Homework 2, CS 6210, Fall 2020
Instructor: Austin R. Benson
Due Friday, October 2, 2020 at 10:19am ET on CMS (before lecture)

Policies

Collaboration. You are encouraged to discuss and collaborate on the homework, but you have to write up your own solutions and write your own code.
Programming language. You can use any programming language for the coding parts of the assignment. Code snippets that we provide and demos in class will use Julia.
Typesetting. Your write-up should be typeset with LaTeX. Handwritten homeworks are not accepted.
Submission. Submit your write-up as a single PDF on CMS: `https://cmsx.cs.cornell.edu`.

Problems

1. *Well-posed but ill-conditioned problems are close to ill-posed problems.*

   Let $A \in \mathbb{R}^{n\times n}$ be nonsingular. Show that the condition number $\kappa_2(A) = \|A\|_2\|A^{-1}\|_2$ is the relative distance to the nearest singular matrix in the 2-norm, i.e.,

$$\frac{1}{\kappa_2(A)} = \min_{\mathrm{rank}(B)<n} \frac{\|A-B\|_2}{\|A\|_2}.$$

2. *Backward stability.*

   Let $x, y \in \mathbb{R}^n$. Should we expect there to exist a backward stable algorithm that computes the outer product $A = xy^T \in \mathbb{R}^{n\times n}$?

3. *Revisiting Sherman–Morrison with Woodbury.*

   Let $V, W \in \mathbb{R}^{n\times k}$ for $k < n$ have full rank and let $A \in \mathbb{R}^{n\times n}$ be nonsingular. For this question, we are interested in solutions to the linear system after a rank-k update:

$$(A + VW^T)x = b.$$

   (a) Note that if $(A + VW^T)x = b$, then

$$\begin{bmatrix} A & V \\ W^T & -I \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}.$$

   Run one step of block Gaussian elimination to factor this matrix into a 2×2 block lower triangular matrix L and a 2×2 block upper triangular matrix U. Some of the block entries of L and U will contain A^{-1}.

   (b) Using forward substitution, derive an expression for $(A + VW^T)^{-1}$ involving A^{-1}, if one exists. Also state when $(A + VW^T)^{-1}$ exists. (Note that the $k = 1$ case is what we studied in Homework 1.)

   (c) Suppose we already computed an LU factorization of A and suppose that $k < n$. Derive an $O(kn^2)$ algorithm for solving $(A + VW^T)x = b$, given L, U, V, W, and b.

4. *Fast banded LU.*

   Let $A \in \mathbb{R}^{n \times n}$ be nonsingular and *banded* with *band width* b, meaning that $A_{ij} = 0$ if $|i - j| > b$. Derive an algorithm to solve $Ax = b$ in $O(nb^2)$ time. *Hint:* Start with Gaussian elimination and consider which entries of L and U become non-zero.

5. *Fill-in with sparse Cholesky.*

   Let $A \in \mathbb{R}^{n \times n}$ be a sparse symmetric matrix, define the matrix

$$S = D - \alpha A \in \mathbb{R}^{n \times n},$$

   where $D \in \mathbb{R}^{n \times n}$ is the *diagonal degree matrix*, $D_{ii} = \sum_j A_{ij}$, and $0 < \alpha < 1$. The matrix S is sparse and strictly diagonally dominant, provided that there is at least one non-zero per row; thus, S is symmetric positive definite as long as there are no zero rows (why?). The matrix S is related to the graph Laplacian matrix, which is used in spectral graph theory and network data analysis, to name a few examples.

   We are interested in finding a permutation matrix P so that the Cholesky factorization

$$PSP^T = LL^T$$

   can be stored sparsely, i.e., L is mostly zero. As discussed in class, if L is sparse, we can more quickly solve $Sx = b$. We will measure the sparsity of L by the relative fill-in:

$$\text{relative fill-in} = 2 \cdot \frac{\text{nnz}(L)}{\text{nnz}(S)}$$

   (the factor two is just to account for the fact that S is symmetric).

   For this problem, we examine a few different ways of constructing the permutation matrix P and the effect on the relative fill-in on two test matrices. Please download the following test matrices from `https://github.com/arbenson/cs6210_2020fa/tree/master/hw2_data`.

   - Minnesota road network. In this matrix, each index corresponds to an intersection and non-zeros entries correspond to roads that connect intersections. In other words, $A_{ij} = 1$ if there is a road connecting intersection i to intersection j and 0 otherwise.
   - Enron emails. In this matrix, each index corresponds to an employee at Enron and non-zero entries correspond to an email communication involving the two employees. (Note: while there is directionality to email communication, we ignore this and just consider the matrix to be symmetric.)

   (a) Implement the algorithm for computing the Cholesky factorization of PSP^T for a given permutation P and S. You do not have to actually maintain a sparse factor L; we will just simulate the fill-in in order to gain a better understanding of an algorithm that reduces fill-in. Also, with Cholesky, we don't have to worry about pivoting — the algorithm is backward stable without pivoting.

   Make sure your algorithm works on the two test matrices. This will require writing some code that reads the text representations of the matrices and creates a sparse matrix.

   (b) For each of the two matrices, permute the rows and columns uniformly at randomly and run the algorithm. Make "spy" plots[1] for PSP^T and L for each matrix. Also report the relative fill-in for each matrix.

[1] `http://docs.juliaplots.org/latest/search/?q=spy`, `https://www.mathworks.com/help/matlab/ref/spy.html`

(c) Implement the *Minimum Degree Ordering Algorithm*[2] to construct P. Again make spy plots for PSP^T and L and report the relative fill-ins.

(d) **Optional, ungraded question that can count towards class participation credit.** Implement the *Reverse Cuthill-McKee Ordering Algorithm* to construct P. Again make spy plots for PSP^T and L and report the relative fill-ins.

[2] https://www.boost.org/doc/libs/1_54_0/libs/graph/doc/sparse_matrix_ordering.html