# Welcome to Ma221! Lecture 16, Sep 29

bcourses announcement: class projects
related to LAPACK

How do we pick best permutation P
i.e. order of row & columns to do Cholesky
on $PAP^T$? Goal: minimize flops + memory

(backward stable for any P)

Use language of graph theory:
- vertices: rows and columns
- edges: locations of nonzeros
  $(v_1, v_2) \to$ nonzero in $A_{v_1,v_2}$
- weights: values of nonzeros

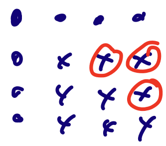
Last Time: RCM: Reverse Cuthill McKee
Breadth First Search: order vertices by distance from starting vertex
$\Rightarrow$ "band" matrix
(block tridiagonal)

Minimum Degree (MD)

Def: degree(v) = # edges touching v
$\Rightarrow$ # nonzeros in row & column v

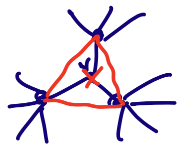
Fact: If I pick v as pivot:
$\Rightarrow$ # flops = $d^2$ muls + add, d = degree(v)

$\Rightarrow$ fill-in (new nonzeros) $\leq d^2$



degree(v) = 3

Greedy Alg: at each step pick v with minimum deg(v)



degree(v) = 3

update graph to show nonzeros of next step

again pick minimum degree(v)

Matlab: amd, symamd, colamd

if $A \neq A^T$, apply to $A + A^T$

ND = Nested Dissection: good if vertices only connected to "nearest neighbors"

$V$ = vertices = $V_1 \cup V_2 \cup V_s$ disjoint

($V_s$ = separator)

① $|V_1| \approx |V_2|$

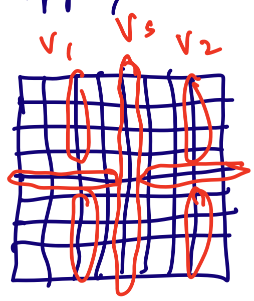
② $|V_s|$ much smaller

③ no edges connecting $V_1$ and $V_2$

Order vertices $V_1$ first, then $V_2$, then $V_s$

$$A = \begin{array}{c} \\ v_1 \\ v_2 \\ v_s \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_s \end{array} \\ \left[\begin{array}{c|c|c} A_{11} & 0 & A_{1s} \\ \hline 0 & A_{22} & A_{2s} \\ \hline A_{s1} & A_{s2} & A_{ss} \end{array}\right] \end{array} \Rightarrow L = \left[\begin{array}{c|c|c} L_{11} & & \\ \hline 0 & L_{22} & \\ \hline L_{s1} & L_{s2} & L_{ss} \end{array}\right]$$

Apply recursively to $A_{11}$ and $A_{22}$




(plots of 7x7 mesh)

Thm (George, Hoffman/Martin/Rose, Gilbert/Tarjan 70s–80s)

Any ordering for Cholesky on $n \times n$ mesh does at least $\Omega(n^3)$ ops

= dense Cholesky on last separator, a dense $n \times n$ matrix, which costs $O(n^3)$

attained by ND

Applies to planar graphs

(draw it on paper with no edge crossings)

(ex: NASA airfoil) (type "load airfoil" in Matlab)

Thm (Ballard, D., Holtz, Schwartz 2009)

#words moved between main mem + cache for LU or Cholesky is

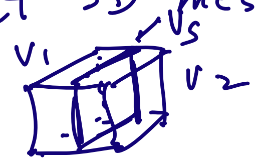
$\Omega(\#\text{flops}/\sqrt{M})$ #flops depends on sparsity pattern

Thm: (David, D., Grigori, Peyronnet, 2010)
attainable by ND, done "carefully"
bottleneck is dense $n \times n$ matrix
for $V_s$

Contrast with BandSolver:
$\#\text{flops} = O(bw^2 \cdot \text{dimension}) = n^4$ for $n \times n$ mesh

(How to pick $V_s$ if not simple mesh:
lots of algs/software, see CS267
METIS, ParMETIS)

What about 3D meshes $n \times n \times n$




ND still good

dimension $= n^3$
dense Chol: costs $O(n^9)$
band Chol: cost $O(n^7)$
ND : costs $O(n^6)$

Steps of sparse Cholesky:
Choose ordering (RCM, MD, ND, ...)
Build data structures for A, L
Perform factorization
Contrast with GE with Partial Pivoting
where data structure dynamic

Ways to deal with Sparse general A

① Threshold pivoting: among pivot choices at each step, pick one with least fill in, within a factor of 2 or 3 of largest (i.e. MD with some stability)

② Static Pivoting: (SuperLU)

(i) reorder and scale A to make diagonal as large as possible

Thm: For any nonsingular A
$\exists$ Perm P and diagonal $D_1$, $D_2$
s.t. $B = D_1 \cdot AP \cdot D_2$ satisfies
$|B(i,i)| = 1$ and $|B(i,j)| \le 1$

$\Rightarrow$ like Cholesky, choose pivots from diagonal

(2) reorder rows and cols of B using same techniques as for Cholesky, Build data structures for B, L, U

(3) During factorization, if a prospective pivot too small, make it bigger (rare) — low rank perturbation of A

$\Rightarrow$ use Sherman-Morrison or use perturbed A as preconditioner in iterative algorithm (GMRES)

Lots of algs, SW... (see "updated survey" by Xiaoye Li on class webpage)