

High Performance {Linear Algebra,Solution of Sparse Linear Systems} & Introduction to Parallel MultiGrid

Lecture 0: Bureaucracy and Contents

Ph.D. program in High Performance Scientific Computing

Fabio Durastante **Pasqua D'Ambra** **Salvatore Filippone**

November 10, 2026



Dipartimento
di Matematica
Università di Pisa



About this course: general information

1 General information

First some *bureaucratic* information about the course:

- Course webpage: fdurastante.github.io/courses/hpla2026.html
- Lecture slides: fdurastante.github.io/courses/hpla2026.html#lectures
- The course consists of **30 hours** of lectures, divided into **15 lectures** of **2 hours** each

The **exam** will consist in a **project work** to be presented at the end of the course. This will involve the implementation and performance analysis of some linear algebra algorithms, or the performance analysis of existing libraries, possibly in relation to a specific application. The choice of the project topic will also depend on your Ph.D. research topic, so to make it more interesting and useful for you.



What is Linear Algebra?

2 Motivation (or maybe demotivation)

Linear Algebra is a branch of mathematics concerned with:

- **Vector spaces** and linear transformations
- Systems of **linear equations**, **matrices**, **vectors**
- Key concepts: determinants, eigenvalues, eigenvectors, singular values

Applications: computer graphics, machine learning, optimization, physics

Numerical Linear Algebra focuses on:

- Solving LA problems using numerical methods on computers
- Development of **efficient**, **stable**, and **accurate** algorithms
- Essential for *large-scale* problems where exact solutions are impractical



Problem 1: Linear Systems

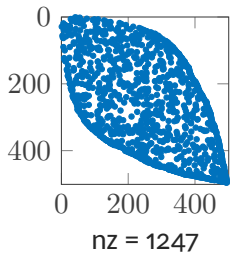
2 Motivation (or maybe demotivation)

Consider the Poisson equation (PDE):

$$-\Delta u = f \quad \text{in } \Omega, \qquad u = 0 \quad \text{on } \partial\Omega$$

Discretization approach:

- Divide domain into grid: $N = n_1 \times n_2 \times n_3$ points
- Use finite difference approximation for derivatives
- Results in sparse linear system: $A\mathbf{u} = \mathbf{f}$
- $A \in \mathbb{R}^{N \times N}$ is **sparse**
- Most elements are zero
- N is typically very large



Sparse matrix pattern



Problem 2: Eigenvalue Problems

2 Motivation (or maybe demotivation)

Find scalar λ and vector $\mathbf{v}$ such that:

$$P\mathbf{v} = \lambda\mathbf{v}$$

Example: Markov Chains

- Transition matrix $P \in \mathbb{R}^{N\times N}$ ($P_{i,j} \geq 0$, rows sum to 1)
- Evolution: $\mathbf{p}_{\ell+1} = P\mathbf{p}_\ell$
- Stationary distribution $\boldsymbol{\pi}$ satisfies:

$$\boldsymbol{\pi}^\top = \boldsymbol{\pi}^\top P, \quad \boldsymbol{\pi}^\top \mathbf{1} = 1$$

- Finding $\boldsymbol{\pi}$ is an eigenvalue problem for large N



Problem 3: Matrix Equations

2 Motivation (or maybe demotivation)

Sylvester equation: $AX + XB = C$

Application: Model Reduction in Control Theory

LTI dynamical system:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \quad \mathbf{y}(t) = C\mathbf{x}(t)$$

Balanced truncation approach:

1. Compute Gramians via Lyapunov equations:

$$AP + PA^\top + BB^\top = 0 \quad \text{(controllability)}$$
$$A^\top Q + QA + C^\top C = 0 \quad \text{(observability)}$$

2. Solve Sylvester equation: $AT + TS = B$
3. Efficient algorithms needed for large dimensions

Problem 4: Machine Learning

2 Motivation (or maybe demotivation)

Linear Regression:

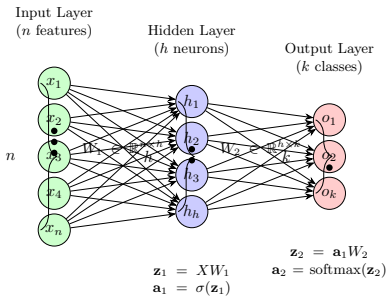
- Data: $X \in \mathbb{R}^{m \times n}$, targets: $\mathbf{y} \in \mathbb{R}^m$
- Find coefficients: $\min_{\boldsymbol{\beta}} \|X\boldsymbol{\beta} - \mathbf{y}\|_2^2$

Neural Networks:

- Weights as matrices: $W_1 \in \mathbb{R}^{n \times h}$, $W_2 \in \mathbb{R}^{h \times k}$
- Forward pass:

$$\mathbf{a}_1 = \sigma(XW_1)$$
$$\mathbf{a}_2 = \text{softmax}(\mathbf{a}_1 W_2)$$

- Training relies on matrix operations



Neural network



Take Home Message

Key Point

Applied mathematics is fundamentally about solving combinations of linear algebra problems.

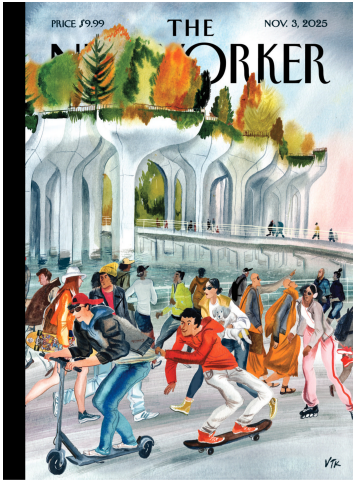
Modern challenges:

- Ever larger problem sizes
- Need for reliable results in reasonable time
- Requirements: *efficient*, *scalable*, *parallel* algorithms

$\Rightarrow$ This motivates high-performance numerical linear algebra!

Straight from the press

2 Motivation (or maybe demotivation)



The math is unwieldy, and is especially dependent on an operation known as matrix multiplication. "Beauty is the first test: there is no permanent place in the world for ugly mathematics," the mathematician G. H. Hardy wrote, in 1940. But ***matrix multiplication****, to which our civilization is now devoting so many of its marginal resources,* ***has all the elegance of a man hammering a nail into a board****. It is possessed of neither beauty nor symmetry: in fact, in matrix multiplication, a times b is not the same as b times a. As the matrices increase in size, the arithmetic requires great computational power to solve.*

S. Witt — The New Yorker, November 3rd, 2025



Nonetheless...

2 Motivation (or maybe demotivation)

I will try to convince you that:



There is both beauty and elegance in Matrix Multiplication!

High Performance Solution of Sparse Linear Systems

3 High Performance Solution of Sparse Linear Systems

Efficient solution of large sparse systems

$$A\mathbf{x} = \mathbf{b}, \qquad A \in \mathbb{R}^{N \times N}.$$

- Iterative methods and effective preconditioners.
- Distributed-memory algorithms for CPUs and GPUs.
- Particular attention to data structure, and software design.

Period: March-April



Salvatore Filippone
Università di Roma
"Tor Vergata"
salvatore.filippone@uniroma2.it

Applications

Computational fluid dynamics, structural analysis, and energy simulations.



Iterative solvers and preconditioning

- **Krylov subspace methods:** Conjugate Gradients for symmetric positive-definite systems, GMRES for nonsymmetric systems.
- **Preconditioning:** accelerate convergence while keeping construction and application costs manageable.
- **Computational kernels:** sparse matrix–vector products, vector updates, and scalar products.

At iteration k, the residual and its preconditioned form are

$$\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k, \qquad \mathbf{z}_k = M^{-1}\mathbf{r}_k.$$

Here $M^{-1}\mathbf{r}_k$ denotes applying the preconditioner, without explicitly forming an inverse.

Performance objective

Reduce the total time to solution, including preconditioner setup and communication.



PSCToolkit and parallel performance

Parallel Sparse Computation Toolkit

- **PSBLAS:** distributed sparse matrices and vectors, parallel sparse operations, and communication.

Practical work

1. Distribute a sparse system and configure an iterative solver.
2. Measure convergence, execution time, and strong and weak scaling on parallel architectures.

Implementation challenges

Data distribution, halo exchanges, global reductions, and load balance on CPU and GPU systems.

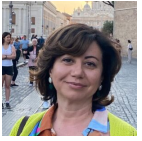
Introduction to Parallel MultiGrid

4 Introduction to Parallel MultiGrid

Solving a problem on several levels

- Fundamentals of geometric and algebraic multigrid.
- Smoothing and coarse-space approximation.
- Theoretical foundations of convergence.
- Design of scalable parallel algebraic multigrid methods.

Period: March-April (4 hours per week over 2 days, for a total of 16 hours)



Pasqua D'Ambra

Istituto per le Applicazioni del Calcolo "Mauro Picone"
National Research Council of Italy

pasqua.dambra@cnr.it

Suggested prerequisites

High-performance linear algebra and introductory knowledge of partial differential equations.



The multigrid idea

1. **Pre-smoothing:** reduce error components efficiently treated by relaxation.
2. **Coarse correction:** approximate the remaining error in a smaller space.
3. **Post-smoothing:** further reduce error after the correction.

For a symmetric positive-definite system, a Galerkin coarse correction uses

$$\mathbf{r} = \mathbf{b} - A\mathbf{x}, \quad A_c = P^\top A P,$$
$$A_c \mathbf{e}_c = P^\top \mathbf{r}, \qquad \mathbf{x} \leftarrow \mathbf{x} + P\mathbf{e}_c.$$

P transfers coarse corrections to the fine level, while $P^\top$ restricts residuals.

Multilevel cycles

Apply the same procedure recursively to the coarse problems.



Geometric and algebraic constructions

Geometric multigrid: mesh geometry defines levels and transfer operators.

Algebraic multigrid: matrix structure and coefficients guide coarsening and interpolation.

Main questions addressed in the course

- **Convergence:** how do smoothing and approximation properties work together?
- **Hierarchy construction:** how do coarsening and aggregation affect accuracy, sparsity, and setup cost?
- **Parallel execution:** how can communication and load imbalance limit scalability?
- **Coarse levels:** how can we address the decreasing amount of work available per process?



PSCToolkit and parallel performance

Parallel Sparse Computation Toolkit

- **PSBLAS:** distributed sparse matrices and vectors, parallel sparse operations, and communication.
- **AMG4PSBLAS:** algebraic multigrid preconditioners.

Practical work

1. Distribute a sparse system and configure an iterative solver.
2. Implement and compare algebraic multilevel preconditioners.
3. Measure convergence, execution time, and strong and weak scaling on parallel architectures.

Implementation challenges

Granularity of the computations, bridging convergence theory and scalability, managing communications as the number of processes increases.