COE 301 Introduction to Computer Programming Fall, 2024

Homework 11 – Due: 11/20/2024 11:59 pm

Problem 1. [35 points] *Chorus*. Many popular guitar effects can be simulated using signal processing techniques. In this problem, you will implement one popular effect known as chorus. A chorus effect adds a delayed version of the signal to itself to approximate the effect of the audio reaching your ears in multiple ways, as would happen if you were playing in a room and some of the sound reached your ears directly while some reflected off the walls before reaching your ears. This gives the subjective effect of making the audio feel less flat and more spacious.

Read in the file "slowguitar.wav" under "Files" tab "Lab11_data" using the Matlab function audioread:

```
[x,Fs]=audioread('slowguitar.wav');
```

Listen to the file (use headphones if you're in a public space) using:

```
sound(x, Fs);
```

Process the audio vector x and generate a new vector, where the nth element of the vector y is computed as:

$$y(n) = x(n) + x(n-D) \qquad \text{for } n-D \geq 1$$

$$y(n) = x(n). \qquad \text{for } n-D < 1$$

Where D is some delay in number of samples (an integer). We set `D=round(Fs*40e-3)`,which corresponding to a time delay `tD` of 40 milliseconds. Listen to the processed audio and comment on the results. **You must implement your solution in two ways: with loops and without loops.**

Submit your .m file as "yourLastName_hw11_pro1.m" and include your comment in the write-up.

Problem 2. [30 points] *Basic image manipulation.* As we discussed in class, a grey-scale image can be represented as a matrix. Download the file 'butterfly.gif' under "Files" tab "Lab11_data".

You may read an image file and display it as follows:

```
% Read an image
img=imread('butterfly.gif');
```

```
% By default, all MATLAB variables you created are double.
% However, you have to pay attention to data types when you read data from a file
% .gif files are stored as uint8 (unsigned 8bit integer, 0 means black and 255 means white)
% Convert to [0,1]
img = double(img)/255;
% Show it, any value larger than 1 will be displayed as white, any value lower than 0 will be displayed as black
imshow(img);
```

Use a MATLAB function to find the size of the image, and then generate (1) a brighter version of the image (2) a darker version of the image (3) an increased contrast version of the image (4) a decreased contrast version of the image. Use `imshow` to show your modified image. **You must implement this problem without any loops.**

Here a brighter image means that the amplitude of each pixel is closer to 1. A darker image means that the amplitude of each pixel is closer to 0. An image with increased contrast means bright pixels (amplitude >0.5) get brighter and dark pixels (amplitude <0.5) get darker. An image with decreased contrast means the amplitude of all pixels are getting closer to 0.5 (greyish).

Submit your .m file as "yourLastName_hw11_pro2.m" and report the image size, and the manipulated images in the write-up.

Problem 3. [35 points] *Electricity bill.* Write **a MATLAB function** called `billCalculator` that takes the monthly electricity unit usage of N families as an input vector x (the length of x is N), and returns a vector y that contains each family's total electricity bill based on the following table:

Electricity usage bracket	**Rate**
usage <= 100	$0.45/unit
100 < usage <= 250	$0.85/unit
Usage > 250	$1.45/unit

*Note: an additional fixed customer charge of $12 is added to the total bill of each family.

Call the function and calculate the electricity bill for 5000 users with their usage stored in dat_hw11_prob3.mat. **You must implement your solution in two ways: with loops and without loops. This means that you need to implement two functions one uses the loop, and one does not use any loop.**

Submit your .m file as "yourLastName_hw11_prob3.m" and report maximum, minimum, mean and variance of the monthly electricity costs derived from the data in the write-up.

Bonus Problem. [+30 points] *Binary data from the Shuttle Radar Topography Mission*. The Shuttle Radar Topography Mission (SRTM) was flown aboard the space shuttle Endeavour February 11-22, 2000. The SRTM mission collected radar data that was used to construct a terrain height map (sometimes called a Digital Elevation Model or DEM) over 80% of the Earth's surface at a spacing of 1 arc-second (1 degree = 3600 arc-seconds, ~ 30 meter pixel spacing).

All SRTM data are divided into tiles extending over 1° x 1° of latitude and longitude, in "geographic" projection. The names of individual data tiles refer to the latitude and longitude of the southwest (lower left) corner of the tile. For example, N30W098 has its lower left corner at 30°N latitude and 98° West (W) longitude. All edge pixels of the tile are centered on whole-degree lines of latitude and/or longitude. This means that the adjacent tiles share a common row or column at the overlapping edge.

Under the folder /Lab11_data, we provided four SRTM tiles (height files with the extension .hgt) over the Austin-San Antonio area. The SRTM data are stored as 16-bit binary signed integer elevations (in meters) in Motorola "big-endian" order. The data are stored in row major order, meaning all the data for the northernmost latitude, row 1, are followed by all the data for row 2, and so on. Each file contains 3601 samples by 3601 samples.

(1) Read all four SRTM tiles and merge them into a single elevation map that covers 2° x 2° of latitude and longitude over Austin-San Antonio. Orient the image so that north is up and east is right. Visualize the elevation map using imagesc(). Choose an appropriate colormap, other than the MATLAB default colormap.

(2) What is the highest elevation and what is the lowest elevation in this area?

(3) Create a "mask" image of the same dimensions as the merged map. Set the mask to 1 wherever the elevation is greater than 300 meters and set it to 0 otherwise. You must implement this task without loops.

Reference: how to read binary data in MATLAB:
https://www.mathworks.com/help/matlab/ref/fread.html#btp1twt-1-precision
Note: if you need help on the bonus problem, please reach out to Ann Chen for hints.

Submission Instructions:
There should be 4 files in your submission:

1. A write up (any type- .txt, .docx, .pdf are all fine) that contains your answers and figures to all questions in problem 1-3.
2. The .m file for problem 1.
3. The .m file for problem 2.

4. The .m file for problem 3.

Please make sure your last name is included in the filename.

Optional Short answers questions. The following questions will not be graded. You may use them for preparing your next week's quiz.

(1) Given that

```
A = [1 3;2 4]
B = [3 0;0 1]
```

What is the output of the following expressions?

```
a)    A * B
b)    A .* B
c)   max(A)
d)   sum(A)
e)   sum(B,2)
f)   sum(B(:))
```

(2) If A is a 4-by 2 matrix and B is a 5-by-2 matrix, which one is a valid MATLAB operation? `A*B` or `A*B'`?

(3) what is the output of the following block of MATLAB code?

```
n = 4;
D = diag(ones(n,1))+diag(-1*ones(n-1,1),1);
D(end,:) = [];
x = [1 2; 3 4; 6 5; -9 -8];
disp(D*x)
```

(4) There are x chickens and y rabbits in a cage. There are 72 heads and 200 feet inside of the cage. How many chickens are there and how many rabbits are there in the cage?
Write the problem as a system of two equations and solve x and y by hand. Now write a snippet of MATLAB code to solve x and y and compare your results.

(5) In linear algebra, the L1-norm of a vector is calculated as the sum of the absolute values of its elements. The L2-norm of a vector is defined as the square root of the sum of the squares of its elements. The infinity-norm of a vector is

defined as $\|x\|_\infty = \max\{|x_1|, \dots, |x_n|\}$. For more information, please check here:
https://en.wikipedia.org/wiki/Norm_(mathematics)

Write a single MATLAB function that compute the L1-norm, L2-norm and the infinity-norm of an input vector. **Note that you are not allowed to use loops or MATLAB built-in function `norm` to solve this problem.**

(6) Write a MATLAB function that returns the estimates of PI using the Monte Carlo method with n samples of (xi, yi). Both xi and yi are random numbers that are uniformly distributed between 0 and 1. Call the function and display the error for n = 10^3,10^4, 10^5, 10^6, and 10^7. **You must implement your solution in two ways: with loops and without loops.**

*****Extra optional practices*****

Short problems to practice MATLAB syntax. Create one MATLAB script, which will have several parts. Note: If you separate parts of a script with "%%", it will create a "section", which you can run separately with Ctrl-enter or clicking "Run section". **We will be giving most of the answers here simply as a way for you to practice using these new commands.**

%% Linear equations
We're going to make the code to create the plots like the one in lecture 15!

Make a new matrix A, and a vector b which will represent the two equations:

```
A = [ 4 3;-3 2];
disp('A:');
disp(A)

b = [5; 2];
disp('b:')
disp(b)

fprintf('This is like %dx + %dy = %d \n', A(1,1), A(1,2), b(1));
fprintf('and %dx + %dy = %d \n', A(2,1), A(2,2), b(2));
```

To plot the lines, we just use a little bit of algebra.
The equations are in the form "cx + dy = b", but we know that we like to plot in slope-intercept form, y = mx + b.

```
disp('If cx + dy = b, dy = b - cx, and y = b/d - (c/d)x')
cCol = A(:, 1);  % First column
dCol = A(:, 2);  % Second column
```

Start with an array of points for x, evenly spaced for the plot
We don't need that many points since it's a line (not a sine curve that needs to be smooth)

```
        x = linspace(-3, 3, 5);  % 5 points between -3 and 3
```

Remember that the dot notion does element-wise multiplication and division

```
        y = b./dCol - (cCol./dCol) .* x;
```

Now we have the two lines this represents!

```
        figure(2);
        % We can specify a specific figure number we want to bring up
        plot(x, y);
        hold on
```

Solve and plot the intersection

```
        solution = A \ b;
        plot(solution(1), solution(2), 'gs', 'MarkerSize', 10)
```

'gs' means "green square".