SPARSITY, ITERATIVE METHODS, AND APPLICATIONS

- **Brief overview of sparsity**
- **Basic iterative schemes**
- **Reordering techniques**
- **Applications**

Typical Problem:

Physical Problem

$\downarrow$

Nonlinear PDEs

$\downarrow$

Discretization

$\downarrow$

Linearization (Newton)

$\downarrow$

Sequence of Sparse Linear Systems $Ax = b$

What are sparse matrices?

Usual definition: "..matrices that allow special techniques to take advantage of the large number of zero elements and the structure."

A few applications which lead to sparse matrices : Structural Engineering, Reservoir simulation, Electrical Networks, optimization problems, ...

➤ Matrices can be **structured** or **unstructured**

✍ Explore sparse matrices in Matlab

✍ Show the pattern of matrices `Sherman5` (structured) and `BP1000` (unstructured) from the Harwell-Boeing collection

➤ **Main goal of Sparse Matrix Techniques:** To perform standard matrix computations economically, i.e., without storing the zeros of the matrix.

➤ **Example:** To add two square dense matrices of size n requires $O(n^2)$ operations. To add two sparse matrices A and B requires $O(nnz(A) + nnz(B))$ where $nnz(X) =$ number of nonzero elements of a matrix X.

➤ For typical Finite Element /Finite difference matrices, number of nonzero elements is $O(n)$.

Observation:

A^{-1} is usually dense, but L and U in the LU factorization may be reasonably sparse (if a good technique is used).

Resources

Matrix Market: `http://math.nist.gov/MatrixMarket`

➤ A large set of test matrices from many applications. (Very useful for testing)

➤ "Harwell-Boeing" collection and *many* other test matrices available.

➤ SPARSKIT: A library of FORTRAN subroutines to work with sparse matrices

`http://www.cs.umn.edu/~saad/software/SPARSKIT`

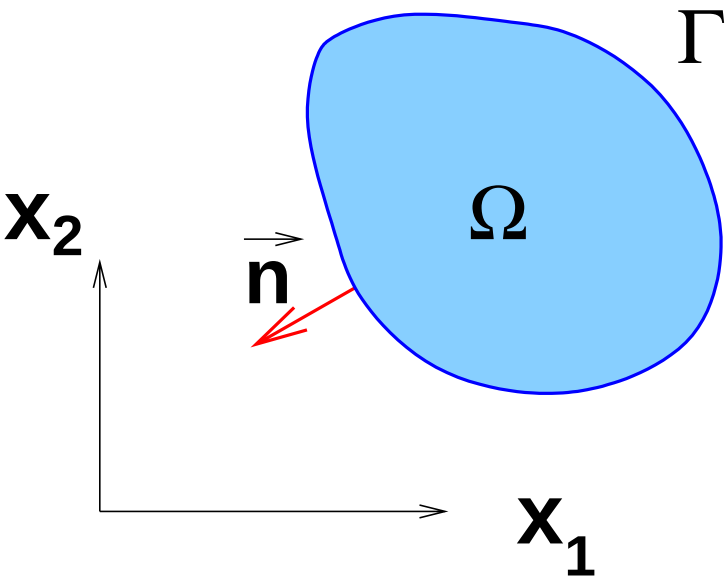
➤ Provides iterative solvers, standard sparse matrix linear algebra routines, etc..

Example: matrices from discretized PDEs

- **Common Partial Differential Equation (PDE) :**

$$\frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} = f, \text{ for } x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \text{ in } \Omega$$

where Ω = bounded, open domain in $\mathbb{R}^2$.



- **+ boundary conditions:**

Dirichlet: $u(x) = \phi(x)$

Neumann: $\frac{\partial u}{\partial \vec{n}}(x) = 0$

Cauchy: $\frac{\partial u}{\partial \vec{n}} + \alpha(x) u = \gamma$

Formulas are derived from Taylor series expansion:

$$u(x+h) = u(x) + h\frac{du}{dx} + \frac{h^2}{2}\frac{d^2u}{dx^2} + \frac{h^3}{6}\frac{d^3u}{dx^3} + \frac{h^4}{24}\frac{d^4u}{dx^4}(\xi_+),$$

- **Simplest scheme: forward difference**

$$\frac{du}{dx} = \frac{u(x+h) - u(x)}{h} - \frac{h}{2}\frac{d^2u(x)}{dx^2} + O(h^2)$$
$$\approx \frac{u(x+h) - u(x)}{h}$$

- **Centered differences for second derivative:**

$$\frac{d^2u(x)}{dx^2} = \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} - \frac{h^2}{12}\frac{d^4u(\xi)}{dx^4},$$

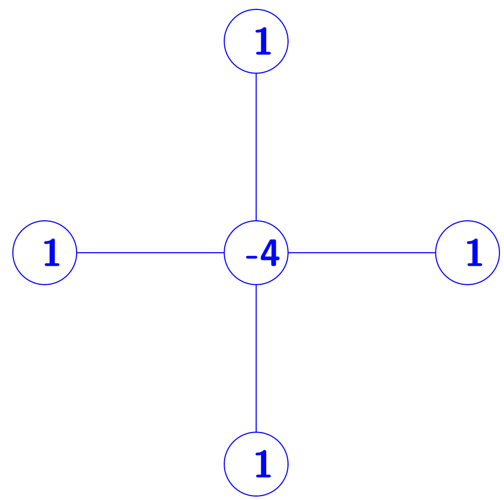
where $\xi_- \le \xi \le \xi_+$.

Difference Schemes for the Laplacian

➤ **Using centered differences for both the $\frac{\partial^2}{\partial x_1^2}$ and $\frac{\partial^2}{\partial x_2^2}$ terms - with mesh sizes $h_1 = h_2 = h$:**

$$\Delta u(x) \approx \frac{1}{h^2}\left[u(x_1+h,x_2)+u(x_1-h,x_2)+ +u(x_1,x_2+h)+u(x_1,x_2-h)-4u(x_1,x_2)\right]$$

The 5-point 'stencil:'



Finite Differences for 2-D Problems

➤ **Consider this simple problem,**

$$-\left(\frac{\partial^2 u}{\partial x_1^2}+\frac{\partial^2 u}{\partial x_2^2}\right)=f \quad \text{in } \Omega \tag{1}$$

$$u = 0 \quad \text{on } \Gamma \tag{2}$$

Ω = **rectangle** $(0, l_1) \times (0, l_2)$ **and** Γ **its boundary.**

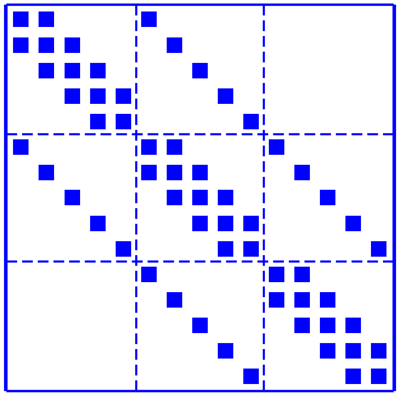
➤ **Discretize uniformly :**

$$x_{1,i} = i \times h_1 \quad i = 0, \ldots, n_1 + 1 \quad h_1 = \frac{l_1}{n_1 + 1}$$

$$x_{2,j} = j \times h_2 \quad j = 0, \ldots, n_2 + 1 \quad h_2 = \frac{l_2}{n_2 + 1}$$

➤ **The resulting matrix has the following block structure:**

$$A = \frac{1}{h^2}\begin{pmatrix} B & -I & \\ -I & B & -I \\ & -I & B \end{pmatrix}$$



Matrix for 7×5 finite difference mesh

With

$$B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & -1 & 4 & -1 \\ & & -1 & 4 \end{pmatrix}$$

Graph Representations of Sparse Matrices

➤ Graph theory is a fundamental tool in sparse matrix techniques.

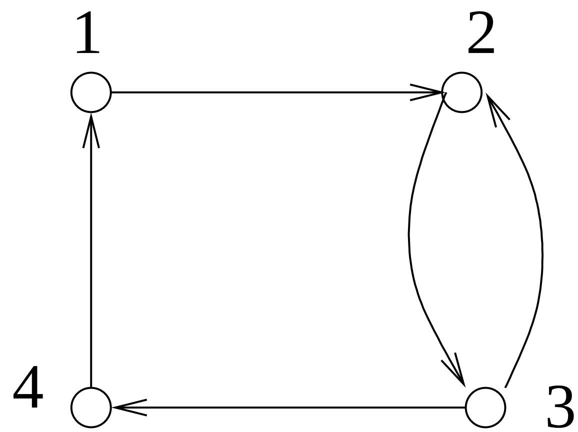
Graph $G = (V, E)$ of an $n \times n$ matrix A defined by

Vertices $V = \{1, 2, ..., N\}$.

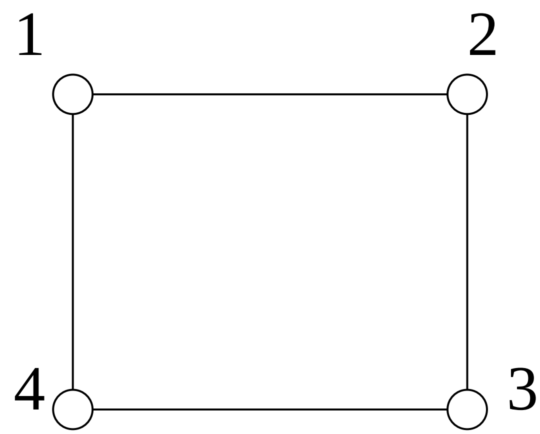
Edges $E = \{(i, j) | a_{ij} \neq 0\}$.

➤ Graph is undirected if matrix has symmetric structure: $a_{ij} \neq 0$ iff $a_{ji} \neq 0$.

x	x		
	x	x	
	x	x	x
x			x



x	x		x
x	x	x	
	x	x	x
x		x	x



Direct versus iterative methods

➤ Direct methods : based on sparse Gaussian eimination

➤ Iterative methods: compute a sequence of iterates which converge to the solution.

Consensus: Direct solvers are often preferred for two-dimensional problems (robust and not too expensive). Direct methods loose ground to iterative techniques for 3-D problems, and problems with many unknowns per grid point.

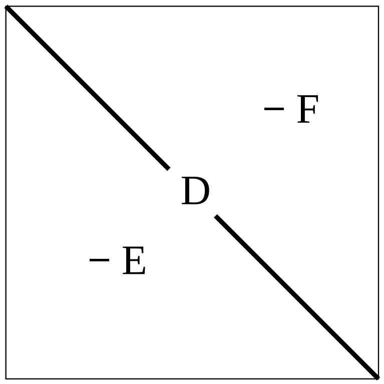
Difficulty:

- No robust 'black-box' iterative solvers.

Iterative methods: Basic relaxation schemes

➤ Relaxation schemes: based on the decomposition

$$A = D - E - F$$



D = diag(A), $-E$ = strict lower part of A and $-F$ its strict upper part.

➤ Simplest method for solving $Ax = b$: Jacobi iteration

$$Dx^{(k+1)} = (E + F)x^{(k)} + b$$

➤ Analyzed using iteration matrix $M_{Jac} = D^{-1}(E + F)$.

- Changes all entries of current approximation to zero out corresponding entries of residual

- Gauss-Seidel: $\xi_i^{new} = \frac{1}{a_{ii}} \left[b_i - \sum_{j<i} a_{ij}\xi_j^{new} - \sum_{j>i} a_{ij}\xi_j \right]$

- Matrix form of Gauss-Seidel:

$$(D - E)x^{(k+1)} = Fx^{(k)} + b$$

Analysed using iteration matrix $M_{GS} = (D - E)^{-1}(F)$.

Can also define a backward Gauss-Seidel Iteration:

$$(D - F)x^{(k+1)} = Ex^{(k)} + b$$

and a Symmetric Gauss-Seidel Iteration: forward sweep followed by backward sweep.

Relaxation: 'relax' Gauss-Seidel iteration:

$$\xi_j^{(k+1)} = \xi_j^{(k)} + \omega(\xi_j^{\mathrm{GS}} - \xi_j^{(k)})$$

- $0 < \omega < 1 \iff$ **Under-relaxation.**
- $\omega = 1 \iff$ **Gauss-Seidel.**
- $1 < \omega < 2 \iff$ **Over-relaxation.**

➤ **Based on the decomposition:**

$$\omega A = (D - \omega E) - (\omega F + (1 - \omega)D)$$

$\to$ **Successive overrelaxation, (SOR, $\omega > 1$):**

$$(D - \omega E)x^{(k+1)} = [\omega F + (1 - \omega)D]x^{(k)} + \omega b$$

Corresponding iteration matrix is:

$$M_{\omega SOR} = (D - \omega E)^{-1}(\omega F + (1 - \omega)D)$$

.

Iteration matrices

➤ **Jacobi, Gauss-Seidel, or SOR, iterations are of the form:**

$$x^{(k+1)} = Mx^{(k)} + f$$

where

- $M_{Jac} = D^{-1}(E + F) = I - D^{-1}A$
- $M_{GS}(A) = (D - E)^{-1}F = I - (D - E)^{-1}A$
- $M_{\omega SOR}(A) = (D - \omega E)^{-1}(\omega F + (1 - \omega)D)$
  $= I - (\omega^{-1}D - E)^{-1}A$

Convergence:

- Jacobi and Gauss-Seidel converge for diagonal dominant matrices
- SOR converges for $0 < \omega < 2$ for SPD matrices
- Optimal ω known for 'consistently ordered matrices' (eig-vals of $\alpha^{-1}D^{-1}E + \alpha D^{-1}F$ indep. of α):

$$\omega_{\text{optimal}} = \frac{2}{1 + \sqrt{1 - \rho(M_{Jac})^2}}.$$

Principle of sparse matrix techniques: Store only the nonzero elements of A. Try to minimize computations and (perhaps more importantly) storage.

- Difficulty in Gaussian elimination: **'fill-in'**

Trivial Example:

- L and U completely full in 1st step of GE

$$A = \begin{pmatrix} + & + & + & + & + & + \\ + & + & & & & \\ + & & + & & & \\ + & & & + & & \\ + & & & & + & \\ + & & & & & + \end{pmatrix}$$

➤ Reorder equations and unknowns in order $n, n-1, ..., 1$:

➤ A stays sparse during Gaussian eliminatin: no fill-in

$$A = \begin{pmatrix} + & & & & & + \\ & + & & & & + \\ & & + & & & + \\ & & & + & & + \\ & & & & + & + \\ + & + & + & + & + & + \end{pmatrix}$$

➤ Finding the best ordering to minimize fill-in is NP-complete but many heuristics were developed. Best known:

- Minimum degree ordering (Tinney Scheme 2)
- Nested Dissection Ordering.

➤ We will come back to reordering methods later if time permits [Also: see course csci8314].