Chapter 4

LU Factorization

In this chapter, 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 (it has ones on the diagonal) and U is upper triangular. When solving the linear system of equations (in matrix notation) $Ax = b$, one can substitute $A = LU$ to find that $(LU)x = b$. Now, using the associative properties of matrix multiplication, we find that $L(Ux) = b$. Substituting in $z = Ux$, this means that $Lz = b$. Thus, solving $Ax = b$ can be accomplished via the steps

- Solve $Lz = b$; followed by
- Solve $Ux = z$.

What we will show is that this process is equivalent to Gaussian elimination with the augmented system $\left(\begin{array}{c|c} A & b \end{array} \right)$ followed by backward substitution.

Next, we will discuss how to overcome some of the conditions under which the precedure breaks down, leading to LU factorization with pivoting.

The reader will notice that this chapter starts where the last chapter ended: with $A = LU$ and how to compute this decomposition more directly. We will show how starting from this point leads directly to the algorithm in Figure 3.6. We then work backwards, exposing once again how Gauss transforms fit into the picture. This allows us to then introduce swapping of equations (pivoting) into the basic algorithm. So, be prepared for seeing some of the same material again, under a slightly different light.

4.1 Gaussian Elimination Once Again

In Figure 4.1 we illustrate how Gaussian Elimination is used to transform a linear system of three equations in three unknowns into an upper triangular system by considering the problem

$$\begin{array}{rrrr} -2\chi_0 - & \chi_1 + & \chi_2 = & 6 \\ 2\chi_0 - & 2\chi_1 - & 3\chi_2 = & 3 \\ -4\chi_0 + & 4\chi_1 + & 7\chi_2 = & -3 \end{array} \tag{4.1}$$

- Step 1: A multiple of the first row is substracted from the second row to eliminate the "χ_0" term. This multiple is computed from the coefficient of the term to be eliminated and the coefficient of the same term in the first equation.
- Step 2: Similarly, a multiple of the first row is substracted from the third row.
- Step 3: A multiple of the second row is substracted from the third row to eliminate the "χ_1" term.

This leaves the upper triangular system

$$\begin{array}{rrrr} -2\chi_0 - & \chi_1 + & \chi_2 = & 6 \\ & -3\chi_1 - & 2\chi_2 = & 9 \\ & & \chi_2 = & 3 \end{array}$$

which is easier to solve, via backward substitution to be discussed later.

In Figure 4.2 we again show how it is not necessary to write down the entire linear equations: it suffices to perform Gaussian elimination on the matrix of coefficients, augmented by the right-hand side.

4.2 LU factorization

Next, let us consider the computation of the LU factorization of a square matrix. We will ignore for now when this factorization can be computed, and focus on the computation itself.

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

$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array} \right), \quad L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array} \right), \quad \text{and} \quad U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array} \right),$$

where we use our usual notation that lower-case Greek letter denote scalars, lower-case Roman letters vectors, and upper-case Roman letters matrices. Now, $A = LU$ implies (using

Step	Current system	Operation
1	$-2\chi_0 - \chi_1 + \chi_2 = 6$ $2\chi_0 - 2\chi_1 - 3\chi_2 = 3$ $-4\chi_0 + 4\chi_1 + 7\chi_2 = -3$	$2\chi_0 - 2\chi_1 - 3\chi_2 = 3$ $-\left(\frac{2}{-2}\right) \times (-2\chi_0 - \chi_1 + \chi_2 = 6)$ $-3\chi_1 - 2\chi_2 = 9$
2	$-2\chi_0 - \chi_1 + \chi_2 = 6$ $-3\chi_1 - 2\chi_2 = 9$ $-4\chi_0 + 4\chi_1 + 7\chi_2 = -3$	$2\chi_0 - 2\chi_1 - 3\chi_2 = 3$ $-\left(\frac{-4}{-2}\right) \times (-4\chi_0 + 4\chi_1 + 7\chi_2 = -3)$ $6\chi_1 + 5\chi_2 = -15$
3	$-2\chi_0 - \chi_1 + \chi_2 = 6$ $-3\chi_1 - 2\chi_2 = 9$ $6\chi_1 + 5\chi_2 = -15$	$6\chi_1 + 5\chi_2 = -15$ $-\left(\frac{6}{-3}\right) \times (-3\chi_1 - 2\chi_2 = 9)$ $\chi_2 = 3$
4	$-2\chi_0 - \chi_1 + \chi_2 = 6$ $-3\chi_1 - 2\chi_2 = 9$ $\chi_2 = 3$	

Figure 4.1: Gaussian Elimination on a linear system of three equations in three unknowns.

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 4.2: Gaussian Elimination with an augmented matrix of coefficients. (Compare and contrast with Figure 4.1.

what we learned about multiplying matrices that have been partitioned into submatrices)

$$\overbrace{\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right)}^{A} = \overbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)}^{L} \overbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)}^{U} = \overbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline l_{21}\upsilon_{11} & l_{21}u_{12}^T + L_{22}U_{22} \end{array}\right)}^{LU}.$$

For two matrices to be equal, their elements must be equal, and therefore, if they are partitioned conformally, their submatrices must be equal:

$$\begin{array}{c|c} \alpha_{11} = \upsilon_{11} & a_{12}^T = u_{12}^T \\ \hline a_{21} = l_{21}\upsilon_{11} & A_{22} = l_{21}u_{12}^T + L_{22}U_{22} \end{array}$$

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 $a_{21} = a_{21}/\alpha_{11} (= l_{21})$.
- Update $A_{22} = A_{22} - a_{21}a_{12}^T (= A_{22} - l_{21}u_{12}^T)$.
- Overwrite A_{22} with L_{22} and U_{22} by continuing recursively 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. (Notice that the diagonal elements of L need not be stored, since they are known to equal one.)

This algorithm is presented in Figure 4.3 as a loop-based algorithm.

Example 4.1 The LU factorization algorithm in Figure 4.3 is illustrated in Figure 4.4 for the coefficient matrix from Figure 4.1. Examination of the computations in Figure 4.2 and Figure 4.4 highlights how Gaussian elimination and LU factorization require the same computations on the coefficient matrix. To make it easy to do this comparison, we repeat these figures in Figures. 4.5–4.6.

Of course, we arrive at the same conclusion (that Gaussian elimination on the coefficient matrix is the same as LU factorization on that matrix) by comparing the algorithms in Figure 4.3 to Figure 3.6.

Remark 4.2 LU factorization and Gaussian elimination with the coefficient matrix are one and the same computation.

$A := \text{LU\_UNB}(A)$

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 4.3: Algorithm for computing the LU factorization. Notice that this is exactly the algorithm in Figure 3.6.

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	$\left(\begin{array}{\|c\|cc} \hline -2 & -1 & 1 \\ \hline 2 & -2 & -3 \\ -4 & 4 & 7 \end{array}\right)$	$\begin{pmatrix} 2 \\ -4 \end{pmatrix} / (-2) = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$	$\begin{pmatrix} -2 & -3 \\ 4 & 7 \end{pmatrix} - \begin{pmatrix} -1 \\ 2 \end{pmatrix} \begin{pmatrix} -1 & 1 \end{pmatrix} = \begin{pmatrix} -3 & -2 \\ 6 & 5 \end{pmatrix}$
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 4.4: LU factorization of a 3×3 matrix. Here, "Step" refers to the corresponding step for Gaussian Elimination in Figure 4.2

Step	Current system	Multiplier	Operation
1	$\left(\begin{array}{rrrr} -2 & -1 & 1 & 6 \\ 2 & -2 & -3 & 3 \\ -4 & 4 & 7 & -3 \end{array}\right)$	$\frac{2}{-2} = -1$	$\begin{array}{rrrrr} & 2 & -2 & -3 & 3 \\ -1\times(& -2 & -1 & 1 & 6) \\ \hline & 0 & -3 & -2 & 9 \end{array}$
2	$\left(\begin{array}{rrrr} -2 & -1 & 1 & 6 \\ 0 & -3 & -2 & 9 \\ -4 & 4 & 7 & -3 \end{array}\right)$	$\frac{-4}{-2} = 2$	$\begin{array}{rrrrr} & -4 & 4 & 7 & -3 \\ -(2)\times(& -2 & -1 & 1 & 6) \\ \hline & 0 & 6 & 5 & -15 \end{array}$
3	$\left(\begin{array}{rrrr} -2 & -1 & 1 & 6 \\ 0 & -3 & -2 & 9 \\ 0 & 6 & 5 & -15 \end{array}\right)$	$\frac{6}{-3} = -2$	$\begin{array}{rrrrr} & 0 & 6 & 5 & -15 \\ -(-2)\times(& 0 & -3 & -2 & 9) \\ \hline & 0 & 0 & 1 & 3 \end{array}$
4	$\left(\begin{array}{rrrr} -2 & -1 & 1 & 6 \\ 0 & -3 & -2 & 9 \\ 0 & 0 & 1 & 3 \end{array}\right)$		

Figure 4.5: Figure 4.2 again: Gaussian Elimination with an augmented matrix of coefficients.

Step	$\left(\begin{array}{ccc} A_{00} & a_{01} & A_{02} \\ a_{10}^T & \alpha_{11} & a_{12}^T \\ A_{20} & a_{21} & A_{22} \end{array}\right)$	a_{21}/α_{11}	$A_{22} - a_{21}a_{12}^T$
1-2	$\left(\begin{array}{rrr} -2 & -1 & 1 \\ 2 & -2 & -3 \\ -4 & 4 & 7 \end{array}\right)$	$\begin{pmatrix} 2 \\ -4 \end{pmatrix} /(-2) = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$	$\begin{pmatrix} -2 & -3 \\ 4 & 7 \end{pmatrix} - \begin{pmatrix} -1 \\ 2 \end{pmatrix} \begin{pmatrix} -1 & 1 \end{pmatrix} = \begin{pmatrix} -3 & -2 \\ 6 & 5 \end{pmatrix}$
3	$\left(\begin{array}{rrr} -2 & -1 & 1 \\ {\scriptstyle -1} & -3 & -2 \\ {\scriptstyle 2} & 6 & 5 \end{array}\right)$	$\begin{pmatrix} 6 \end{pmatrix} /(-3) = \begin{pmatrix} -2 \end{pmatrix}$	$\begin{pmatrix} 5 \end{pmatrix} - \begin{pmatrix} -2 \end{pmatrix} \begin{pmatrix} -2 \end{pmatrix} = \begin{pmatrix} 1 \end{pmatrix}$
	$\left(\begin{array}{rrr} -2 & -1 & 1 \\ {\scriptstyle -1} & -3 & -2 \\ {\scriptstyle 2} & {\scriptstyle -2} & 1 \end{array}\right)$		

Figure 4.6: LU factorization of a 3×3 matrix. Compare with the above Gaussian elimination with the coefficient matrix!

Step	Stored multipliers and right-hand side	Operation
1	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 3 \\ 2 & -2 & - & -3 \end{array}\right)$	$\begin{array}{r} 3 \\ -(-1)\times(\underline{\ \ 6}) \\ 9 \end{array}$
2	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 9 \\ 2 & -2 & - & -3 \end{array}\right)$	$\begin{array}{r} -3 \\ -(2)\times(\underline{\ \ 6}) \\ -15 \end{array}$
3	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 9 \\ 2 & -2 & - & -15 \end{array}\right)$	$\begin{array}{r} -15 \\ -(-2)\times(\underline{\ \ 9}) \\ 3 \end{array}$
4	$\left(\begin{array}{ccc\|c} - & - & - & 6 \\ -1 & - & - & 9 \\ 2 & -2 & - & 3 \end{array}\right)$	

Figure 4.7: Forward substitution with the multipliers computed for a linear system in Figure 4.2. Compare to what happens to the right-hand side (the part to the right of the |) in Figure 4.2.

4.3 Forward Substitution = Solving a Unit Lower Triangular System

It is often the case that the coefficient matrix for the linear system is available *a priori* and the right-hand side becomes available later. In this case, one may want to perform Gaussian elimination without augmenting the system with the right-hand-side or, equivalently, LU factorization on the coefficient matrix. In Figure 4.7 we illustrate that if the multipliers are stored, typically over the elements that were zeroed when a multiplier was used, then the computations that were performed during Gaussian Elimination can be applied *a postiori* (afterwards), once the right-hand side becomes available. This process is often referred to as *forward substitution*.

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 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

$$L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), \quad z \rightarrow \left(\begin{array}{c} \zeta_1 \\ \hline z_2 \end{array}\right), \quad \text{and} \quad b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right).$$

(Note: the horizontal line here partitions the result. It is *not* a division.) Now, $Lz = b$

implies

$$\overbrace{\left(\begin{array}{c}\beta_1\\ \hline b_2\end{array}\right)}^{b} = \overbrace{\left(\begin{array}{c|c}1 & 0\\ \hline l_{21} & L_{22}\end{array}\right)\left(\begin{array}{c}\zeta_1\\ \hline z_2\end{array}\right)}^{L} = \overbrace{\left(\begin{array}{c}\zeta_1\\ \hline l_{21}\zeta_1 + L_{22}z_2\end{array}\right)}^{Lz}$$

so that

$$\left(\begin{array}{c}\beta_1 = \zeta_1\\ \hline b_2 = l_{21}\zeta_1 + L_{22}z_2\end{array}\right) \quad \text{or} \quad \left(\begin{array}{c}\zeta_1 = \beta_1\\ \hline z_2 = b_2 - l_{21}\zeta_1\end{array}\right).$$

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 recursively with $L = L_{22}$ and $b = b_2$.

This algorithm is presented as an iteration using our notation in Figure 4.8. It is illustrated for the matrix L that results from Equation (4.1) in Figure 4.9. Examination of the computations in Figure 4.7 on the right-hand-side and 4.9 highlights how forward substitution and the solution of $Lz = b$ are related: they are the same!

Exercise 4.3 Use `http://www.cs.utexas.edu/users/flame/Spark/` to write a FLAME@lab code for computing the solution of $Lx = b$, overwriting b with the solution and assuming that L is unit lower triangular.

Exercise 4.4 Modify the algorithm in Figure 4.8 so that it solves $Lz = b$ when L is lower triangular matrix (not unit lower triangular). Next implement it using FLAME@lab.

4.4 Backward Substitution = Solving an Upper Triangular System

Next, let us consider how to solve a linear system $Ux = b$ and how it is the same as backward substitution.

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

$$U \rightarrow \left(\begin{array}{c|c}\upsilon_{11} & u_{12}^T\\ \hline 0 & U_{22}\end{array}\right), \quad x \rightarrow \left(\begin{array}{c}\chi_1\\ \hline x_2\end{array}\right) \quad \text{and} \quad b \rightarrow \left(\begin{array}{c}\beta_1\\ \hline b_2\end{array}\right).$$

Algorithm: $[b] := \textsc{Ltrsv\_unb}(L, b)$

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 4.8: Algorithm for triangular solve with unit lower triangular matrix.

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	$\left(\begin{array}{r\|rr} 1 & 0 & 0 \\ \hline -1 & 1 & 0 \\ 2 & -2 & 1 \end{array}\right)$	$\left(\begin{array}{r} 6 \\ \hline 3 \\ -3 \end{array}\right)$	$\left(\begin{array}{r} 3 \\ -3 \end{array}\right) - \left(\begin{array}{r} -1 \\ 2 \end{array}\right)(6) = \left(\begin{array}{r} 9 \\ -15 \end{array}\right)$
3	$\left(\begin{array}{r\|r\|r} 1 & 0 & 0 \\ \hline -1 & 1 & 0 \\ \hline 2 & -2 & 1 \end{array}\right)$	$\left(\begin{array}{r} 6 \\ \hline 9 \\ \hline -15 \end{array}\right)$	$(-15) - (-2)(9) = (3)$
	$\left(\begin{array}{rr\|r} 1 & 0 & 0 \\ -1 & 1 & 0 \\ \hline 2 & -2 & 1 \end{array}\right)$	$\left(\begin{array}{r} 6 \\ 9 \\ \hline 3 \end{array}\right)$	

Figure 4.9: Triangular solve with unit lower triangular matrix computed in Figure 4.4.

Algorithm: $[b] := \textsc{Utrsv\_unb}(U, b)$

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

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & \upsilon_{11} & u_{12}^T \\ \hline 0 & 0 & U_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

where υ_{11} is 1×1 , β_1 has 1 row

$\beta_1 := (\beta_1 - u_{12}^T b_2)/\upsilon_{11}$

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 4.10: Algorithm for triangular solve with upper triangular matrix.

Now, $Ux = b$ implies

$$\overbrace{\left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)}^{b} = \overbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)}^{U} \overbrace{\left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right)}^{x} = \overbrace{\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 \\ \hline U_{22} x_2 \end{array}\right)}^{Ux}$$

so that

$$\left(\begin{array}{c} \beta_1 = \upsilon_{11}\chi_1 + u_{12}^T x_2 \\ \hline b_2 = U_{22} x_2 \end{array}\right) \quad \text{or} \quad \left(\begin{array}{c} \chi_1 = (\beta_1 - u_{12}^T x_2)/\upsilon_{11} \\ \hline U_{22} x_2 = b_2 \end{array}\right)$$

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 suggests the algorithms in Figure 4.10.

Exercise 4.5 Side-by-side, solve the upper triangular linear system

$$\begin{array}{rcrcrcr} -2\chi_0 & - & \chi_1 & + & \chi_2 & = & 6 \\ & & -3\chi_1 & - & 2\chi_2 & = & 9 \\ & & & & \chi_2 & = & 3 \end{array}$$

using the usual approach and apply the algorithm in Figure 4.10 with

$$U = \begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 6 \\ 9 \\ 3 \end{pmatrix}.$$

In other words, for this problem, give step-by-step details of what both methods do, much like Figures. 4.5 and 4.6.

Exercise 4.6 Use `http://www.cs.utexas.edu/users/flame/Spark/` to write a FLAME@lab code for computing the solution of $Ux = b$, overwriting b with the solution and assuming that U is upper triangular.

4.5 Solving the Linear System

What we have seen is that Gaussian Elimination can be used to convert a linear system into an upper triangular linear system, which can then be solved. We also showed that computing the LU factorization of a matrix is the same as performing Gaussian Elimination on the matrix of coefficients. Finally, we showed that forward substitution is equivalent to solving $Lz = b$, where L is the unit lower triangular matrix that results from the LU factorization. We can now show how the solution of the linear system can computed using the LU factorization.

Let $A = LU$ and assume that $Ax = b$, where A and b are given. Then $(LU)x = b$ or $L(Ux) = b$. Let us introduce a dummy vector $z = Ux$. Then $Lz = b$ and z can be computed as described in the previous section. Once z has been computed, x can be computed by solving $Ux = z$ where now U and z are known.

4.6 When LU Factorization Breaks Down

A question becomes "Does Gaussian elimination always solve a linear system?" Or, equivalently, can an LU factorization always be computed?

What we do know is that *if* an LU factorization 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. The reason is that if the LU factorization can be computed, then $A = LU$ for some unit lower triangular matrix L and upper triangular matrix U. Now, if you look at the algorithm for forward substitition (solving $Lz = b$), you will see that the only computations that are encountered are multiplies and adds. Thus, the algorithm will complete. Similarly, the backward substitution algorithm (for solving $Ux = z$) can only break down if the division causes an error. And that can only happen if U has a zero on its diagonal.

Are there examples where LU (Gaussian elimination as we have presented it so far) can break down? The answer is *yes*. A simple example is the matrix $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. In the first step, the algorithm for LU factorization will try to compute the multiplier $1/0$, which will cause an error.

Now, $Ax = b$ is given by the set of linear equations

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \chi_1 \\ \chi_0 \end{pmatrix}$$

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 $x = \begin{pmatrix} \beta_1 \\ \beta_0 \end{pmatrix}$.

To motivate the solution, consider applying Gaussian elimination to the following example:

$$\begin{array}{rrrr} 2\chi_0 + & 4\chi_1 + & (-2)\chi_2 = & -10 \\ 4\chi_0 + & 8\chi_1 + & 6\chi_2 = & 20 \\ 6\chi_0 + & (-4)\chi_1 + & 2\chi_2 = & 18 \end{array}$$

Recall that 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, we get

$$\begin{array}{rrrr} 2\chi_0 + & 4\chi_1 + & (-2)\chi_2 = & -10 \\ 0\chi_0 + & 0\chi_1 + & 10\chi_2 = & 40 \\ 6\chi_0 + & (-4)\chi_1 + & 2\chi_2 = & 18 \end{array}$$

- By subtracting $(6/2) = 3$ times the first row from the third row, we get

$$\begin{array}{rcl} 2\chi_0 + 4\chi_1 + (-2)\chi_2 &=& -10 \\ 0\chi_0 + 0\chi_1 + 10\chi_2 &=& 40 \\ 0\chi_0 + (-16)\chi_1 + 8\chi_2 &=& 48 \end{array}$$

- 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:

$$\begin{array}{rcl} 2\chi_0 + 4\chi_1 + (-2)\chi_2 &=& -10 \\ 0\chi_0 + (-16)\chi_1 + 8\chi_2 &=& 48 \\ 0\chi_0 + 0\chi_1 + 10\chi_2 &=& 40 \end{array}$$

  at which point we are done transforming our system into an upper triangular system, and the backward substition can commence to solve the problem.

Another example:

$$\begin{array}{rcl} 0\chi_0 + 4\chi_1 + (-2)\chi_2 &=& -10 \\ 4\chi_0 + 8\chi_1 + 6\chi_2 &=& 20 \\ 6\chi_0 + (-4)\chi_1 + 2\chi_2 &=& 18 \end{array}$$

Now,

- We start by trying to subtract $(4/0)$ times the first row from the second row, which leads to an error. So, instead, we swap the first row with any of the other two rows:

$$\begin{array}{rcl} 4\chi_0 + 8\chi_1 + 6\chi_2 &=& 20 \\ 0\chi_0 + 4\chi_1 + (-2)\chi_2 &=& -10 \\ 6\chi_0 + (-4)\chi_1 + 2\chi_2 &=& 18 \end{array}$$

- By subtracting $(6/4) = 3/2$ times the first row from the third row, we get

$$\begin{array}{rcl} 4\chi_0 + 8\chi_1 + 6\chi_2 &=& 20 \\ 0\chi_0 + 4\chi_1 + (-2)\chi_2 &=& -10 \\ 0\chi_0 + (-16)\chi_1 + (-7)\chi_2 &=& -22 \end{array}$$

- Next, we subtract $(-16)/4 = -4$ times the second row from the third to obtain

$$\begin{array}{rcl} 4\chi_0 + 8\chi_1 + 6\chi_2 &=& 20 \\ 0\chi_0 + 4\chi_1 + (-2)\chi_2 &=& -10 \\ 0\chi_0 + 0\chi_1 + (-15)\chi_2 &=& -62 \end{array}$$

  at which point we are done transforming our system into an upper triangular system, and the backward substition can commence to solve the problem.

The above discussion suggests that the LU factorization in Fig. 4.11 needs to be modified to allow for row exchanges. But to do so, we need to create some machinery.

4.7 Permutations

Example 4.7 Consider

$$\underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}}_{P} \underbrace{\begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}}_{A} = \begin{pmatrix} 3 & 2 & 1 \\ -1 & 0 & -3 \\ -2 & 1 & 2 \end{pmatrix}.$$

Notice that multiplying A by P from the left permuted the order of the rows in that matrix.

Examining the matrix P in Example 4.7 we see that each row of P appears to equals a unit basis vector. This leads us to the following definitions:

Definition 4.8 A vector $p = (k_0, k_1, \ldots, k_{n-1})^T$ is said to be a permutation (vector) if $k_j \in \{0, \ldots, n-1\}$, $0 \leq j < n$, and $k_i = k_j$ implies $i = j$.

We will below write $(k_0, k_1, \ldots, k_{n-1})^T$ to indicate a column vector, for space considerations. This permutation is just a rearrangement of the vector $(0, 1, \ldots, n-1)^T$.

Definition 4.9 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation. Then

$$P = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}$$

is said to be a *permutation matrix*.

In other words, P is the identity matrix with its rows rearranged as indicated by the n-tuple $(k_0, k_1, \ldots, k_{n-1})$. We will frequently indicate this permutation matrix as $P(p)$ to indicate that the permutation matrix corresponds to the permutation vector p.

Theorem 4.10 Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation. 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

$$Px = \begin{pmatrix} \chi_{k_0} \\ \chi_{k_1} \\ \vdots \\ \chi_{k_{n-1}} \end{pmatrix}, \quad \text{and} \quad PA = \begin{pmatrix} a_{k_0}^T \\ a_{k_1}^T \\ \vdots \\ a_{k_{n-1}}^T \end{pmatrix}.$$

In other words, Px and PA rearrange the elements of x and the rows of A in the order indicated by permutation vector p.

Proof: Recall that unit basis vectors have the property that $e_j^T A = \check{a}_j^T$.

$$PA = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} A = \begin{pmatrix} e_{k_0}^T A \\ e_{k_1}^T A \\ \vdots \\ e_{k_{n-1}}^T A \end{pmatrix} = \begin{pmatrix} \check{a}_{k_0}^T \\ \check{a}_{k_1}^T \\ \vdots \\ \check{a}_{k_{n-1}}^T \end{pmatrix}.$$

The result for Px can be proved similarly or, alternatively, by viewing x as a matrix with only one column.

Exercise 4.11 Let $p = (2, 0, 1)^T$. Compute

$$P(p) \begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix} \quad \text{and} \quad P(p) \begin{pmatrix} -2 & 1 & 2 \\ 3 & 2 & 1 \\ -1 & 0 & -3 \end{pmatrix}.$$

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.

Corollary 4.12 Let $p = (k_0, k_1, \ldots, k_{n-1})^T$ be a permutation and $P = P(p)$. Consider $A = \left(a_0 \mid a_1 \mid \cdots \mid a_{n-1} \right)$. Then $AP^T = \left(a_{k_0} \mid a_{k_1} \mid \cdots \mid a_{k_{n-1}} \right)$.

Proof: Recall that unit basis vectors have the property that $Ae_k = a_k$.

$$\begin{aligned} AP^T &= A \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}^T = A \left(\begin{array}{c|c|c|c} e_{k_0} & e_{k_1} & \cdots & e_{k_{n-1}} \end{array} \right) \\ &= \left(\begin{array}{c|c|c|c} Ae_{k_0} & Ae_{k_1} & \cdots & Ae_{k_{n-1}} \end{array} \right) = \left(\begin{array}{c|c|c|c} a_{k_0} & a_{k_1} & \cdots & a_{k_{n-1}} \end{array} \right). \end{aligned}$$

Corollary 4.13 If P is a permutation matrix, then so is P^T.

This follows from the observation that if P can be viewed either as a rearrangement of the rows or as a (usually different) rearrangement of the columns.

Corollary 4.14 Let P be a permutation matrix. Then $PP^T = P^T P = I$

Proof: Let $p = (k_0, k_1, \ldots, k_{n-1})^T$ be the permutation that defines P. Then

$$\begin{aligned} PP^T &= \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}^T = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \left(\begin{array}{c|c|c|c} e_{k_0} & e_{k_1} & \cdots & e_{k_{n-1}} \end{array} \right) \\ &= \begin{pmatrix} e_{k_0}^T e_{k_0} & e_{k_0}^T e_{k_1} & \cdots & e_{k_0}^T e_{k_{n-1}} \\ e_{k_1}^T e_{k_0} & e_{k_1}^T e_{k_1} & \cdots & e_{k_1}^T e_{k_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ e_{k_{n-1}}^T e_{k_0} & e_{k_{n-1}}^T e_{k_1} & \cdots & e_{k_{n-1}}^T e_{k_{n-1}} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} = I. \end{aligned}$$

Now, we already argued that P^T is also a permutation matrix. Thus, $I = P^T(P^T)^T = P^T P$, which proves the second part of the corollary.

Definition 4.15 Let us call the special permutation matrix of the form

$$\tilde{P}(\pi) = \begin{pmatrix} \boxed{e_\pi^T} \\ e_1^T \\ \vdots \\ e_{\pi-1}^T \\ \boxed{e_0^T} \\ e_{\pi+1}^T \\ \vdots \\ e_{n-1}^T \end{pmatrix} = \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

a *pivot matrix.*

Theorem 4.16 When $\tilde{P}(\pi)$ multiplies a matrix from the left, it swaps rows 0 and π. When $\tilde{P}(\pi)$ multiplies a matrix from the right, it swaps columns 0 and π.

4.8 Back to "When LU Factorization Breaks Down"

Let us reiterate the algorithmic steps that were exposed for the LU factorization in Section 4.2:

- Partition
$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right).$$
- Update $a_{21} = a_{21}/\alpha_{11}(= l_{21})$.
- Update $A_{22} = A_{22} - a_{21}a_{12}^T(= A_{22} - l_{21}u_{12}^T)$.
- Overwrite A_{22} with L_{22} and U_{22} by continuing recursively with $A = A_{22}$.

Instead of overwriting A with the factors L and U, we can compute L separately and overwrite A with U, and letting the elements below its diagonal become zeroes. This allows us to get back to formulating the algorithm using Gauss transforms

- Partition
$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right).$$
- Compute $l_{21} = a_{21}/\alpha_{11}$.

$[L,A] := \text{LU\_UNB\_VAR5\_ALT}(A)$

Partition $L := I$

$$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), \; L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$$

where A_{TL} is 0×0

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

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \; \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right)$$

where α_{11} is 1×1

$l_{21} := a_{21}/\alpha_{11}$

$$\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) := \left(\begin{array}{c|c} 1 & 0 \\ \hline -l_{21} & I \end{array}\right) \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline 0 & A_{22} - l_{21}a_{12}^T \end{array}\right)$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \; \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right)$$

endwhile

Figure 4.11: Algorithm for computing the LU factorization, exposing the update of the matrix as multiplication by a Gauss transform.

- Update $\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) := \left(\begin{array}{c|c} 1 & 0 \\ \hline -l_{21} & I \end{array}\right) \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline 0 & A_{22} - l_{21}a_{12}^T \end{array}\right)$.

- Overwrite A_{22} with L_{22} and U_{22} by continuing recursively with $A = A_{22}$.

This leads to the equivalent LU factorization algorithm in Fig. 4.11. In that algorithm the elements below the diagonal of A are overwritten with zeroes, so that it eventually equals the upper triangular matrix U. The unit lower triangular matrix L is now returned as a separate matrix. The matrix $\left(\begin{array}{c|c} 1 & 0 \\ \hline -l_{21} & I \end{array}\right)$ is known as a Gauss transform.

Example 4.17 In Fig. 4.12 we illustrate the above alternative description of LU factorization with the same matrix that we used in Fig. 4.4.

Let us explain this in one more, slightly different, way:

Step	$\left(\begin{array}{c\|c\|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$	$\left(\begin{array}{c\|c\|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right)$	$l_{21} := a_{21}/\alpha_{11}$	$\left(\begin{array}{c\|c} 1 & 0 \\ \hline -l_{21} & I \end{array}\right)\left(\begin{array}{c\|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right)$
1-2	$\begin{pmatrix} -2 & -1 & 1 \\ 2 & -2 & -3 \\ -4 & 4 & 7 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 2 \\ -4 \end{pmatrix}/(-2) = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix}\begin{pmatrix} -2 & -1 & 1 \\ 2 & -2 & -3 \\ -4 & 4 & 7 \end{pmatrix} = \begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 6 & 5 \end{pmatrix}$
3	$\begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 6 & 5 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 0 & 1 \end{pmatrix}$	$(6)/(-3) = (-2)$	$\begin{pmatrix} 1 & 0 \\ -(-2) & 1 \end{pmatrix}\begin{pmatrix} -3 & -2 \\ 6 & 5 \end{pmatrix} = \begin{pmatrix} -3 & -2 \\ 0 & 1 \end{pmatrix}$
	$\begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & -2 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & -2 & 1 \end{pmatrix}$		

Figure 4.12: LU factorization based on Gauss transforms of a 3×3 matrix. Here, "Step" refers to the corresponding step for Gaussian Elimination in Fig. 4.2.

- Partition

$$A \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & a_{01} & A_{02} \end{array}\right) \text{ and } L \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & 0 & I \end{array}\right)$$

  where the thick line indicates, as usual, how far we have gotten into the computation. In other words, the elements below the diagonal of A in the columns that contain A_{00} have already been replaced by zeroes and the corresponding columns of L have already been computed.

- Compute $l_{21} = a_{21}/\alpha_{11}$.

- Update $\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & 0 & A_{02} \end{array}\right) := \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right) \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & a_{01} & A_{02} \end{array}\right).$

- Continue by moving the thick line forward one row and column.

This leads to yet another equivalent LU factorization algorithm in Fig. 4.13. Notice that upon completion, A is an upper triangular matrix, U. The point of this alternative explanation is to show that if $\check{L}^{(i)}$ represents the ith Gauss transform, computed during the ith iteration of the algorithms, then the final matrix stored in A, the upper triangular matrix U, satisfies $U = \check{L}^{(n-2)}\check{L}^{(n-3)} \cdots \check{L}^{(0)}\hat{A}$, where $\hat{A}$ is the original matrix stored in A.

Example 4.18 Let us illustrate these last observations with the same example as in Fig. 4.4:

- Start with $A = A^{(0)} = \begin{pmatrix} -2 & -1 & 1 \\ 2 & -2 & -3 \\ -4 & 4 & 7 \end{pmatrix}$ and $L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

- In the first step,

  – Partition A and L to expose the first row and column:

$$\left(\begin{array}{||c|cc} -2 & -1 & 1 \\ \hline 2 & -2 & -3 \\ -4 & 4 & 7 \end{array}\right) \text{ and } \left(\begin{array}{||c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right);$$

  – Compute

$$l_{21} = \begin{pmatrix} 2 \\ -4 \end{pmatrix} / (-2) = \begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

  – Update A with

$$A^{(1)} = \underbrace{\left(\begin{array}{||c|cc} 1 & 0 & 0 \\ \hline 1 & 1 & 0 \\ -2 & 0 & 1 \end{array}\right)}_{\check{L}^{(0)}} \underbrace{\left(\begin{array}{||c|cc} -2 & -1 & 1 \\ \hline 2 & -2 & -3 \\ -4 & 4 & 7 \end{array}\right)}_{A^{(0)}} = \left(\begin{array}{||c|cc} -2 & -1 & 1 \\ \hline 0 & -3 & -2 \\ 0 & 6 & 5 \end{array}\right).$$

$[L, A] := \text{LU\_UNB\_VAR5\_ALT}(A)$

Partition $L := I$

$$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), \; L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$$

where A_{TL} is 0×0

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

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \; \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right)$$

where α_{11} is 1×1

$l_{21} := a_{21} / \alpha_{11}$

$$\left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{22} & 0 & A_{02} \end{array}\right) := \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right) \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{01} & A_{02} \end{array}\right)$$

$$= \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right) \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & a_{01} & A_{02} \end{array}\right) = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & 0 & A_{02} - l_{21} a_{12}^T \end{array}\right)$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \; \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right)$$

endwhile

Figure 4.13: Algorithm for computing the LU factorization.

We emphasize that now $A^{(1)} = \check{L}^{(0)} A^{(0)}$.

- In the second step,
  - Partition A (which now contains $A^{(1)}$ and L to expose the second row and column:

$$\left(\begin{array}{c|c|c} -2 & -1 & 1 \\ \hline 0 & -3 & -2 \\ \hline 0 & 6 & 5 \end{array}\right) \text{ and } \left(\begin{array}{c|c|c} 1 & 0 & 0 \\ \hline -1 & 1 & 0 \\ \hline 2 & 0 & 1 \end{array}\right);$$

  - Compute

$$l_{21} = \left(\ 6 \ \right) / (-3) = \left(\ -2 \ \right).$$

  - Update A with

$$A^{(2)} = \underbrace{\left(\begin{array}{c|c|c} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 2 & 1 \end{array}\right)}_{\check{L}^{(1)}} \underbrace{\left(\begin{array}{c|c|c} -2 & -1 & 1 \\ \hline 0 & -3 & -2 \\ \hline 0 & 6 & 5 \end{array}\right)}_{A^{(1)}} = \left(\begin{array}{c|c|c} -2 & -1 & 1 \\ \hline 0 & -3 & -2 \\ \hline 0 & 0 & 1 \end{array}\right).$$

We emphasize that now

$$\begin{aligned} A \ &= \underbrace{\begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 0 & 1 \end{pmatrix}}_{A^{(2)}} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix}}_{\check{L}^{(1)}} \underbrace{\begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 6 & 5 \end{pmatrix}}_{A^{(1)}} \\ &= \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix}}_{\check{L}^{(1)}} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & 0 & 1 \end{pmatrix}}_{\check{L}^{(0)}} \underbrace{\begin{pmatrix} -2 & -1 & 1 \\ 2 & -2 & -3 \\ -4 & 4 & 7 \end{pmatrix}}_{A^{(0)}} \end{aligned}$$

The point of this last example is to show that LU factorization can be viewed as the computation of a sequence of Gauss transforms so that, upon completion $U = \check{L}^{(n-1)} \check{L}^{(n-2)} \check{L}^{(n-3)} \cdots \check{L}^{(0)} A$. (Actually, $\check{L}^{(n-1)}$ is just the identity.)

Now, let us consider the following property of a typical Gauss transform:

$$\underbrace{\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)}_{\check{L}^{(i)}} \underbrace{\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right)}_{L^{(i)}} = \underbrace{\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & I \end{array}\right)}_{I}$$

The inverse of a Gauss transform can be found by changing $-l_{21}$ to l_{21}!!!

This means that if $U = \check{L}^{(n-2)}\check{L}^{(n-3)}\cdots\check{L}^{(0)}A$, then if $L^{(i)}$ is the inverse of $\check{L}^{(i)}$, then $L^{(0)}\ldots L^{(n-3)}L^{(n-2)}U = A$. In the case of our example,

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 0 & 1 \end{pmatrix}}_{L^{(0)}} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{pmatrix}}_{L^{(1)}} \underbrace{\begin{pmatrix} -2 & -1 & 1 \\ 0 & -3 & -2 \\ 0 & 0 & 1 \end{pmatrix}}_{U} = \underbrace{\begin{pmatrix} -2 & -1 & 1 \\ 2 & -2 & -3 \\ -4 & 4 & 7 \end{pmatrix}}_{A}$$

Finally, note that

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 0 & 1 \end{pmatrix}}_{L^{(0)}} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{pmatrix}}_{L^{(1)}} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & -2 & 1 \end{pmatrix}}_{\check{L}^{(1)}}$$

In other words, the LU factor L can be constructed from the Gauss transforms by setting the jth column of L to the jth column of the jth Gauss transform, and then "flipping" the sign of the elements below the diagonal. One can more formally prove this by noting that

$$\left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & 0 & I \end{array}\right) \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right) = \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & 1 & 0 \\ \hline L_{20} & l_{21} & I \end{array}\right)$$

as part of an inductive argument.

Now, we are ready to add pivoting (swapping of rows, in other words: swapping of equations) into the mix.

Example 4.19 Consider again the system of linear equations

$$\begin{array}{rcrcrcr} 2\chi_0 & + & 4\chi_1 & + & (-2)\chi_2 & = & -10 \\ 4\chi_0 & + & 8\chi_1 & + & 6\chi_2 & = & 20 \\ 6\chi_0 & + & (-4)\chi_1 & + & 2\chi_2 & = & 18 \end{array}$$

and let us focus on the matrix of coefficients

$$A = \begin{pmatrix} 2 & 4 & -2 \\ 4 & 8 & 6 \\ 6 & -4 & 2 \end{pmatrix}.$$

Let us start the algorithm in Figure 4.11. In the first step, we apply a pivot to ensure that the diagonal element in the first column is not zero. In this example, no pivoting is required, so the first pivot matrix, $\tilde{P}^{(0)} = I$.

$$\underbrace{\left(\begin{array}{|c|cc} \hline 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right)}_{\tilde{P}^{(0)}} \left(\begin{array}{|c|cc} \hline 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}\right) = \underbrace{\left(\begin{array}{|c|cc} \hline 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}\right)}_{\tilde{A}^{(0)}}$$

$[L, A] := \text{LU\_UNB\_VAR5\_PIV}(A)$

Partition $L := I$

$$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), \; L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right), \; p \rightarrow \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right)$$

where A_{TL} is 0×0

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

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \cdots, \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \cdots, \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right) \rightarrow \left(\begin{array}{c} p_0 \\ \hline p_1 \\ \hline p_2 \end{array}\right)$$

where α_{11} is 1×1

$$\pi_1 = \text{PIVOT}\left(\left(\begin{array}{c} \alpha_{11} \\ \hline a_{21} \end{array}\right)\right)$$

$$\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) := P(\pi_1) \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right)$$

$$l_{21} := a_{21} / \alpha_{11}$$

$$\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) := \left(\begin{array}{c|c} 1 & 0 \\ \hline -l_{21} & I \end{array}\right) \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline 0 & A_{22} - l_{21} a_{12}^T \end{array}\right)$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \cdots, \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \cdots, \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right) \leftarrow \left(\begin{array}{c} p_0 \\ \hline p_1 \\ \hline p_2 \end{array}\right)$$

endwhile

Figure 4.14: Algorithm for computing the LU factorization, exposing the update of the matrix as multiplication by a Gauss transform and adding pivoting.

Next, a Gauss transform is computed and applied:

$$\underbrace{\left(\begin{array}{r|rr} 1 & 0 & 0 \\ \hline -2 & 1 & 0 \\ -3 & 0 & 1 \end{array}\right)}_{\check{L}^{(0)}} \underbrace{\left(\begin{array}{r|rr} 2 & 4 & -2 \\ \hline 4 & 8 & 6 \\ 6 & -4 & 2 \end{array}\right)}_{\tilde{A}^{(0)}} = \underbrace{\left(\begin{array}{r|rr} 2 & 4 & -2 \\ \hline 0 & 0 & 10 \\ 0 & -16 & 8 \end{array}\right)}_{A^{(1)}}.$$

In the second step, we apply a pivot to ensure that the diagonal element in the second column is not zero. In this example, the second and third row must be swapped by pivot matrix $\tilde{P}^{(1)}$:

$$\underbrace{\left(\begin{array}{r|r|r} 1 & 0 & 0 \\ \hline 0 & 0 & 1 \\ \hline 0 & 1 & 0 \end{array}\right)}_{\left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & \check{P}^{(1)} \end{array}\right)} \underbrace{\left(\begin{array}{r|r|r} 2 & 4 & -2 \\ \hline 0 & 0 & 10 \\ \hline 0 & -16 & 8 \end{array}\right)}_{A^{(1)}} = \underbrace{\left(\begin{array}{r|r|r} 2 & 4 & -2 \\ \hline 0 & -16 & 8 \\ \hline 0 & 0 & 10 \end{array}\right)}_{\tilde{A}^{(1)}}$$

Next, a Gauss transform is computed and applied:

$$\underbrace{\left(\begin{array}{r|r|r} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 1 \end{array}\right)}_{\check{L}^{(1)}} \underbrace{\left(\begin{array}{r|r|r} 2 & 4 & -2 \\ \hline 0 & -16 & 8 \\ \hline 0 & 0 & 10 \end{array}\right)}_{\tilde{A}^{(1)}} = \underbrace{\left(\begin{array}{r|r|r} 2 & 4 & -2 \\ \hline 0 & -16 & 8 \\ \hline 0 & 0 & 10 \end{array}\right)}_{A^{(2)}}.$$

Notice that at each step, some permutation matrix is used to swap two rows, after which a Gauss transform is computed and then applied to the resulting (permuted) matrix. One can describe this as $U = \check{L}^{(n-2)}\check{P}^{(n-2)}\check{L}^{(n-3)}\check{P}^{(n-3)}\cdots\check{L}^{(0)}\check{P}^{(0)}A$, where $P^{(i)}$ represents the permutation applied during iteration i. Now, once an LU factorization with pivoting is computed, one can solve $Ax = b$ by noting that $Ux = \check{L}^{(n-2)}\check{P}^{(n-2)}\check{L}^{(n-3)}\check{P}^{(n-3)}\cdots\check{L}^{(0)}\check{P}^{(0)}Ax = \check{L}^{(n-2)}\check{P}^{(n-2)}\check{L}^{(n-3)}\check{P}^{(n-3)}\cdots\check{L}^{(0)}\check{P}^{(0)}b$. In other words, the pivot matrices and Gauss transforms, in the proper order, must be applied to the right-hand side,

$$z = \check{L}^{(n-2)}\check{P}^{(n-2)}\check{L}^{(n-3)}\check{P}^{(n-3)}\cdots\check{L}^{(0)}\check{P}^{(0)}b,$$

after which x can be obtained by solving the upper triangular system $Ux = z$.

Remark 4.20 If the LU factorization with pivoting completes without encountering a zero pivot, then given any right-hand size b this procedure produces a unique solution x. In other words, the procedure computes the net effect of applying A^{-1} to the right-hand side vector b, and therefore A has an inverse. If a zero pivot is encountered, then there exists a vector $x \neq 0$ such that $Ax = 0$, and hence the inverse does not exist.

4.9 The Inverse of a Matrix

4.9.1 First, some properties

Definition 4.21 *Given $A \in \mathbb{R}^{n\times n}$, a matrix B that has the property that $BA = I$, the identity, is called the inverse of matrix A and is denoted by A^{-1}.*

Remark 4.22 We will later see that not every square matrix has an inverse! The inverse of a nonsquare matrix is not defined. Indeed, we will periodically relate other properties of a matrix to the matrix having an inverse as these notes unfold.

Notice that A^{-1} is the matrix that "undoes" the transformation A: $A^{-1}(Ax) = x$. It acts as the inverse function of the function $F(x) = Ax$.

Example 4.23 *Let's start by looking at some matrices for which it is easy to determine the inverse:*

- **The identity matrix:** $I^{-1} = I$, *since* $I \cdot I = I$. ¿
- **Diagonal matrices:**

$$\text{if } D = \begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix} \text{ then } D^{-1} = \begin{pmatrix} \frac{1}{\delta_0} & 0 & \cdots & 0 \\ 0 & \frac{1}{\delta_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\delta_{n-1}} \end{pmatrix}.$$

  In particular, if $D = \delta I$ (all elements on the diagonal equal δ) then $D^{-1} = \frac{1}{\delta} I$.

- **Zero matrix:** *Let O denote the $n \times n$ matrix of all zeroes. This matrix does not have an inverse. Why? Pick any vector $x \neq 0$ (not equal to the zero vector). Then $O^{-1}(Ox) = 0$ and, if O^{-1} existed, $(O^{-1}O)x = Ix = x$, which is a contradiction.*

Theorem 4.24 Let $Ax = b$ and assume that A has an inverse, A^{-1}. Then $x = A^{-1}b$.

Proof: If $Ax = b$ then $A^{-1}Ax = A^{-1}b$ and hence $Ix = x = A^{-1}b$.

Corollary 4.25 Assume that A has an inverse, A^{-1}. Then $Ax = 0$ implies that $x = 0$.

Proof: If A has an inverse and $Ax = 0$, then $x = Ix = (A^{-1}A)x = A^{-1}(Ax) = A^{-1}0 = 0$.

Theorem 4.26 If A has an inverse A^{-1}, then $AA^{-1} = I$.

Theorem 4.27 (Uniqueness of the inverse) If A has an inverse, then that inverse is unique.

Proof: Assume that $AB = BA = I$ and $AC = CA = I$. Then by associativity of matrix multiplication $C = CI = C(AB) = (CA)B = B$.

Let us assume in this section that A has an inverse and let us assume that we would like to compute $C = A^{-1}$. Matrix C must satisfy $AC = I$. Partition matrices C and I by columns:

$$A \underbrace{\left(b_0 \mid b_1 \mid \cdots \mid b_{n-1} \right)}_{C} = \left(Ac_0 \mid Ac_1 \mid \cdots \mid Ac_{n-1} \right) = \underbrace{\left(e_0 \mid e_1 \mid \cdots \mid e_{n-1} \right)}_{I},$$

where e_j equals the jth column of I. (Notice that we have encountered e_j before in Section **??**. Thus, the jth column of C, c_j, must solve $Ac_j = e_j$.

Now, let us recall how *if* Gaussian elimination works (and in the next section we will see it doesn't always!) then you can solve $Ax = b$ by applying Gaussian elimination to the augmented system $\left(A \mid b \right)$, leaving the result as $\left(U \mid z \right)$ (where we later saw that z solves $Lz = b$), after which backward substitution could be used to solve the upper triangular system $Ux = z$.

So, this means that we should do this for each of the equations $Ac_j = e_j$: Append $\left(A \mid e_j \right)$, leaving the result as $\left(U \mid z_j \right)$ and then perform back substitution to solve $Uc_j = z_j$.

4.9.2 That's about all we will say about determinants

Example 4.28 *Consider the* 2×2 *matrix* $\begin{pmatrix} 2 & -1 \\ 1 & 1 \end{pmatrix}$. *How would we compute its inverse? One way is to start with* $A^{-1} = \begin{pmatrix} \beta_{0,0} & \beta_{0,1} \\ \beta_{1,0} & \beta_{1,1} \end{pmatrix}$ *and note that*

$$\begin{pmatrix} 2 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} \\ \beta_{1,0} & \beta_{1,1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

which yields two linear systems:

$$\begin{pmatrix} 2 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \beta_{0,0} \\ \beta_{1,0} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } \begin{pmatrix} 2 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \beta_{0,1} \\ \beta_{1,1} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Solving these yields $A^{-1} = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} \end{pmatrix}$.

Exercise 4.29 Check that $\begin{pmatrix} 2 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{3} & \frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

One can similarly compute the inverse of any 2×2 matrix: Consider

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} \\ \beta_{1,0} & \beta_{1,1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

which yields two linear systems:

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \begin{pmatrix} \beta_{0,0} \\ \beta_{1,0} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix} \begin{pmatrix} \beta_{0,1} \\ \beta_{1,1} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Solving these yields

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}^{-1} = \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{0,1}\alpha_{1,0}} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix}.$$

Here the expression $\alpha_{0,0}\alpha_{1,1} - \alpha_{0,1}\alpha_{1,0}$ is known as the *determinant*. The inverse of the 2×2 matrix exists if and only if this expression is not equal to zero.

Similarly, a determinant can be defined for any $n \times n$ matrix A and there is even a method for solving linear equations, known as Kramer's rule and taught in high school algebra classes, that requires computation the determinants of various matrices. But this method is completely impractical and therefore does not deserve any of our time.

4.9.3 Gauss-Jordan method

There turns out to be a convenient way of computing all columns of the inverse matrix simultaneously. This method is known as the Gauss-Jordan method. We will illustrate this for a specific matrix and relate it back to the above discussion.

Consider the matrix $A = \begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix}$. Computing the columns of the inverse matrix could start by applying Gaussian elimination to the augmented systems

$$\left(\begin{array}{ccc|c} 2 & 4 & -2 & 1 \\ 4 & -2 & 6 & 0 \\ 6 & -4 & 2 & 0 \end{array}\right), \left(\begin{array}{ccc|c} 2 & 4 & -2 & 0 \\ 4 & -2 & 6 & 1 \\ 6 & -4 & 2 & 0 \end{array}\right), \text{ and } \left(\begin{array}{ccc|c} 2 & 4 & -2 & 0 \\ 4 & -2 & 6 & 0 \\ 6 & -4 & 2 & 1 \end{array}\right).$$

Why not apply them all at once by creating an augmented system with all three right-hand side vectors:

$$\left(\begin{array}{ccc|ccc} 2 & 4 & -2 & 1 & 0 & 0 \\ 4 & -2 & 6 & 0 & 1 & 0 \\ 6 & -4 & 2 & 0 & 0 & 1 \end{array}\right).$$

Then, proceeding with Gaussian elimination:

- By subtracting $(4/2) = 2$ times the first row from the second row, we get

$$\left(\begin{array}{rrr|rrr} 2 & 4 & -2 & 1 & 0 & 0 \\ 0 & -10 & 10 & -2 & 1 & 0 \\ 6 & -4 & 2 & 0 & 0 & 1 \end{array}\right)$$

- By subtracting $(6/2) = 3$ times the first row from the third row, we get

$$\left(\begin{array}{rrr|rrr} 2 & 4 & -2 & 1 & 0 & 0 \\ 0 & -10 & 10 & -2 & 1 & 0 \\ 0 & -16 & 8 & -3 & 0 & 1 \end{array}\right)$$

- By subtracting $((-16)/(-10)) = 1.6$ times the second row from the third row, we get

$$\left(\begin{array}{rrr|rrr} 2 & 4 & -2 & 1 & 0 & 0 \\ 0 & -10 & 10 & -2 & 1 & 0 \\ 0 & 0 & -8 & 0.2 & -1.6 & 1 \end{array}\right)$$

Exercise 4.30 Apply the LU factorization in Figure 4.3 to the matrix

$$\left(\begin{array}{rrr} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{array}\right).$$

Compare and contrast it to the Gauss-Jordan process that we applied to the appended system

$$\left(\begin{array}{rrr|rrr} 2 & 4 & -2 & 1 & 0 & 0 \\ 4 & -2 & 6 & 0 & 1 & 0 \\ 6 & -4 & 2 & 0 & 0 & 1 \end{array}\right).$$

Next, one needs to apply backward substitution to each of the columns. It turns out that the following procedure has the same net effect:

- Look at the "10" and the "-8" in last column on the left of the |. Subtract $(10)/(-8)$ times the last row from the second row, producing

$$\left(\begin{array}{rrr|rrr} 2 & 4 & -2 & 1 & 0 & 0 \\ 0 & -10 & 0 & -1.75 & -1 & 1.25 \\ 0 & 0 & -8 & 0.2 & -1.6 & 1 \end{array}\right)$$

- Now take the "-2" and the "-8" in last column on the left of the | and subtract $(-2)/(-8)$ times the last row from the first row, producing

$$\left(\begin{array}{rrr|rrr} 2 & 4 & 0 & 0.95 & 0.4 & -0.25 \\ 0 & -10 & 0 & -1.75 & -1 & 1.25 \\ 0 & 0 & -8 & 0.2 & -1.6 & 1 \end{array}\right)$$

- Finally take the "4" and the "-10" in second column on the left of the | and subtract $(4)/(-10)$ times the second row from the first row, producing

$$\left(\begin{array}{rrr|rrr} 2 & 0 & 0 & 0.25 & 0 & 0.25 \\ 0 & -10 & 0 & -1.75 & -1 & 1.25 \\ 0 & 0 & -8 & 0.2 & -1.6 & 1 \end{array}\right)$$

- Finally, divide the first, second, and third row by the diagonal elements on the left, respectively, yielding

$$\left(\begin{array}{rrr|rrr} 1 & 0 & 0 & 0.125 & 0 & 0.125 \\ 0 & 1 & 0 & 0.175 & 0.1 & -0.125 \\ 0 & 0 & 1 & -0.025 & 0.2 & -0.125 \end{array}\right)$$

Lo and behold, the matrix on the right is the inverse of the original matrix:

$$\begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} \begin{pmatrix} 0.125 & 0 & 0.125 \\ 0.175 & 0.1 & -0.125 \\ -0.025 & 0.2 & -0.125 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Notice that this procedure works only if no divide by zero is encountered.

(Actually, more accurately, no zero "pivot" (diagonal element of the upper triangular matrix, U, that results from Gaussian elimination) can be encountered. The elements by which one divides in Gauss-Jordan become the diagonal elements of U, but notice that one never divides by the last diagonal element. But if that element equals zero, then the backward substitution breaks down.)

Exercise 4.31 Apply the Gauss-Jordan method to the matrix in Example 4.28 to compute its inverse.

Remark 4.32 Just like Gaussian elimination and LU factorization could be fixed if a zero pivot were encountered by swapping rows (pivoting), Gauss-Jordan can be fixed similarly. It is only if in the end swapping of rows does not yield a nonzero pivot that the process fully breaks down. More on this, later.

Although we don't state the above remark as a formal theorem, let us sketch a proof anyway:

Proof: We previous showed that LU factorization with pivoting could be viewed as computing a sequence of Gauss transforms and pivoting matrices that together transform $n \times n$ matrix A to an upper triangular matrix:

$$\check{L}^{(n-2)}\check{P}^{(n-2)}\check{L}^{(n-3)}\check{P}^{(n-3)}\cdots\check{L}^{(0)}\check{P}^{(0)}A = U,$$

where $\check{L}^{(i)}$ and $P^{(i)}$ represents the permutation applied during iteration i.

Now, if U has no zeroes on the diagonal (no zero pivots were encountered during LU with pivoting) then it has an inverse. So,

$$\underbrace{U^{-1}\check{L}^{(n-2)}\check{P}^{(n-2)}\check{L}^{(n-3)}\check{P}^{(n-3)}\cdots\check{L}^{(0)}\check{P}^{(0)}}_{A^{-1}}A = I,$$

which means that A has an inverse:

$$A^{-1} = U^{-1}\check{L}^{(n-2)}\check{P}^{(n-2)}\check{L}^{(n-3)}\check{P}^{(n-3)}\cdots\check{L}^{(0)}\check{P}^{(0)}.$$

What the first stage of Gauss-Jordon process does is to compute

$$U = \left(\check{L}^{(n-2)}\left(\check{P}^{(n-2)}\left(\check{L}^{(n-3)}\left(\check{P}^{(n-3)}\cdots\left(\left(\check{L}^{(0)}\left(\check{P}^{(0)}A\right)\right)\right)\cdots\right)\right)\right),$$

applying the computed transformations also to the identity matrix:

$$B = \left(\check{L}^{(n-2)}\left(\check{P}^{(n-2)}\left(\check{L}^{(n-3)}\left(\check{P}^{(n-3)}\cdots\left(\left(\check{L}^{(0)}\left(\check{P}^{(0)}I\right)\right)\right)\cdots\right)\right)\right)$$

The second stage of Gauss-Jordan (where the elements of A above the diagonal are eliminated) is equivalent to applying U^{-1} from the left to both U and B.

By viewing the problems as the appended (augmented) system $\left(\begin{array}{c|c} A & I \end{array}\right)$ is just a convenient way for writing all the intermediate results, applying each transformation to both A and I.

4.9.4 Inverting a matrix using the LU factorization

An alternative to the Gauss-Jordan method illustrated above is to notice that one can compute the LU factorization of matrix A, $A = LU$, after which each $Ab_j = e_j$ can be solved by instead solving $Lz_j = e_j$ followed by the computation of the solution to $Ub_j = z_j$. This is an example of how an LU factorization can be used to solve multiple linear systems with different right-hand sides.

One could also solve $AB = I$ for the matrix B by solving $LZ = I$ for matrix Z followed by a computation of the solution to $UB = Z$. This utilizes what are known as "triangular solves with multiple right-hand sides", which go beyond the scope of this document.

Notice that, like for the Gauss-Jordan procedure, this approach works only if no zero pivot is encountered.

Exercise 4.33 The answer to Exercise 4.31 is

$$\begin{pmatrix} 2 & 4 & -2 \\ 4 & -2 & 6 \\ 6 & -4 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 1.6 & 1 \end{pmatrix}\begin{pmatrix} 2 & 4 & -2 \\ 0 & -10 & 10 \\ 0 & 0 & -8 \end{pmatrix}.$$

Solve the three lower triangular linear systems

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 1.6 & 1 \end{pmatrix} \begin{pmatrix} \zeta_{0,0} \\ \zeta_{1,0} \\ \zeta_{2,0} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 1.6 & 1 \end{pmatrix} \begin{pmatrix} \zeta_{0,1} \\ \zeta_{1,1} \\ \zeta_{2,1} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix},$$

$$\text{and } \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 1.6 & 1 \end{pmatrix} \begin{pmatrix} \zeta_{0,2} \\ \zeta_{1,2} \\ \zeta_{2,2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Check (using octave if you are tired of doing arithmetic) that $LZ = I$:

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 1.6 & 1 \end{pmatrix} \begin{pmatrix} \zeta_{0,0} & \zeta_{0,1} & \zeta_{0,2} \\ \zeta_{1,0} & \zeta_{1,1} & \zeta_{1,2} \\ \zeta_{2,0} & \zeta_{2,1} & \zeta_{2,2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Next, solve

$$\begin{pmatrix} 2 & 4 & -1 \\ 0 & -10 & 10 \\ 0 & 0 & 8 \end{pmatrix} \begin{pmatrix} \beta_{0,0} \\ \beta_{1,0} \\ \beta_{2,0} \end{pmatrix} = \begin{pmatrix} \zeta_{0,0} \\ \zeta_{1,0} \\ \zeta_{2,0} \end{pmatrix}, \begin{pmatrix} 2 & 4 & -1 \\ 0 & -10 & 10 \\ 0 & 0 & 8 \end{pmatrix} \begin{pmatrix} \beta_{0,1} \\ \beta_{1,1} \\ \beta_{2,1} \end{pmatrix} = \begin{pmatrix} \zeta_{0,1} \\ \zeta_{1,1} \\ \zeta_{2,1} \end{pmatrix},$$

$$\text{and } \begin{pmatrix} 2 & 4 & -1 \\ 0 & -10 & 10 \\ 0 & 0 & 8 \end{pmatrix} \begin{pmatrix} \beta_{0,2} \\ \beta_{1,2} \\ \beta_{2,2} \end{pmatrix} = \begin{pmatrix} \zeta_{0,2} \\ \zeta_{1,2} \\ \zeta_{2,2} \end{pmatrix}.$$

Check (using octave if you are tired of doing arithmetic) that $UB = Z$ and $AB = I$.

Compare and contrast this process to the Gauss-Jordan process that we applied to the appended system

$$\left(\begin{array}{ccc|ccc} 2 & 4 & -2 & 1 & 0 & 0 \\ 4 & -2 & 6 & 0 & 1 & 0 \\ 6 & -4 & 2 & 0 & 0 & 1 \end{array}\right).$$

You probably conclude that the Gauss-Jordan process is a more convenient way for computing the inverse of a matrix by hand because it organizes the process more conveniently.

Theorem 4.34 Let $L \in \mathbb{R}^{n\times n}$ be a lower triangular matrix with (all) nonzero diagonal elements. Then its inverse L^{-1} exists and is lower triangular.

Proof: Proof by Induction. Let L be a lower triangular matrix with (all) nonzero diagonal elements.

- **Base case:** Let $L = \left(\lambda_{11} \right)$. Then, since $\lambda_{11} \neq 0$, we let $L^{-1} = \left(1/\lambda_{11} \right)$, which is lower triangular and well-defined.

- **Inductive step:** Inductive hypothesis: Assume that for a given $k \geq 0$ the inverse of all $k \times k$ lower triangular matrices with nonzero diagonal elements exist and are lower triangular. We will show that the inverse of a $(k+1) \times (k+1)$ lower trianglar matix with (all) nonzero diagonal elements exists and is lower triangular.

  Let $(k+1)\times(k+1)$ matrix L be lower triangular with (all) nonzero diagonal elements. We will construct a matrix B that is its inverse and is lower triangular. Partition

  $$L \rightarrow \left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right) \quad \text{and} \quad B \rightarrow \left(\begin{array}{c|c} \beta_{11} & b_{12}^T \\ \hline b_{21} & B_{22} \end{array}\right),$$

  where L_{22} is lower triangular (why?) and has (all) nonzero diagonal elements (why?), and $\lambda_{11} \neq 0$ (why?). We will try to construct B such that

  $$\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right) \left(\begin{array}{c|c} \beta_{11} & b_{12}^T \\ \hline b_{21} & B_{22} \end{array}\right) = \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & I \end{array}\right).$$

  (Note: we don't know yet that such a matrix exist.) Equivalently, employing blocked matrix-matrix multiplication, we want to find β_{11}, b_{12}^T, b_{21}, and B_{22} such that

  $$\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right) \left(\begin{array}{c|c} \beta_{11} & b_{12}^T \\ \hline b_{21} & B_{22} \end{array}\right) = \begin{array}{c|c} \lambda_{11}\beta_{11} & \lambda_{11}b_{12}^T \\ \hline \beta_{11}l_{21} + L_{22}b_{21} & l_{21}b_{12}^T + L_{22}B_{22} \end{array}. = \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & I \end{array}\right).$$

  Thus, the desired submatrices must satisfy

  $$\begin{array}{c|c} \lambda_{11}\beta_{11} = 1 & \lambda_{11}b_{12}^T = 0 \\ \hline \beta_{11}l_{21} + L_{22}b_{21} = 0 & l_{21}b_{12}^T + L_{22}B_{22} = I \end{array}.$$

  Now, let us choose β_{11}, b_{12}^T, b_{21}, and B_{22} so that

  - $\beta_{11} = 1/\lambda_{11}$ (why?);
  - $\beta_{12}^T = 0$ (why?);
  - $L_{22}B_{22} = I$. By the inductive hypothesis such a B_{22} exists and is lower triangular; and finally
  - $l_{21}\beta_{11} + L_{22}b_{21} = 0$, or, equivalently, $b_{21} = -L_{22}^{-1}l_{21}/\lambda_{11}$ (which is well-defined because $B_{22} = L_{22}^{-1}$ exists).

  Indeed,

  $$\left(\begin{array}{c|c} \lambda_{11} & 0 \\ \hline l_{21} & L_{22} \end{array}\right) \left(\begin{array}{c|c} 1/\lambda_{11} & 0 \\ \hline -L_{22}^{-1}l_{21}/\lambda_{11} & L_{22}^{-1} \end{array}\right) = \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & I \end{array}\right).$$

- **By the Principle of Mathematical Induction** the result holds for all lower triangular matrices with nonzero diagonal elements.

Theorem 4.35 Let $L \in \mathbb{R}^{n \times n}$ be a unit lower triangular matrix with (all) nonzero diagonal elements. Then its inverse L^{-1} exists and is unit lower triangular.

Theorem 4.36 Let $U \in \mathbb{R}^{n \times n}$ be an upper triangular matrix with (all) nonzero diagonal elements. Then its inverse U^{-1} exists and is upper triangular.

Theorem 4.37 Let $U \in \mathbb{R}^{n \times n}$ be a unit upper triangular matrix with (all) nonzero diagonal elements. Then its inverse U^{-1} exists and is unit upper triangular.

Exercise 4.38 Prove Theorems 4.35–4.37.

Exercise 4.39 Use the insights in the proofs of Theorems 4.34 4.37 to formulate algorithms for

1. a lower triangular matrix.
2. a unit lower triangular matrix.
3. an upper triangular matrix.
4. a unit upper triangular matrix.

Hints for part 1:

- Overwrite the matrix L with its inverse.
- In the proof for Theorem 4.34, $b_{21} = L_{22}^{-1} l_{21} / \lambda_{11}$. When computing this, first update $b_{21} := l_{21}/\lambda_{11}$. Next, *do not invert* L_{22}. Instead, recognize that the operation $l_{21} := L_{22}^{-1} l_{21}$ can be instead viewed as solving $L_{22} x = l_{21}$, overwriting the vector l_{21} with the result x. Then use the algorithm in Figure 4.8 (modified as in Exercise 4.4).

Corollary 4.40 Let $L \in \mathbb{R}^{n \times n}$ be a lower triangular matrix with (all) nonzero diagonal elements. Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array} \right)$, where L_{TL} is $k \times k$. Then $L^{-1} = \left(\begin{array}{c|c} L_{TL}^{-1} & 0 \\ \hline -L_{BR}^{-1} L_{BL} L_{TL}^{-1} & L_{BR}^{-1} \end{array} \right)$.

Proof: Notice that both L_{TL} and L_{BR} are themselves lower triangular matrices with (all) nonzero diagonal elements. Here, their inverses exists. To complete the proof, multiply out LL^{-1} for the partitioned matrices.

Exercise 4.41 Formulate and prove a similar result for the inverse of a partitioned upper triangular matrix.

4.9.5 Inverting the LU factorization

Yet another way to compute A^{-1} is to compute its LU factorization, $A = LU$ and to then note that $A^{-1} = (LU)^{-1} = U^{-1}L^{-1}$. But that requires us to discuss algorithms for inverting a triangular matrix, which is also beyond the scope of this document. This is actually (closer to) how matrices are inverted in practice.

Again, this approach works only if no zero pivot is encountered.

4.9.6 In practice, do not use inverted matrices!

Inverses of matrices are a wonderful theoretical tool. They are not a practical tool.

We noted that if one wishes to solve $Ax = b$, and A has an inverse, then $x = A^{-1}b$. Does this mean we should compute the inverse of a matrix in order to compute the solution of $Ax = b$? The answer is a resounding "no".

Here is the reason: In the previous sections, we have noticed that as part of the computation of A^{-1}, one computes the LU factorization of matrix A. This costs $2/3n^3$ flops for an $n \times n$ matrix. There are many additional computations that must be performed. Indeed, although we have not shown this, it takes about $2n^3$ flops to invert a matrix. After this, in order to compute $b = A^{-1}x$, one needs to perform a matrix-vector multiplication, at a cost of about $2n^2$ flops. Now, solving $Lz = b$ requires about n^2 flops, as does solving $Ux = b$. Thus, simply using the LU factors of matrix A to solve the linear system costs about as much as does computing $b = A^{-1}x$ but avoids the additional $4/3n^3$ flops required to compute A^{-1} after the LU factorization has been computed.

Remark 4.42 If anyone ever indicates they invert a matrix in order to solve a linear system of equations, they either are (1) very naive and need to be corrected; or (2) they really mean that they are just solving the linear system and don't really mean that they invert the matrix.

4.9.7 More about inverses

Theorem 4.43 Let $A, B, C \in \mathbb{R}^{n\times n}$ assume that A^{-1} and B^{-1} exist. Then $(AB)^{-1}$ exists and equals $B^{-1}A^{-1}$.

Proof: Let $C = AB$. It suffices to find a matrix D such that $CD = I$ since then $C^{-1} = D$. Now,

$$C(B^{-1}A^{-1}) = (AB)(B^{-1}A^{-1}) = A \underbrace{(BB^{-1})}_{I} A^{-1} = AA^{-1} = I$$

and thus $D = B^{-1}A^{-1}$ has the desired property.

Theorem 4.44 Let $A \in \mathbb{R}^{n\times n}$ and assume that A^{-1} exists. Then $(A^T)^{-1}$ exists and equals $(A^{-1})^T$.

Proof: We need to show that $A^T(A^{-1})^T = I$ or, equivalently, that $(A^T(A^{-1})^T)^T = I^T = I$. But

$$(A^T(A^{-1})^T)^T = ((A^{-1})^T)^T(A^T)^T = A^{-1}A = I,$$

which proves the desired result.

Theorem 4.45 Let $A \in \mathbb{R}^{n\times n}$, $x \in \mathbb{R}^n$, and assume that A has an inverse. Then $Ax = 0$ if and only if $x = 0$.

Proof:

- Assume that A^{-1} exists. If $Ax = 0$ then

$$x = Ix = A^{-1}Ax = A^{-1}0 = 0.$$

- Let $x = 0$. Then clearly $Ax = 0$.

Theorem 4.46 Let $A \in \mathbb{R}^{n\times n}$. Then A has an inverse if and only if Gaussian elimination with row pivoting does not encounter a zero pivot.

Proof:

- Assume Gaussian elimination with row pivoting does not encounter a zero pivot. We will show that A then has an inverse.

  Let $\check{L}_{n-1}P_{n-1}\cdots\check{L}_0P_0A = U$, where P_k and $\check{L}_k$ are the permutation matrix that swaps the rows and the Gauss transform computed and applied during the kth iteration of Gaussian elimination, and U is the resulting upper triangular matrix. The fact that Gaussian elimination does not encounter a zero pivot means that all these permutation matrices and Gauss transforms exist and it means that U has only nonzeroes on the diagonal. We have already seen that the inverse of a permutation matrix is its transpose and that the inverse of each $\check{L}_k$ exists (let us call it L_k). We also have seen that a triangular matrix with only nonzeroes on the diagonal has an inverse. Thus,

  $$\underbrace{U^{-1}\check{L}_{n-1}P_{n-1}\cdots\check{L}_0P_0}_{A^{-1}}\ A = U^{-1}U = I,$$

  and hence A has an inverse.

- Let A have an inverse, A^{-1}. We will next show that then Gaussian elimination with row pivoting will execute to completion without encountering a zero. We will prove this by contradiction: Assume that Gaussian elimination with column pivoting encounters a zero pivot. Then we will construct a nonzero vector x so that $Ax = 0$, which contradicts the fact that A has an inverse by Theorem 4.45.

  Let us assume that k steps of Gaussian elimination have proceeded without enountering a zero pivot, and now there is a zero pivot. Using the observation that this process can be explained with pivot matrices and Gauss transforms, this means that

  $$\check{L}_{k-1}P_{k-1}\cdots\check{L}_0P_0A = U = \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & 0 & \tilde{a}_{12}^T \\ \hline 0 & 0 & \tilde{A}22 \end{array}\right),$$

  where U_{00} is a $k \times k$ upper triangular matrices with only nonzeroes on the diagonal (meaning that U_{00}^{-1} exists). Now, let

  $$x = \left(\begin{array}{c} -U_{00}^{-1}u_{01} \\ \hline 1 \\ \hline 0 \end{array}\right) \neq 0 \quad \text{so that } Ux = 0.$$

  Then

  $$Ax = P_0^T L_0 \cdots P_{k-1}^T L_{k-1} U x = P_0^T L_0 \cdots P_{k-1}^T L_{k-1} 0 = 0.$$

  As a result, A^{-1} cannot exist, by Theorem 4.45.

Exercise 4.47 Show that

$$\left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline 0 & 0 & \tilde{a}_{12}^T \\ \hline 0 & 0 & \tilde{A}22 \end{array}\right), \left(\begin{array}{c} -U_{00}^{-1} u_{01} \\ \hline 1 \\ \hline 0 \end{array}\right) = \left(\begin{array}{c} 0 \\ \hline 0 \\ \hline 0 \end{array}\right) = 0.$$