Lecture 07 – Solving Linear Systems (Part 1)

Prof. Brendan Kochunas

NERS/ENGR 570 - Methods and Practice of Scientific Computing (F20)

COLLEGE OF ENGINEERING
NUCLEAR ENGINEERING & RADIOLOGICAL SCIENCES
UNIVERSITY OF MICHIGAN

Outline

- Solution of Linear Systems
- ~~Direct Methods and Matrix Factorizations~~
- Iterative Methods
- ~~Third Party Libraries~~

Learning Objectives: By the end of Today's Lecture you should be able to

- ~~*(Skill)* Perform Conflict Resolution (but just in git)~~
- ~~*(Knowledge)* Interpretate meaning of some vector norms~~
- ~~(Value/Knowledge) explain how to think about programming equations in linear algebra~~
- *(Knowledge)* know when to use direct vs. iterative solution algorithms
- (*Knowledge*) implement LU factorization
- (Knowledge) how to implement some iter. & understand convergence

Basic Linear Algebra Operations

Residual and Norms of Vectors

$\mathbf{r} = \mathbf{A}\mathbf{x} - \mathbf{b}$ residual

$\|\mathbf{r}\|_1 = \sum_i |r_i|$ 1-norm

$\|\mathbf{r}\|_2 = \sqrt{\sum_i r_i^2}$ 2-norm ("average error")

$\|\mathbf{r}\|_\infty = \max_i(|r_i|)$ ∞-norm ("max local error")

$\|\mathbf{r}\|_p = \left(\sum_i |r_i|^p\right)^{1/p}$ *p*-norm

Inner/Dot Product (vector-vector multiply)

$$\mathbf{u}^T \cdot \mathbf{v} = \sum_i u_i v_i$$

Matrix-vector Multiply

$$\mathbf{A}\mathbf{x} = \mathbf{b} \rightarrow b_i = \sum_j a_{i,j} x_j$$

Matrix-Matrix Multiply

$$\mathbf{A}\mathbf{B} = \mathbf{C} \rightarrow \boxed{c_{i,j}} = \sum_k a_{i,k} b_{k,j}$$

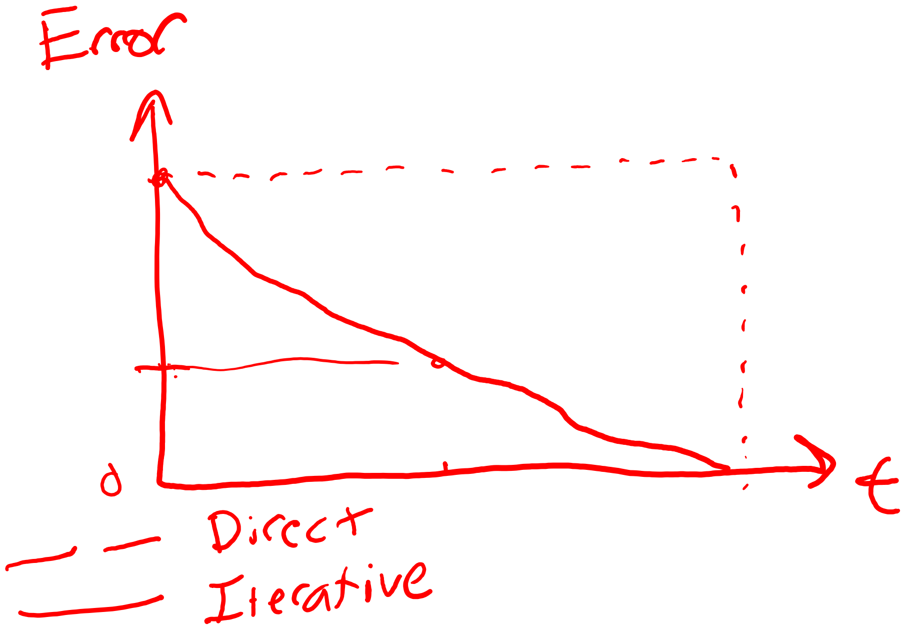
Solving Linear Systems

How do we solve linear systems?

MATLAB $A\backslash b = x$

$A^{-1}Ax = A^{-1}b \;\Rightarrow\; x = A^{-1}b \qquad A^{-1}A = I = AA^{-1}$

Ap.Solv(A, b, x)



Direct Methods: Matrix Factorizations

Operation count $\sim O(?\, n^3)$

Iterative methods

repetitive application of Mat-vec $\sim O(?\, n^2) \times m$

$m < n$

$(8\text{-bytes})(10^{12}) = 8$ TB GB

Overview of Solution Methods

$\boxed{Ax = b}$

Iterative

- Fixed Point (Classical Iterative Methods)
- Multigrid Methods (1970's-1980's)
- Krylov Methods
  - GMRES
  - Lanczos & Arnoldi (1950's)
  - Conjugate Gradient

Direct

- Gaussian Elim. / LU
- QR decomposition

Types of Matrices

1. Identity Matrix
2. Triangular
3. Diagonal
4. Symmetric
5. Tri-diagonal
6. Diagonally Dominant
7. Sparse/Dense
8. Square
9. Unitary
10. Positive definite
11. Invertible
12. scalar
13. Hermitian
14. Orthonormal

Direct Solution Methods SuperLU

aka Matrix Factorizations

LU Factorization

$A = \underbrace{LU}$ $\quad L = \begin{bmatrix} \Delta & 0 \end{bmatrix}$ $\quad U = \begin{bmatrix} 0 & \Delta \end{bmatrix}$

$Ax = b \rightarrow LUx = b$

$\cancel{L^{-1}L}Ux = \underline{L^{-1}b} = y$

$$\begin{cases} Ux = y & \text{- Backward Subs. (2)} \\ Ly = b & \text{- Foward Elm. (1)} \end{cases}$$

LU: Forward Elimination

$Ly = b$

$L = \begin{bmatrix} & 0 \end{bmatrix}$

$a_{1,1} = l_{1,1} y_1 = b_1 \Rightarrow y_1 = \frac{b_1}{l_{1,1}}$

$l_{1,1} y_1 + l_{2,1} y_2 = b_2 \Rightarrow y_2 = \frac{b_2 - l_{1,1} y}{l_{2,1}}$

$y_i = \left(b_i - \sum_{j=1, i-1} l_{i,j} y_j \right) / l_{i,i}$

LU: Backward Substitution

$$Ux = y \qquad U =$$

$$U_{n,n} x_n = y_n \Rightarrow x_n = \frac{y_n}{u_{n,n}}$$

$$x_{n-1} = \frac{y_{n-1} - U_{n,n} x_n}{u_{n-1,n-1}}$$

$$x_i = \left(y_i - \sum_{j=n,i+1,-1} u_{i,j} x_j \right) / u_{i,i}$$

Classical Iterative Methods

aka Fixed Point Iteration Schemes

Classical Iteration Schemes

$$A = L + D + U$$

$$L = \begin{bmatrix} \diagdown & \\ \text{lower} & \diagdown \end{bmatrix} \quad D = \begin{bmatrix} \diagdown \end{bmatrix} \quad U = \begin{bmatrix} \diagdown & \text{upper} \\ & \diagdown \end{bmatrix}$$

Jacobi

$$x_i^{(\ell+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{\substack{j=1 \\ i \neq j}}^{n} a_{ij} x_j^{(\ell)} \right]$$

$$- \sum_{j=1}^{i-1} a_{ij} x_j^{(\ell)} - \sum_{i+1}^{n} a_{ij} x_j^{(\ell)}$$

$$x^{(\ell+1)} = \underbrace{-D^{-1}(L + U)}\, x^{(\ell)} + D^{-1} b$$

$$E = -D^{-1}(L + U)$$

Gauss-Siedel

$$x_i^{(\ell+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(\ell+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(\ell)} \right]$$

$$x^{(\ell+1)} = \underbrace{-(D+L)^{-1} U}\, x^{(\ell)} + (D+L)^{-1} b$$

$$x^{(\ell+1)} = E\, x^{(\ell)} + c \qquad -(D+L)^{-1} U$$

Do they converge?

- Fixed point iteration

$$\mathbf{x}^{(\ell+1)} = \mathbf{F}\mathbf{x}^{(\ell)} + \mathbf{c}$$

- Express iterate as combination of exact solution and error

$$\mathbf{x} + \boldsymbol{\varepsilon}^{(\ell+1)} = \mathbf{F}\left(\mathbf{x} + \boldsymbol{\varepsilon}^{(\ell)}\right) + \mathbf{c}$$

$x = Fx + c$

- If the method converges then:

$$\boxed{\lim_{\ell \to \infty} \boldsymbol{\varepsilon}^{(\ell)} = 0}$$

Hands on Python Examples