

Math 543, Spring 2020

Midterm #1, Due Friday April 10 at 11:59pm.


Submission: UPLOAD ALL PAGES TO GRADESCOPE, attach extra page(s)
AFTER the 10 numbered pages

Tools: Pen/Pencil/Eraser/Paper.

Rules: This is a take-home midterm; see below...

Problem	Pts Possible	Pts Scored
1(a)	5	
1(b)-i	5	
1(b)-ii	5	
1(b)-iii	5	
1(b)-iv	5	
1(c)-i	5	
1(c)-ii	5	
1(c)-iii	5	
2(a)	5	
2(b)	5	
2(c)	5	
2(d)	5	
2(e)	5	
2(f)-i	5	
2(f)-ii	5	
2(f)-iii	5	
2(f)-iv	5	
3(a)	5	
3(b)	5	
3(c)	5	
3(d)	5	
4	5	
5	0	
Total	110	

I, Stephen Giang, pledge that this exam is *completely my own work*, and that I did not take, borrow or steal any portions from any other person, including "*Questionable Cousin Chegg*." Any and all references I used are clearly cited in my solutions. I understand that if I violate this honesty pledge, I am subject to disciplinary action pursuant to the appropriate sections of the San Diego State University Policies.


Signature

Rules: This midterm is open-book, open-notes. You cannot consult any other human. If you refer to results from books (including the class text), research papers, or the web (other than the class web page(s)) carefully cite your source(s).

Midterm 1
Numerical Matrix Analysis
Math 543
Stephen Giang

Problem 1:

- (a) For the function $f : X \rightarrow Y$, define the relative condition number $\kappa(x)$

Let $x \in X$ and $y \in Y$, such that $f(x) = y$

$$\kappa(x) = \sup_{\delta x} \left[\frac{\|\delta f\|}{\|f(x)\|} \bigg/ \frac{\|\delta x\|}{\|x\|} \right]$$

- (b) (i) What is the smallest (in magnitude) real perturbation $\epsilon_{\mathbb{R}} \in \mathbb{R} \setminus \{0\}$?

Because there does not exist a smallest positive number in the set of $\mathbb{R} \setminus \{0\}$, the following does not exist.

- (ii) What is the smallest (in magnitude) integer perturbation $\epsilon_{\mathbb{Z}} \in \mathbb{Z} \setminus \{0\}$?

Because the smallest positive integer is 1, the smallest integer perturbation is in fact 1

- (iii) What is the smallest (in magnitude) IEEE 754-1985 64-bit floating point \mathbb{F}_{64} perturbation $\epsilon_{\mathbb{F}_{64}} \in \mathbb{F}_{64} \setminus \{0\}$?

For IEEE 754-1985 64-bit, the format is $sc_{10}c_9...c_0m_{51}m_{50}...m_0$. We can use the formula

$$r = (-1)^s 2^{c-1023} (1 + f) \quad c = \sum_{n=0}^{10} c_n 2^n, f = \sum_{k=0}^{51} \frac{m_k}{2^{52-k}}$$

So we get $c = 0$ and $f = 2^{-52}$

Which means that the exponent is 2^{-1023} and the gap is $(2^{-1023})(2^{-52})$, the relative gap $\epsilon_{\mathbb{F}_{64}} = 2^{-52} = 2.22044604925031 * 10^{-16}$

- (iv) What is the smallest (in magnitude) IEEE 754-1985 128-bit floating point \mathbb{F}_{128} perturbation $\epsilon_{\mathbb{F}_{128}} \in \mathbb{F}_{128} \setminus \{0\}$?

For IEEE 754-1985 128-bit, the format is $sc_{14}c_{13}...c_0m_{111}m_{110}...m_0$. We can use the formula

$$r = (-1)^s 2^{c-16383} (1 + f) \quad c = \sum_{n=0}^{14} c_n 2^n, f = \sum_{k=0}^{111} \frac{m_k}{2^{112-k}}$$

So we get $c = 0$ and $f = 2^{-112}$

Which means that the exponent is 2^{-16383} and the gap is $(2^{-16383})(2^{-112})$, the relative gap $\epsilon_{\mathbb{F}_{128}} = 2^{-112} = 1.92592994438724 * 10^{-34}$

Link: https://en.wikipedia.org/wiki/IEEE_754 (clickable link)

- (c) (i) For the specific function $f(x) = \begin{cases} -55 & x < 0.5 \\ 55 & x \geq 0.5 \end{cases}$ where $x \in \mathbb{R}$ is a **real variable** in the interval $[0, 1]$, what is the relative condition number $\kappa(x)$, for all values of x ? (i.e. as a function of x)

$$\begin{aligned} \kappa(x) &= \sup_{\delta x} \left[\frac{||\delta f||}{||f(x)||} \bigg/ \frac{||\delta x||}{||x||} \right] \\ &= \sup_{\delta x} \left[\frac{110}{55} \bigg/ \frac{||\delta x||}{||x||} \right] \\ &= \sup_{\delta x} \frac{2||x||}{||\delta x||} \\ &= DNE \end{aligned}$$

- (ii) For the specific function $f(x) = \begin{cases} -55 & x < 0.5 \\ 55 & x \geq 0.5 \end{cases}$ where $x \in \mathbb{F}_{64}$ is a **64-bit floating point variable** in the interval $[0, 1]$, what is the relative condition number $\kappa(x)$, for all values of x ? (i.e. as a function of x)

$$\begin{aligned} \kappa(x) &= \sup_{\delta x} \left[\frac{||\delta f||}{||f(x)||} \bigg/ \frac{||\delta x||}{||x||} \right] \\ &= \sup_{\delta x} \left[\frac{110}{55} \bigg/ \frac{||\delta x||}{||x||} \right] \\ &= \sup_{\delta x} \frac{2||x||}{||\delta x||} \\ &= \frac{2|x|}{2^{-52}} \\ &= \frac{|x|}{2^{-53}} \end{aligned}$$

- (iii) For the specific function $f(x) = \begin{cases} -55 & x \leq 9,989,224 \\ 55 & x \geq 9,989,225 \end{cases}$ where $x \in \mathbb{Z}$ is an **integer variable**, what is the relative condition number $\kappa(x)$, for all values of x ? (i.e. as a function of x)

$$\begin{aligned}
\kappa(x) &= \sup_{\delta x} \left[\frac{\|\delta f\|}{\|f(x)\|} \bigg/ \frac{\|\delta x\|}{\|x\|} \right] \\
&= \sup_{\delta x} \left[\frac{110}{55} \bigg/ \frac{\|\delta x\|}{\|x\|} \right] \\
&= \sup_{\delta x} \frac{2\|x\|}{\|\delta x\|} \\
&= \frac{2\|x\|}{1} \\
&= 2|x|
\end{aligned}$$

Problem 2: Derive the matrix least squares problem for fitting a data set $\{y(t_i), t_i\}_{i=1, \dots, m}$ by

(a) a constant

Notice: a constant is $y(t_i) = c = y_i$

$$\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} [c] = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$$

(b) a straight line $z(t) = a + bt$

Notice: a straight line is $y(t_i) = a + bt_i = y_i$

$$\begin{bmatrix} 1 & t_1 \\ \vdots & \vdots \\ 1 & t_m \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$$

(c) What is the solution in part (a)?

Solving using the formula: $A^* A \vec{c} = A^* \vec{y}$

$$\begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} [c] = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$$

$$mc = y_1 + \cdots + y_m$$

$$c = \frac{y_1 + \cdots + y_m}{m}$$

Notice this is the average of the y values of the points given, which makes sense because the least squares fit should be a line that goes through a majority of the points.

- (d) Explain why, in general, solving the least squares problem using the normal equations is not such a good idea.

The problem with solving the least squares problem using normal equations is that it is unstable when $\kappa(A)$ is not uniformly bounded.

- (e) Suggest one alternative approach, what are its advantages and disadvantages?

SVD is an alternative approach. Advantages to it is that it is stable for full rank matrices, where normal equations is unstable. A disadvantage to it is that it is the most expensive however between all the other methods.

- (f) **Part (ii.):** Identify period of exponential growth (a period where the growth looks linear on the log scale)

The graph looks linear from 39 days to 70 days after 1/22.

Specified Date Range: { 3/1/20 - 4/1/20 }

Part (iv): Use the model to extrapolate 7 days (add to the plot); what is the count?

$$\begin{Bmatrix} 140,427 \\ 169,617 \\ 204,874 \\ 247,461 \\ 298,900 \\ 361,031 \\ 436,077 \end{Bmatrix}$$

Problem 3: Let $f(x)$ denote the exact solution to the exact problem, let $\tilde{f}(x)$ denote the solution computed by an algorithm

- (a) Complete the statement, "We say that an algorithm \tilde{f} for a problem $f(x)$ is stable if for each $x \in X$ "

$$\frac{\|\tilde{f}(x) - f(\tilde{x})\|}{\|f(\tilde{x})\|} = \mathcal{O}(\epsilon_{mach})$$

for some \tilde{x} with

$$\frac{\|\tilde{x} - x\|}{\|x\|} = \mathcal{O}(\epsilon_{mach})$$

- (b) Complete the statement, "We say that an algorithm \tilde{f} for a problem $f(x)$ is backward stable if for each $x \in X$ "

$$\tilde{f}(x) = f(\tilde{x})$$

for some \tilde{x} with

$$\frac{\|\tilde{x} - x\|}{\|x\|} = \mathcal{O}(\epsilon_{mach})$$

- (c) Show that the obvious algorithm for solving (for x) the 1-by-1 system $rx = b$ is backward stable if executed in matlab on a modern-day computer. Explain your notation and any results / definitions / axioms you are invoking.

Let

$$\begin{array}{lll} f(x) = rx = b & \tilde{f}(x) = rx(1 + \epsilon) = b & f(\tilde{x}) = r(x(1 + \epsilon)) = b \\ x = \frac{b}{r} & x = \frac{b}{r(1 + \epsilon)} & x = \frac{b}{r(1 + \epsilon)} \end{array}$$

Thus for $f(x) = b$, because $\tilde{f}(x) = f(\tilde{x})$ with $\frac{\|\tilde{x} - x\|}{\|x\|} = \mathcal{O}(\epsilon_{mach})$, the algorithm to solve this which is division is backwards stable

- (d) Explain how backward stability, the condition number, and the available “computational precision” limit how well we can solve a problem numerically. (You may state a theorem, but it is not necessary (but highly recommended)).

With backwards stability, condition numbers, and the available “computational precision”, it limits how well we can solve a problem numerically. Because a computer can not calculate an exact smallest number or an exact greatest number, there will always be round-off errors that will give the slightest change in our answers. Because of this as well, we always have to consider the δ in all our solutions such that $(1 + 0) \neq (1 + 0)$ when computing as $(1 + 0) = (1 + 0)(1 + \epsilon)$. As well, with these limitations, we sometimes get instability which can lead to very bad results.

Problem 4: The sloped line is most likely a least squares fit (of some kind). Do you have any comments?

We see that from the least square fit that the approximate line most likely seems to hit the majority of the points given, and almost completely ignores the few outliers. This might be because of the two bigger outliers, NYSE and Deutsche Börse, seem to cancel each other out as they're both approximately equidistant to the line. The line also tells these companies how they are doing in retrospect to the other companies, almost like an 'average' line.

```

clear
figure(1)
clf
hold off

grid on
hold on

usCovidData = ✓
[1,1,2,2,5,5,5,5,5,7,8,8,11,11,11,11,11,11,11,11,12,12,13,13,13,13,13,13,13,15,15,15,5 ✓
1,51,57,58,60,68,74,98,118,149,217,262,402,518,583,959,1281,1663,2179,2727,3499,4632,6421 ✓
,7783,13677,19100,25489,33276,43847,53740,65778,83836,101657,121478,140886,161807,188172, ✓
213372];
usCovidData = transpose(usCovidData);
logUSCovidData = log10(usCovidData);

time = 0 : size(logUSCovidData) - 1;
time = transpose(time);

%Semi-Log Plot
plot(time,logUSCovidData, 'r');

%QR Factorization for Least-Squares Linear Fit
yint = ones(size(time));
A = [yint time];
[Q, R] = qr(A);
x = linsolve(R, Q*logUSCovidData);
yInt = x(1);
slope = x(2);
plot(time, line(yInt,slope,time), 'b');

xticks(0:2:70);
yticks(-1:.5:6);
xlabel('Days Since Jan 22. ');
ylabel('$\log_{10}$ of Confirmed Cases', 'interpreter', 'latex');
title('Confirmed Cases since Jan 22. ');
legend({'Actual Data', 'Linear Fit'}, 'location', 'southeast');

extrapData = ones(7,1);
for i = 1:7
    logData = line(yInt,slope,size(time,1) + i);
    extrapData(i,:) = 10^logData;
end

function y = line(yint,slope,x)
    y = yint + slope*x;
end

```

