Updated on February 3, 2022

Spring 2022 Math 3607: Exam 1

Due: 6:00PM, Friday, February 4, 2022

Please read the statements below and sign your name.

Disclaimers and Instructions

- You are **not** allowed to use MATLAB commands and functions **other than** the ones discussed in lectures, accompanying live scripts, textbooks, and homework/practice problem solutions.
- 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 for this exam.
- You are **not** allowed to collaborate with classmates, unlike for homework.
- If any code is found to be plagiarized from the internet or another person, you will receive a zero on the *entire* exam and will be reported to the COAM.
- Do not carry out computations using *Symbolic Math Toolbox*. Any work done using `sym`, `syms`, `vpa`, and such will receive NO credit.
- **Notation.** Problems marked with ✏ are to be done by hand; those marked with 💻 are to be solved using a computer.
- Answers to analytical questions (ones marked with ✏) without supporting work or justification will not receive any credit.

Academic Integrity Statements

- All of the work shown on this exam is my own.
- I will not consult with any resources (MathWorks website, 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 (COAM) for their review.

Signature ____________________________

1 Surface Plot [20 points]

Plot the surface represented (parametrically) by

$$\begin{cases} x(\theta,\varphi) = (R + r\cos\theta)\cos\varphi \\ y(\theta,\varphi) = (R + r\cos\theta)\sin\varphi \\ z(\theta,\varphi) = r\sin\theta \end{cases} \qquad \text{for } \theta,\varphi \in [0, 2\pi].$$

Use $0 < r < R$ of your own choice. Do this in a single code block; you do not need to write a script for this problem. Begin your code block with `clf`. At the end of the code block, include

```
axis equal, axis off
```

Suggestions.

- Use sufficiently many points so that the generated figure looks reasonably smooth. Too many points, however, will make it look unaesthetic, and your code will run slow.
- You are free to modify the color theme using `colormap` function.
- For visually pleasing/familiar results, it is recommended that you pick r and R such that R/r is about $3/2$.

2 Birthday Problem [25 points]

This problem is adapted from LM 3.9–22 which contains a useful hint. It is also a continuation of a recent homework problem.

(a) Write a script `threeBdayMatch.m` which generates a group of `n` people randomly and determines if there are at least **three** people with the same birthday. This script should take `n` as an input. Do this without using a loop nor an if-statement. Then print out the content of your script using `type`:

```
type threeBdayMatch.m
```

(b) Write another script `threeBdayMatchSims.m` which runs the previous simulation multiple times and calculates an approximate probability of having at least **three** people with the same birthday. This script should take `n` and the number of simulations `n_sims` as inputs. Do this without using a loop nor an if-statement. Then print out the content of your script using `type`.

```
type threeBdayMatchSims.m
```

(c) Call the script from part (b) with $n = 30, 40, \ldots, 100$, each with 10 000 simulations by running the following code block.

```
n_sims = 10000;
for n = 30:10:100
    threeBdayMatchSims
end
```

Note. If your script contains lines using the `input` function such as

```
n = input('Enter the number of people: ');
n_sims = input('Enter the number of simulations: ');
```

comment them out, just as you were instructed for previous homework assignments.

3 Approximation of π [25 points]

Each of the following sequences converges to π:

$$a_n = \frac{6}{\sqrt{3}} \sum_{k=0}^{n} \frac{(-1)^k}{3^k(2k+1)}, \qquad b_n = 16 \sum_{k=0}^{n} \frac{(-1)^k}{5^{2k+1}(2k+1)} - 4 \sum_{k=0}^{n} \frac{(-1)^k}{239^{2k+1}(2k+1)}.$$

This is a continuation of a recent homework problem. Here our focus is to use vectorized codes (no loops allowed) and to produce visual illustrations of the convergence behavior. Answer each question in a single code block; you do not need to write a script for this problem.

(a) Generate two row vectors `a` $= (a_0, a_1, a_2, \ldots, a_{30})$ and `b` $= (b_0, b_1, b_2, \ldots, b_{30})$, without using a loop. Use semicolons to suppress outputs.

(b) Using `a` and `b` from the previous part, plot a_n and b_n against n for $n = 0, \ldots, 30$ on a single graph. Circle the data points and connect them with lines. Give the plot a title, label axes, and create legends as shown in the example figure below. Begin your code block with `clf`.

Note. The expression "plot a_n against n" means that n is along the horizontal axis and a_n is along the vertical axis.

(c) Now plot $|a_n - \pi|$ and $|b_n - \pi|$ against n for $n = 0, \ldots, 30$ on a single log-linear graph. To draw on a log-linear graph, just replace `plot` by `semilogy`. Circle the data points and connect them with lines. Give the plot a title, label axes, and create legends as shown in the example figure below. Begin your code block with `clf`.

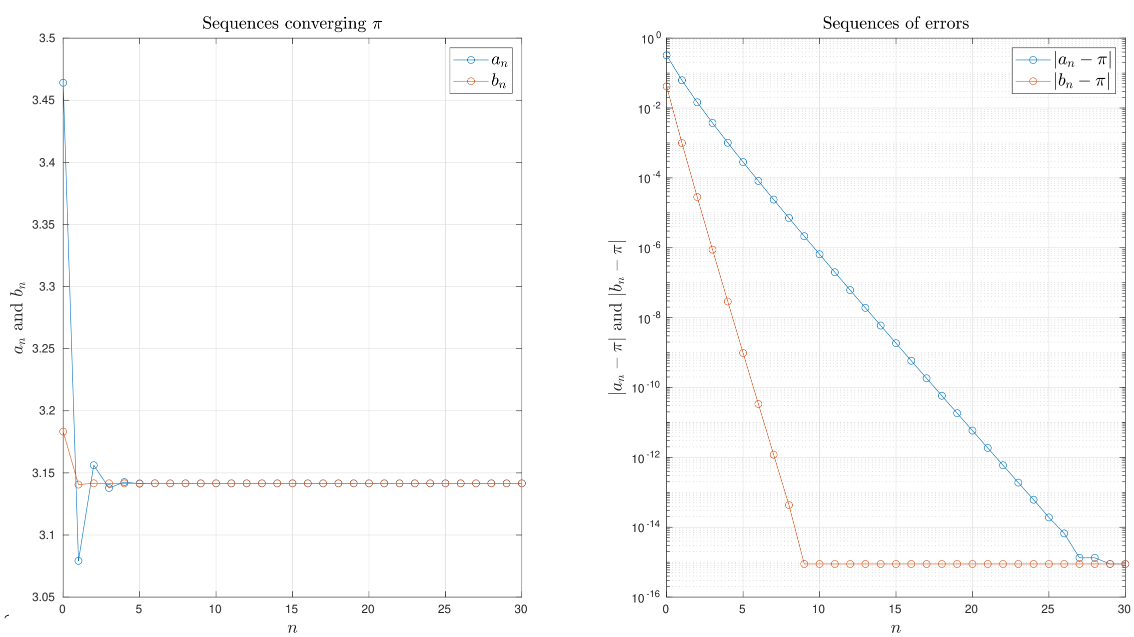

Figure 1: Example outputs for part (b) on the left and part (c) on the right.

4 Array Operations [30 points]

Load the stock price data for Company A from January 2000 through May 2018 by

```
T = load('stocks.dat');
```

The data file contains opening price, daily high/low, closing price, etc. In this problem, we will be working with the *adjusted close price* which is found on the 5-th column of `T`. Note that the stock data are ordered from most recent to oldest.

Answer each of parts (a) through (e) using ONE MATLAB statement in a single code block. Do not use a loop nor an if-statement for parts (a) through (e).

(a) Extract all adjusted close price into a single column vector with the oldest price appearing first and the most recent price appearing last. Name the column vector as `adjclose`.

Instruction. For this part, put a semicolon at the end to suppress the output, `adjclose`, since it is very long.

(b) Save the number of the data in the vector `adjclose` as `n`. Show the output.

From this point onward, *adjusted close price* will be referred to simply as *stock price*.

(c) Calculate the absolute gain[1] in stock price by taking the difference between the oldest and the most recent stock prices. Show the output.

(d) Calculate the relative gain *in percentage* by dividing the absolute gain by the initial stock price and multiplying it by 100. Show the output.

(e) Using `adjclose`, construct another column vector `monthlyAvg` whose elements are 30-day average stock prices, that is,

$$\texttt{monthlyAvg} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n/30} \end{bmatrix},$$

where a_1 is the average stock prices of the first 30 days, a_2 is the average stock prices of the second 30 days, etc.

Instruction. For this part, put a semicolon at the end to suppress the output, `monthlyAvg`, since it is very long.

[1] An absolute gain is positive if it is the case that the most recent stock price is higher than the oldest stock price; it may be negative if it is the case that the most recent stock price is lower than the oldest stock price. The word *absolute* here has nothing to do with the absolute value function $|\cdot|$.

Answer each of parts (f) and (g) in a single code block. For these parts, you are allowed to use loops.

(f) Using `adjclose`, construct the column vector `monthlyMovingAvg` whose elements are 30-day *moving average* stock prices, that is,

$$\texttt{monthlyMovingAvg} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}, \quad \text{where} \quad b_j = \begin{cases} \dfrac{1}{j} \displaystyle\sum_{k=1}^{j} p_k, & \text{if } 1 \le j \le 30 \\ \dfrac{1}{30} \displaystyle\sum_{k=j-29}^{j} p_k, & \text{if } j > 30 \end{cases}$$

in which p_k is the kth element of `adjclose`. You may use a loop. You may also use an if-statement. But to earn full mark on this part, do it without using an if-statement. Correct solutions using if-statements will receive partial credits.

Instruction. Do not show the output.

Warning! Be sure to use MATLAB functions that were introduced in class. A use of specialized functions that were not discussed in lectures, lecture notes, textbooks, or supplementary materials will not earn any credit.

(g) Using `adjclose` and `monthlyMovingAvg`, plot the stock prices and their 30-day moving averages in a single graph. Create a title, label the graph, and create a legend as shown below. Begin your code block with `clf`. At the end of your code block, include

```
xlim([n-729, n])
```

to show only the last two years of the data. Here `n` is the variable created in part (b).

Hint. You may use `1:n` as x-data for plotting, but it is not necessary.