Department of Mathematics, University of Utah

Analysis of Numerical Methods I

MATH 6610 – Section 001 – Fall 2020

Homework 3

LU **and Cholesky factorizations**

Due Friday, November 6, 2020 by 11:59pm MT

Submission instructions:

Create a private repository on `github.com` named `math6610-homework-3`. Add your LaTeX source files and your Matlab/Python code and push to Github. To submit: grant me (username `akilnarayan`) write access to your repository.

You may grant me write access before you complete the assignment. I will not look at your submission until the due date+time specified above. If you choose this route, I will only grade the assignment associated with the last commit before the due date.

<u>All commits timestamped after the due date+time will be ignored</u>

<u>All work in commits before the final valid timestamped commit will be ignored.</u>

Problem assignment:

Trefethen & Bau III, Lecture 20: # 20.1
Trefethen & Bau III, Lecture 21: # 21.6
Trefethen & Bau III, Lecture 23: # 23.1

Additional problems:

P1. (LU with partial pivoting) Let $A \in \mathbb{C}^{n\times n}$ be invertible. Prove that the LU decomposition algorithm with partial pivoting always successfully computes $PA = LU$.

P2. (Schur complements) Consider the block matrix

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix},$$

where $A \in \mathbb{C}^{m\times m}$, $D \in \mathbb{C}^{n\times n}$, and B and C have appropriate rectangular size. Throughout this problem, we will assume that both A and D are invertible. This problem concerns, among other things, computing the solution vectors x, y. The *Schur complement* of the block A of the matrix M is defined as

$$M/A \coloneqq D - CA^{-1}B,$$

Similarly, $M/D \coloneqq A - BD^{-1}C$ is the Schur complement of D of the matrix M. For this problem, you will be performing block matrix operations; in particular, block matrix multiplication works like matrix multiplication with scalars. Perform the following exercises:

(a) If $B = 0$, prove that $\det M = (\det A)(\det D)$. (It is tempting, but incorrect, to use the familiar 2×2 matrix determinant formula. Instead, perform block LU-type elimination on the C block of M.)

(b) Prove, in general, that $\det M = \det A \det(M/A)$ (Use the procedure as in part a, but with $B \neq 0$, and then utilize part a.)

(c) If $C = B^*$ and both A and D are Hermitian, show that M is (Hermitian) positive definite if and only if both A and M/A are (Hermitian) positive definite. (Perform a symmetric LU, i.e., Cholesky-type, transformation on M similar to part a.)

(d) Given vectors $f \in \mathbb{C}^m$ and $g \in \mathbb{C}^n$, consider the following linear system:

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}, \qquad\qquad x \in \mathbb{C}^m, \;\; y \in \mathbb{C}^n.$$

Give a formula for x and y that utilizes inverses of only A and M/A. (In particular, show that a solution to the system exists if A and M/A are both invertible. Again, perform block LU-type elimination.)

P3. (Sherman-Morrison-Woodbury identity) Consider the matrix M in the previous problem, and assume that A, D, M/A, and M/D are all invertible. Consider the following matrix system:

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} 0_{m\times n} \\ I_{n\times n} \end{pmatrix}, \tag{1}$$

where $X \in \mathbb{C}^{m\times n}$ and $Y \in \mathbb{C}^{n\times n}$. Using this to prove the (Sherman-Morrison-)Woodbury matrix identity:

$$(M/A)^{-1} = D^{-1} + D^{-1}C(M/D)^{-1}BD^{-1}.$$

One way to accomplish this is to solve the 2 x 2 block system above in 2 ways that result in 2 different expressions for the solution Y: first eliminate X and solve for Y, and second eliminate Y and solve for X.

P4. (Column-pivoted QR) Given $A \in \mathbb{C}^{m\times n}$, consider a *column-pivoted* QR decomposition, i.e., a factorization of the form,

$$AP = QR,$$

where P is a permutation matrix that is chosen in the following way: At step j in the orthogonalization process (say step j of Gram-Schmidt), the columns $j, j+1, \ldots, n$ are permuted/pivoted so that r_{jj} will be as large as possible. Note that the vector p defined as

$$p \coloneqq P^T \begin{pmatrix} 1 \\ 2 \\ 3 \\ \vdots \\ n \end{pmatrix} \in \mathbb{R}^n$$

has entries that identify the *column pivots*, i.e., the ordered column indices of A chosen by the pivoting process.

(a) (Column-pivoted QR decompositions are *rank-revealing*, in a sense.) Prove that the number of nonzero diagonal entries in R equals the rank of A.

(b) (Column-pivoted QR is greedy determinant maximization.) Assume $n = \operatorname{rank}(A)$. For S any subset of $\{1, 2, \ldots, n\}$, let A_S denote the $m \times |S|$ submatrix of A formed by selecting the column indices in S. Furthermore, let $G_S \in \mathbb{C}^{|S|\times|S|}$ be defined as

$$G_S = (A_S)^* A_S$$

Set $S_0 = \{\}$, and consider the following iterative, greedy, determinant maximization for $j = 1, \ldots n$:

$$s_j = \operatorname{argmax}_{k\in[n]} \det G_{S_{j-1}\cup\{k\}}, \qquad\qquad S_j \coloneqq S_{j-1} \cup \{s_j\},$$

where $[n] \coloneqq \{1, \ldots, n\}$. Assuming each maximization yields a unique s_j, show that $p_j = s_j$ for $j = 1, \ldots, n$ (where p_j are the QR column pivots).

P5. (Partial LU pivoting is greedy determinant maximization.) Let $A \in \mathbb{C}^{n\times n}$, with $\operatorname{rank}(A) = n$. Show that the LU factorization with partial row pivoting,

$$PA = LU,$$

selects pivots via another kind of greedy determinant maximization. I.e., with S a subset of $[n] = \{1, \ldots, n\}$ as in the previous problem, let ${}_SA$ denote the $|S| \times n$ matrix formed by selecting the rows with indices S from A. Combining notation, ${}_SA_R$, for some $R \subset [n]$ is a $|S| \times |R|$ matrix formed by selecting from A the subblock corresponding to the rows S and columns R.

Again with $S_0 = \{\}$, then consider the optimization problem for $j = 1, \ldots, n$:

$$s_j = \operatorname{argmax}_{k\in[n]} \left|\det \;\; {}_{S_{j-1}\cup\{k\}}A_{[j]}\right|$$

for $j \geq 1$ where again $S_0 = \{\}$. Assuming each maximization yields a unique s_j, then show that s_j, for $j = 1, \ldots, n$, equals the jth entry of the vector p defined by

$$p \coloneqq P \begin{pmatrix} 1 \\ 2 \\ 3 \\ \vdots \\ m \end{pmatrix} \in \mathbb{R}^m$$

Computing assignment:

C1. (Low-rank approximation) In a programming language of your choice, program and test several algorithms for computing low-rank approximations to matrices: Consider $A \in \mathbb{C}^{n \times n}$; we've seen that the 2-norm optimal rank-k approximation to A is a truncated SVD:

$$A_k = \underset{\mathrm{rank}(M) \leq k}{\mathrm{argmin}} \left\| A - M \right\|_2, \qquad\qquad A_k \coloneqq \sum_{j=1}^{k} \sigma_j u_j v_j^*,$$

where $(u_j)_{j=1}^n$ and $(v_j)_{j=1}^n$ are the ordered left- and right-singular vectors of A, respectively, and $(\sigma_j)_{j=1}^n$ are ordered (decreasing) singular values. Consider two other rank-k approximations to A:

- (Column skeletonizations) Let $S =\subset \{1, \ldots, n\}$ be a set of size k given by the first k ordered pivots in a column-pivoted QR decomposition of A. As in previous problems, A_S denotes the $n \times k$ matrix formed by the columns S of A. Then a rank-k column skeletonization B_k of A can be formed by

  $$B_k = P(S)A,$$

  where $P(S) \in \mathbb{C}^{n \times n}$ is the orthogonal projection operator onto range(A_S).

- (Interpolative decompositions) Let $S \subset \{1, \ldots, n\}$ be a set of size k given by the first k ordered pivots in a partial-pivoting LU decomposition of A. As in previous problems, ${}_SA$ denotes the $k \times n$ matrix formed by the rows S of A. Then a rank-k interpolative decomposition C_k of A can be formed by

  $$C_k = A_{[k]} \left({}_S A_{[k]}\right)^{-1} \left({}_S A\right)$$

  where $[k] = \{1, \ldots, k\}$. Note that this is an oblique projection of the columns of A onto range$(A_{[k]})$ defined by enforcing interpolation in each column on elements in rows S.

Report the errors committed by A_k, B_k, and C_k (say in the 2-norm) as a function of k. For test matrices to consider, you may either randomly generate A, or use an A from another application (e.g., from the Yale face database). Why might one prefer to use B_k or C_k as approximations instead of A_k?

(If you're interested, replace C_k above with a *full*-pivoted version, i.e., $C_k = A_R \left({}_S A_R\right)^{-1} \left({}_S A\right)$, where (S, R) are the size-k row and column pivots, respectively from a full-pivoting LU decomposition of A.)

C2. (Eigenvalue algorithms) In a programming language of your choice, program and test several algorithms for computing eigenvalues:

a. Power iteration

b. Rayleigh iteration

c. The (unshifted) QR algorithm

d. The QR algorithm with shifts

For this problem, only consider Hermitian matrices A (for simplicity). Generate the appropriate $A \in \mathbb{C}^{n \times n}$ matrices via randomization. (E.g., set $A \leftarrow A + A^*$ for a random, non-symmetric matrix A; it's ok if you specialize to real-valued matrices.) Compare results from the above algorithms to a baseline, trusted algorithm. (E.g., Matlab's `eig` or Python's `numpy.linalg.eigh`) Plot and evaluate the following metrics:

- Time required as a function of n
- Accuracy as a function of n (e.g., stacking the eigenvalues in a vector, the ℓ^2 or ℓ^∞ norm of the difference between the exact and true vectors)

You may also test accuracy of eigenvectors if you wish, but this is a little more technical since you have to normalize/scale them appropriately.

The purpose of this exercise is to gain familiarity with these algorithms without worrying too much about stability or optimization. E.g., you can use simple (but generally unstable) Hotelling deflation if necessaray at all, you need not reduce A to triangular structure, you may use an unsophisticated choice of shifts (e.g., Rayleigh shifts), etc.

You are encouraged to exercise modularity in your code: build routines that accomplish specific, very particular tasks, and then combine these routines in your iterative schemes.