*Updated on July 14, 2021*

---

# Math 3607: Exam 2

## Due: 11:59PM, Saturday, July 17, 2021

Please read the statements below and sign your name.

### Disclaimers and Instructions

- You may use any of the book functions (quote), any of functions provided in lectures, accompanying live scripts, and homework or practice problem solutions, and any of your own helper functions. As stated in the syllabus, you must be able to explain how they are supposed to work. You may be requested to explain your code to me, in which case a proper and satisfactory explanation must be provided to receive any credits on relevant parts.
- You are not allowed to search online forums or even MathWorks website. All problems can be solved with what has been taught in class. If you need to look something up, then you must be doing it incorrectly.
- Unlike for homework, you are not allowed to collaborate with classmates.
- If any code is found to be plagiarized from the internet or another person, you will receive a zero on the *entire* exam (both parts).

---

### Academic Integrity Statements

- All of the work shown on this exam is my own.
- I will not consult with any resources (MATLAB documentation, online searches, etc.) other than the textbooks, lecture notes, and supplementary resources provided on the course Carmen pages.
- I will not discuss any part of this exam with anyone, online or offline.
- I understand that academic misconduct during an exam at The Ohio State University is very serious and can result in my failing this class or worse.
- I understand that any suspicious activity on my part will be automatically reported to the OSU Committee on Academic Misconduct for their review.

Signature ____________________________

**Notation.** Problems marked with ✏ are to be done by hand; those marked with 💻 are to be solved using a computer.

## 1 PLU Factorization [15 points]

✏ Find the PLU factorization of the matrix

$$A = \begin{bmatrix} 2 & -1 & 3 \\ 10 & -7 & 10 \\ -6 & 4 & -5 \end{bmatrix}$$

by Gaussian elimination *with partial pivoting*. That is, find matrices $P$ (permutation matrix), $L$ (unit lower triangular matrix), and $U$ (upper triangular matrix) such that $PA = LU$. Justify all your steps.

*Hint.* You may (and should) use, without proof, the properties of Gaussian transformation matrices presented in Problem 7 of Module 2 practice problem set. Be sure to give a clear reference to the properties used in your solution.

## 2 Catastrophic Cancellation [20 points]

Let

$$f(x) = \begin{cases} 2\,\dfrac{1-\cos x}{x^2} & \text{if } x \neq 0, \\ 1 & \text{if } x = 0. \end{cases}$$

(a) Find the condition number $\kappa_f(x)$; simplify it as much as you can. Then compute

$$\lim_{x\to 0} \kappa_f(x).$$

(b) For small $x$, the "obvious" evaluation algorithm

```
f1 = @(x) 2*(1-cos(x))./(x.^2);
```

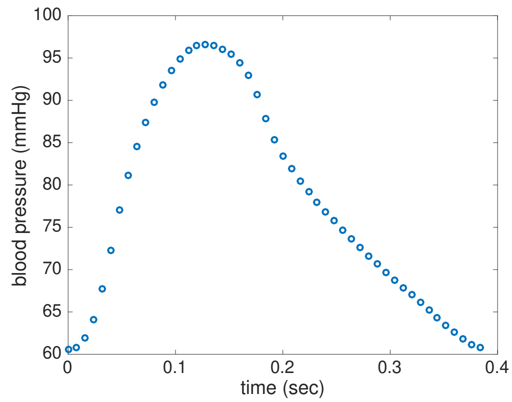
suffers from catastrophic cancellation. Explain why.

(c) Using the first 3 terms of the Taylor series expansion of $f(x)$, establish a second algorithm `f2` to compute $f(x)$ stably for small $x$. Fully justify the derivation of your algorithm.

(d) Evaluate $f(x)$ for $x = 10^{-k}$ for $k \in \mathbb{N}[1, 10]$ using the two algorithms `f1` and `f2`. Tabulate the results neatly. The table should have three columns with the first being `x`, the second being `f1`, and the third being `f2`. Use either `format long g` or an appropriate `fprintf` statement to display full accuracy.

## 3 Least Squares for Periodic Data [30 points]

The graph below represents arterial blood pressure collected at 8 ms (milliseconds) intervals over one heart beat from an infant patient:




Denote the data points by $(t_i, y_i)$ for $i = 1, \ldots, m$. The data can be fit using a low-degree polynomial of the form

$$f(t) = c_1 + c_2 t + \cdots + c_n t^{n-1}, \quad n < m. \tag{1}$$

In the most general terms, the fitting function takes the form

$$f(t) = c_1 f_1(t) + \cdots + c_n f_n(t), \tag{2}$$

where $f_1, \ldots, f_n$ are known functions while $c_1, \ldots, c_n$ are to be determined to optimize the fit to the data. This optimization can be formulated as an $m \times n$ LLS problem of minimizing the 2-norm of the residual $\|\mathbf{y} - A\mathbf{c}\|_2$, where $A_{i,j} = f_j(t_i)$.

(a) Download the data file `pressuredata.mat` and load them into MATLAB using

```
load pressuredata
```

This creates two vectors `t` and `y` containing time and blood pressure data, respectively. Use them to regenerate the plot above.

(b) Fit the data to a straight line, $f(t) = c_1 + c_2 t$. Solve for the coefficients using backslash. Superimpose the graph of the fitting line on your graph from the previous step, and compute the 2-norm of the residual $\|\mathbf{y} - A\mathbf{c}\|_2$, where $A_{i,j} = t_i^{j-1}$ is a *Vandermonde*-type matrix.

(c) Repeat part (b) for a quadratic and a cubic polynomial. The residual norm will get smaller in each case, but there is still a room for improvement.

(d) Exploiting the fact that the data come from a periodic phenomenon (heart beats), adapt (2) to a periodic fitting function

$$f(t) = c_1 + c_2 \cos\frac{2\pi t}{\tau} + c_3 \sin\frac{2\pi t}{\tau} + c_4 \cos\frac{4\pi t}{\tau} + c_5 \sin\frac{4\pi t}{\tau}, \quad \text{where } \tau = t_m - t_1. \tag{3}$$

As in the previous parts, solve for the coefficients using backslash, superimpose the graph of $f(t)$ to the plot of data points, and compute the residual norm. Comment on your observation.

## 4 Visualization of Matrix Norms in 3-D [35 points]

Recall that

$$\|A\|_p = \max_{\|\mathbf{x}\|_p=1} \|A\mathbf{x}\|_p , \quad p \in [1, \infty].$$

In this problem, we generate three-dimensional visualization of this definition. This is a direct extension of a recent homework problem.

(a) Complete the following program[1] which, given $p \in [1, \infty]$ and $A \in \mathbb{R}^{3\times 3}$, approximates $\|A\|_p$ and plots the unit sphere in the $p$-norm and its image under $A$. Avoid using loops as much as possible.

```
function norm_A = visMatrixNorms3D(A, p)
    %% Basic checks
    if size(A,1)~=3 || size(A,2)~=3
        error('A must be a 3-by-3 matrix.')
    elseif p < 1
        error('p must be >= 1.')
    end

    %% Step 1: Initialization
    nr_th = 41; nr_ph = 31;
    th = linspace(0, 2*pi, nr_th);
    ph = linspace(0, pi, nr_ph);
    [T, P] = meshgrid(th, ph);
    x1 = cos(T).*sin(P);
    x2 = sin(T).*sin(P);
    x3 = cos(P);
    X = [x1(:), x2(:), x3(:)]';

    %% Step 2: [FILL IN] Normalize columns of X into unit vectors



    %% Step 3: [FILL IN] Form Y = A*X; calculate norms of columns of Y



    %% Step 4: [FILL IN] Calculate p-norm of A (approximate)



    %% Step 5: [FILL IN] Generate surface plots


end
```

[1] The function must be written at the very end of your Live Script.

The following steps are carried out by the program.

**Step 1.** Create 3-vectors

$$\mathbf{x}_k = \begin{bmatrix} \cos\theta_i \sin\phi_j \\ \sin\theta_i \sin\phi_j \\ \cos\phi_j \end{bmatrix}, \quad \text{for } 1 \le i \le 41,\ 1 \le j \le 31 \tag{4}$$

using 41 evenly distributed $\theta_i$ in $[0, 2\pi]$ and 31 evenly distributed $\phi_j$ in $[0, \pi]$. Note the use of `meshgrid`, which is useful for surface plots later.

**Step 2.** Normalize $\mathbf{x}_k$ into a unit vector in $p$-norm by $\mathbf{x}_k \leftarrow \mathbf{x}_k / \|\mathbf{x}_k\|_p$ (that is, replacing $\mathbf{x}_k$ with $\mathbf{x}_k / \|\mathbf{x}_k\|_p$).

**Step 3.** For each $k$, let $\mathbf{y}_k = A\mathbf{x}_k$. Calculate and store $\|\mathbf{y}_k\|_p$.

**Step 4.** Approximate $\|A\|_p$ based on the norms $\|\mathbf{y}_k\|_p$ calculated in the previous step.

**Step 5.** Generate surface plots of the unit sphere in the $p$-norm and its image under $A$. Use `surf` function; see pp. 125-128 of Module 1 lecture slides. Use `subplot` to put two graphs side by side.

(b) Run the program with $p = 1, \frac{3}{2}, 2, 4$, all with the same matrix

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 0 & \cos(\pi/12) & -\sin(\pi/12) \\ 0 & \sin(\pi/12) & \cos(\pi/12) \end{bmatrix}. \tag{5}$$

```
A = [2 0 0;
     0 cos(pi/12) -sin(pi/12);
     0 sin(pi/12)  cos(pi/12)];
visMatrixNorms3D(A, 1);        % Depending on how you write the code,
visMatrixNorms3D(A, 3/2);      % you may need to use 'clf' or 'hold off'
visMatrixNorms3D(A, 2);        % in between function calls.
visMatrixNorms3D(A, 4);
```

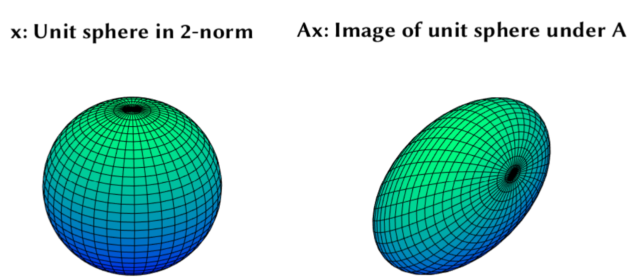



Figure 1: Example output.