

Lecture 20 Sparse Matrix Algorithms

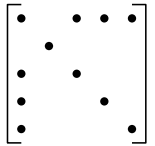
MIT 18.335J / 6.337J
Introduction to Numerical Methods

Per-Olof Persson (persson@mit.edu)
November 26, 2007

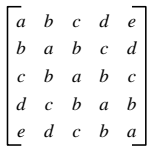
1

Sparse vs. Dense Matrices

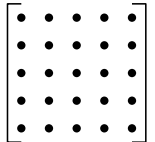
- A *sparse matrix* is a matrix with enough zeros that it is worth taking advantage of them [Wilkinson]



- A *structured matrix* has enough structure that it is worthwhile to use it (e.g. Toeplitz)



- A *dense matrix* is neither sparse nor structured



2

MATLAB Sparse Matrices: Design Principles

- Most operations should give the same results for sparse and full matrices
- Sparse matrices are never created automatically, but once created they propagate
- Performance is important – but usability, simplicity, completeness, and robustness are more important
- Storage for a sparse matrix should be $O(\text{nonzeros})$
- Time for a sparse operation should be close to $O(\text{flops})$

3

Data Structures for Matrices

Full:

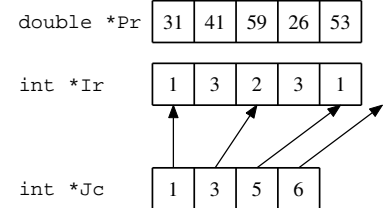
- Storage: Array of real (or complex) numbers
- Memory: `nrows*ncols`

31	0	53
0	59	0
41	26	0

double *A

Sparse:

- Compressed column storage
- Memory: About $1.5 * \text{nnz} + .5 * \text{ncols}$



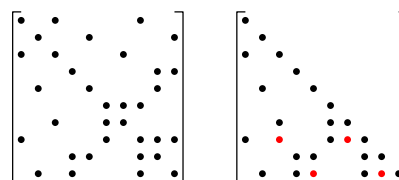
4

Compressed Column Format - Observations

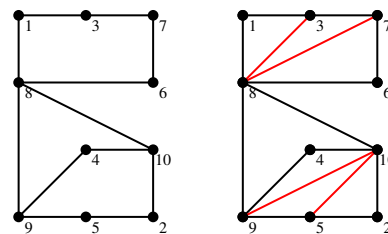
- Element look-up: $O(\log \# \text{elements in column})$ time
- Insertion of new nonzero very expensive
- Sparse vector = Column vector (not Row vector)

5

Graphs and Sparsity: Cholesky Factorization



Fill: New nonzeros in factor



$G(A)$

$G^+(A)$

Symmetric Gaussian
Elimination:

for $j = 1$ to N
Add edges between j 's
higher-numbered neighbors

6

Permutations of the 2-D Model Problem

- 2-D Model Problem: Poisson's Equation on $n \times n$ finite difference grid
- Total number of unknowns $n^2 = N$
- Theoretical results for the fill-in:
 - With natural permutation: $O(N^{3/2})$ fill
 - With any permutation: $\Omega(N \log N)$ fill
 - With a *nested dissection* permutation: $O(N \log N)$ fill

7

Nested Dissection Ordering

- A *separator* in a graph G is a set S of vertices whose removal leaves at least two connected components
- A *nested dissection* ordering for an N -vertex graph G numbers its vertices from 1 to N as follows:
 - Find a separator S , whose removal leaves connected components T_1, T_2, \dots, T_k
 - Number the vertices of S from $N - |S| + 1$ to N
 - Recursively, number the vertices of each component: T_1 from 1 to $|T_1|$, T_2 from $|T_1| + 1$ to $|T_1| + |T_2|$, etc
 - If a component is small enough, number it arbitrarily
- It all boils down to finding good separators!

8

Heuristic Fill-Reducing Matrix Permutations

- Banded orderings (Reverse Cuthill-McKee, Sloan, etc):
 - Try to keep all nonzeros close to the diagonal
 - Theory, practice: Often wins for “long, thin” problems
- Minimum degree:
 - Eliminate row/col with fewest nonzeros, add fill, repeat
 - Hard to implement efficiently – current champion is “Approximate Minimum Degree” [Amestoy, Davis, Duff]
 - Theory: Can be suboptimal even on 2-D model problem
 - Practice: Often wins for medium-sized problems

9

Heuristic Fill-Reducing Matrix Permutations

- Nested dissection:
 - Find a separator, number it *last*, proceed recursively
 - Theory: Approximately optimal separators \implies approximately optimal fill and flop count
 - Practice: Often wins for very large problems
- The best modern general-purpose orderings are ND/MD hybrids

10

Fill-Reducing Permutations in Matlab

- Reverse Cuthill-McKee:
 - `p=symrcm(A)` ;
 - Symmetric permutation: $A(p, p)$ often has smaller bandwidth than A
- Symmetric approximate minimum degree:
 - `p=symamd(A)` ;
 - Symmetric permutation: $\text{chol}(A(p, p))$ sparser than $\text{chol}(A)$
- Nonsymmetric approximate minimum degree:
 - `p=colamd(A)` ;
 - Column permutation: $\text{lu}(A(:, p))$ sparser than $\text{lu}(A)$
- Symmetric nested dissection:
 - Not built into MATLAB, several versions in the MESHPART toolbox

11

Complexity of Direct Methods

- Time and space to solve any problem on any well-shaped finite element mesh with N nodes:

	1-D	2-D	3-D
Space (fill):	$O(N)$	$O(N \log N)$	$O(N^{4/3})$
Time (flops):	$O(N)$	$O(N^{3/2})$	$O(N^2)$

12