we have constructed a nonzero vector x such that $Ax = \lambda x$ and therefore $\lambda \in \Lambda(A)$.

Case 2: $\lambda \notin \Lambda(A_{0,0})$. Then there exists $x_1 \neq 0$ s.t. that $A_{1,1}x_1 = \lambda x_1$ (since $\lambda \in \Lambda(A_{1,1})$) *and* $A_{0,0} - \lambda I$ is nonsingular (and hence its inverse exists). Observe that

$$\underbrace{\begin{pmatrix} A_{0,0} - \lambda I & A_{0,1} \\ 0 & A_{1,1} - \lambda I \end{pmatrix}}_{A - \lambda I} \underbrace{\begin{pmatrix} -(A_{0,0} - \lambda I)^{-1} A_{0,1} x_1 \\ x_1 \end{pmatrix}}_{x} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Hence we have constructed a nonzero vector x such that $(A - \lambda I)x = 0$ and therefore $\lambda \in \Lambda(A)$.

☛ BACK TO TEXT

Homework 12.3.4.2 Let $A \in \mathbb{R}^{n \times n}$ be symmetric, $\lambda_i \neq \lambda_j$, $Ax_i = \lambda_i x_i$ and $Ax_j = \lambda_j x_j$.

$x_i^T x_j = 0$

Always/Sometimes/Never

Answer: Always

$$\left\{\begin{array}{rcl} Ax_i & = & \lambda_i x_i \\ Ax_j & = & \lambda_j x_j \end{array}\right.$$

implies $<$ Multiplying both sides by transpose of same vector maintains equivalence $>$

$$\left\{\begin{array}{rcl} x_j^T A x_i & = & x_j^T(\lambda_i x_i) \\ x_i^T A x_j & = & x_i^T(\lambda_j x_j) \end{array}\right.$$

implies $<$ Move scalar to front $>$

$$\left\{\begin{array}{rcl} x_j^T A x_i & = & \lambda_i x_j^T x_i \\ x_i^T A x_j & = & \lambda_j x_i^T x_j \end{array}\right.$$

implies $<$ Transposing both sides maintains equivalence $>$

$$\left\{\begin{array}{rcl} (x_j^T A x_i)^T & = & (\lambda_i x_j^T x_i)^T \\ x_i^T A x_j & = & \lambda_j x_i^T x_j \end{array}\right.$$

implies $<$ Property of transposition of product $>$

$$\left\{\begin{array}{rcl} x_i^T A^T x_j & = & \lambda_i x_i^T x_j \\ x_i^T A x_j & = & \lambda_j x_i^T x_j \end{array}\right.$$

implies $< A = A^T >$

$$\begin{array}{rcl} x_i^T A^T x_j & = & \lambda_i x_i^T x_j \\ \| & & \\ x_i^T A x_j & = & \lambda_j x_i^T x_j \end{array}$$

implies $<$ Transitivity of equivalence $>$

$$\lambda_i x_i^T x_j = \lambda_j x_i^T x_j$$

implies $<$ Since $\lambda_i \neq \lambda_j$ $>$

$$x_i^T x_j = 0$$

☛ BACK TO TEXT

Homework 12.3.4.3 If $Ax = \lambda x$ then $AAx = \lambda^2 x$. (AA is often written as A^2.)

Always/Sometimes/Never

Answer: Always

$AAx = A(\lambda x) = \lambda Ax = \lambda\lambda x = \lambda^2 x.$

☛ BACK TO TEXT

Homework 12.3.4.4 Let $Ax = \lambda x$ and $k \geq 1$. Recall that $A^k = \underbrace{AA\cdots A}_{k \text{ times}}$.

$A^k x = \lambda^k x.$

Always/Sometimes/Never

Answer: Always

Proof by induction.

Base case: $k = 1$.

$$A^k x = A^1 x = Ax = \lambda x = \lambda^k x.$$

Inductive hypothesis: Assume that $A^k x = \lambda^k x$ for $k = K$ with $K \geq 1$.

We will prove that $A^k x = \lambda^k x$ for $k = K+1$.

$$
\begin{array}{ll}
A^k x & \\
\quad = & < k = K+1 > \\
A^{K+1} x & \\
\quad = & < \text{Definition of } A^k > \\
(AA^K)x & \\
\quad = & < \text{Associativity of matrix multiplication} > \\
A(A^K x) & \\
\quad = & < \text{I.H.} > \\
A(\lambda^K x) & \\
\quad = & < Ax \text{ is a linear transformation} > \\
\lambda^K Ax & \\
\quad = & < Ax = \lambda x > \\
\lambda^K \lambda x & \\
\quad = & < \text{Algebra} > \\
\lambda^{K+1} x & \\
\quad = & < k = K+1 > \\
\lambda^k x &
\end{array}
$$

We conclude that $A^k x = \lambda^k x$ for $k = K+1$.

By the Principle of Mathematical Induction the result holds for $k \geq 1$.

☛ BACK TO TEXT

Homework 12.3.4.5 $A \in \mathbb{R}^{n \times n}$ is nonsingular if and only if $0 \notin \Lambda(A)$.

True/False

Answer: True

($\Rightarrow$) Assume A is nonsingular. Then $Ax = 0$ only if $x = 0$. But that means that there is no nonzero vector x such that $Ax = 0x$. Hence $0 \notin \Lambda(A)$.

($\Leftarrow$) Assume $0 \notin \Lambda(A)$. Then $Ax = 0$ must imply that $x = 0$ since otherwise $Ax = 0x$. Therefore A is nonsingular.

☛ BACK TO TEXT

12.4 Practical Methods for Computing Eigenvectors and Eigenvalues

12.4.1 Predicting the Weather, One Last Time

Homework 12.4.1.1 If $\lambda \in \Lambda(A)$ then $\lambda \in \Lambda(A^T)$.

True/False

Answer: True

$$
\begin{aligned}
&\lambda \in \Lambda(A) \quad \Rightarrow \quad (A - \lambda I) \text{ is singular} \\
&\quad \Rightarrow \quad < \text{ equivalent conditions } > \\
&\dim(\mathcal{N}(A - \lambda I)) = k > 0 \\
&\quad \Rightarrow \quad < A \text{ is square, fundamental space picture } > \\
&\dim(\mathcal{N}((A - \lambda I)^T)) = k > 0 \\
&\quad \Rightarrow \quad < (A - \lambda I)^T = A^T - \lambda I > \\
&\dim(\mathcal{N}(A^T - \lambda I)) = k > 0 \\
&\quad \Rightarrow \quad < \text{ equivalent conditions } > \\
&(A^T - \lambda I) \text{ is singular} \\
&\quad \Rightarrow \quad < \text{ property of eigenvalue } > \\
&\lambda \in \Lambda(A^T)
\end{aligned}
$$

☛ BACK TO TEXT

Homework 12.4.1.2 $\lambda \in \Lambda(A)$ if and only if $\lambda \in \Lambda(A^T)$.

True/False

Answer: True

This follows immediately from the last homework and the fact that $(A^T)^T = A$.

☛ BACK TO TEXT

12.4.2 The Power Method

Homework 12.4.2.1 Let $A \in \mathbb{R}^{n \times n}$ and $\mu \neq 0$ be a scalar. Then $\lambda \in \Lambda(A)$ if and only if $\lambda/\mu \in \Lambda(\frac{1}{\mu}A)$.

True/False

Answer: True

☛ BACK TO TEXT

Homework 12.4.2.2 We now walk you through a simple implementation of the Power Method, referring to files in directory `LAFF-2.0xM/Programming/Week12`.

We want to work with a matrix A for which we know the eigenvalues. Recall that a matrix A is diagonalizable if and only if there exists a nonsingular matrix V and diagonal matrix Λ such that $A = V\Lambda V^{-1}$. The diagonal elements of Λ then equal the eigenvalues of A and the columns of V the eigenvectors.

Thus, given eigenvalues, we can create a matrix A by creating a diagonal matrix with those eigenvalues on the diagonal and a random nonsingular matrix V, after which we can compute A to equal $V\Lambda V^{-1}$. This is accomplished by the function

[A, V] = CreateMatrixForEigenvalueProblem(eigs)

(see file `CreateMatrixForEigenvalueProblem.m`).

The script in `PowerMethodScript.m` then illustrates how the Power Method, starting with a random vector, computes an eigenvector corresponding to the eigenvalue that is largest in magnitude, and via the Rayleigh quotient (a way for computing an eigenvalue given an eigenvector that is discussed in the next unit) an approximation for that eigenvalue.

To try it out, in the Command Window type

```
>> PowerMethodScript
input a vector of eigenvalues. e.g.: [ 4; 3; 2; 1 ]
[ 4; 3; 2; 1 ]
```

The script for each step of the Power Method reports for the current iteration the length of the component orthogonal to the eigenvector associated with the eigenvalue that is largest in magnitude. If this component becomes small, then the vector lies approximately in the direction of the desired eigenvector. The Rayleigh quotient slowly starts to get close to the eigenvalue that is largest in magnitude. The slow convergence is because the ratio of the second to largest and the largest eigenvalue is not much smaller than 1.

Try some other distributions of eigenvalues. For example, `[ 4; 1; 0.5; 0.25]`, which should converge faster, or `[ 4; 3.9; 2; 1 ]`, which should converge much slower.

You may also want to try `PowerMethodScript2.m`, which illustrates what happens if there are two eigenvalues that are equal in value and both largest in magnitude (relative to the other eigenvalues).

☛ BACK TO TEXT

12.4.3 In Preparation for this Week's Enrichment

Homework 12.4.3.1 Let $A \in \mathbb{R}^{n\times n}$ and x equal an eigenvector of A. Assume that x is real valued as is the eigenvalue λ with $Ax = \lambda x$.

$\lambda = \frac{x^T Ax}{x^T x}$ is the eigenvalue associated with the eigenvector x.

Always/Sometimes/Never

Answer: Always

$Ax = \lambda x$ implies that $x^T Ax = x^T(\lambda x) = \lambda x^T x$. But $x^T x \neq 0$ since x is an eigenvector. Hence $\lambda = x^T Ax/(x^T x)$.

☛ BACK TO TEXT

Homework 12.4.3.2 Let $A \in \mathbb{R}^{n\times n}$ be nonsingular, $\lambda \in \Lambda(A)$, and $Ax = \lambda x$. Then $A^{-1}x = \frac{1}{\lambda}x$.

True/False

Answer: True

$Ax = \lambda x$ means that $\frac{1}{\lambda}A^{-1}Ax = \frac{1}{\lambda}A^{-1}\lambda x$ which means that $\frac{1}{\lambda}x = A^{-1}x$.

☛ BACK TO TEXT

Homework 12.4.3.3 Let $A \in \mathbb{R}^{n\times n}$ and $\lambda \in \Lambda(A)$. Then $(\lambda - \mu) \in \Lambda(A - \mu I)$.

True/False

Answer: True

Let $Ax = \lambda x$ for $x \neq 0$. Then

$$(A - \mu I)x = Ax - \mu Ix = Ax - \mu x = \lambda x - \mu x = (\lambda - \mu)x.$$

☛ BACK TO TEXT

Appendix A

LAFF Routines (FLAME@lab)

Figure A summarizes the most important routines that are part of the `laff` FLAME@lab (MATLAB) library used in these materials.

!!

Operation Abbrev.	Definition	Function	Approx. cost	
			flops	memops
Vector-vector operations				
Copy (COPY)	$y := x$	`y = laff_copy( x, y )`	0	$2n$
Vector scaling (SCAL)	$x := \alpha x$	`x = laff_scal( alpha, x )`	n	$2n$
Vector scaling (SCAL)	$x := x/\alpha$	`x = laff_invscal( alpha, x )`	n	$2n$
Scaled addition (AXPY)	$y := \alpha x + y$	`y = laff_axpy( alpha, x, y )`	$2n$	$3n$
Dot product (DOT)	$\alpha := x^T y$	`alpha = laff_dot( x, y )`	$2n$	$2n$
Dot product (DOTS)	$\alpha := x^T y + \alpha$	`alpha = laff_dots( x, y, alpha )`	$2n$	$2n$
Length (NORM2)	$\alpha := \|x\|_2$	`alpha = laff_norm2( x )`	$2n$	n
Matrix-vector operations				
General matrix-vector multiplication (GEMV)	$y := \alpha Ax + \beta y$ $y := \alpha A^T x + \beta y$	`y = laff_gemv( 'No transpose', alpha, A, x, beta, y )` `y = laff_gemv( 'Transpose', alpha, A, x, beta, y )`	$2mn$ $2mn$	mn mn
Rank-1 update (GER)	$A := \alpha x y^T + A$	`A = laff_ger( alpha, x, y, A )`	$2mn$	mn

Triangular matrix solve (TRSV)	$b := L^{-1}b, b := U^{-1}b$
	$b := L^{-T}b, b := U^{-T}b$
		example:
		`b = laff_trsv( 'Upper triangular', 'No transpose',`
		`'Nonunit diagonal', U, b )`
			n^2	$n^2/2$

Triangular matrix-vector multiply (TRMV)	$x := Lx, x := Ux$
	$x := L^T x, x := U^T x$
		example:
		`x = laff_trmv( 'Upper triangular', 'No transpose',`
		`'Nonunit diagonal', U, x )`
			n^2	$n^2/2$

Matrix-matrix operations				

General matrix-matrix multiplication (GEMM)	$C := \alpha AB + \beta C$
	$C := \alpha A^T B + \beta C$
	$C := \alpha AB^T + \beta C$
	$C := \alpha A^T B^T + \beta C$
		example:
		`C = laff_gemm( 'Transpose', 'No transpose,`
		`alpha, A, B, beta, C )`
			$2mnk$	$2mn + mk + nk$

Triangular solve with MRHs (TRSM)	$B := \alpha L^{-1} B$
	$B := \alpha U^{-T} B$
	$B := \alpha B L^{-1}$
	$B := \alpha B U^{-T}$
		example:
		`B = laff_trsm( 'Left', 'Lower triangular',`
		`'No transpose', 'Nonunit diagonal',`
		`alpha, U, B )`
			$m^2 n$
			$m^2 n$
				$m^2 + mn$
				$m^2 + mn$

Appendix B

“What You Will Learn” Check List

Estimate to what level you have mastered the items in each week’s “What You Will Learn”. Often, it is best to move on even if you feel you are still somewhat unsure about a given item. You can always revisit the subject later.

Printable Version (https://s3.amazonaws.com/ulaff/Documents/WhatYouWillLearn.pdf)

1 - not yet mastered. 5 - mastered.

Week 1: Vectors in Linear Algebra

Represent quantities that have a magnitude and a direction as vectors.	1 2 3 4 5
Read, write, and interpret vector notations.	1 2 3 4 5
Visualize vectors in $\mathbb{R}^2$.	1 2 3 4 5
Perform the vector operations of scaling, addition, dot (inner) product.	1 2 3 4 5
Reason and develop arguments about properties of vectors and operations defined on them.	1 2 3 4 5
Compute the (Euclidean) length of a vector.	1 2 3 4 5
Express the length of a vector in terms of the dot product of that vector with itself.	1 2 3 4 5
Evaluate a vector function.	1 2 3 4 5
Solve simple problems that can be represented with vectors.	1 2 3 4 5
Create code for various vector operations and determine their cost functions in terms of the size of the vectors.	1 2 3 4 5
Gain an awareness of how linear algebra software evolved over time and how our programming assignments fit into this (enrichment).	1 2 3 4 5
Become aware of overflow and underflow in computer arithmetic (enrichment).	1 2 3 4 5

Week 2: Linear Transformations and Matrices

Determine if a given vector function is a linear transformation.	1 2 3 4 5
Identify, visualize, and interpret linear transformations.	1 2 3 4 5
Recognize rotations and reflections in 2D as linear transformations of vectors.	1 2 3 4 5
Relate linear transformations and matrix-vector multiplication.	1 2 3 4 5
Understand and exploit how a linear transformation is completely described by how it transforms the unit basis vectors.	1 2 3 4 5
Find the matrix that represents a linear transformation based on how it transforms unit basis vectors.	1 2 3 4 5
Perform matrix-vector multiplication.	1 2 3 4 5
Reason and develop arguments about properties of linear transformations and matrix vector multiplication.	1 2 3 4 5
Read, appreciate, understand, and develop inductive proofs. (Ideally you will fall in love with them! They are beautiful. They don't deceive you. You can count on them. You can build on them. The perfect life companion! But it may not be love at first sight.)	1 2 3 4 5 1 2 3 4 5
Make conjectures, understand proofs, and develop arguments about linear transformations.	1 2 3 4 5
Understand the connection between linear transformations and matrix-vector multiplication.	1 2 3 4 5
Solve simple problems related to linear transformations.	1 2 3 4 5

Week 3: Linear Transformations and Matrices

Recognize matrix-vector multiplication as a linear combination of the columns of the matrix.	1 2 3 4 5
Given a linear transformation, determine the matrix that represents it.	1 2 3 4 5
Given a matrix, determine the linear transformation that it represents.	1 2 3 4 5
Connect special linear transformations to special matrices.	1 2 3 4 5
Identify special matrices such as the zero matrix, the identity matrix, diagonal matrices, triangular matrices, and symmetric matrices.	1 2 3 4 5
Transpose a matrix.	1 2 3 4 5
Scale and add matrices.	1 2 3 4 5
Exploit properties of special matrices.	1 2 3 4 5
Extrapolate from concrete computation to algorithms for matrix-vector multiplication.	1 2 3 4 5
Partition (slice and dice) matrices with and without special properties.	1 2 3 4 5
Use partitioned matrices and vectors to represent algorithms for matrix-vector multiplication.	1 2 3 4 5
Use partitioned matrices and vectors to represent algorithms in code.	1 2 3 4 5

Week 4: From Matrix-Vector Multiplication to Matrix-Matrix Multiplication

Apply matrix vector multiplication to predict the probability of future states in a Markov process.	1 2 3 4 5
Make use of partitioning to perform matrix vector multiplication.	1 2 3 4 5
Transpose a partitioned matrix.	1 2 3 4 5
Partition conformally, ensuring that the size of the matrices and vectors match so that matrix-vector multiplication works.	1 2 3 4 5
Take advantage of special structures to perform matrix-vector multiplication with triangular and symmetric matrices.	1 2 3 4 5
Express and implement various matrix-vector multiplication algorithms using the FLAME notation and FlamePy.	1 2 3 4 5
Make connections between the composition of linear transformations and matrix-matrix multiplication.	1 2 3 4 5
Compute a matrix-matrix multiplication.	1 2 3 4 5
Recognize scalars and column/row vectors as special cases of matrices.	1 2 3 4 5
Compute common vector-vector and matrix-vector operations as special cases of matrix-matrix multiplication.	1 2 3 4 5
Compute an outer product xy^T as a special case of matrix-matrix multiplication and recognize that • The rows of the resulting matrix are scalar multiples of y^T. • The columns of the resulting matrix are scalar multiples of x.	1 2 3 4 5

Week 5: Matrix-Matrix Multiplication

Recognize that matrix-matrix multiplication is not commutative.	1 2 3 4 5
Relate composing rotations to matrix-matrix multiplication.	1 2 3 4 5
Fluently compute a matrix-matrix multiplication.	1 2 3 4 5
Perform matrix-matrix multiplication with partitioned matrices.	1 2 3 4 5
Identify, apply, and prove properties of matrix-matrix multiplication, such as $(AB)^T = B^T A^T$.	1 2 3 4 5
Exploit special structure of matrices to perform matrix-matrix multiplication with special matrices, such as identity, triangular, and diagonal matrices.	1 2 3 4 5
Identify whether or not matrix-matrix multiplication preserves special properties in matrices, such as symmetric and triangular structure.	1 2 3 4 5
Express a matrix-matrix multiplication in terms of matrix-vector multiplications, row vector times matrix multiplications, and rank-1 updates.	1 2 3 4 5
Appreciate how partitioned matrix-matrix multiplication enables high performance. (Optional, as part of the enrichment.)	1 2 3 4 5

Week 6: Gaussian Elimination

Apply Gaussian elimination to reduce a system of linear equations into an upper triangular system of equations.	1 2 3 4 5
Apply back(ward) substitution to solve an upper triangular system in the form Ux = b.	1 2 3 4 5
Apply forward substitution to solve a lower triangular system in the form Lz = b.	1 2 3 4 5
Represent a system of equations using an appended matrix.	1 2 3 4 5
Reduce a matrix to an upper triangular matrix with Gauss transforms and then apply the Gauss transforms to a right-hand side.	1 2 3 4 5
Solve the system of equations in the form Ax = b using LU factorization.	1 2 3 4 5
Relate LU factorization and Gaussian elimination.	1 2 3 4 5
Relate solving with a unit lower triangular matrix and forward substitution.	1 2 3 4 5
Relate solving with an upper triangular matrix and back substitution.	1 2 3 4 5
Create code for various algorithms for Gaussian elimination, forward substitution, and back substitution.	1 2 3 4 5
Determine the cost functions for LU factorization and algorithms for solving with triangular matrices.	1 2 3 4 5

Week 7: More Gaussian Elimination and Matrix Inversion

Determine, recognize, and apply permutation matrices.	1 2 3 4 5
Apply permutation matrices to vectors and matrices.	1 2 3 4 5
Identify and interpret permutation matrices and fluently compute the multiplication of a matrix on the left and right by a permutation matrix.	1 2 3 4 5
Reason, make conjectures, and develop arguments about properties of permutation matrices.	1 2 3 4 5
Recognize when Gaussian elimination breaks down and apply row exchanges to solve the problem when appropriate.	1 2 3 4 5
Recognize when LU factorization fails and apply row pivoting to solve the problem when appropriate.	1 2 3 4 5
Recognize that when executing Gaussian elimination (LU factorization) with $Ax = b$ where A is a square matrix, one of three things can happen: 1. The process completes with no zeroes on the diagonal of the resulting matrix U. Then $A = LU$ and $Ax = b$ has a unique solution, which can be found by solving $Lz = b$ followed by $Ux = z$. 2. The process requires row exchanges, completing with no zeroes on the diagonal of the resulting matrix U. Then $PA = LU$ and $Ax = b$ has a unique solution, which can be found by solving $Lz = Pb$ followed by $Ux = z$. 3. The process requires row exchanges, but at some point no row can be found that puts a nonzero on the diagonal, at which point the process fails (unless the zero appears as the last element on the diagonal, in which case it completes, but leaves a zero on the diagonal of the upper triangular matrix). In Week 8 we will see that this means $Ax = b$ does not have a unique solution.	1 2 3 4 5
Reason, make conjectures, and develop arguments about properties of inverses.	1 2 3 4 5
Find the inverse of a simple matrix by understanding how the corresponding linear transformation is related to the matrix-vector multiplication with the matrix.	1 2 3 4 5
Identify and apply knowledge of inverses of special matrices including diagonal, permutation, and Gauss transform matrices.	1 2 3 4 5
Determine whether a given matrix is an inverse of another given matrix.	1 2 3 4 5
Recognize that a 2×2 matrix $A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}$ has an inverse if and only if its determinant is not zero: $\det(A) = \alpha_{0,0}\alpha_{1,1} - \alpha_{0,1}\alpha_{1,0} \neq 0$.	1 2 3 4 5
Compute the inverse of a 2×2 matrix A if that inverse exists.	1 2 3 4 5

Week 8: More on Matrix Inversion

Determine with Gaussian elimination (LU factorization) when a system of linear equations with n equations in n unknowns does not have a unique solution.	1 2 3 4 5
Understand and apply Gauss Jordan elimination to solve linear systems with one or more right-hand sides and to find the inverse of a matrix.	1 2 3 4 5
Identify properties that indicate a linear transformation has an inverse.	1 2 3 4 5
Identify properties that indicate a matrix has an inverse.	1 2 3 4 5
Create an algorithm to implement Gauss-Jordan elimination and determine the cost function.	1 2 3 4 5
Recognize and understand that inverting a matrix is not the method of choice for solving a linear system.	1 2 3 4 5
Identify specialized factorizations of matrices with special structure and/or properties and create algorithms that take advantage of this (enrichment).	1 2 3 4 5

Week 9: Vector Spaces

Determine when systems do not have a unique solution and recognize the general solution for a system.	1 2 3 4 5
Use and understand set notation.	1 2 3 4 5
Determine if a given subset of $\mathbb{R}^n$ is a subspace.	1 2 3 4 5
For simple examples, determine the null space and column space for a given matrix.	1 2 3 4 5
Identify, apply, and prove simple properties of sets, vector spaces, subspaces, null spaces and column spaces.	1 2 3 4 5
Recognize for simple examples when the span of two sets of vectors is the same.	1 2 3 4 5
Determine when a set of vectors is linearly independent by exploiting special structures. For example, relate the rows of a matrix with the columns of its transpose to determine if the matrix has linearly independent rows.	1 2 3 4 5
For simple examples, find a basis for a subspace and recognize that while the basis is not unique, the number of vectors in the basis is.	1 2 3 4 5

Week 10:. Vector Spaces, Orthogonality, and Linear Least Squares

Determine when linear systems of equations have a unique solution, an infinite number of solutions, or only approximate solutions.	1 2 3 4 5
Determine the row-echelon form of a system of linear equations or matrix and use it to • find the pivots, • decide the free and dependent variables, • establish specific (particular) and general (complete) solutions, • find a basis for the column space, the null space, and row space of a matrix, • determine the rank of a matrix, and/or • determine the dimension of the row and column space of a matrix.	1 2 3 4 5
Picture and interpret the fundamental spaces of matrices and their dimensionalities.	1 2 3 4 5
Indicate whether vectors are orthogonal and determine whether subspaces are orthogonal.	1 2 3 4 5
Determine the null space and column space for a given matrix and connect the row space of A with the column space of A^T.	1 2 3 4 5
Identify, apply, and prove simple properties of vector spaces, subspaces, null spaces and column spaces.	1 2 3 4 5
Determine when a set of vectors is linearly independent by exploiting special structures. For example, relate the rows of a matrix with the columns of its transpose to determine if the matrix has linearly independent rows.	1 2 3 4 5
Approximate the solution to a system of linear equations of small dimension using the method of normal equations to solve the linear least-squares problem.	1 2 3 4 5

Week 11:. Orthogonal Projection, Low Rank Approximation, and Orthogonal Bases

Given vectors a and b in $\mathbb{R}^m$, find the component of b in the direction of a and the component of b orthogonal to a.	1 2 3 4 5
Given a matrix A with linear independent columns, find the matrix that projects any given vector b onto the column space A and the matrix that projects b onto the space orthogonal to the column space of A, which is also called the left null space of A.	1 2 3 4 5
Understand low rank approximation, projecting onto columns to create a rank-k approximation.	1 2 3 4 5
Identify, apply, and prove simple properties of orthonormal vectors.	1 2 3 4 5
Determine if a set of vectors is orthonormal.	1 2 3 4 5
Transform a set of basis vectors into an orthonormal basis using Gram-Schmidt orthogonalization.	1 2 3 4 5
Compute an orthonormal basis for the column space of A.	1 2 3 4 5
Apply Gram-Schmidt orthogonalization to compute the QR factorization.	1 2 3 4 5
Solve the Linear Least-Squares Problem via the QR Factorization.	1 2 3 4 5
Make a change of basis.	1 2 3 4 5
Be aware of the existence of the Singular Value Decomposition and that it provides the “best” rank-k approximation.	1 2 3 4 5

Week 12:. Eigenvalues, Eigenvectors, and Diagonalization

Determine whether a given vector is an eigenvector for a particular matrix.	1 2 3 4 5
Find the eigenvalues and eigenvectors for small-sized matrices.	1 2 3 4 5
Identify eigenvalues of special matrices such as the zero matrix, the identity matrix, diagonal matrices, and triangular matrices.	1 2 3 4 5
Interpret an eigenvector of A, as a direction in which the "action" of A, Ax, is equivalent to x being scaled without changing its direction. (Here scaling by a negative value still leaves the vector in the same direction.) Since this is true for any scalar multiple of x, it is the direction that is important, not the length of x.	1 2 3 4 5
Compute the characteristic polynomial for 2×2 and 3×3 matrices.	1 2 3 4 5
Know and apply the property that a matrix has an inverse if and only if its determinant is nonzero.	1 2 3 4 5
Know and apply how the roots of the characteristic polynomial are related to the eigenvalues of a matrix.	1 2 3 4 5
Recognize that if a matrix is real valued, then its characteristic polynomial has real valued coefficients but may still have complex eigenvalues that occur in conjugate pairs.	1 2 3 4 5
Link diagonalization of a matrix with the eigenvalues and eigenvectors of that matrix.	1 2 3 4 5
Make conjectures, reason, and develop arguments about properties of eigenvalues and eigenvectors.	1 2 3 4 5
Understand practical algorithms for finding eigenvalues and eigenvectors such as the power method for finding an eigenvector associated with the largest eigenvalue (in magnitude).	1 2 3 4 5

Index