# CS 450: Numerical Anlaysis

## Lecture 28
## Chapter 11 Partial Differential Equations
### Solving Sparse Linear Systems

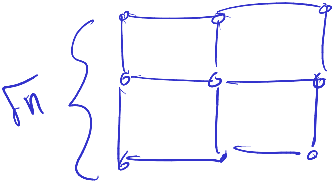
Edgar Solomonik

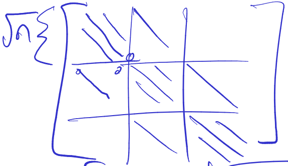
Department of Computer Science
University of Illinois at Urbana-Champaign

April 27, 2018

# Sparse Linear Systems

- Finite-difference and finite-element methods for time-independent PDEs give rise to sparse linear systems:




$I \otimes T + T \otimes I$ $\quad \boxed{nnz = O(n)}$

$T = \begin{bmatrix} -2 & 1 & \\ 1 & -2 & \\ & & \ddots \end{bmatrix}$ matrix dimension

often Toeplitz

- *Direct methods* apply LU or other factorization to $\boldsymbol{A}$, while *iterative methods* refine $\boldsymbol{x}$ by minimizing $\boldsymbol{r} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$, e.g. via Krylov subspace methods.

- direct - factorize $A$, e.g. $A = LU$ $\quad b = \sqrt{n}$

$O(n^3)$ $\quad O(n^2)$

$O(nb^2)$ for banded

- iterative - matrix-vector products

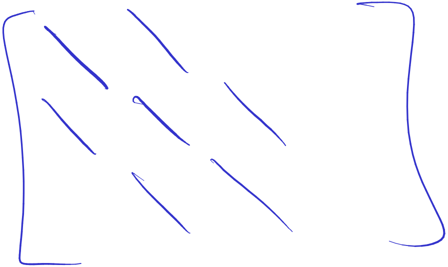
$m = \sqrt{n}$

$$T \otimes I = m \begin{bmatrix} t_{11} I & t_{12} I & \\ t_{21} I & t_{22} I & \\ 0 & t_{32} I & \ddots \end{bmatrix}$$

$$T \in \mathbb{R}^{m \times m}$$

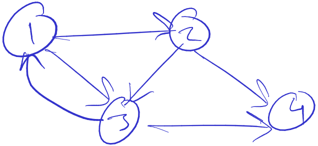
$$T = \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & & \\ & & \ddots & 1 \\ & & 1 & -2 \end{bmatrix}$$

$$=$$



# Direct Methods for Sparse Linear Systems

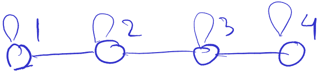
- It helps to think of $\boldsymbol{A}$ as the adjacency matrix of graph $G = (V, E)$ where $V = \{1, \dots n\}$ and $a_{ij} \neq 0$ if and only if $(i, j) \in E$:

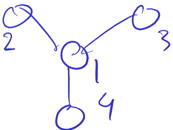


$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & & & \times & \\ 2 & \times & & & \\ 3 & \times & \times & & \\ 4 & & \times & \times & \end{array}$$

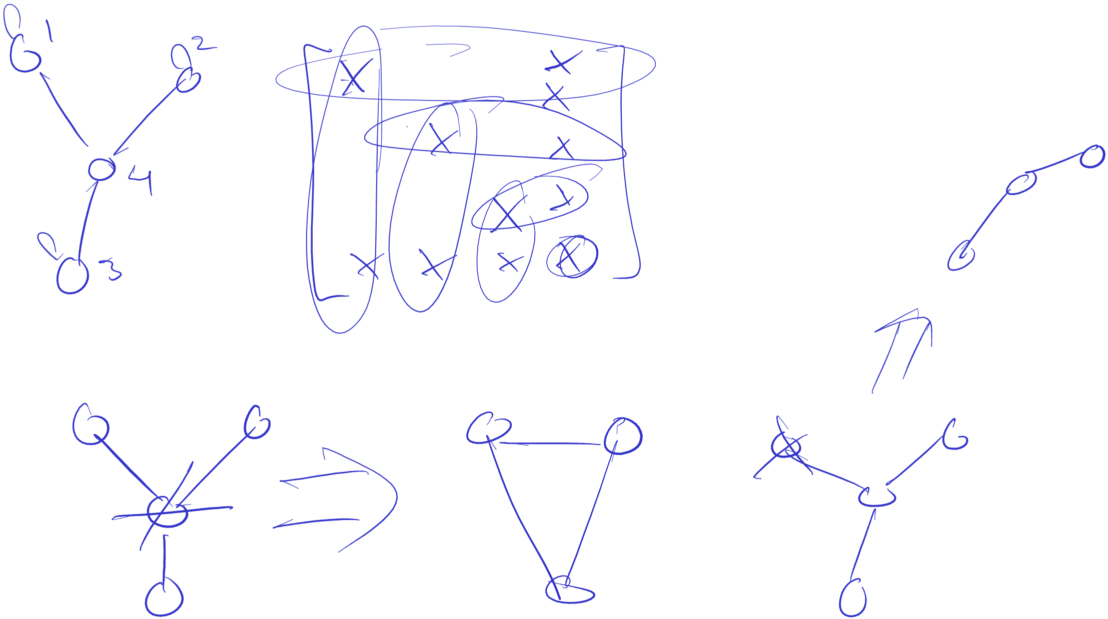
- Factorizing the $l$th row/column in Gaussian elimination corresponds to removing node $i$, with nonzeros (new edges) introduces for each $k, l$ such that $(i, k)$ and $(i, l)$ are in the graph.

$$\begin{bmatrix} \times & \times & & \\ \times & \times & \times & \\ & \times & \times & \times \\ & & \times & \times \end{bmatrix} \Rightarrow$$





$$\begin{bmatrix} \times & \times & \times & \times \\ \times & & & \\ \times & & & \\ \times & & & \end{bmatrix} \Rightarrow \begin{bmatrix} & \end{bmatrix} - \begin{bmatrix} a_{21}/a_{11} \\ a_{31}/a_{11} \\ a_{41}/a_{11} \end{bmatrix}$$


1
2
3
4

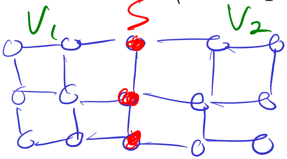
# Vertex Orderings for Sparse Direct Methods

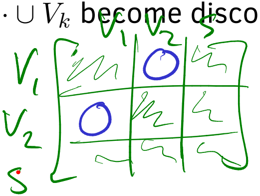
- Select the node of minimum degree at each step of factorization:

  row/col with minimum nonzeros

  minimize number of entries update

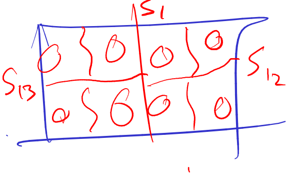
- Graph partitioning also serves to bound fill, remove vertex separator $S \subset V$ so that $V \setminus S = V_1 \cup \cdots \cup V_k$ become disconnected, then order $V_1, \ldots, V_k, S$:

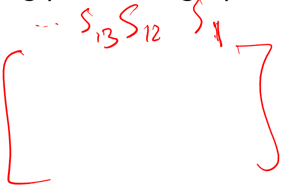




- *Nested dissection* ordering partitions graph into halves recursively, ordering each separator last.



# Sparse Iterative Methods

- Direct sparse factorization is ineffective in memory usage and/or cost for many typical sparsity matrices, motivating iterative methods:

$$Mx_{k+1} = Nx_k + b,$$

choose $M, N$ so that $Mx = Nx + b$ for $x$ satisfying $Ax = b$

$$g(x) = M^{-1}Nx + M^{-1}b$$

$$Mx = Nx + b$$

splitting of $A$

$$(M-N)x = b \Rightarrow \underbrace{(M-N)}_{A}x = Ax$$

# Sparse Iterative Methods

- The *Jacobi method* is the simplest iterative solver:

$$Dx_{k+1} = -(L+U)x_k + b$$

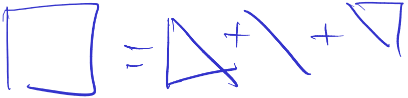
$$x_{k+1} = -D^{-1}(L+U)x_k + b$$

upper tri

$$A = L + D + U$$

strictly lower-tri

diagonal



$$M = D, \quad N = -(L+U)$$

- The Jacobi method converges if $A$ is strictly row-diagonally-dominant:

$$|a_{ii}| > \sum_{j\neq i} |a_{ij}|$$

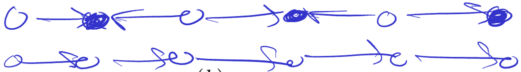
Jacobian of fixed point scheme

$$g(x) = M^{-1}Nx + M^{-1}b$$

$$\rho(M^{-1}N) < 1$$

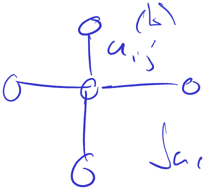
$$D^{-1}(L+U) = \underbrace{M^{-1}N}_{B} \Rightarrow \sum_j B_{ij} = \sum_{j\neq i} a_{ij}/a_{ii} < 1$$

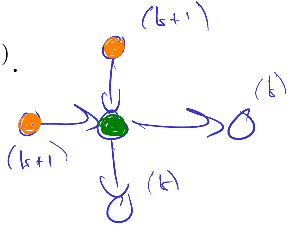
# Gauss-Seidel Method



- The Jacobi method takes weighted sums of $\boldsymbol{x}^{(k)}$ to produce each entry of $\boldsymbol{x}^{(k+1)}$, while Gauss-Seidel uses the latest available values, i.e. to compute $x_i^{(k+1)}$ it uses a weighted sum of

$$\underbrace{x_1^{(k+1)}, \dots x_{i-1}^{(k+1)}}, x_i^{(k)}, \dots, x_n^{(k)}.$$




Jacobi iteration averages neighbors

$$u_{i,j}^{(k+1)} = \frac{u_{i,j-1}^{(k+1)} + u_{i-1,j}^{(k+1)} + u_{i,j+1}^{(k)} + u_{i+1,j}^{(k)}}{4}$$

$M = D + L$

$N = -U$

- Gauss-Seidel provides somewhat better convergence than Jacobi:

ordering of vertices is important for efficiency | $\underbrace{D}_{\text{diagonal}} + \underbrace{L}_{\text{lower-triangular}}) x^{(k+1)} = -Ux^{k} + b$

# Successive Over-Relaxation

- The *successive over-relaxation* (SOR) method seeks to improve the spectral radius achieved by Gauss-Seidel, by choosing

$$M = \frac{1}{\omega}D + L, \quad N = \boxed{\left(\frac{1}{\omega} - 1\right)} D - U$$

$$x^{(k+1)} = \omega x^{(k+1)}_{GS} + (1-\omega)x^{(k)}$$

↑ Gauss-Seidel

- The parameter $\omega$ in SOR controls the 'step-size' of the iterative method:

$\omega$ has effect $\rho(M^{-1}N)$, but choosing the ideal one is hard.

# Conjugate Gradient

- The solution to $\boldsymbol{Ax} = \boldsymbol{b}$ is a minima of the quadratic optimization problem,

$$\min_{\boldsymbol{x}} ||\boldsymbol{Ax} - \boldsymbol{b}||_2^2$$

when $A$ is S.P.D.

objective function quantifies error

- Conjugate gradient works by picking $\boldsymbol{A}$-orthogonal descent directions

  - steepest descent is suboptimal due to reuse of same directions
  - orthogonalize search directions making sure the dirs. are A-orthogonal & cheap

$x_i^T A x_j = \delta_{ij}$

- The convergence rate of CG is linear with coefficient $\boxed{\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}}$ where $\kappa = \text{cond}(\boldsymbol{A})$:

preconditioning
– improve conditioning of A

# Preconditioning

- Preconditioning techniques choose matrix $\boldsymbol{M} \approx \boldsymbol{A}$ and solve the linear system

$$\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{M}^{-1}\boldsymbol{b}$$

if $M = A$, then $x = M^{-1}b$

so seek $M \approx A$, and where $M$ is easy to solve linear systems with

- $\boldsymbol{M}$ is a usually chosen to be an effective approximation to $\boldsymbol{A}$ with a simple structure

diagonal $M$ (Jacobi), lower-triangular $M$

at every iteration, replace $Ax^{(k)}$ with $M^{-1}Ax^{(k)}$

$\kappa(M^{-1}A) < \kappa(A)$