18.335 Midterm Exam Solutions: Spring 2019

Problem 1: (10 points)

The general formula for $\kappa(A)$, from the book, is the supremum of the condition number $\|A\| \cdot \|x\| / \|Ax\|$ for all x, i.e.

$$\kappa(A) = \|A\| \left(\sup_{x \neq 0} \frac{\|x\|}{\|Ax\|} \right) = \left(\sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} \right) \left(\sup_{x \neq 0} \frac{\|x\|}{\|Ax\|} \right).$$

Since A (n columns) is a subset of the columns of B($n' \geq n$ columns), then for every $x \in \mathbb{C}^n$ there is an $x' \in \mathbb{C}^{n'}$ such that $Ax = Bx'$ — that is, x' is simply x padded with zeros for the extra columns of B. Furthermore, in any of our L_p norms we have $\|x\| = \|x'\|$. So, if x_* is a vector where $\frac{\|Ax\|}{\|x\|}$ achieves its supremum, there is an x' such that $\frac{\|Ax_*\|}{\|x_*\|} = \frac{\|Bx'\|}{\|x'\|}$, and hence

$$\sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} \leq \sup_{x' \neq 0} \frac{\|Bx'\|}{\|x'\|}.$$

Similarly for $\frac{\|x\|}{\|Ax\|}$. Hence $\kappa(A) \leq \kappa(B)$.

Problem 2: (5+5 points)

For a diagonalizable $m \times m$ matrix $A = X \Lambda X^{-1}$, the matrix square root is

$$A^{\frac{1}{2}} = X \Lambda^{\frac{1}{2}} X^{-1} = X \begin{pmatrix} \sqrt{\lambda_1} & & & \\ & \sqrt{\lambda_2} & & \\ & & \ddots & \\ & & & \sqrt{\lambda_m} \end{pmatrix} X^{-1}.$$

(a) A may be nearly defective, in which case X is badly conditioned and multiplying by X or X^{-1} will be inaccurate. (Being exactly defective is exceedingly rare — a set of measure zero among all matrices, so you might ignore this case, but you can't ignore the possibility of being nearly defective.)

(b) One possible answer is that all her matrices are Hermitian (or anti-Hermitian/skew-Hermitian).

For the $X \Lambda^{\frac{1}{2}} X^{-1}$ formula to be accurate, you need X to be well-conditioned, and the best case for this is if A is normal ($AA^* = A^*A$), in which case X can be chosen unitary (condition number 1). The only cases where you can typically see that A is normal by inspection are the Hermitian or anti-Hermitian cases. (Another possibility would be diagonal matrices A, but you were told that the matrices were non-sparse.)

Problem 3: (10 points)

If one of the x_i values is sufficiently large and positive ($\gtrsim 710$ in double precision), then e^{x_i} will overflow and you will get +Inf. Alternatively, if *all* of the x_i values are sufficiently large in magnitude and negative ($\lesssim -745$ in double precision), then e^{x_i} will underflow to $+0.0$ and the log will give you $-$Inf. To start with, we want to avoid both of these cases.

8/10 points: A simple solution is to compute $X = \max_i x_i$, and then use the identity

$$f(x) = \log \left(\sum_{i=1}^{n} e^{x_i} \right) = \log \left(e^X \sum_{i=1}^{n} e^{x_i - X} \right) = X + \log \left(\sum_{i=1}^{n} e^{x_i - X} \right).$$

This solves the overflow problem, because $x_i - X \le 0$ and hence $e^{x_i - X}$ can only be small, not large. What about underflow? Without loss of generality, let's suppose that $X = x_1$. Then we have

$$f(x) = X + \log\left(1 + \sum_{i=2}^{n} e^{x_i - X}\right).$$

Notice that $e^{x_i - X}$ in the sum may underflow to zero, but we will never get zero as the argument of the log because we have $1 + \cdots \ge 1$. So we won't get $-\mathrm{Inf}$ even if the x_i are large negative numbers.

10/10 points: However, there is still a subtle problem: if $\sum_{i=2}^{n} e^{x_i - X} \ll 1$, then in floating-point arithmetic we may get

$$X + \log\left(1 \oplus \sum_{i=2}^{n} e^{x_i - X}\right) = X + \log(1) = X,$$

so the contribution of the $\sum_{i=2}^{n} e^{x_i - X}$ is lost. Recall the Taylor expansion

$$\log(1+y) = y - \frac{y^2}{2} + \frac{y^3}{3} - \cdots,$$

so even if $0 < y \ll 1$, we are not supposed to get zero from the log. This can lead to an inaccurate result. For example, consider the case of $n = 2$ with $x_1 = 10^{-20} > x_2 = \log 10^{-20} \approx -46.0517$. Then the correct answer is

$$x_1 + \log(1 + e^{x_2}) = 10^{-20} + \log(1 + 10^{-20}) \approx 2 \times 10^{-20}.$$

but in floating-point arithmetic we will get $x_1 \oplus \log(1 \oplus e^{x_2}) = x_1 \oplus \log(1) = x_1 \approx 10^{-20}$, which is off by a factor of 2! The solution is that we need to compute $\mathrm{log1p}(y) = \log(1+y)$ accurately even for very small y, and fortunately most math libraries (including Julia's) provide a built-in "`log1p`" function that does just that. So, in summary, if we want an accurate result we really need to use a floating-point version of the expression:

$$f(x) = X + \mathrm{log1p}\left(\sum\nolimits' e^{x_i - X}\right),$$

where $\sum'$ denotes the sum omitting a single term with $x_i = X = \max_j x_j$. If we want, we could implement this sum with pairwise summation or similar, for even more accuracy. If we didn't have a "`log1p`" function available, to accurately compute $\mathrm{log1p}(y) = \log(1+y)$, we could implement it ourselves using the Taylor series when $|y|$ is sufficiently small (although it turns out that there are more clever ways to do it).

Problem 4: (10 points)

8/10: We can use the Hessenberg factorization $A = QHQ^*$, which can be computed in $\Theta(m^3)$ operations from class, and for which H is tridiagonal if A is Hermitian. Then

$$f(z) = \det(A - zI) = \det(QHQ^* - zI) = \det\left[Q(H - zI)Q^*\right] = \det(Q)\det(H - zI)\det(Q^*) = \det(H - zI)$$

by elementary properties of determinants. Since $H - zI$ is tridiagonal, as mentioned in class we can find its LU factorization in $\Theta(m)$ operations, from which the determinant is simply the product of the diagonal entries of U. A little care is needed for the case where $H - zI$ is nearly singular, though.

10/10: Since in neither the book nor in class did we explicitly study the LU decomposition of tridiagonal matrices — I only stated in passing that it was $\Theta(m)$ — and some care is needed in the singular case, to get full marks on this problem you need to do a bit more work to convince me of how you would compute $\det H$. In particular, there are lots of ways to derive nice explicit formulas here. (Outside of an exam you would just google "determinant tridiagonal matrix," of course.) For example, if we write:

$$H = \begin{pmatrix} a_1 & \overline{b_1} & & & \\ b_1 & a_2 & \overline{b_2} & & \\ & b_2 & a_3 & \ddots & \\ & & \ddots & \ddots & \overline{b_{m-1}} \\ & & & b_{m-1} & a_m \end{pmatrix},$$

then each step of Gaussian elimination transforms the 2×2 diagonal block

$$\begin{pmatrix} d_k & \overline{b_k} \\ b_k & a_{k+1} \end{pmatrix} \longrightarrow \begin{pmatrix} d_k & \overline{b_k} \\ 0 & a_{k+1} - \frac{b_k \overline{b_k}}{d_k} \end{pmatrix},$$

so that the diagonal entries satisfy the recurrence relation

$$\begin{aligned} d_1 &= a_1 \\ d_{k+1} &= a_{k+1} - \frac{|b_k|^2}{d_k}. \end{aligned}$$

and once it is reduced to upper-triangular form then the determinant is simply the product of the pivots $\prod d_k$. This recurrence may look slightly dangerous at first — what if $d_k = 0$? However, this division by zero goes away when you multiply the entries together — consider the term $d_k d_{k+1}$ — and after a little thought you can see that the the product

$$p_k = \prod_{i=1}^{k} d_k$$

satisfies a simpler recurrence (called the "continuant" in linear algebra):

$$\begin{aligned} p_0 &= 1 \\ p_1 &= a_1 \\ p_{k+1} &= d_{k+1} p_k = p_k a_{k+1} - p_{k-1} |b_k|^2, \end{aligned}$$

which has no possibility of division by zero, giving $\det H = p_m$ in $\Theta(m)$ operations. Finally, get $\det(H - zI)$, we simply modify this recurrence to subtract z from the diagonals:

$$\begin{aligned} p_0 &= 1 \\ p_1 &= a_1 - z \\ p_{k+1} &= p_k (a_{k+1} - z) - p_{k-1} |b_k|^2. \end{aligned}$$

This recurrence can also be derived in other ways, e.g. by cofactor formulas. For the case of real b_i (real-symmetric A and H), the same recurrence is given in equation (30.9) of the Trefethen & Bau textbook.

Another possible $\Theta(m)$ determinant algorithm is to do the QR factorization of $H - zI$, which can be accomplished in $\Theta(m)$ operations by Givens rotations as you showed in pset 3. Then the determinant is simply $\det R$ (since $\det Q = 1$ for Givens rotations), which is the product of the diagonal entries of R.