18.335 Midterm, Fall 2012

Problem 1: (25 points)

(a) Your friend Alyssa P. Hacker claims that the function \( f(x) = \sin x \) can be computed accurately (small forward relative error) near \( x = 0 \), but not near \( x = 2\pi \), despite the fact that the function is periodic in exact arithmetic. True or false? Why?

(b) Matlab provides a function \( \log(1 + x) \) that computes \( \log(1 + x) \). What is the point of providing such a function, as opposed to just letting the user compute \( \log(1 + x) \) herself? (Hint: not performance.) Outline a possible implementation of \( \log(1 + x) \) [rough pseudocode is fine].

(c) Matlab provides a function \( \text{gamma}(x) \) that computes the “Gamma” function \( \Gamma(x) = \int_0^\infty e^{-t}t^{x-1}dt \), which is a generalization of factorials, since \( \Gamma(n+1) = n! \). Matlab also provides a function \( \text{gammaLn}(x) \) that computes \( \ln[\Gamma(x)] \). What is the point of providing a separate \( \text{gammaLn} \) function? (Hint: not performance.)

Problem 2: (5+10+10 points)

Recall that a floating-point implementation \( \tilde{f}(x) \) of a function \( f(x) \) (between two normed vector spaces) is said to be backwards stable if, for every \( x \), there exists some \( \tilde{x} \) such that \( \tilde{f}(x) = f(\tilde{x}) \) for \( \|x - \tilde{x}\| \leq O(\epsilon_{\text{machine}}) \). Consider how you would apply this definition to a function \( f(x, y) \) of two arguments \( x \) and \( y \). Two possibilities are:

- First: The most direct application of the original definition would be to define a single vector space on pairs \( v = (x, y) \) in the obvious way \((x_1, y_1) + (x_2, y_2) = (x_1 + x_2, y_1 + y_2)\) and \( \alpha \cdot (x, y) = (\alpha x, \alpha y) \), with some norm \( \|(x, y)\| \) on pairs. Then \( \tilde{f} \) is backwards stable if for every \( (x, y) \) there exist \( (\tilde{x}, \tilde{y}) \) with \( \tilde{f}(x, y) = f(\tilde{x}, \tilde{y}) \) and \( \|(x, y) - (\tilde{x}, \tilde{y})\| = O(\epsilon_{\text{machine}}) \).

- Second: Alternatively, we could say \( \tilde{f} \) is backwards stable if for every \( x, y \) there exist \( \tilde{x}, \tilde{y} \) with \( \tilde{f}(x, y) = f(\tilde{x}, \tilde{y}) \) and \( \|\tilde{x} - x\| = O(\epsilon_{\text{machine}}) \), and \( \|y - \tilde{y}\| = \|y\|O(\epsilon_{\text{machine}}) \).

(a) Given norms \( \|x\| \) and \( \|y\| \) on \( x \) and \( y \), give an example of a valid norm \( \|(x, y)\| \) on the vector space of pairs \( (x, y) \).

(b) Does First \( \implies \) Second, or Second \( \implies \) First, or both, or neither? Why?

(c) In class, we proved that summation of \( n \) floating-point numbers, in some sequential order, is backwards stable. Suppose we sum \( m + n \) floating point numbers \( x \in \mathbb{R}^m \) and \( y \in \mathbb{R}^n \) by \( \tilde{f}(x, y) = x_1 \oplus x_2 \oplus x_3 \oplus \cdots \oplus x_m \oplus y_1 \oplus y_2 \oplus \cdots \oplus y_n \), doing the floating-point additions (\( \oplus \)) sequentially from left to right. Is this backwards stable in the First sense? In the Second sense? (No complicated proof required, but give a brief justification if true and a counterexample if false.)

Problem 3: (25 points)

Say \( A \) is an \( m \times m \) diagonalizable matrix with eigenvectors \( x_1, x_2, \ldots, x_m \) (normalized to \( \|x_k\| = 1 \) for convenience) and distinct-magnitude eigenvalues \( \lambda_k \) such that \( |\lambda_1| > |\lambda_2| > \cdots > |\lambda_m| \). In class, we showed that \( n \) steps of the QR algorithm produce a matrix \( A_n = Q(n)^n A Q(n) \) where \( Q(n)^n \) is equivalent (in exact arithmetic) to QR factorizing \( A^n = Q(n)^n R(n) \). This proof was general for all \( A \). For the specific case of \( A = A^* \) where the eigenvectors are orthonormal, we concluded that as \( n \to \infty \) we obtain \( Q(n)^n \to \text{eigenvectors} (x_1 \cdots x_m) \) and \( A_n \to \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_m) \).

Show that if \( A \neq A^* \) (so that the eigenvectors \( x_k \) are no longer in generally orthogonal), the QR algorithm approaches \( A_n \to T \) and \( Q(n)^n \to Q \) where \( T = Q' A Q \) is the Schur factorization of \( A \). (Hint: show that \( q_k = Q(n)^n e_k \), the \( k \)-th column of \( Q(n)^n \), is in the span \( \langle x_1, x_2, \ldots, x_k \rangle \) as \( n \to \infty \), by considering \( v_k = A^n e_k \), the \( k \)-th column of \( A^n \). Similar to class, think about the power method \( A^n e_k \), and what Gram-Schmidt does to this.)