Lecture 05 - Graph Structure of Matrices; Matrix Re-Ordering

June 17, 2025

Outline

Graph Structure of Matrices

- This lecture considers the graph representation of (symmetric) matrices.
- We will discuss the effect of factorization with respect to fill-in and the graph itself.
- Remember that fill-in during factorizations increases storage and flop costs, so lower is better!
- Common matrix reordering methods that reduce fill-in will be discussed in following lectures.

Graph Structure

- Given a square matrix A we can create a directed graph $G(A)$.
- The graph has one node i for each row i in A and an edge connecting $i \to j$ if the matrix entry $a_{ij} \neq 0$.
- If A is symmetric we can take an undirected graph, i.e., edges $i \leftrightarrow j$.
- Definition 1.1 gives a more formal definition of the graph structure from a matrix.

Definition 1.1

Let $A \in \mathbb{R}^{n \times n}$. We associate A with a (directed) graph $G = (V, E)$ that has n nodes: one node per row of A. When $i \neq j$, an edge $(i, j) \in E$ connects nodes i and j iff $a_{ij} \neq 0$. Mathematically,

$$i \in V, \forall 1 \leq i \leq n \text{ and } \forall i \neq j, (i, j) \in E \text{ iff } a_{ij} \neq 0.$$

Graph Structure

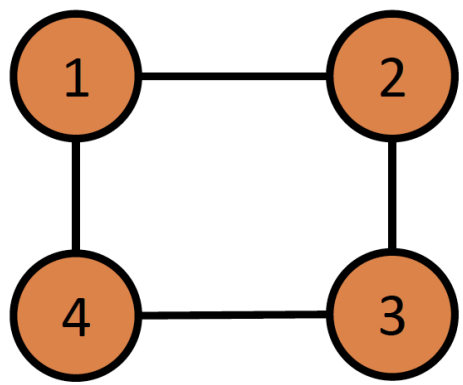
Remarks:

1. This is most useful when A is sparse.
2. Note that we exclude drawing self-cycles (i.e. when $i = j$) in the graph, even though $a_{ii} \neq 0$ is possible.
3. We will consider mostly undirected (symmetric) graphs in this course, unless otherwise noted.

Graph Structure

An example graph from a matrix is give below:

$$A = \begin{bmatrix} \times & \times & & \times \\ \times & \times & \times & \\ & \times & \times & \times \\ \times & & \times & \times \end{bmatrix}$$

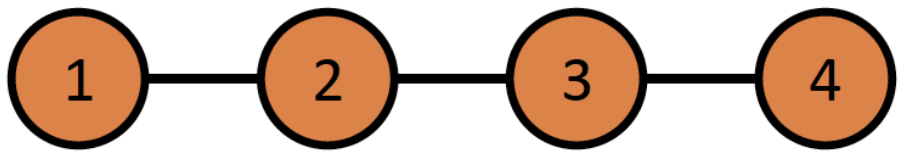
$\rightarrow G(A) =$ 

Graph Structure

1. The graph structure often has a physical/geometric interpretation.
2. For example, the Laplacian matrix from the Poisson equation recovers the underlying grid structure.
3. For the 1D Laplacian matrix, which is tridiagonal, we obtain a graph consists of nodes connected consecutively:

$$A = \begin{bmatrix} \times & \times & & \\ \times & \times & \times & \\ & \times & \times & \times \\ & & \times & \times \end{bmatrix}$$

$$\rightarrow G(A) =$$

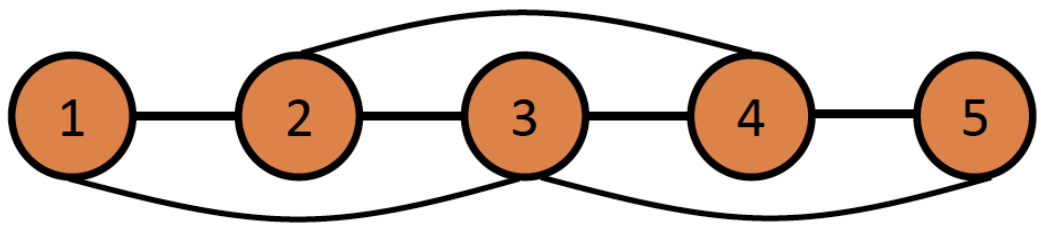


Graph Structure

Another 1D example is shown below.
Exercise: what shape of finite difference "stencil" would produce the graph below?

$$A = \begin{bmatrix} \times & \times & \times & & \\ \times & \times & \times & \times & \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \end{bmatrix}$$

$\rightarrow G(A) =$



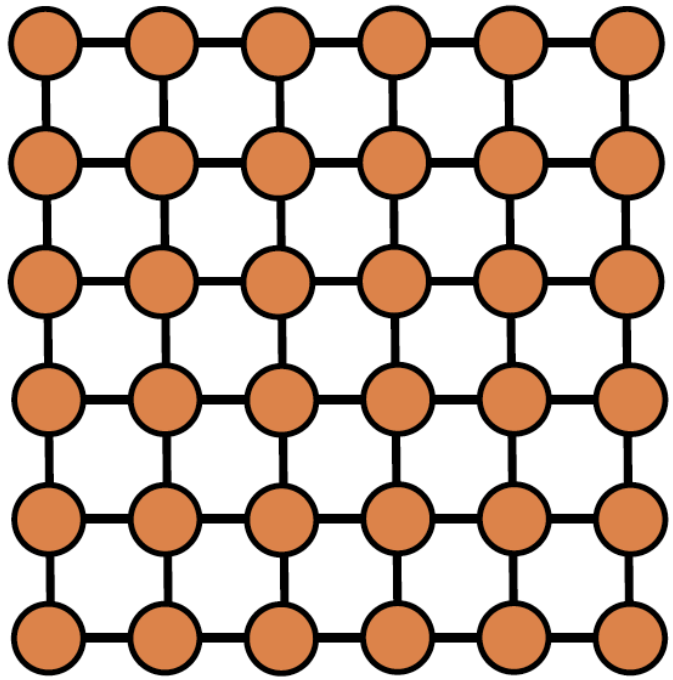
Graph Structure

The graph of the 2D Laplacian matrix also recovers the original grid structure. With

$$A = \begin{bmatrix}
4 & -1 & & & -1 & & & & & & & & & & & \\
-1 & \ddots & \ddots & & & \ddots & & & & & & & & & & \\
 & \ddots & \ddots & -1 & & & \ddots & & & & & & & & & \\
 & & -1 & 4 & & & & -1 & & & & & & & & \\
-1 & & & & 4 & -1 & & & -1 & & & & & & & \\
 & \ddots & & & -1 & \ddots & \ddots & & & \ddots & & & & & & \\
 & & \ddots & & & \ddots & \ddots & -1 & & & \ddots & & & & & \\
 & & & -1 & & -1 & 4 & & & -1 & & & & & & \\
 & & & & -1 & & & 4 & -1 & & & -1 & & & & \\
 & & & & & \ddots & & \ddots & \ddots & & & & \ddots & & & \\
 & & & & & & \ddots & & \ddots & \ddots & -1 & & & \ddots & & \\
 & & & & & & & -1 & & -1 & 4 & & & & -1 & \\
 & & & & & & & & -1 & & & 4 & -1 & & & \\
 & & & & & & & & & \ddots & & \ddots & \ddots & & & \\
 & & & & & & & & & & \ddots & & \ddots & \ddots & & \\
 & & & & & & & & & & & -1 & & -1 & 4
\end{bmatrix},$$

Graph Structure

we have

$G(A) =$ 

Graph Structure

Explanation:

1. By symmetry, we can analyze the rows and know that the columns will behave the same way.
2. There are m^2 rows. Hence the graph has m^2 nodes, i.e. it is a grid with m nodes on each of its rows and columns.
3. For the block of the first m rows, and the block of the last m rows (2 such blocks):
    1. 2 rows with 2 off-diagonal non-zero entries - these are the "corner" nodes of the graph.
    2. $m-2$ rows with 3 off-diagonal non-zero entries - these are "outside, non-corner" nodes of the graph.
4. For each of the inner blocks of m rows ($m-2$ such blocks):
    1. 2 rows with 3 off-diagonal non-zero entries - these are "outside, non-corner" nodes of the graph.
    2. $m-2$ rows with 4 off-diagonal non-zero entries - these are "inside" nodes of the graph.

Graph Structure

5. From here, it is just a matter of mapping the rows of the matrix to the nodes of the graph and checking that the structure is as described.
6. It is an exercise to verify that this numbering of the graph vertices agrees with the structure decribed:

$(m-1)m+1$	$(m-1)m+2$	$\cdots$	$(m-1)m+(m-1)$	m^2
$\vdots$				$\vdots$
$m+1$	$m+2$	$\cdots$	$2m-1$	$2m$
1	2	$\cdots$	$m-1$	m

Graph Structure

Q & A

1. What if the matrix A is not symmetric?
   A: We must write the graph in a directed way.
2. Can we still write the graph directed, even of A is symmetric?
   A: Yes! In this case, each edge has an arrowhead on both ends.

Fill-in During Factorization

Recall that factorization can destroy the nice sparsity pattern of a matrix. For example, consider the LU factorization of this arrow matrix

$$
\begin{aligned}
A &= \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & & & \\ \times & & \times & & \\ \times & & & \times & \\ \times & & & & \times \end{bmatrix} \\
&= \underbrace{\begin{bmatrix} 1 & & & & \\ \times & 1 & & & \\ \times & \times & 1 & & \\ \times & \times & \times & 1 & \\ \times & \times & \times & \times & 1 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix}}_{U}.
\end{aligned}
$$

Fill-in During Factorization

In this section we consider the relationship between factorization of sparse matrices, fill-in, and the graph structure.
Suppose we want to compute the Cholesky factorization of the matrix

$$A = \begin{bmatrix} \times & \times & & \times \\ \times & \times & \times & \\ & \times & \times & \times \\ \times & & \times & \times \end{bmatrix}. \tag{1}$$

Fill-in During Factorization

Fill-in occurs if we compute the Cholesky factorization of A in (1). For Cholesky factorization we have

$$v = \begin{bmatrix} \times \\ 0 \\ \times \end{bmatrix}, \qquad \Rightarrow \quad \frac{vv^T}{\alpha} = \begin{bmatrix} \times & 0 & \times \\ 0 & 0 & 0 \\ \times & 0 & \times \end{bmatrix},$$

$$\Rightarrow \quad B - \frac{vv^T}{\alpha} = \begin{bmatrix} \times & \times & \\ \times & \times & \times \\ & \times & \times \end{bmatrix} - \begin{bmatrix} \times & 0 & \times \\ 0 & 0 & 0 \\ \times & 0 & \times \end{bmatrix} = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}.$$

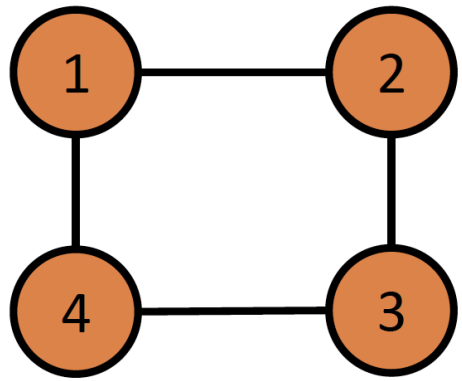
Fill-in During Factorization

- Therefore, non-zero entries are introduced in the same places as with LU factorization. This is because we are deleting the same node $i = (1)$, which causes nodes (2) and (4) to connect with an edge.
- For the 1^{st} iteration, we add a multiple of the 1^{st} row to the rows below it. This reduces $a_{2:n,1}$ to zeros, indicated in green.
- While introducing the zeros in $a_{2:n,1}$ we unfortunately introduce some new nonzeros (indicated in blue, called "fill-in").
- Note that some nonzero entries may also become zero, but we do not take these into account. These new zeros will depend on the actual values in A, which we would like to abstract away for greater generality.

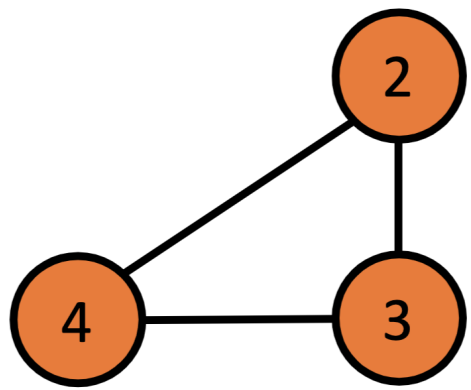
Fill-in During Factorization

Now consider what happens to the graph structure of (1) after one step of Cholesky factorization. The new graph deletes node (1) and connects nodes (2) and (4) together.

$G(A) =$



$G\left(A^{(1)}\right) =$



Fill-in During Factorization

However, nodes (2) and (4) were not connected before, which corresponds to the fill-in. In general, elimination of node i yields a new graph with:

1. Node i and all its edges deleted,
2. New edges $j \leftrightarrow k$ added if there were edges (j, i) and (i, k), i.e., new edges between all node pairs connected to i in the old graph (corresponds to fill-in!).

Convention: Don't display diagonal entries, if they are not connected to anything else. A complete graph would display node #1, with no edges connecting it to anything else.

Fill-in During Factorization

Exercise: what are the graphs of the LU-factorizations of the arrow matrices below? From the graph, can you see why A_1 produces dense LU factors, while A_2 suffers no fill-in?

$$A_1 = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & & & \\ \times & & \times & & \\ \times & & & \times & \\ \times & & & & \times \end{bmatrix}$$

$$A_2 = \begin{bmatrix} \times & & & & \times \\ & \times & & & \times \\ & & \times & & \times \\ & & & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix}$$

Matrix Reordering

Earlier, we saw that general sparse matrices may have **dense LU factors**, e.g.,

$$
\begin{aligned}
A &= \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & & & \\ \times & & \times & & \\ \times & & & \times & \\ \times & & & & \times \end{bmatrix} \\
&= \begin{bmatrix} 1 & & & & \\ \times & 1 & & & \\ \times & \times & 1 & & \\ \times & \times & \times & 1 & \\ \times & \times & \times & \times & 1 \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix} \\
&= LU.
\end{aligned}
$$

Matrix Reordering

We will now start discussing matrix reorderings, which can produce LU factors without fill-in. For example, reordering the same arrow matrix above can give

$$
\begin{aligned}
A &= \begin{bmatrix} \times & & & & \times \\ & \times & & & \times \\ & & \times & & \times \\ & & & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} \\
&= \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ \times & \times & \times & \times & 1 \end{bmatrix} \begin{bmatrix} \times & & & & \times \\ & \times & & & \times \\ & & \times & & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix} \\
&= LU.
\end{aligned}
$$

Key Idea

- We saw earlier that the graph structure of a matrix expresses the underlying relationships among variables.
- The ordering (numbering) of the nodes/variables impacts the matrix layout, but **not** its graph or the solution.
- The graph structure of a symmetric matrix is clearly unchanged by just renumbering its nodes.
- However, different matrices with the same graph can suffer vastly different levels of fill-in during factorization.
- **Goal of matrix reordering:** Renumber the graph nodes to produce a matrix that minimizes fill-in during factorization.

Key Idea

Reordering a matrix can be written mathematically in terms of **permutation matrices**.
A permutation matrix is the identity matrix I with (some) rows/columns swapped.

- Permuting the rows is equivalent to multiplying A by a permutation matrix P on the left: PA. For example,

$$\begin{bmatrix} & 1 & \\ & & 1 \\ 1 & & \end{bmatrix} \begin{bmatrix} 3 & 2 & 5 \\ 2 & 4 & 1 \\ 5 & 1 & 3 \end{bmatrix} = \begin{bmatrix} 2 & 4 & 1 \\ 5 & 1 & 3 \\ 3 & 2 & 5 \end{bmatrix}.$$

  Note that this multiplication is only conceptual. In implementations one never multiplies or stores permutation matrices explicitly.

Key Idea

- Similarly, permuting the columns is equivalent as multiplying by a permutation matrix Q on the right: AQ. For example,

$$\begin{bmatrix} 3 & 2 & 5 \\ 2 & 4 & 1 \\ 5 & 1 & 3 \end{bmatrix} \begin{bmatrix} & 1 & \\ & & 1 \\ 1 & & \end{bmatrix} = \begin{bmatrix} 5 & 3 & 2 \\ 1 & 2 & 4 \\ 3 & 5 & 1 \end{bmatrix}.$$

- Of course, we can permute the rows and the columns simultaneously: PAQ.

Key Idea

A Nice Fact About Any Permutation Matrix, Q: $QQ^T = I$.
The effect of permutation matrices on solving linear systems is as follows. Suppose we are interested in solving the linear system

$$Ax = b.$$

Key Idea

Key Question: How can we correctly keep track of permutations of the rows/columns of A?

- If we permute A to $\tilde{A} = PAQ$, then we need to reorder entries of x and b to match the changes applied to A. Hence

$$\begin{aligned} Ax &= b && (2)\\ A(QQ^T)x &= b && (3)\\ AQ(Q^Tx) &= b && (4)\\ PAQ(Q^Tx) &= Pb && (5)\\ \tilde{A}\tilde{x} &= \tilde{b} && (6) \end{aligned}$$

where $\tilde{x} = Q^Tx$ and $\tilde{b} = Pb$.

- After solving (6) for $\tilde{x}$, we can recover the original solution $x = Q\tilde{x}$.
- We are "unpermuting" $\tilde{x}$ to recover x.

Key Idea

Q & A

1. Why is Q needed?
   A: To permute the columns of A, if needed.
2. Can we have $P = I$?
   A: Yes, if no row permutations are required for A. The setup simplifies considerably in this case.
3. In the 3×3 case, can P swap two rows, leaving the 3^{rd} row untouched?
   A: Yes!
4. Is I a permutation matrix?
   A: Yes! It's the matrix of the identity permutation.