Midterm Exam

Math 651
Fall 2019

Due Friday, November 1

1 Instructions

Please do not collaborate with your fellow students, and do not search the internet for answers. You may refer to your course notes and the texts, and you may ask me questions.

Problem 4 is open-ended. That is, it admits many different answers. I have listed some questions that you may want to address as part of your own answer. You should not feel that you have to address all of these questions. In fact, you do not have to address any of the proposed questions—I encourage you to develop your own. I only want you to think critically about what we have been doing.

2 Exam

Problem 1 (10 points). *Let $f : \mathbb{R} \to \mathbb{R}$ be four times continuously differentiable. Assume that the fourth derivative of f is bounded. Show that*

$$\left| \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} - f''(x) \right| \leq C \|f^{(4)}\|_\infty h^2$$

for some constant $C > 0$ that does not depend on f or h.

You might think that the determinant and the condition number of a matrix are related, since the determinant is zero if and only if the matrix is not invertible. In fact, the determinant and condition number are totally unrelated, as you will show in this next problem.

Problem 2 (10 points). *Find a sequence of square matrices $M_n \in \mathbb{R}^{5\times 5}$ so that $\lim_{n\to\infty} \det(M_n) = 0$ but $\kappa_2(M_n) = 1$ for all n.*

Define

$$x_+^3 := \begin{cases} x^3, & x \geq 0, \\ 0, & x < 0. \end{cases}$$

Recall that the $n+3$ functions

$$\begin{aligned} u_k(x) &:= (x-x_0)^k, && \text{for } k = 0,\dots,3, \\ v_k(x) &:= (x-x_k)_+^3, && \text{for } k = 1,\dots,n-1, \end{aligned}$$

are a basis for the set $\mathscr{S}^3$ of cubic splines on the knots $x_0 < x_1 < \cdots < x_n$. Also, recall that the natural cubic interpolating spline of a function $f : \mathbb{R} \to \mathbb{R}$ is the unique cubic spline $s \in \mathscr{S}^3$ so that

$$\begin{aligned} s(x_i) &= f(x_i) \text{ for all } i = 0,\dots,n, \text{ and} \\ s''(x_0) &= s''(x_n) = 0. \end{aligned} \tag{1}$$

To calculate the interpolating spline, one could use the basis $\{u_0,\dots,u_3,v_1,\dots,v_{n-1}\}$. That is, one could write

$$s = \sum_{k=0}^{3} \alpha_k u_k + \sum_{j=1}^{n-1} \beta_j v_j,$$

and interpret (1) as a linear system determining the coefficients α_k and β_j. In matrix form, the linear system reads

$$V(x_0,\dots,x_n) \begin{pmatrix} \alpha_0 \\ \vdots \alpha_3 \\ \beta_1 \\ \vdots \\ \beta_{n-1} \end{pmatrix} = \begin{pmatrix} f(x_0) \\ \vdots \\ f(x_n) \\ 0 \\ 0 \end{pmatrix},$$

where

$$V(x_0,\dots,x_n) = \begin{pmatrix} u_0(x_0) & \dots & u_3(x_0) & v_1(x_0) & \dots & v_{n-1}(x_0) \\ u_0(x_1) & \dots & u_3(x_1) & v_1(x_1) & \dots & v_{n-1}(x_1) \\ & \vdots & & & \vdots & \\ u_0(x_n) & \dots & u_3(x_n) & v_1(x_n) & \dots & v_{n-1}(x_n) \\ u_0''(x_0) & \dots & u_3''(x_0) & v_1''(x_0) & \dots & v_{n-1}''(x_0) \\ u_0''(x_n) & \dots & u_3''(x_n) & v_1''(x_n) & \dots & v_{n-1}''(x_n) \end{pmatrix}. \tag{2}$$

Here, the first $n+1$ rows of the matrix V correspond to the $n+1$ conditions $s(x_i) = f(x_i)$, and the last two rows correspond to the natural boundary condition $s''(x_0) = s''(x_n) = 0$.

Problem 3 (10+10+2+2 points)**.** *For each $N \in \mathbb{N}$ define a set of equally spaced knots*

$$x_0^n = 0, x_1^n = 1/n, \dots, x_n^n = 1$$

covering the interval $[0,1]$. Let $V_n = V(x_0^n,\dots,x_n^n)$.

1. *Use the numerical SVD to estimate the condition number* $\kappa_2(V_n)$ *for*

$$n \in \{5, 10, 20, 50, 75, 100, 150\}.$$

   Report the following plots:

   - $\kappa_2(v_n)$ *vs.* n
   - $\log_{10}(\kappa_2(v_n))$ *vs.* n
   - $\log_{10}(\kappa_2(v_n))$ *vs.* $\log_{10}(n)$

   You will see that $\kappa_2(V_n)$ *increases with* n*. Do you think it increases polynomially or exponentially? If polynomially, what is the exponent? If exponentially, what is the rate?*

2. *Define*

$$f(x) = \sin^2(2x) + \exp(x).$$

   Calculate the coefficients α_j *and* β_k *for the natural cubic interpolating spline of* f *with* $n = 2000$ *using the* QR*-factorization and back substitution. (To perform back substitution in python, you can use the function np.linalg.solve_triangular.) Compute the error vector*

$$err = V_{2000} \begin{pmatrix} \alpha_0 \\ \vdots \alpha_3 \\ \beta_1 \\ \vdots \\ \beta_{n-1} \end{pmatrix} - \begin{pmatrix} f(x_0) \\ \vdots \\ f(x_n) \\ 0 \\ 0 \end{pmatrix}.$$

   Plot the entries of err*. Report* $\|err\|_\infty$*. Observe that* $\|err\|_\infty$ *is actually quite small, so even though* V_{2000} *is ill-conditioned, the coefficients computed by inverting* V_{2000} *numerically yield a cubic spline that very nearly interpolates* f *on the knots. Explain why this is the case. Your answer should draw on concepts from class, e.g. backwards stability.*

3. *Since* V_{2000} *is ill-conditioned, the error in the calculated coefficients* α_j *and* β_k *is probably very large. (You don't have to try to demonstrate this.) But which is more important, the error in the coefficients or the error vector* err*?*

4. *Is it a bad idea to compute interpolating splines using this very ill-conditioned basis? Why or why not?*

Problem 4 (20 points)**.** *Please criticize and extend the analysis of finite differences developed in Homework 5, Problem 2. (For your convenience, I have included the statement of this problem below in Appendix A.) To get you started, here are a few points that you might want to address:*

1. *Carry out a floating point error analysis for the estimate of the second derivative given in Problem 1 above. Does the error for the optimal value of* h *decrease faster or slower than* $O(\sqrt{\varepsilon_m})$ *with machine precision?*

2. *The sum* $x + h$ *is not necessarily a floating point number. How does the analysis change if you take that into account?*

3. *The estimates in Homework 5, Problem 2 are not very precise. As an alternative, one could develop an analysis based on asymptotic formulas such as*
$$\frac{f(x+h) - f(x)}{h} - f'(x) = hf''(x) + o(h).$$
In what respects would an analysis based on asymptotics be better or worse than one based on upper bounds?

4. *In your homework you proved a theoretical result concerning the limit as* ε_m *tends to zero. However, you only did calculations in 64-bit arithmetic, which corresponds to taking* $\varepsilon_m = 2^{-53}$*. It would have been better to do calculations for a decreasing sequence of values of* ε_m*. How might you simulate floating point of less than 64-bit precision? That is, given a precision* ε_m*, can you come up with a "fake" version of floating point arithmetic that satisfies the fundamental axiom of floating point? Are there readily available software packages that actually perform arbitrary precision floating arithmetic?*

A Homework 5, Problem 2

I have included Homework 5, Problem 2 in this appendix for reference. You do not have to turn in a solution to this problem.

Problem 5 (2+2+2+2 points)**.**

1. *Compute*
$$d(h) := \frac{exp(h) - exp(0)}{h}$$
for $h \in \{2^{-k} : k = 0, \ldots, 60\}$*. Now do the following:*

   - *Plot* $-\log_{10}(h)$ *vs.* $|d(h) - 1|$.
   - *Plot* $-\log_{10}(h)$ *vs.* $\log_{10}(|d(h) - 1|)$*. (Most likely, when you generate this plot, numpy.log10 will return an error message. Make sure to understand and fix the error.)*
   - *Compute and report*
$$\min_{h \in \{2^{-k}:k=0,\ldots,60\}} |d(h) - 1|.$$

2. *Let $f : \mathbb{R} \to \mathbb{R}$. Assume for convenience that f is bounded. Let $x, h \in \mathbb{F}$, and let ε_m denote machine precision. Assume that $\varepsilon_m \leq 1$. Show that*

$$\left| (fl(f(x+h)) \ominus fl(f(x))) \oslash h - \frac{f(x+h) - f(x)}{h} \right| \leq \frac{C\varepsilon_m \|f\|_\infty}{|h|}$$

for some constant $C > 0$ that does not depend on ε_m or f. Hint: *Use the fundamental axiom of floating point. It is probably easiest to do a forwards analysis instead of a backwards analysis.*

3. *Now let $f \in C^2$. That is, assume that f is twice continuously differentiable and that f, f', and f'' are bounded. Show that*

$$\left| \frac{f(x+h) - f(x)}{h} - f'(x) \right| \leq |h| \|f''\|_\infty.$$

.

4. *Combining the results above, we see that*

$$|(fl(f(x+h)) \ominus fl(f(x))) \oslash h - f'(x)| \leq \frac{C\varepsilon_m \|f\|_\infty}{|h|} + |h| \|f''\|_\infty.$$

Define

$$E(\varepsilon_m, h) := \frac{C\varepsilon_m \|f\|_\infty}{|h|} + |h| \|f''\|_\infty.$$

Now let $h(\varepsilon_m)$ be the value of h that minimizes $E(\varepsilon_m, h)$ for fixed ε_m. That is, define

$$h(\varepsilon_m) := \operatorname*{argmin}_h E(\varepsilon_m, h).$$

Show that $E(\varepsilon_m, h(\varepsilon_m)) = O(\sqrt{\varepsilon_m})$ as $\varepsilon_m \to 0$, so even in the best case the error of the finite difference approximation decreases only like $\sqrt{\varepsilon_m}$ with machine precision. Do the computations in part 1 support this theoretical result?