

Sparsity in Linear Least Squares

Graph Theoretic Approaches to Sparse Factorization

Manmohan Krishna Chandraker

CSE 252C, Fall 2004, UCSD

Linear Least Squares Problem

- Find $\mathbf{x} \in \mathbb{R}^n$ that minimizes

$$\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|, \quad \mathbf{A} \in \mathbb{R}^{m \times n}, \quad \mathbf{b} \in \mathbb{R}^m, \quad m \geq n.$$

- Residual vector, $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$.
- Sparse LLS : \mathbf{A} is sparse.

Is Sparsity Useful?

- Electric grids
- Geodetic measurements
- Bundle adjustment

Characterization of LS solutions

- Normal Equations

- \mathbf{x} is a solution to the Least Squares problem if and only if

$$\mathbf{A}^\top \mathbf{A} \mathbf{x} = \mathbf{A}^\top \mathbf{b}$$

- Solution method : Cholesky Decomposition

- QR decomposition

- $\min_{\mathbf{x}} \|\mathbf{A} \mathbf{x} - \mathbf{b}\| = \min_{\mathbf{x}} \|\mathbf{Q}^\top (\mathbf{A} \mathbf{x} - \mathbf{b})\|$ for $\mathbf{Q} \in \mathcal{SO}(m)$.

Time Complexity of Direct Methods

- Structure of A influences choice of algorithm.

$$Ax = b$$

- Dense - Gaussian elimination : $\frac{2}{3}n^3 + \mathcal{O}(n^2)$ flops.
- Symmetric, positive definite - Cholesky decomposition : $\frac{1}{3}n^3 + \mathcal{O}(n^2)$ flops.
- Triangular - Simple substitution : $n^2 + \mathcal{O}(n)$.

What kind of sparsity is useful?

- When there are $\mathcal{O}(n)$ non-zero entries.
 - Sparse data structures include more storage overhead.
 - Arithmetic operations are slower (due to indirect addressing).
- When the sparsity has a pattern.

Sparse Data Structures

Static, compressed row storage

$$\mathbf{A} = \begin{bmatrix} a_{11} & 0 & a_{13} & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 & 0 \\ 0 & 0 & a_{33} & 0 & 0 \\ 0 & a_{42} & 0 & a_{44} & 0 \\ 0 & 0 & 0 & a_{54} & a_{55} \\ 0 & 0 & 0 & 0 & a_{65} \end{bmatrix}$$

$$AC = (a_{11}, a_{13} \mid a_{22}, a_{21} \mid a_{33} \mid a_{42}, a_{44} \mid a_{54}, a_{55} \mid a_{65})$$

$$IA = (1, 3, 5, 6, 8, 10, 11)$$

$$JA = (1, 3, 2, 1, 3, 2, 4, 4, 5, 5)$$

Gaussian Elimination: Fill-in

Fill-in : Non-zero elements created by Gaussian elimination.

$$\mathbf{A} = \mathbf{A}^{(1)} = \begin{bmatrix} a & \mathbf{r}^\top \\ \mathbf{c} & \bar{\mathbf{A}} \end{bmatrix} \text{ where } a \in \mathbb{R}^{1 \times 1}, \bar{\mathbf{A}} \in \mathbb{R}^{(n-1) \times (n-1)}$$

$$\mathbf{A}^{(1)} = \begin{bmatrix} 1 & \mathbf{0} \\ \frac{\mathbf{c}}{a} & \mathbf{I} \end{bmatrix} \begin{bmatrix} a & \mathbf{r}^\top \\ \mathbf{0} & \mathbf{A}^{(2)} \end{bmatrix} \Rightarrow \mathbf{A}^{(2)} = \bar{\mathbf{A}} - \left(\frac{\mathbf{c}\mathbf{r}^\top}{a} \right)$$

Repeat same for $\mathbf{A}^{(2)}$, say, $\mathbf{A}^{(2)} = \mathbf{L}_2\mathbf{U}_2$.

Processing Order and Fill-in

- Column order greatly affects fill-in.

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

- Find the column order that minimizes fill-in :

Processing Order and Fill-in

- Column order greatly affects fill-in.

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

- Find the column order that minimizes fill-in :
NP-complete!!

Steps in a Sparse LS Problem

Symbolic factorization

- Find the structure of $\mathbf{A}^T \mathbf{A}$.
- Determine a column order that reduces fill-in.
- Predict the sparsity structure of the decomposition and allocate storage.

Numerical solution

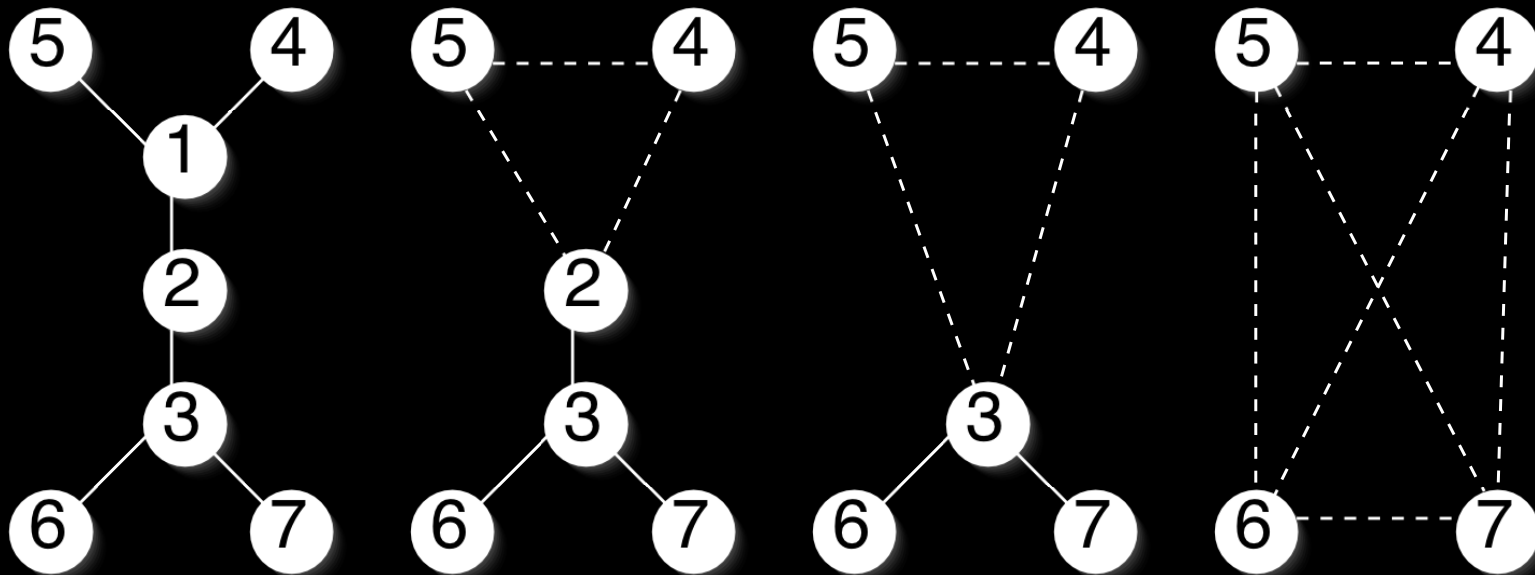
- Read numerical values into the data structure.
- Do a numerical factorization and solve.

Predicting Structure of $\mathbf{A}^\top \mathbf{A}$

- $\mathbf{A}^\top \mathbf{A} = \sum_{i=1}^m \mathbf{a}_i \mathbf{a}_i^\top$ where $\mathbf{a}_i^\top = i$ -th row of \mathbf{A} .
- $G(\mathbf{A}^\top \mathbf{A}) =$ direct sum of $G(\mathbf{a}_i^\top \mathbf{a}_i)$,
 $i = 1, \dots, m$.
- Non-zeros in \mathbf{a}_i^\top form a clique subgraph.

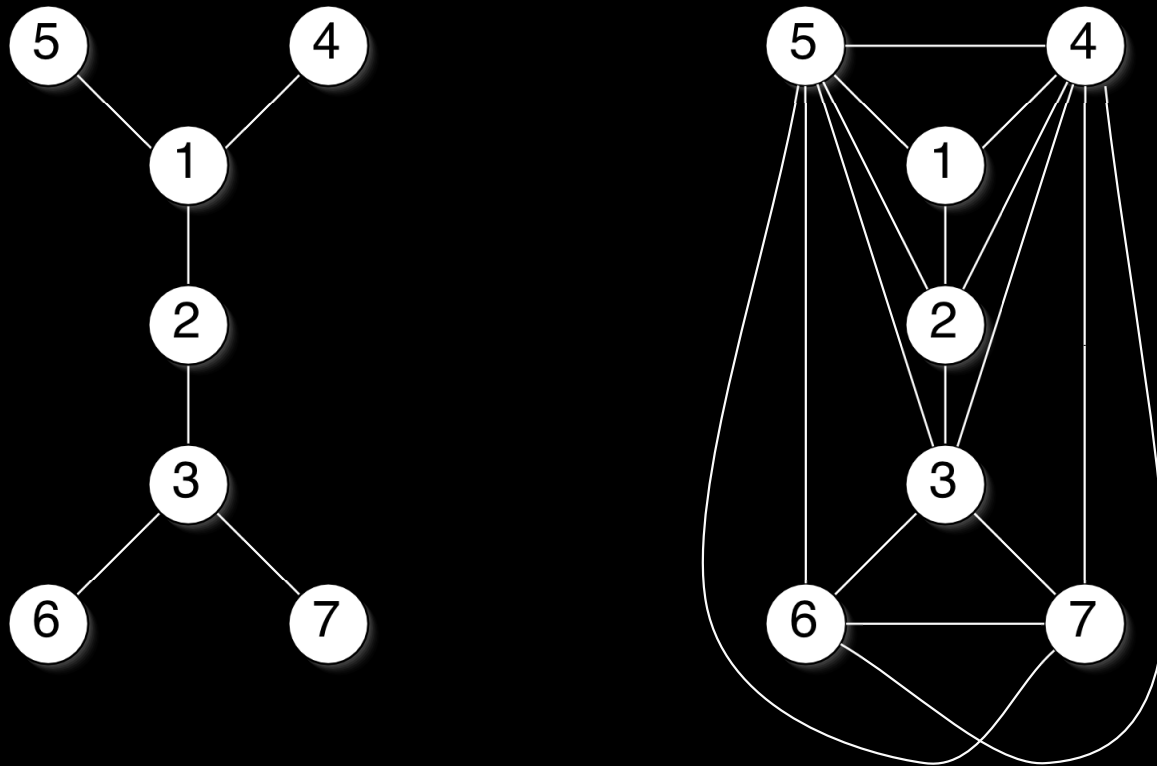
Predicting Structure of Cholesky Factor R

Elimination Graphs : Represent fill-in during factorization.



Filled Graph

- $G_F(\mathbf{A})$: Direct sum of elimination graphs.



Structure of Cholesky Factor

- Filled graph bounds structure of Cholesky factor

$$G(\mathbf{R} + \mathbf{R}^T) \subset G_F(\mathbf{A})$$

- Equality when no-cancellation holds.

\mathbf{A}						\mathbf{R}						
×	×		×	×		×	×		×	×		
×	×	×					×	×	×			
	×	×			×	×			×	×	×	×
×			×						×	×	×	×
×				×					×	×	×	×
		×			×					×	×	×
		×									×	×

Efficient Computation of Filled Graph

- Theorem : *Let $G(\mathbf{A}) = (V, E)$ be an ordered graph of \mathbf{A} . Then (v_i, v_j) is an edge of the filled graph $G_F(\mathbf{A})$ if and only if $(v_i, v_j) \in E$ or there is a path in $G(\mathbf{A})$ from vertex v_i to v_j passing only through vertices with numbering less than $\min\{i, j\}$.*
- Allows construction of filled graph in $\mathcal{O}(n|E|)$ time.

Fill Minimizing Column Orderings

- Reordering rows of A does not affect $A^T A$.
- Column reordering in A keeps number of non-zeros same in $A^T A$.
- Greatly affects number of non-zeros in R .
- Heuristics to reduce fill-in :
 - Minimum degree ordering
 - Nested dissection orderings.

Minimum Degree Ordering

Let $G^{(0)} = G(\mathbf{A})$.

for $i = 1, \dots, (n - 1)$:

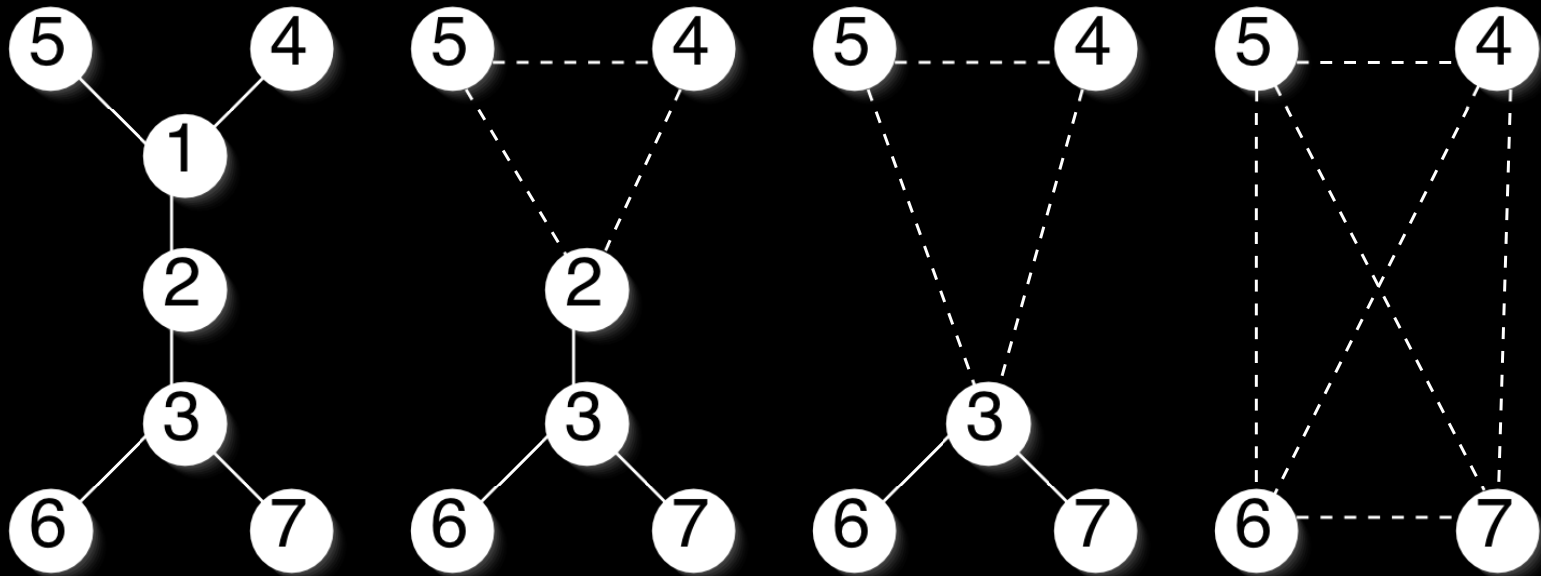
 Select a node v in $G^{(i-1)}$ of minimal degree.

 Choose v as next pivot.

 Update elimination graph to get $G^{(i)}$.

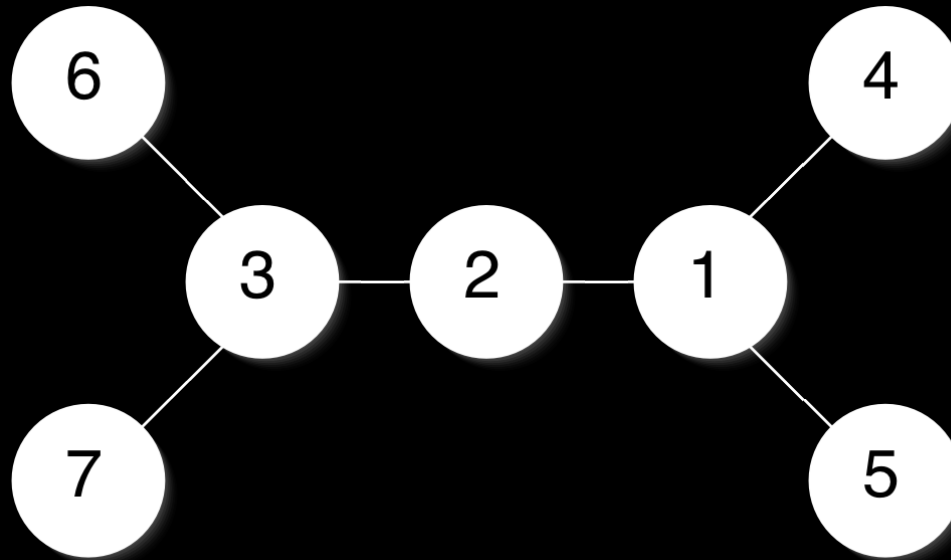
end

Without Ordering



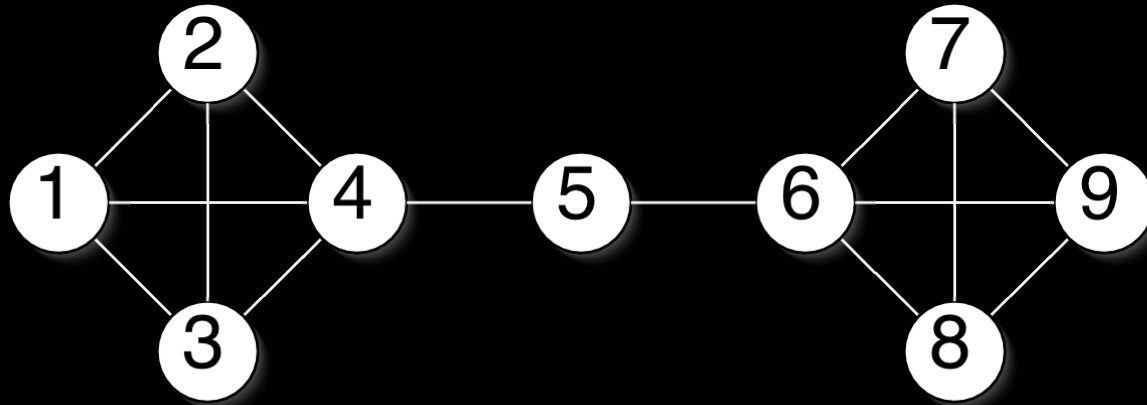
Order 1, 2, 3, 4, 5, 6, 7 \rightarrow fill-in = 10.

With Minimum Degree Ordering



Order 4, 5, 6, 7, 1, 2, 3 \rightarrow fill-in = 0 !!

Does not always work!

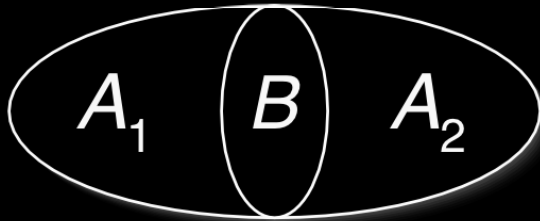


- Order 1, 2, \dots , 9 \rightarrow fill-in = 0.
- Minimum degree node : 5 \rightarrow fills in (4, 6).

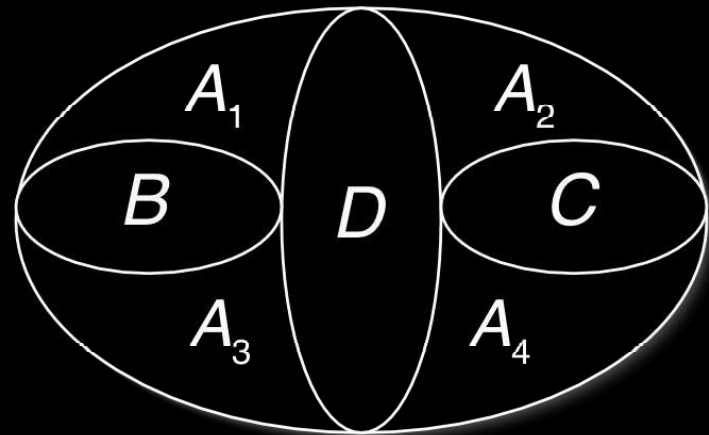
Nested Dissection Ordering

- Reorder to obtain block angular sparse matrix.

Level 1 Dissection



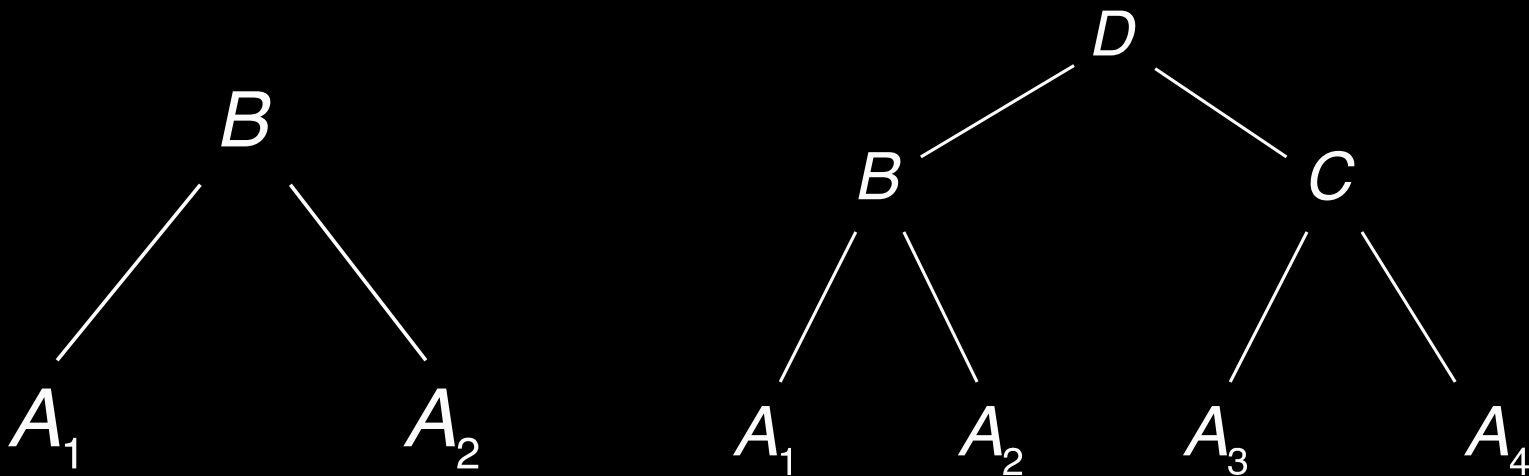
Level 2 Dissection



Nested Dissection: Block Structure and Elimination Tree

$$A = \begin{bmatrix} A_1 & B_1 \\ & A_2 & B_2 \end{bmatrix}$$

$$A = \begin{bmatrix} A_1 & & & B_1 & D_1 \\ & A_2 & & B_2 & D_2 \\ & & A_3 & & C_3 & D_3 \\ & & & A_4 & C_4 & D_4 \end{bmatrix}$$



Numerical Factorization

- Mathematically, Cholesky factor of $A^T A$ same as R in QR-decomposition of A .
- Numerical issues govern choice of algorithm.
- Symbolic factorization same for both.

Numerical Cholesky Factorization

Symbolic Phase

1. Find symbolic structure of $\mathbf{A}^\top \mathbf{A}$.
2. Find column permutation \mathbf{P}_c such that $\mathbf{P}_c^\top \mathbf{A}^\top \mathbf{A} \mathbf{P}_c$ has sparse Cholesky factor \mathbf{R} .
3. Perform symbolic factorization and generate storage structure for \mathbf{R} .

Numerical Phase

1. Compute $\mathbf{B} = \mathbf{P}_c^\top \mathbf{A}^\top \mathbf{A} \mathbf{P}_c$ and $\mathbf{c} = \mathbf{P}_c^\top \mathbf{A}^\top \mathbf{b}$ numerically.
2. Compute \mathbf{R} numerically and solve $\mathbf{R}^\top \mathbf{z} = \mathbf{c}$, $\mathbf{R}\mathbf{y} = \mathbf{z}$, $\mathbf{x} = \mathbf{P}_c \mathbf{y}$.

Cholesky vs QR

- Symbolic computation: No pivoting for needed for Cholesky.
- Loss of information in $A^T A$ and $A^T b$.
- Condition number squared in $A^T A$.
- Inefficient memory utilization: both rows and columns accessed in elimination.

Sparse QR algorithm

Symbolic Phase

1. Determine structure of \mathbf{R} and allocate storage.
2. Determine row permutation \mathbf{P}_r and reorder rows to get $\mathbf{P}_r \mathbf{A} \mathbf{P}_c$.

Numerical Phase

1. Apply orthogonal transformations to $(\mathbf{P}_r \mathbf{A} \mathbf{P}_c, \mathbf{P}_r \mathbf{b})$ to get \mathbf{R} .
2. Solve $\mathbf{R} \mathbf{y} = \mathbf{c}$ and $\mathbf{x} = \mathbf{P}_c \mathbf{y}$.

QR-Decomposition

- Dense problems: Sequence of Householder reflections.
- Sparse problems: Intermediate fill-in.
- Expensive computation and storage.
- Row sequential QR-decomposition.

Row Sequential QR Algorithm

- Row-oriented Givens rotations for orthogonalization avoid intermediate fill-in.

$$\begin{bmatrix} \mathbf{R}_{k-1} \\ \mathbf{a}_k^\top \end{bmatrix} = \begin{bmatrix} \times & 0 & \times & 0 & 0 & \times & 0 & 0 \\ & \otimes & 0 & \oplus & \otimes & 0 & 0 & 0 \\ & & \times & 0 & \times & 0 & 0 & 0 \\ & & & \otimes & \oplus & 0 & \otimes & 0 \\ & & & & \otimes & \oplus & 0 & 0 \\ & & & & & \times & 0 & 0 \\ & & & & & & \otimes & \otimes \\ & & & & & & & \otimes \\ 0 & \otimes & 0 & \otimes & \oplus & 0 & \oplus & \oplus \end{bmatrix}$$

Summary

- Exploiting sparsity: storage and time savings.
- A sparse LS problem can be subdivided into a symbolic phase and a numerical phase.
- The symbolic phase:
 - Determines a column ordering that makes the Cholesky factor sparse.
 - Determines the structure of the Cholesky factor.
- The numerical phase:
 - Uses specialized orthogonalization algorithms to determine the numerical factorization.