18.335 Take-Home Midterm Exam: Spring 2023

Posted Friday 12:30pm April 14, due **11:59pm Monday April 17.**

Problem 0: Honor code

Copy and sign the following in your solutions:

I have not used any resources to complete this exam other than my own 18.335 notes, the textbook, running my own Julia code, and posted 18.335 course materials.

____

your signature

Problem 1: (32 points)

Given two real vectors $u = (u_1, u_2, \ldots, u_n)^T$ and $v = (v_1, v_2, \ldots, v_n)^T$, computing the dot product $f(u,v) = u_1v_1 + u_2v_2 + \cdots + u_nv_n = u^Tv$ in floating point arithmetic with left to right summation is backward stable. The computed dot product $\hat{f}(u,v)$ satisfies the *component-wise* backward error criteria

$$\hat{f}(u,v) = (u+\delta u)^T v, \qquad \text{where} \qquad |\delta u| \le n\varepsilon_{\text{mach}}|u| + \mathscr{O}(\varepsilon_{\text{mach}}^2).$$

The notation $|w|$ indicates the vector $|w| = (|w_1|, |w_2|, \ldots, |w_n|)^T$, i.e., the vector obtained by taking the absolute value of each entry of w.

(a) Using the dot product algorithm $\hat{f}(u,v)$, derive an algorithm $\hat{g}(A,b)$ for computing the matrix-vector product $g(A,b) = Ab$ in floating point arithmetic, and show that it satisfies the component-wise backward stability criteria

$$\hat{g}(A,b) = (A+\delta A)b, \qquad \text{where} \qquad |\delta A| \le n\varepsilon_{\text{mach}}|A| + \mathscr{O}(\varepsilon_{\text{mach}}^2),$$

where the notation $|B|$ indicates the matrix obtained by taking the absolute value of each entry of B.

Solution: The i^{th} entry of the matrix-vector product Ab is the dot product of the i^{th} row of A with the vector b. Using the floating-point algorithm $\hat{f}(u,v)$ for each of these dot products results in a computed vector $\hat{g}(A,b)$ whose i^{th} entry is $\hat{f}(A_{i,:},b) = (A_{i,:} + \delta A_i)b$. Denoting the matrix whose i^{th} row is δA_i by δA, we have that $\hat{g}(A,b) = (A+\delta A)b$ as desired. The componentwise bounds on $|\delta A|$ follow immediately from the component-wise backward error bounds for $\hat{f}(A_{:,i},b)$, i.e., the component-wise bounds on the rows δA_i, for $1 \le i \le n$.

(b) Suppose the algorithm $\hat{g}(A,b)$ is used to compute matrix-matrix products $C = AB$ by computing one column of the matrix C at a time. Is the resulting floating-point algorithm $\hat{h}(A,B)$ component-wise backward stable in the sense that there is a matrix δA such that

$$\hat{h}(A,B) = (A+\delta A)B, \qquad \text{where} \qquad |\delta A| \le n\varepsilon_{\text{mach}}|A| + \mathscr{O}(\varepsilon_{\text{mach}}^2)?$$

Explain why or why not. **Solution:** We can apply the matrix-vector product algorithm $\hat{g}(A,b)$ from part (a) to compute one column of $C = AB$ at a time. The columns of the computed matrix $\hat{C} = \hat{h}(A,b)$ then satisfy $\hat{C}_{:,j} = \hat{g}(A, B_{:,j}) = (A+\delta A_j)B_{:,j}$. The problem here is that the j^{th} computed column of $\hat{C}$ is the result of multiplying a column of B by a *different* perturbed matrix $A+\delta A_j$, so it is impossible to express $\hat{C}$ as a product of B with a single perturbed matrix: $\hat{C} \ne (A+\delta A)B$ for some δA. Matrix-matrix multiplication is *not* backward stable. See Higham's book (chapter 3.5) for more discussion and complimentary forward error bounds.

Problem 2: (32 points)

Given an n-dimensional subspace $\mathscr{V}$, the standard Rayleigh–Ritz projection approximates a few ($n \ll m$) eigenvalues of an $m \times m$ matrix A by finding a scalar λ and $x \in \mathscr{V}$ such that $Ax - \lambda x \perp \mathscr{V}$, i.e., the residual is perpendicular to the subspace. A *two-sided* Rayleigh–Ritz projection uses a second subspace $\mathscr{W}$ (not orthogonal to $\mathscr{V}$) and searches for a scalar λ and $x \in \mathscr{V}$ such that

$$Ax - \lambda x \perp \mathscr{W}, \qquad \text{and} \qquad x \in \mathscr{V}, \tag{1}$$

i.e., the residual is perpendicular to the *second* subspace. In this problem, A is diagonalizable.

(a) Let V and W be a pair of bases for $\mathscr{V}$ and $\mathscr{W}$, and let λ (finite) and w solve the eigenvalue problem $Bw = \lambda Mw$, where $B = W^TAV$ and $M = W^TV$. Show that λ and $x = Vw$ satisfy the criteria in (1). **Solution:** Since the columns of V form a basis for $\mathscr{V}$, the vector $x = Vw \in \mathscr{V}$ as it is a linear combination of the columns of V. On the other hand, we have that

$$Bw - \lambda Mw = W^TAVw - \lambda W^TVw = W^T(Ax - \lambda x) = 0,$$

which means that the residual $Ax - \lambda x$ is orthogonal to the columns of W. Since the columns of W form a basis for $\mathscr{W}$, the residual is orthogonal to the whole subspace $\mathscr{W}$, i.e., $Ax - \lambda x \perp \mathscr{W}$.

(b) Suppose that $\mathscr{V} = \text{span}\{x_1, \ldots, x_n\}$ and $\mathscr{W} = \text{span}\{y_1, \ldots, y_n\}$, where $Ax_i = \lambda_i x_i$ and $A^T y_i = \lambda_i y_i$ for $i = 1, \ldots, n$, are a pair of n-dimensional *right and left invariant subspaces* of A. If the bases V and W are chosen to be *bi-orthonormal*, meaning that $W^TV = I$, show that λ and x from part (a) are an eigenpair of the full $m \times m$ matrix A, i.e., that $Ax = \lambda x$. **Solution:** If the bases V and W are biorthonormal, the generalized eigenvalue problem from part (a) becomes the standard eigenvalue problem $Bw = \lambda w$. Following the first hint, we consider the similarity transform

$$D = \begin{pmatrix} W & W_2 \end{pmatrix}^T A \begin{pmatrix} V & V_2 \end{pmatrix} = \begin{pmatrix} W^TAV & W^TAV_2 \\ W_2^TAV & W_2^TAV_2 \end{pmatrix}.$$

First, we can verify that this is indeed a similarity transform because $[W \quad W_2]^T[V \quad V_2] = I$ by the biorthogonality conditions and, therefore, $[W \quad W_2]^T = [V \quad V_2]^{-1}$. Similar matrices have the same eigenvalues, so D and A have the same eigenvalues. Second, notice that the upper left block is the matrix $W^TAV = B$. What about the remaining blocks? By the second hint, $\mathscr{V}$ and $\mathscr{W}$ are orthogonal to $\mathscr{W}_2$ and $\mathscr{V}_2$, respectively. Now, $\mathscr{V}$ and $\mathscr{W}$ are right and left invariant subspaces of A so the columns of AV are vectors in $\mathscr{V}$ and the rows of W^TA are vectors in $\mathscr{W}$. Therefore, the off-diagonal blocks vanish because the columns of AV are orthogonal to the rows of W_2^T and the rows of W^TA are orthogonal to the columns of V_2. The eigenvalues of a block diagonal matrix are the eigenvalues of the diagonal blocks, so the eigenavalues of the upper left block B are eigenvalues of the full matrix D, which are eigenvalues of A by similarity. Thefore, if λ is an eigenvalue of B, it is also an eigenvalue of A. How are the eigenvectors of B related to eigenvectors of A? First, by similarity, the right eigenvectors of A are related to those of D by

$$x_i = \begin{pmatrix} V & V_2 \end{pmatrix} \chi_i, \qquad \text{where} \qquad D\chi_i = \lambda_i \chi_i.$$

Consider the vector $\chi = [w \quad 0]^T$ of length m, and, using that $Bw = \lambda w$, calculate directly that

$$D\chi = \begin{pmatrix} W^TAV & \\ & W_2^TAV_2 \end{pmatrix} \begin{pmatrix} w \\ 0 \end{pmatrix} = \begin{pmatrix} Bw \\ 0 \end{pmatrix} = \lambda \begin{pmatrix} w \\ 0 \end{pmatrix}.$$

So, $\chi = [w \quad 0]^T$ is an eigenvector of D with eigenvalue λ, and therefore, using the connection between eigenvectors of similar matrices from above, we have that

$$\begin{pmatrix} V & V_2 \end{pmatrix} \begin{pmatrix} w \\ 0 \end{pmatrix} = Vw = x$$

is an eigenvector of A with eigenvalue λ. There is an alternative elegant way to prove the statement using orthogonality relations for the residual. From part (a) we know that $Ax - \lambda x \perp \mathscr{W}$ when $x = Vw$ and (λ, w) solves $Bw = \lambda Mw$. If $\mathscr{V}$ is also invariant under A, then we also have that $Ax - \lambda x \in \mathscr{V}$. This implies $Ax - \lambda x \perp \mathscr{W}_2$ because $\mathscr{V}$ and $\mathscr{W}_2$ are orthogonal subspaces. Since A is diagonalizable, $\mathscr{W} \bigcup \mathscr{W}_2 = \mathbb{R}^m$ so the only vector orthogonal to both is the zero vector, which means that $Ax - \lambda x = 0$.

Hint 1: In part (b), consider the similarity transform $[W \quad W_2]^T A [V \quad V_2]$,where V_2 and W_2 are biorthonormal bases for the subspaces $\mathscr{V}_2 = \{x_{n+1}, \ldots, x_m\}$ and $\mathscr{W}_2 = \{y_{n+1}, \ldots, y_m\}$, respectively. **Hint 2:** The right and left eigenvectors of a diagonalizable matrix can be made biorthonormal (why?), so $\mathscr{V}$ and $\mathscr{W}_2$ are orthogonal subspaces.

Problem 3: (36 points)

The method of Generalized Minimal RESiduals (GMRES) uses n iterations of the Arnoldi method to construct a sequence of approximate solutions $x_1, x_2, \ldots, x_n$ to the $m \times m$ linear system $Ax = b$. At the n^{th} iteration, the approximate solution $x_n = Q_n y_n$ is constructed by solving the least-squares problem,

$$y_n = \text{argmin}_y \|\tilde{H}_n y - \|b\| e_1\|,$$

where $\tilde{H}_n$ is an $(n+1) \times n$ upper Hessenberg matrix and Q_n is the usual orthonormal basis for the Krylov subspace $\mathscr{K}_n(A, b) = \text{span}\{b, Ab, A^2 b, \ldots, A^{n-1} b\}$.

(a) Describe an algorithm based on Givens rotations that exploits the upper Hessenberg structure of $\tilde{H}_n$ to solve the $(n+1) \times n$ least-squares problem in $\mathscr{O}(n^2)$ flops. **Solution:** The $(n+1) \times n$ upper Hessenberg matrix $\tilde{H}_n$ has n (potentially) nonzero entries on the subdiagonal. We can compute its QR factorization efficiently by applying Givens rotations to eliminate these subdiagonal entries and triangularize $\tilde{H}_n$. We begin by applying a Givens rotation, G_1, that mixes the first two rows in order to eliminate the $(2,1)$ entry:

$$\tilde{H}_n = \begin{pmatrix} \times & \times & \times & \cdots & \times \\ \times & \times & \times & \cdots & \times \\ & \times & \times & \cdots & \times \\ & & \ddots & \ddots & \vdots \\ & & & \times & \times \\ & & & & \times \end{pmatrix} \qquad \Longrightarrow \qquad G_1 \tilde{H}_n = \begin{pmatrix} \boxtimes & \boxtimes & \boxtimes & \cdots & \boxtimes \\ 0 & \boxtimes & \boxtimes & \cdots & \boxtimes \\ & \times & \times & \cdots & \times \\ & & \ddots & \ddots & \vdots \\ & & & \times & \times \\ & & & & \times \end{pmatrix}.$$

Note that only the first two rows are affected by the first Givens rotation and no new nonzeros appear below the first subdiagonal. Next, we apply a Givens rotation, G_2, that mixes the second two rows in order to eliminate the $(3,2)$ entry:

$$G_1 \tilde{H}_n = \begin{pmatrix} \times & \times & \times & \cdots & \times \\ 0 & \times & \times & \cdots & \times \\ & \times & \times & \cdots & \times \\ & & \ddots & \ddots & \vdots \\ & & & \times & \times \\ & & & & \times \end{pmatrix} \qquad \Longrightarrow \qquad G_2 G_1 \tilde{H}_n = \begin{pmatrix} \times & \times & \times & \cdots & \times \\ 0 & \boxtimes & \boxtimes & \cdots & \boxtimes \\ & 0 & \boxtimes & \cdots & \boxtimes \\ & & \ddots & \ddots & \vdots \\ & & & \times & \times \\ & & & & \times \end{pmatrix}.$$

Note that only the second and third row are affected by the second Givens rotation and there is no fill-in (the introduction of "unwanted" nonzeros) below the diagonal. We continue applying Givens rotations, eliminating the $(k+1, k)$ entry with G_k, which mixes rows k and $k+1$ at the kth step. After

$n-1$ Givens rotations, we apply a final Givens rotation to eliminate the single nonzer entry in the last row of the rectangular Hessenberg matrix $\tilde{H}_n$:

$$G_{n-1}G_1\tilde{H}_n = \begin{pmatrix} \times & \times & \times & \cdots & \times \\ 0 & \times & \times & \cdots & \times \\ & 0 & \times & \cdots & \times \\ & & \ddots & \ddots & \vdots \\ & & & 0 & \times \\ & & & & \times \end{pmatrix} \implies G_2G_1\tilde{H}_n = \begin{pmatrix} \times & \times & \times & \cdots & \times \\ 0 & \times & \times & \cdots & \times \\ & 0 & \times & \cdots & \times \\ & & \ddots & \ddots & \vdots \\ & & & 0 & \boxtimes \\ & & & & 0 \end{pmatrix}.$$

Now, $G_n \ldots G_1\tilde{H}_n = R_n$ is an $(n+1)\times n$ upper triangular matrix, $\Omega_n = G_1^T \ldots G_n^T$ is an $(n+1)\times(n+1)$ orthogonal matrix (usually *not* stored explicitly), and $\tilde{H}_n = \Omega_n R_n$. We can use the QR factorization to solve the least squares problem in the usual way by appying the Givens rotations to the right-hand side, $d = \|b\|\Omega_n^T e_1 = \|b\|G_n \ldots G_1 e_1$, and solving the $n\times n$ triangular system $(R_n)_{1:n,n}y_n = d_{1:n}$ with backsubstitution. The kth step of the QR factorization of $\tilde{H}_n$ requires $\mathcal{O}(n-k)$ flops because rows of length $n-k+1$ are combined by the Givens rotation G_k. After n steps, the total flop count is $\mathcal{O}(n^2)$. Applying the n Givens rotations to e_1 costs $\mathcal{O}(n)$ flops and backsubstitution for the triangular system costs $\mathcal{O}(n^2)$ flops. Therefore, the total cost of computing the least-squares solution is $\mathcal{O}(n^2)$.

(b) If the QR factorization $\tilde{H}_{n-1} = \Omega_{n-1}R_{n-1}$ is known from the previous iteration, explain how to update the QR factorization to $\tilde{H}_n = \Omega_n R_n$ cheaply using a single Givens rotation. **Solution:** If the QR factorization is known at the previous iteration, we can write $\tilde{H}_n$ in the block form

$$\tilde{H}_n = \begin{pmatrix} \Omega_{n-1}R_{n-1} & h_{1:n,n} \\ & h_{n,n+1} \end{pmatrix} = \begin{pmatrix} \Omega_{n-1} & \\ & 1 \end{pmatrix}\begin{pmatrix} R_{n-1} & \Omega_{n-1}^T h_{1:n,n} \\ & h_{n,n+1} \end{pmatrix}.$$

Using the full QR decomposition (as in part (a)), note that R_{n-1} is a $n\times(n-1)$ rectangular upper triangular matrix and Ω_{n-1} is a $n\times n$ orthogonal matrix. Therefore, the first factor is an $(n+1)\times(n+1)$ orthogonal matrix and the first $n-1$ columns of the second factor are already upper triangular. It remains to apply a single additional Givens rotation to the second factor, mixing the last two rows to eliminate the single subdiagonal entry $h_{n,n+1}$. We start with the structure

$$\begin{pmatrix} R_{n-1} & \Omega_{n-1}^T h_{1:n,n} \\ & h_{n,n+1} \end{pmatrix} = \begin{pmatrix} \times & \times & \times & \cdots & \times \\ 0 & \times & \times & \cdots & \times \\ & 0 & \times & \cdots & \times \\ & & \ddots & \ddots & \vdots \\ & & & 0 & \times \\ & & & & \times \end{pmatrix},$$

and end up with the structure

$$G\begin{pmatrix} R_{n-1} & \Omega_{n-1}^T h_{1:n,n} \\ & h_{n,n+1} \end{pmatrix} = \begin{pmatrix} \times & \times & \times & \cdots & \times \\ 0 & \times & \times & \cdots & \times \\ & 0 & \times & \cdots & \times \\ & & \ddots & \ddots & \vdots \\ & & & 0 & \boxtimes \\ & & & & 0 \end{pmatrix}$$

Since Givens rotations are orthogonal matrices, we have that $G^TG = I$, and we can reformulate

$$\tilde{H}_n = \begin{pmatrix} \Omega_{n-1}R_{n-1} & h_{1:n,n} \\ & h_{n,n+1} \end{pmatrix} = \begin{pmatrix} \Omega_{n-1} & \\ & 1 \end{pmatrix} G^TG\begin{pmatrix} R_{n-1} & \Omega_{n-1}^T h_{1:n,n} \\ & h_{n,n+1} \end{pmatrix}.$$

The product of the first two matrices on the right is the orthogonal matrix Ω_n and the product of the second two matrices on the right is the triangular matrix R_n. Note that computing the updated QR factorization means applying the previous Givens rotations to the new column $h_{1:n,n}$, i.e., computing $\Omega_{n-1}^T h_{1:n,n}$, and then computing and applying the new Givens rotation G. The total cost of the update is $\mathcal{O}(n)$ flops.

(c) Using your result from part (b), explain how the solution to the least-squares problem can also be updated cheaply from the solution at the previous iteration. **Solution:** After computing $\tilde{H}_n = \Omega_n R_n$ using the fast update in part (b), we simply solve the triangular system $(R_n)_{1:n,n} y_n = d_{1:n}^{(n)}$, where

$$d^{(n)} = \|b\|\Omega_n^T e_1 = \|b\| G \begin{pmatrix} \Omega_{n-1}^T & \\ & 1 \end{pmatrix} e_1 = G \begin{pmatrix} d^{(n-1)} \\ 0 \end{pmatrix}.$$

In other words, we apply the new Givens rotation G (from the QR update) to update the right-hand side from $d^{(n-1)}$ to $d^{(n)}$ and then solve the new triangular system by backsubstitution as usual.

(d) What is the approximate flop count for updating the least-squares solution at the n^{th} step of GMRES? You may use big-O notation to express the asymptotic scaling in n. **Solution:** In part (a), both the Hessenberg QR factorization and the solution of the triangular system were $\mathcal{O}(n^2)$ flops. Using the fast QR update from part (b), we can reduce the cost of the QR factorization, but the solution of the triangular system remains at $\mathcal{O}(n^2)$ flops. Therefore, updating the least-squares solution at the nth step of of GMRES remains $\mathcal{O}(n^2)$. Note that both the $m \times m$ matrix-vector multiplication and the $\mathcal{O}(mn^2)$ orthogonalization cost of the Arnoldi process are typically much more expensive than the $\mathcal{O}(n^2)$ cost of the least-squares update in GMRES, since $n \ll m$ in almost all practical situations.