18.335 Problem Set 2 Solutions

Problem 1: (14+(10+5) points)

(a) Trefethen, exercise 15.1. In the following, I abbreviate $\epsilon_{\text{machine}} = \epsilon_m$, and I use the fact (which follows trivially from the definition of continuity) that we can replace any Lipshitz-continuous $g(O(\epsilon))$ with $g(0) + g'(0)O(\epsilon)$. I also assume that $\text{fl}(x)$ is deterministic—by a stretch of Trefethen's definitions, it could conceivably be nondeterministic in which case one of the answers changes as noted below, but this seems crazy to me (and doesn't correspond to any real machine). Note also that, at the end of lecture 13, Trefethen points out that the same axioms hold for complex floating-point arithmetic as for real floating-point arithmetic (possibly with ϵ_m increased by a constant factor), so we don't need to do anything special here for $\mathbb{C}$ vs. $\mathbb{R}$.

(i) Backward stable. $x \oplus x = \text{fl}(x) \oplus \text{fl}(x) = [x(1+\epsilon_1) + x(1+\epsilon_1)](1+\epsilon_2) = 2\tilde{x}$ for $|\epsilon_i| \le \epsilon_m$ and $\tilde{x} = x(1+\epsilon_1+\epsilon_2+2\epsilon_1\epsilon_2) = x[1+O(\epsilon_m)]$.

(ii) Backward stable. $x \otimes x = \text{fl}(x) \otimes \text{fl}(x) = [x(1+\epsilon_1) \times x(1+\epsilon_1)](1+\epsilon_2) = \tilde{x}^2$ for $|\epsilon_i| \le \epsilon_m$ and $\tilde{x} = x(1+\epsilon_1)\sqrt{1+\epsilon_2} = x[1+O(\epsilon_m)]$.

(iii) Stable but not backwards stable. $x \oslash x = [\text{fl}(x)/\,\text{fl}(x)](1+\epsilon) = 1+\epsilon$ (not including $x = 0$ or ∞, which give NaN). This is actually forwards stable, but there is no $\tilde{x}$ such that $\tilde{x}/\tilde{x} \ne 1$ so it is not backwards stable. (Under the stronger assumption of correctly rounded arithmetic, this will give exactly 1, however.)

(iv) Backwards stable. $x \ominus x = [\text{fl}(x) - \text{fl}(x)](1+\epsilon) = 0$. This is the correct answer for $\tilde{x} = x$. (In the crazy case where fl is not deterministic, then it might give a nonzero answer, in which case it is unstable.)

(v) Unstable. It is definitely not backwards stable, because there is no data (and hence no way to choose $\tilde{x}$ to match the output). To be stable, it would have to be forwards stable, but it isn't because the errors decrease more slowly than $O(\epsilon_m)$. More explicitly, $1 \oplus \frac{1}{2} \oplus \frac{1}{6} \oplus \cdots$ summed from left to right will give $((1+\frac{1}{2})(1+\epsilon_1)+\frac{1}{6})(1+\epsilon_2)\cdots = e + \frac{3}{2}\epsilon_1 + \frac{10}{6}\epsilon_2 + \cdots$ dropping terms of $O(\epsilon^2)$, where the coefficients of the ϵ_k factors converge to e. The number of terms is n where n satisfies $n! \approx 1/\epsilon_m$, which is a function that grows very slowly with $1/\epsilon_m$, and hence the error from the additions alone is bounded above by $\approx n\epsilon_m$. The key point is that the errors grow at least as fast as $n\epsilon_m$ (not even counting errors from truncation of the series, approximation of $1/k!$, etcetera), which is *not* $O(\epsilon_m)$ because n grows slowly with decreasing ϵ_m.

(vi) Stable. As in (e), it is not backwards stable, so the only thing is to check forwards stability. Again, there will be n terms in the series, where n is a slowly growing function of $1/\epsilon_m$ ($n! \approx 1/\epsilon_m$). However, the summation errors no longer grow as n. From right to left, we are summing $\frac{1}{n!} \oplus \frac{1}{(n-1)!} \oplus \cdots \oplus 1$. But this gives $((\frac{1}{n!} + \frac{1}{(n-1)!})(1+\epsilon_{n-1}) + \frac{1}{(n-2)!})(1+\epsilon_{n-2})\cdots$,and the linear terms in the ϵ_k are then bounded by

$$\left|\sum_{k=1}^{n-1} \epsilon_k \sum_{j=k}^{n} \frac{1}{j!}\right| \le \epsilon_m \sum_{k=1}^{n-1}\sum_{j=k}^{n} \frac{1}{j!} = \epsilon_m \left[\frac{n-1}{n!} + \sum_{j=1}^{n-1} \frac{j}{j!}\right] \approx \epsilon_m e = O(\epsilon_m).$$

The key point is that the coefficients of the ϵ_k coefficients grow smaller and smaller with k, rather than approaching e as for left-to-right summation, and the sum of the coefficients converges. The truncation error is of $O(\epsilon_m)$, and we assume $1/k!$ can also be calculated to within $O(\epsilon_m)$,[1] so the overall error is $O(\epsilon_m)$ and the algorithm is forwards stable.

[1] For example, one could imagine simply having a table of the exactly rounded values of $k!$ for a given precision up to overflow (which occurs at 171! in double precision). One could even imagine some some fancier computational

(vii) Forwards stable. Not backwards stable since no data, but what about forwards stability? Supposing $\sin(x)$ is computed in a stable manner, then $\widetilde{\sin}(x) = \sin(x+\delta)\cdot[1+O(\epsilon_m)]$ for $|\delta| = |x|O(\epsilon_m)$. It follows that, in the vicinity of $x=\pi$, the $\widetilde{\sin}$ function can only change sign within $|\delta| = \pi O(\epsilon_m)$ of $x=\pi$. (Note also that $a\otimes b = (a\times b)(1+\epsilon)$ has the same sign as $a\times b$, so we don't have to worry about the rounding error in $\sin\otimes\sin$.) Hence, checking for $\widetilde{\sin}(x)\otimes\widetilde{\sin}(x')\le 0$, where x' is the floating-point successor to x (`nextfloat(x)` in Julia) yields $\pi[1+O(\epsilon_m)]$, a forwards-stable result.

(b) Trefethen, exercise 16.1. Note that we are free to switch norms as needed, by norm equivalence. *Notation:* the floating-point algorithm for computing $f(A) = QA$ will be denoted $\tilde{f}(A) = \widetilde{QA}$; I will assume that we simply use the obvious three-loop algorithm, i.e. computing the row–column dot products with in-order ("recursive") summation, allowing us to re-use the summation error analysis from pset 1.

(i) We will proceed by induction on k: first, we will prove the base case, that multiplying A by a *single* Q is backwards stable, and then we will do the inductive step (assume it is true for k, prove it for $k+1$).

First, the base case: we need to find a δA with $\|\delta A\| = \|A\|O(\epsilon_{\text{machine}})$ such that $\widetilde{QA} = Q(A+\delta A)$. Since $\|\delta A\| = \|Q^*\widetilde{QA} - A\| = \|Q(Q^*\widetilde{QA}-A)\| = \|\widetilde{QA} - QA\|$ in the L_2 norm, however, this is equivalent to showing $\|\widetilde{QA}-QA\| = \|A\|O(\epsilon_{\text{machine}})$; that is, we can look at the *forwards* error, which is a bit easier. It is sufficient to look at the error in the ij-th element of QA, i.e. the error in computing $\sum_k q_{ik}a_{kj}$. Assuming we do this sum by a straightforward loop, the analysis is exactly the same as in problem 2, except that there is an additional $(1+\epsilon)$ factor in each term for the error in the product $q_{ik}a_{kj}$ [or $(1+2\epsilon)$ if we include the rounding of q_{ik} to $\tilde{q}_{ik} = \text{fl}(q_{ik})$]. Hence, the error in the ij-th element is bounded by $mO(\epsilon_{\text{machine}})\sum_k |q_{ik}a_{kj}|$, and (using the unitarity of Q, which implies that $|q_{ik}|\le 1$) this in turn is bounded by $mO(\epsilon_{\text{machine}})\sum_k |a_{kj}| \le mO(\epsilon_{\text{machine}})\sum_{kj}|a_{kj}| \le mO(\epsilon_{\text{machine}})\|A\|$ (since $\sum_{kj}|a_{kj}|$ is just an L_1 Frobenius norm of A, which is within a constant factor of any other norm). Summing m^2 of these errors in the individual elements of QA, again using norm equivalence, we obtain $\|\widetilde{QA}-QA\| = O(\sum_{ij}|(\widetilde{QA}-QA)_{ij}|) = m^3 O(\epsilon_{\text{machine}})\|A\|$. Thus, we have proved backwards stability for multiplying by one unitary matrix (with a overly pessimistic m^3 coefficient, but that doesn't matter here).

Now, we will show by induction that multiplying by k unitary matrices is backwards stable. Suppose we have proved it for k, and want to prove for $k+1$. That, consider $QQ_k\cdots Q_1 A$. By assumption, $Q_k\cdots Q_1 A$ is backwards stable, and hence $\tilde{B} = \widetilde{Q_k\cdots Q_1 A} = Q_k\cdots Q_1(A+\delta A_k)$ for some $\|\delta A_k\| = O(\epsilon_{\text{machine}})\|A\|$. Also, from above, $\widetilde{Q\tilde{B}} = Q(\tilde{B}+\delta\tilde{B})$ for some $\|\delta\tilde{B}\| = O(\epsilon_{\text{machine}})\|\tilde{B}\| = \|Q_k\cdots Q_1(A+\delta A_k)\|O(\epsilon_{\text{machine}}) = \|A+\delta A_k\|O(\epsilon_{\text{machine}}) \le \|A\|O(\epsilon_{\text{machine}}) + \|\delta A_k\|O(\epsilon_{\text{machine}}) = \|A\|O(\epsilon_{\text{machine}})$. Hence, $\widetilde{QQ_k\cdots Q_1 A} = \widetilde{Q\tilde{B}} = Q[Q_k\cdots Q_1(A+\delta A_k)+\delta\tilde{B}] = QQ_k\cdots Q_1(A+\delta A)$ where $\delta A = \delta A_k + [Q_1^*\cdots Q_k^*]\delta\tilde{B}$ and $\|\delta A\| \le \|\delta A_k\| + \|\delta\tilde{B}\| = O(\epsilon_{\text{machine}})\|A\|$. Q.E.D.

(ii) Consider $f(A) = XA$, where X is some rank-1 matrix xy^* and A has rank >1. The product XA has rank 1 in exact arithmetic, but after floating-point errors it is unlikely that $\tilde{f}(A) = \widetilde{XA}$ will be exactly rank 1. Hence it is not backwards stable, because $X\tilde{A}$ will be rank 1 regardless of $\tilde{A}$, and thus is $\neq \widetilde{XA}$. (See also example 15.2 in the text.)

technique [e.g. in computing the $\Gamma(k+1)$ function at large arguments, which generalizes the factorial function to non-integer k, one often uses Stirling's asymptotic series]. In this particular case, however, since we truncate the series when $1/k! < e\cdot\epsilon_m$, we should never have $k!$ greater than the largest representable integer, which means it can be computed exactly by repeated multiplication. Then we just compute $1\oslash k! = (1+\epsilon)/k!$ where $|\epsilon|<\epsilon_m$.

Problem 2: (10+10 points)

(a) Denote the rows of A by $a_1^T, \ldots, a_m^T$. Consider the unit ball in the L_∞ norm, the set $\{x \in \mathbb{C}^n : \|x\|_\infty \leq 1\}$. Any vector Ax in the image of this set satisfies:

$$\|Ax\|_\infty = \max_{j \in 1:m} |a_j^T x| = \max_{j \in 1:m} \left| \sum_{k \in 1:n} a_{j,k} x_k \right| \leq \max_{j \in 1:m} \sum_{k \in 1:n} |a_{j,k}| = \max_{j \in 1:m} \|a_j\|,$$

since $|x_k| \leq 1$ in the L_∞ unit ball. Furthermore, this bound is achieved when $x_k = \text{sign}(a_{j,k})$ where $j = \text{argmax}_j \|a_j\|$. Hence $\|A\|_\infty = \max_j \|a_j\|$, corresponding to (3.10). Q.E.D.

If we look in the Julia source code, we find that this norm is computed by summing the absolute values of each row of `A` and then takes the maximum, exactly as in (3.10).

(b) To obtain $\mu \times \nu$ submatrix B of the $m \times n$ matrix A by selecting a subset of the rows and columns of A, we simply multiply A on the left and right by $\mu \times m$ and $n \times \nu$ matrices as follows:

$$B = \begin{pmatrix} 1 & & & & \\ & & & 1 & \\ & & & & \ddots \end{pmatrix} A \begin{pmatrix} 1 & & \\ & & \\ & 1 & \\ & & \\ & & \ddots \\ & & \end{pmatrix}$$

where there are 1's in the columns/rows to be selected. More precisely, if we want a subset $\mathcal{R}$ of the rows of A and a subset $\mathcal{C}$ of the columns of A, then we compute $B = D_\mathcal{R} A D_\mathcal{C}^T$, where the "deletion matrix" for an ordered set $\mathcal{S}$ of indices is given by $(D_\mathcal{S})_{ij} = 1$ if j equals the i-th element of $\mathcal{S}$ and $(D_\mathcal{S})_{ij} = 0$ otherwise; $D_\mathcal{R}$ is $\mu \times m$ and $D_\mathcal{C}$ is $\nu \times n$.

From Trefethen, chapter 3, we have $\|B\|_p \leq \|A\|_p \|D_\mathcal{R}\|_p \|D_\mathcal{C}\|_p$. So, we merely need to show $\|D_\mathcal{S}\|_p \leq 1$ and the result follows. But this is trivial: $\|D_\mathcal{S} x\|_p = \left[\sum_{i \in \mathcal{S}} |x_i|^p\right]^{1/p} \leq \left[\sum_i |x_i|^p\right]^{1/p} = \|x\|_p$, so $\|D_\mathcal{S}\|_p \leq 1$ and we obtain $\|B\|_p \leq \|A\|_p$.

In Julia, we construct a random 10×7 A by `A=randn(10,7)`, and an arbitrary 3×4 subset of this matrix by `B = A[[1,3,4],[2,3,5,6]]`. Then `norm(B) <= norm(A)` (the $p = 2$ norm) returns `true`. As a more careful test, we can also try computing thousands of such random matrices and check that the maximum of `norm(B)/norm(A)` is < 1; a one-liner to do this in Julia is `maximum(Float64[let A=randn(10,7); norm(A[1:3,1:4])/norm(A); end for i=1:10000])`, which returns roughly 0.92. However, a quick check with a single matrix is acceptable here—such numerical "spot checks" are extremely useful to catch gross errors, but of course they aren't a substitute for proof, only a supplement (or sometimes a suggestive guide, if the numerical results precede the proof).

Problem 3: (10+10+10 points)

(a) Trefethen, probem 4.5. Consider first the columns of V, which are also eigenvectors of A^*A — the nonzero eigenvalues correspond to the columns of $\hat{V}$, and the zero eigenvalues correspond to the remaining columns of V which span $N(A^*A) = N(A)$. So we just need to prove that the eigenvectors v of a real-symmetric matrix $B = A^*A$ can be chosen real. Say $Bv = \lambda v$. Then the real and imaginary parts of v are themselves eigenvectors (if they are nonzero) with eigenvalue λ (proof: take the real and imaginary parts of $Bv = \lambda v$, since B and λ are real). Hence, taking either the real or imaginary parts of an eigenvector v (whichever is nonzero) and normalizing it to unit length, we obtain a new purely real eigenvector. (There is a slight wrinkle for eigenvalues, e.g. $\lambda_1 = \lambda_2$, because the real or imaginary parts of v_1 and v_2 might not be orthogonal. However, taken together, the real and imaginary parts of any multiple eigenvalues must span the same space, and hence we can find a real orthonormal basis with Gram-Schmidt or whatever.) It is now simple to show that U can also be chosen real. For singular values $\sigma_i > 0$, corresponding to columns of $\hat{U}$, we saw in class that $\hat{u}_i = A\hat{v}_i/\sigma_i$, so the columns of $\hat{U}$ are real if V is real. The remaining columns of U are eigenvectors of the real-symmetric matrix AA^* with eigenvalue zero, which can also be chosen real as for V. Q.E.D.

(b) Trefethen, problem 5.2. We just need to show that, for any $A \in \mathbb{C}^{m\times n}$ with rank $< n$ and for any $\epsilon > 0$, we can find *any* sequence of full-rank matrices B that eventually satisfies $\|A - B\|_2 < \epsilon$, i.e. becomes arbitrarily close to A. Form the SVD $A = U\Sigma V^*$ with singular values $\sigma_1, \ldots, \sigma_r$ where $r < n$ is the rank of A. Choose $B = U\tilde{\Sigma}V^*$ where $\tilde{\Sigma}$ is the same as Σ except that it has $n - r$ additional nonzero singular values $\sigma_{k>r} = \epsilon/2$. From equation 5.4 in the book, $\|B - A\|_2 = \sigma_{r+1} = \epsilon/2 < \epsilon$, noting that $A = B_r$ in the notation of the book. We can then make a sequence of such matrices e.g. by letting $\epsilon_k = \sigma_r 2^{-k}$ for $k = 1, 2, \ldots$, such that $\|B_k - A\|_2 = \epsilon_k \to 0$ as $k \to \infty$.

(c) Trefethen, problem 5.4. From $A = U\Sigma V^*$, recall that $AV = U\Sigma$ and $A^*U = V\Sigma$. Therefore,

$$\begin{pmatrix} & A^* \\ A & \end{pmatrix}\begin{pmatrix} V \\ \pm U \end{pmatrix} = \begin{pmatrix} \pm A^*U \\ AV \end{pmatrix} = \pm\begin{pmatrix} V\Sigma \\ \pm U\Sigma \end{pmatrix} = \pm\begin{pmatrix} V \\ \pm U \end{pmatrix}\Sigma$$

and hence $(v_i; \pm u_i)$ is an eigenvector of $\begin{pmatrix} & A^* \\ A & \end{pmatrix}$ with eigenvalue $\pm\sigma_i$. Noting that these vectors $(v_i; \pm u_i)$ are orthogonal by construction and only need to be divided by $\sqrt{2}$ to be normalized, we immediately obtain the diagonalization

$$\begin{pmatrix} & A^* \\ A & \end{pmatrix} = Q\begin{pmatrix} +\Sigma & \\ & -\Sigma \end{pmatrix}Q^*$$

for

$$Q = \begin{pmatrix} V & V \\ +U & -U \end{pmatrix}/\sqrt{2}.$$

Problem 4: (10+10 points)

Trefethen, problem 11.2. See the pset 2 solution notebook for Julia code and accompanying explanations.