18.335 Midterm, Spring 2015

Problem 1: (10+10+10 points)

(a) Suppose you have a forwards-stable algorithm $f$ to compute $f(x) \in \mathbb{R}$ for $x \in \mathbb{R}$, i.e. $\|f(x) - f(x)\| = O(\epsilon_{\text{mach}})$. Suppose $f$ is bounded below and analytic (has a convergent Taylor series) everywhere; suppose $f$ has some global minimum $f_{\min} > 0$ at $x_{\min}$. Suppose that we compute $x_{\min}$ in floating-point arithmetic by exhaustive search: we just evaluate $f$ for all $x \in \mathbb{F}$ and return the $x$ where $f$ is smallest. Is this procedure stable or unstable? Why? (Hint: look at a Taylor series of $f$.)

(b) Consider the function $f(x) = Ax$ where $A \in \mathbb{C}^{m \times n}$ is an $m \times n$ matrix.

(i) In class, we proved that naive summation (by the obvious in-order loop) is stable, and in the book it was similarly proved that the function $g(x) = b^{T}x$ (dot products of $x$ with $b$) is backwards stable for $x \in \mathbb{C}^{n}$ when computed in the obvious loop $g$ (that is: for each $x$ there exists an $\tilde{x}$ such that $g(x) = g(\tilde{x})$ and $\|\tilde{x} - x\| = \|x\| O(\epsilon_{\text{mach}})$). Your friend Simplicio points out that each component $f_{k}$ of $f(x)$ is simply a dot product $f_{k}(x) = a_{k}^{T}x$ (where $a_{k}$ is the $i$-th row of $A$)—so, he argues, since each component of $f$ is backwards stable, $f(x)$ must be backwards stable (when computed by the same obvious dot-product loop for each component). What is wrong with this argument (assuming $m > 1$)?

(ii) Give an example $A$ for which $f(x)$ is definitely not backwards stable for the obvious $f$ algorithm.

Problem 2: (10+10+10 points)

In figure 1 are shown, from class, the classical/modified Gram–Schmidt (CGS/MGS) and Householder algorithms to compute the QR factorization $A = QR$ (reduced: $Q$ is $m \times n$) or $A = QR$ ($Q$ is $m \times m$) respectively of an $m \times n$ matrix $A$. Recall that, using the QR factorization, we can solve the least-squares problem $\min \|Ax - b\|_{2}$ by $\hat{x} = \hat{Q}^{*}b$. Recall that we can compute the right-hand side $\hat{Q}^{*}b$ by forming an augmented $m \times (n + 1)$ matrix $\hat{A} = (A, b)$, finding its QR factorization $\hat{A} = \hat{QR}$ and obtaining $\hat{Q}^{*}b$ from the last column of $\hat{R} = \hat{Q}^{*}\hat{A}$.

Figure 1: Left: Classical/Modified Gram-Schmidt algorithm. Right: Householder QR algorithm. (Figures borrowed from Per Persson’s 18.335 slides.)

Explain whether this procedure is better than computing $\hat{Q}^{*}b$ directly for:

(a) Classical Gram–Schmidt.

(b) Modified Gram–Schmidt.

(c) Householder QR. (Recall that, for Householder QR, we don’t actually compute $Q$ explicitly, but instead store the reflectors $v_{k}$ and re-use them as needed to multiply by $Q$ or $Q^{*}$.)

That is, each of the above three algorithms computes the QR factorization of $A$—for each of the three algorithms is it an improvement to compute $\hat{Q}^{*}b$ via that algorithm on $\hat{A}$ compared with computing $Q$ (or its equivalent) by that algorithm and then performing the $Q^{*}b$ multiplication?

Problem 3: (10+20+10 points)

Suppose $A$ and $B$ are $m \times m$ matrices, $A = A^{*}$, $B = B^{*}$, and $B$ is positive-definite. Consider the “generalized” eigenproblem of finding solutions $x \neq 0$ and $\lambda$ to $Ax = \lambda Bx$, or equivalently solve the ordinary eigenproblem $B^{-1}Ax = \lambda x$. (In general, $B^{-1}A$ is not
Hermitian.) Suppose that there are \( m \) distinct eigenvalues \(|\lambda_1| > |\lambda_2| > \cdots > |\lambda_m|\) and corresponding eigenvectors \( x_1, \ldots, x_m \).

(a) Show that the \( \lambda_k \) are real and that \( x_i^* B x_j = 0 \) for \( i \neq j \). (Hint: multiply both sides of \( Ax = \lambda Bx \) by \( x^* \), similar to the derivation for Hermitian problems in class.)

(b) Explain how to generalize the modified Gram–Schmidt algorithm (figure 1) to compute an “SR” factorization \( B^{-1}A = SR \) where \( S^* BS = I \). (That is, the columns \( s_k \) of \( S \) form a basis for the columns of \( B^{-1}A \) as in QR, but orthogonalized so that \( s_i^* B s_j = 0 \) for \( i \neq j \) and \( = 1 \) for \( i = j \).) Make sure your algorithm still requires \( \Theta(m^3) \) operations!

(c) In exact arithmetic, what would \( S \) in the SR factorization of \((B^{-1}A)^k\) converge to as \( k \to \infty \), and why? (Assume the “generic” case where none of the eigenvectors happen to be orthogonal to the columns of \( B \).)