Math 425 Spring 2022 Project 1 due: 5PM on Fri April 22

Project Directions

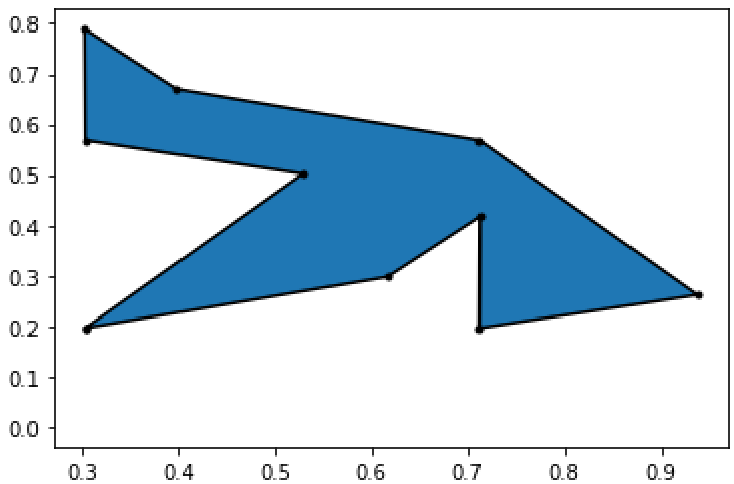
- Include a report on every group member's contribution.
- Submit the group's well commented code used for the project with instructions on how to compile and run.
- Make a **15** to **20** minute video presentation of your results.

The project consists of three problems

Problem 1

Escher in the Matrix - Tiling the plane

Consider the tile in the figure below. We'll call this the model bird.



The ten endpoints of the line segments are (0.3036, 0.1960), (0.6168, 0.2977), (0.7128, 0.4169), (0.7120, 0.1960), (0.9377, 0.2620), (0.7120, 0.5680), (0.3989, 0.6697), (0.3028, 0.7889), (0.3036, 0.5680), and (0.5293, 0.5020). Our goal is to create an Escher like artwork by tiling the model bird.

(a) Take four tiles of the model bird and fit them together according to the instructions below:

   i. Create the first tile by rotating the model bird through π radians about the point $(0.7120, 0.4320)$. Provide the matrix for the transformation in homogeneous coordinates.

   ii. Form the second tile by reflecting the model bird through the horizontal line $y = 0.6180$ and then translating this image by 0.4084 units along the x-axis. Provide the matrix for the transformation in homogeneous coordinates.

iii. To create the third tile, reflect the model bird through the vertical line $x = 0.5078$ and translate the image by 0.1000 along the y-axis. Provide the matrix for the transformation in homogeneous coordinates.

iv. Create the fourth tile by translating the model bird along the y-axis by 0.4720. Provide the matrix for the transformation in homogeneous coordinates.

v. Graph all four tiles together to produce the base pattern.

(b) Repeat the pattern of four model birds in part (a) but translate the entire pattern by $0.7441n$, $n = 1, 2, 3$ along the y-axis to produce a column of the tilings.

(c) Repeat parts (b) and translate the column of tilings by $-0.8168n$, $n = 1, 2, 3, 4, 5$ to produce the final pattern.

Problem 2

You are given part of the Wisconsin Diagnostic Breast Cancer (WDBC) dataset[1]. For each patient, you are given a vector **a** giving features computed from digitized images of a fine needle aspirate (FNA) of a breast mass for that patient. The features describe characteristics of the cell nuclei present in the image. The goal is to decide whether the cells are malignant or benign.

Here is a brief description of the way the features were computed. Ten real-valued quantities are computed for each cell nucleus:

- radius (mean of distances from center to points on the perimeter)
- texture (standard deviation of gray-scale values)
- perimeter
- area
- smoothness (local variation in radius lengths)
- compactness (perimeter^2 / area - 1.0)
- concavity (severity of concave portions of the contour)
- concave points (number of concave portions of the contour)
- symmetry
- fractal dimension (“coastline approximation” - 1)

The mean, standard error (stderr), and a measure of the largest (worst) (mean of the largest values) of each of the features were computed for each image. Thus each specimen is represented by a vector **a** with thirty entries. The domain D consists of thirty strings identifying these features, e.g. `‘‘radius (mean)"`, `‘‘ radius (stderr)"`, `‘‘radius (worst)"`,

[1](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic))

``area (mean)", and so on. Two files are provided containing data, `train.data` and `validate.data`. Also provided is the module `efficient_cancer_data`.
The procedure in `read_training_data` in the `efficient_cancer_data` module takes a single argument, a string giving the pathname of a file. It reads the data in the specified file and returns a pair $(A, \mathbf{b})$ where:

- A is a matrix whose rows correspond to the data for each patient in the data set. The elements in a row correspond to the 30 features measured for a patient.
- $\mathbf{b}$ is a vector whose domain is the set of patients and $\mathbf{b}[\mathbf{r}]$ is 1 if the specimen of patient r is malignant and it's -1 if the specimen is benign.

Use `read_training_data` to read the data in the file `train.data` into the variables A, $\mathbf{b}$.

(a) Use the QR algorithm to find the least-squares linear model for the data.

(b) Apply the linear model from (a) to the data set `validate.data` and predict the malignancy of the tissues. You will have to define a classifier function

$$C(\mathbf{y}) = \begin{cases} +1 & \text{if the prediction is non-negative} \\ -1 & \text{otherwise} \end{cases}$$

(c) What is the percentage of samples that are incorrectly classified? Is it greater or smaller than the success rate on the training data?

Problem 3

Classification of Handwritten Digits

On iLearn you will find the following data files of handwritten digits[2]:

- `handwriting_training_set.txt`: 4000 training examples of handwritten digits. Each training example is a 20 pixel by 20 pixel grayscale image of a digit reshaped into a 400-dimensional vector. Each pixel is represented by a floating point number that indicates the grayscale intensity at that location. Thus the set is a 4000 by 400 matrix.
- `handwriting_training_set_labels.txt`: This data set contains the labels of the corresponding digits in the training set. The digits "1" to "9" are labeled as they are. However, because MATLAB has no zero index, the digit zero is represented as the value ten, i.e. "0" is labeled as "10."
- `handwriting_test_set.txt`: 1000 test set of handwritten digits with the same format as the training set. Thus this set is a 1000 by 400 matrix.
- `handwriting_test_set_labels.txt`: The labels for the test set.

[2] This is a subset of the MNIST handwritten digit dataset (http://yann.lecun.com/exdb/mnist)

A. Construct an algorithm for classification of handwritten digits. Use the training set and compute the SVD of each class/digit matrix. Note that in the training set, the first 400 are examples of the digit 0, the next 400 are examples of the digit 1, etc.. Identify the unknown test digits by using the singular value decomposition of the each digit matrix. Do the classification using 5, 10, 15 and 20 singular vectors as a basis.

SPECIFIC TASKS:

i. Give a table or graph of the percentage of correctly classified digits as a function of the number of basis vectors.

ii. Check if all digits are equally easy or difficult to classify. Also look at some of the difficult ones, and see that in may cases they are very badly written.

iii. Check the singular values of the different classes. Is there evidence to support using different number of basis for different digits?

B. Implement the following **two-stage** algorithm:

In the first stage compare the unknown digit only to the first singular vector in each class. If for one class/digit the residual is significantly smaller than for the others, classify as that digit. Otherwise perform the algorithm above. Is it possible to get as good a result for this version? How frequently is the second stage necessary?