

# **A History of Modern Numerical Linear Algebra**

Gene H. Golub  
Stanford University

## What is Numerical Analysis? (a.k.a Scientific Computing)

The definitions are becoming shorter...

- **Webster's New Collegiate Dictionary (1973):**  
"The study of quantitative approximations to the solutions of mathematical problems including consideration of the errors and bounds to the errors involved."
- **The American Heritage Dictionary (1992):**  
"The study of approximate solutions to mathematical problems, taking into account the extent of possible errors."
- **L. N. Trefethen, Oxford University (1992):**  
"The study of algorithms for the problems of continuous mathematics."

## Numerical Analysis: How It All Started

Numerical analysis motivated the development of the earliest computers.

- Ballistics
- Solution of PDE's
- Data Analysis

Early pioneers included:

J. von Neumann, A. M. Turing

In the beginning...

von Neumann & Goldstine (1947):

”Numerical Inversion of Matrices of High Order”

## Numerical Linear Algebra

Numerical Linear Algebra (NLA) is a small but active area of research: Less than 200 active, committed persons. But the community involves many scientists.

## Top Ten Algorithms in Science (Jack Dongarra, 2000)

1. Metropolis Algorithm for Monte Carlo
2. Simplex Method for Linear Programming
3. Krylov Subspace Iteration Methods
4. The Decompositional Approach to Matrix Computations
5. The Fortran Optimizing Compiler
6. QR Algorithm for Computing Eigenvalues
7. Quicksort Algorithm for Sorting
8. Fast Fourier Transform
9. Integer Relation Detection
10. Fast Multipole Method
  - Red: Algorithms within the exclusive domain of NLA research.
  - Blue: Algorithms strongly (though not exclusively) connected to NLA research.

## Three important components in solving NLA problems

- Development and analysis of numerical algorithms.
- Perturbation theory.
- Software.

## A Fundamental Problem

Problem: Suppose

$$Ax = b + r,$$

where  $A$  is an  $m \times n$  matrix, and  $b$  is a given vector.

Goal: Determine  $r$  such that

$$\|r\| = \min.$$

## Important Parameters

- The relationship between  $m$  and  $n$ :
  - Overdetermined vs. 'square' vs. Underdetermined.
  - Uniqueness of solution.
- The rank of the matrix  $A$ . (Difficult to handle if a small perturbation in  $A$  will change rank.)
- Choice of norm.
- Structure of  $A$ :
  - Sparsity.
  - Specialized matrices such as Hankel or Toeplitz.
- Origin of problem: ideally, can make use of this in developing an algorithm.



## Some Perturbation Theory

Given  $Ax = b$ ,

and the perturbed system  $(A + \Delta A)y = b + \delta$ ,

it can be shown that if

$$\frac{\|\Delta\|}{\|A\|} \leq \varepsilon, \quad \frac{\|\delta\|}{\|b\|} \leq \varepsilon, \quad \rho < 1,$$

then

$$\frac{\|x - y\|}{\|x\|} \leq \frac{2\varepsilon}{1 - \rho} \cdot \kappa(A),$$

where  $\rho = \|\Delta\| \cdot \|A^{-1}\| = \|\Delta\| \cdot \kappa(A) / \|A\|$ ,

and  $\kappa(A) = \|A\| \cdot \|A^{-1}\|$ .

## The Condition Number

The quantity

$$\kappa(A) = \|A\| \cdot \|A^{-1}\|$$

is called **the condition number** of  $A$  (or the condition number of the linear system).

Note:

- Even if  $\varepsilon$  is small, a large  $\kappa$  can be destructive.
- A special relationship between  $A$  and  $b$  may further determine the conditioning of the problem.

A detailed theory of condition numbers:

John Rice, 1966.

## Stability: Algorithm vs. Problem

- Fundamentally important to distinguish between the conditioning of the problem and the stability of the algorithm.
- Even if an algorithm is stable, not all problems can be solved using it.
- Making the **problem** well-posed → responsibility of modeller.  
Making the **algorithm** stable → responsibility of numerical analyst.
- A good algorithm is one for which a small change in the input of a well-posed problem causes a small change in the output.

## Solving Linear Systems

1. Gaussian Elimination.
2. The QR decomposition.
3. The Cholesky method.
4. Iterative solution.

## A Little Bit About Gaussian Elimination

Not off to a good start....:

The famous statistician Hotelling derived bounds so pessimistic that he recommended not to use it for large problems.

But this is a story with a happy end:

- Goldstine and von Neumann's analysis of the Cholesky method for fixed point arithmetic.
- Wilkinson's complete round-off error analysis of Gaussian Elimination in 1961.

Those developments were turning points for GE and it has become one of the most commonly used algorithms.

## The Silent Killer (a.k.a round-off errors)

Round-off errors are quietly accumulated in every computation, and should not be overlooked!

There is an error inherent in any computer's most basic arithmetic operations:

$$fl(x + y) = x(1 + \varepsilon) + y(1 + \varepsilon) = \bar{x} + \bar{y}.$$

Gaussian Elimination with pivoting is equivalent to performing the decomposition

$$\Pi A = L \cdot U.$$

$\Pi$  is a permutation matrix,  $L$  and  $U$  are lower and upper triangular. This algorithm guarantees that

$$\max_{i \geq j} |\ell_{i,j}| = 1 \quad \text{and} \quad \max_{j \geq i} |u_{i,j}| \leq 2^{n-1}.$$

## Wilkinson's Backward Error Analysis

By Wilkinson, GE with pivoting is equivalent to solving

$$(A + E)y = b,$$

where

$$\|E\|_{\infty} \leq 8n^3 G \|A\|_{\infty} u + O(u^2)$$

and

$$|u_{i,j}| \leq G.$$

$u$  is the machine roundoff unit.

**Backward Error Analysis:** shows how the original data matrix has been perturbed in the presence of round-off errors.

Importance: the error can be bounded.
---------------------------------------

## The Simplex Algorithm

Wilkinson's work enabled development of algorithms for many classes of problems.

Consider Linear Programming problem: Given

$$Ax = b \quad \text{where } A \text{ is } m \times n, \text{ with } m < n,$$

determine  $x$  such that  $x \geq 0$  and  $c^T x = \min$ .

Basic algorithm (due to Dantzig): the basis

$A^{(k)} = [a_{i_1}, a_{i_2}, \dots, a_{i_m}]$  is replaced by

$$A^{(k+1)} = [a_{i_1}, \dots, a_{i_{p-1}}, a_{i_q}, a_{i_{p+1}}, \dots, a_{i_m}],$$

so that  $A^{(k+1)}$  differs from  $A^{(k)}$  by one column.

The approximants  $x^{(k)}$  and  $x^{(k+1)}$  satisfy

$$A^{(k)} x^{(k)} = b; \quad A^{(k+1)} x^{(k+1)} = b.$$



## Simplex (cont.)

Given  $\Pi^{(k)} A^{(k)} = L^{(k)} U^{(k)}$ , we seek a method for computing

$$\Pi^{(k+1)} A^{(k+1)} = L^{(k+1)} U^{(k+1)},$$

within  $\mathcal{O}(m^2)$  operations.

Bartels & G. :

A stable algorithm for applying the method.

Classical algorithm is based on Gauss-Jordan elimination, which is stable for limited classes of matrices. Here we encounter the classical problem of **sparsity vs. stability**.

## Linear Algebra and Optimization

NLA plays an important role in optimization:

- Strong connection between a formulation of a linear system, and a minimization formulation. (Example: CG, which we will talk about soon.)
- Even in nonlinear optimization, the majority of computing time is spent on solving the linear systems involved!

## Example: Quadratic Programming

Equality-constrained quadratic programs:

$$\text{Minimize } \frac{1}{2}x^T Ax - x^T c \text{ subject to } B^T x = d.$$

Lagrange Multipliers formulation: define

$$\phi(x, \lambda) = x^T Ax - x^T c + \lambda^T (B^T x - d)$$

and compute its stationary points:

$$\nabla \phi = 0.$$

$$\mathcal{K}u = \begin{pmatrix} A & B \\ B^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} c \\ d \end{pmatrix}$$

## Updating and Downdating

Shows up in a variety of applications. Examples:

- Data fitting
- Kalman filters
- Signal processing

## Example: Least Squares

Determine  $x$  such that

$$\|b - Ax\|_2 = \min.$$

Popular method: The QR factorization. (Better than forming the Normal Equations!)

How to generate an orthogonal matrix  $Q$ ? — Use the modified Gram-Schmidt method, Householder Transformations or Givens Rotations.

Don't use classical Gram-Schmidt!

Frequently a row or a column of  $A$  are added or deleted: important and delicate theory.

## The Singular Value Decomposition (SVD)

Let  $A$  be an  $m \times n$  matrix. The singular value decomposition is

$$A = U\Sigma V^T, \quad \text{where } U^T U = I_m; \quad V^T V = I_n;$$

and

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \ddots & \vdots \\ \vdots & 0 & \ddots & 0 \\ \vdots & & \ddots & \sigma_n \\ \vdots & & & 0 \\ \vdots & & & \vdots \\ 0 & & & 0 \end{pmatrix}.$$

## SVD (cont.)

- The singular values are typically ordered monotonically:

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0.$$

- The non-zero singular values of  $A$  are the square roots of the non-zero eigenvalues of  $A^T A$ :

$$\sigma_i(A) = (\lambda_i(A^T A))^{1/2}.$$

- Stable algorithm for computing the decomposition: G. & Kahan, 1965.

## The SVD is very useful in numerous applications of great importance

In addition to its enormous importance in NLA algorithms, the SVD is useful in areas of applications of importance to the whole scientific community, and has influenced many people's lives!

- Vision and motion analysis
- Signal processing
- Search engines and data mining (Google!)



## Examples of Use of the SVD

1. Truncated SVD as an optimal low rank approximation
2. The least squares problem
3. Determining the rank of a matrix

### Example: low rank approximation

Let  $A$  be an  $m \times n$  matrix of rank  $r$ . The matrix  $A_k$  such that

$$\|A - A_k\|_2 = \min$$

is simply the matrix given by  $A_k = U\Sigma_k V^T$ , where

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \sigma_k & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & & & & 0 & 0 \end{pmatrix}.$$

## Computing the SVD

Many ways...

A popular method (G. & Kahan, 1965):

**Bi-diagonalization.**

Find  $X$  such that  $X^T X = I_m$  ;  $Y$  such that  $Y^T Y = I_n$  and

$$B = \begin{pmatrix} \alpha_1 & \beta_1 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \beta_{n-1} \\ 0 & \cdots & 0 & \alpha_n \end{pmatrix},$$

such that

$$X^T A Y = (B \ 0)^T.$$

By using a variant of the QR method, the matrix  $B$  is diagonalized.

## Cyclic Reduction

Consider the system

$$\begin{pmatrix} I & F \\ F^T & I \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} g \\ h \end{pmatrix}.$$

The matrix of the system is said to have **Property A**.

Easy to eliminate  $u$ :

$$(I - F^T F)v = h - F^T g.$$

The matrix  $I - F^T F$  can be reordered in some cases, to have the same structure as above: can repeat this procedure again and again, eliminating half of the remaining unknowns at each step.

Resulting algorithm reminiscent of FFT! —  $\mathcal{O}(N^2 \log N)$  operations to solve the system.

## Example: Poisson's equation (1D)

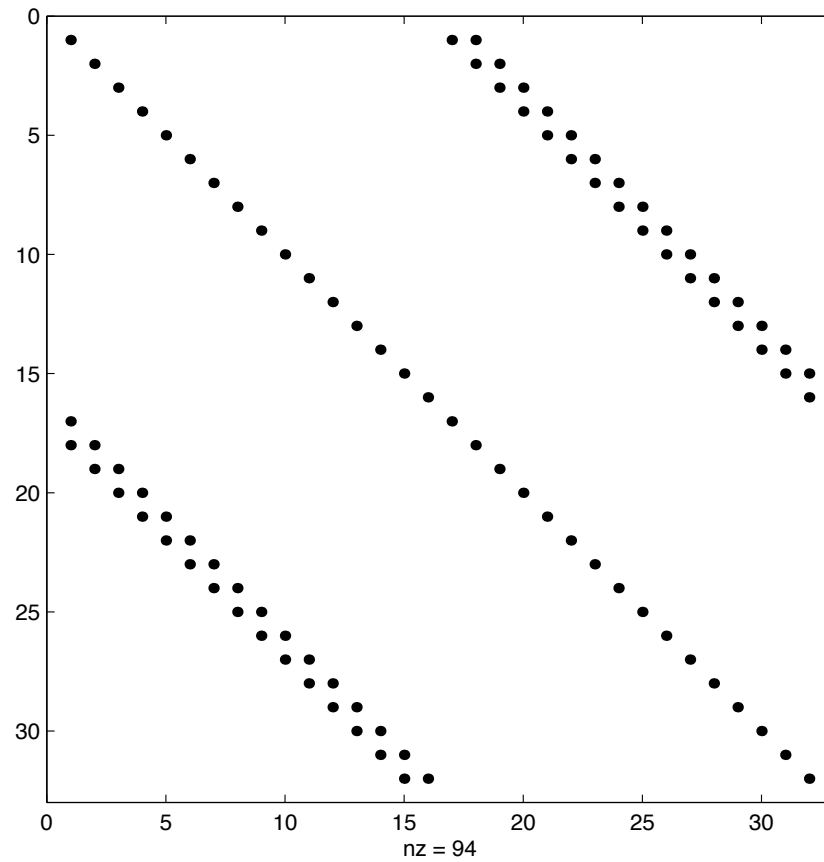
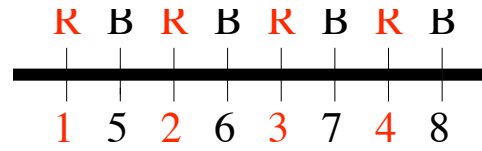
Consider the ordinary differential equation on  $[0,1]$ :

$$\begin{aligned} -u''(x) &= f(x) \quad , \\ u(0) &= a, \quad u(1) = b. \end{aligned}$$

Discretizing using centered schemes on a uniform mesh, the matrix associated with the linear system is:

$$A = \begin{pmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -1 & 2 & -1 & \\ & & & -1 & 2 \end{pmatrix} .$$

## Red/black re-ordering



## Red/black re-ordering (cont.)

After scaling by  $1/2$ , the linear system can be written as:

$$\begin{pmatrix} I & F \\ F^T & I \end{pmatrix} \begin{pmatrix} u^{(r)} \\ u^{(b)} \end{pmatrix} = \begin{pmatrix} s^{(r)} \\ s^{(b)} \end{pmatrix}.$$

And now, do it again...

## Iterative Methods

For large sparse linear systems, Gaussian Elimination may not be a good algorithm, due to the **fill-in**.

Iterative methods are based on computing a sequence of approximations  $x^{(k)}$ , starting with an initial guess  $x^{(0)}$ , and ideally converging 'sufficiently close' to the solution after a 'reasonable' number of iterations.



## The (50 years old!) Conjugate Gradient

The celebrated Conjugate Gradient algorithm (Hestenes & Stiefel [1952]) is an optimal approximation in the following sense: At the  $n$ th iteration,

$$\begin{aligned}x^{(n)} - x^{(0)} &\in K_n(A) \\ &= \text{span}\{r^{(0)}, Ar^{(0)}, A^2r^{(0)}, \dots, A^{n-1}r^{(0)}\}\end{aligned}$$

such that

$$\begin{aligned}\|x - x^{(n)}\|_A &= \|b - Ax^{(n)}\|_{A^{-1}} = \|r^{(n)}\|_{A^{-1}} \\ &= \min_{u \in K_n(A)} \|x - u\|_A\end{aligned}$$

The idea is based on picking directions  $p^{(k)}$  such that

$$p^{(i)T} Ap^{(j)} = 0 \quad \text{for } i \neq j.$$

## CG (cont.)

The iterations are computed by

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)}.$$

The residual satisfies

$$r^{(k+1)} = r^{(k)} - \alpha_k A p^{(k)}.$$

The CG method is optimal in that the **error** is minimized over the Krylov subspace in the energy norm  $\|e\|_A \equiv e^T A e$ . The sequence of errors satisfies

$$\|e_n\|_A \leq \|e_{n-1}\|_A.$$

The beauty of the method is that  $p^{(k)}$  can be chosen so that the iterate  $x^{(k+1)}$  really minimizes the error over the whole Krylov subspace, not only over  $\text{span}(x^{(k)}, p^{(k)})$ .

## CG as an Optimization Algorithm

The  $p^{(k)}$  can be thought of as search directions in a nonlinear optimization problem!

The problem is:

$$\text{minimize } \phi(x) = \frac{1}{2}x^T Ax - x^T b$$

Equating the gradient of  $\phi$  to zero takes us (obviously) to the beloved combination of characters

$$Ax = b.$$

The **direction**  $p^{(k)}$  and the **step length**  $\alpha_k$  can be determined mathematically by the formulation of the problem.

## Iterative Solution

Consider the linear system  $Ax = b$ . The splitting

$$A = M - N; \quad M\mathbf{x} = \mathbf{N}\mathbf{x} + \mathbf{b}$$

leads to the ‘**fundamental**’ iterative scheme:  $M\mathbf{x}^{k+1} = \mathbf{N}\mathbf{x}^k + \mathbf{b}$  (\*).

Define the error and the iteration matrix:  $\mathbf{e}^k = \mathbf{x} - \mathbf{x}^k$ ;  $\mathbf{K} = \mathbf{M}^{-1}\mathbf{N}$ .

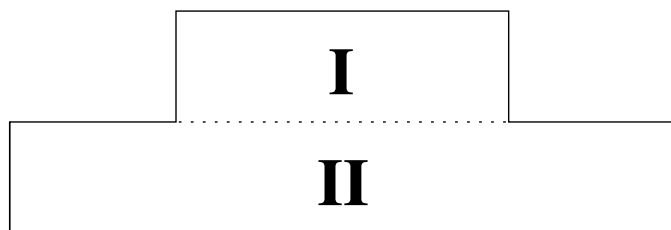
We obtain  $\mathbf{e}^k \rightarrow 0$  as  $k \rightarrow \infty$ , if  $\rho(\mathbf{K}) < 1$ .

We assume it is “easy” to solve (\*), which is equivalent to

$$(**) \quad \begin{cases} M\mathbf{z}^k = \mathbf{r}^k \equiv \mathbf{b} - \mathbf{A}\mathbf{x}^k \equiv \mathbf{A}\mathbf{e}^k \\ \mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{z}^k \end{cases} .$$

## Examples of Splittings

### 1. Domain Decomposition:



$$A = \begin{pmatrix} A_1 & & & B_1 \\ & \ddots & & \vdots \\ & & A_r & B_r \\ B_1^T & \dots & B_r^T & Q \end{pmatrix}$$

$$M = \begin{pmatrix} A_1 & & & \\ & \ddots & & \\ & & A_r & \\ & & & Q \end{pmatrix}, N = - \begin{pmatrix} & & & B_1 \\ & & & \vdots \\ & & & B_r \\ B_1^T & \dots & B_r^T & 0 \end{pmatrix}$$

## Examples of Splittings (cont.)

2. Non-symmetric problems:

$$\begin{aligned} A &= \frac{A + A^T}{2} + \frac{A - A^T}{2} \\ &= M - N \end{aligned}$$

Concus & G.: The Conjugate gradient and Chebyshev methods will converge, provided that for any real vector  $\mathbf{u}$  we have

$$\mathbf{u}^T M \mathbf{u} > 0.$$

## Software

A large number of software packages are available, and their quality gets better and better.

Where should you start your journey? —

An invaluable source is [Netlib](#), a numerical software distribution system.

- LINPACK
- EISPACK
- FISHPACK
- MATLAB

Recent important package:

ATLAS: Automatically Tuned Linear Algebra Software.

## Hot Areas of Research

- Model reduction problems
- Polynomial eigenvalue problems
- PDE solvers in 3D
- Parallel processing



## Parallel Processing

### Architecture aspects:

mesh processing, blocking sizes, software pipelining strategies, register allocations, memory hierarchy, memory distribution, ...

### Algorithmic aspects:

Multi-color orderings, recursive approach, 'locality', ...

### What goes around comes around...:

In some cases old methods that have been 'dead' for a long time have been resurrected, since they are good for parallel environments. (Simple example: the Jacobi algorithm.)

## Future

Scientific Computing is driven by technology:

1. New devices and environments.
2. New problems that will require new techniques.

## Personalities



## Founding Fathers

The wonderful people who were the 'founding fathers' of the field:

1. J. H. Wilkinson
2. A. S. Householder
3. G. E. Forsythe
4. H. Rutishauser

and many others...