Overreaching with Outreach:
Is HPC in Reach?

Maggie Myers
Robert van de Geijn
Are we crazy?

Three Massive Open Online Courses:

• Linear Algebra: Foundations to Frontiers
• LAFF-On Programming for Correctness
• (LAFF-On) Programming for HPC

Educating the world about our world.
Insights for HPC we want to share

• When presenting algorithms, abstract away from details.
• When representing algorithms in code, use APIs.
• When optimizing or parallelizing, don’t just start with legacy code. Derive families of algorithms.
• When implementing software, layer.
• When sharing gained insights use the simplest examples.
Where are we in our journey?

• LAFF
  – Sixth offering on edX.

• LAFF-On Programming for Correctness
  – First offered Spring 2017

• Programming for HPC
  – Planning
What is LAFF?

- Linear Algebra: Foundations to Frontiers (LAFF)
- A full semester class
- Connects the abstractions in mathematics with algorithms and their instantiation in code.
  - Layering of matrix-matrix/matrix-vector/vector-vector operations linked to layering with BLAS.
- Targets novices and those who want a refresher.
- Enriches fundamental linear algebra with HPC issues.
- Introduces FLAME notation and, in enrichments, acquaints the world with our research.
You will learn to

**Slice and Dice**

✧ Recognize, apply, verify, and make connections between properties of vectors and their operations including geometric interpretations, orthogonality, length, vector algebra, and representations as linear combinations of unit basis vector.

✧ Develop and evaluate algorithms for matrix and vector operations.

✧ Make use of decomposition and blocking methods as well as recognize, develop and apply special characteristics of matrices including whether a matrix is triangular, symmetric, diagonal, and invertible.

**Transform and Solve**

✧ Recognize, apply, and verify properties of linear transformations, including their connection to matrices and systems of linear equations.

✧ Determine solutions for systems of equations by applying a variety of methods including Gaussian elimination as well as various LU and QR factorization methods, and compare and contrast the different approaches.

✧ Find eigenvalues and eigenvectors of a matrix.

**Abstract**

✧ Develop and evaluate a variety of algorithms for determining the solutions of systems.

✧ Understand and apply notions such as span, column space, null space, rank, and linear independence in the context of vector spaces and subspaces.

✧ Recognize, develop, and apply various characterizations of linear independence.

✧ Create a small library of basic linear algebra subroutines.
How’s it goin’?
How’s it goin’?
LAFF-On Programming for Correctness

• For many HPC applications, including dense matrix computations, implementations of key operations are loop-based.
• Classic techniques developed in the 1960s by Dijkstra et al. use a loop invariant and the Principle of Mathematical Induction to prove a loop correct.
• FLAME Notation hides indices and facilitates formal derivation of families of algorithms.
• A priori determination of loop invariants.
• Important for HPC: best algorithm for situation can be chosen.
Annotate programs with assertions about the states of variables.

Prove programs correct with Hoare triples and weakest preconditions.

Apply goal oriented programming techniques to derive programs hand-in-hand with their proofs of correctness.

Avoid indexing errors by avoiding indexing.

Determine loop invariants a priori.

Develop families of algorithms for matrix and vector operations.
How’s it goin’?

<table>
<thead>
<tr>
<th>WATCH TIME (MINUTES)</th>
<th>VIEWS</th>
</tr>
</thead>
<tbody>
<tr>
<td>47,170</td>
<td>16,060</td>
</tr>
</tbody>
</table>

On campus course

First offering

First six weeks
The beat goes on...

And now (LAFF-On) Programming for HPC

• Illustrate issues in HPC with simple example: matrix-matrix multiplication (gemm)
Envision three parts:

• Optimizing gemm (for single core and multicore)
• Data movement on distributed memory architectures (MPI and collective communication)
• Parallelizing gemm to distributed memory
Resources

• J. Huang et al. Strassen's Algorithm Reloaded. SC'16
• https://github.com/flame/how-to-optimize-gemm/wiki
• https://github.com/flame/blislab
Challenges

• Funding
• Access to interesting architectures
• Keeping our sanity
Summary

• MOOCs are a way for us to reach a broad audience.

• Benefits?
  – Open source, open platform, open education, open HPC?
  – Crowd source optimizing DLA?
  – Demonstrate broad impact to funders!

• Can they democratize HPC?
Are we Crazy?

TO DO LIST:
1. Wait for Tonight
2. Try to take over the World!
Acknowledgments

Linear Algebra: Foundations to Frontiers
• Partially funded by
  – First NSF SSI grant
  – University of Texas System

LAFF-On Programming for Correctness
• Partially funded by
  – Second NSF SSI grant
  – A gift from Mathworks
  – Profit sharing with edX

(LAFF-On) Programming for HPC
• Partially funded by
  – NSF grant on Strassen’s algorithm
  – ...