

## MATH3290 Mathematical Modeling 2017/2018

### Assignment 2

Due Date: 5pm, November 1st

You can either turn in the assignment to the assignment box in LSB or pack all your files, name it **[your name your student ID].zip** and send it to **xwang@math.cuhk.edu.hk**.

1. Construct and perform a **Monte Carlo simulation** of the popular casino game of craps. The rules are as follows: There are two basic bets in craps, pass and don't pass. In the pass bet, you wager that the shooter (the person throwing the dice) will win; in the don't pass bet, you wager that the shooter will lose. We will play by the rule that on an initial roll of 12 ("boxcars"), both pass and don't pass bets are losers. Both are even-money bets.

#### Conduct of the game:

- (a) Roll a 7 or 11 on the first roll: Shooter wins (pass bets win and don't pass bets lose).
- (b) Roll a 12 on the first roll: Shooter loses (boxcars; pass and don't pass bets lose).
- (c) Roll a 2 or 3 on the first roll: Shooter loses (pass bets lose, don't pass bets win).
- (d) Roll 4, 5, 6, 8, 9, 10 on the first roll: This becomes the point. The object then becomes to roll the point again before rolling a 7.
- (e) The shooter continues to roll the dice until the point or a 7 appears. Pass bettors win if the shooter rolls the point again before rolling a 7. Don't pass bettors win if the shooter rolls a 7 before rolling the point again.
- (f) (Optional) Compute (by hand) the probability of winning and **pass bet** and **don't pass bet**.

Write an algorithm and code it in the computer language of your choice and it is recommended to use MATLAB. Run the simulation  $n$  times ( $n = 1000, 5000, 10000$ ) to estimate the probability of winning a pass bet and the probability of winning a don't pass bet. Which is the better bet? As the number of trials increases, to what do the probabilities converge?

2. In this exercise, you will use  $k$ -means to compress an image by reducing the number of colors it contains. In a straightforward 24-bit color representation of image, each pixel is represented as three 8-bit integers (ranging from 0 to 255) that specify red, green and blue intensity values ([https://en.wikipedia.org/wiki/RGB\\_color\\_model](https://en.wikipedia.org/wiki/RGB_color_model)). One photo may contain thousands of colors, but we would like to reduce that number to a small number  $k$ . By making this reduction, it would be possible to represent the photo in a more efficient way by storing only the RGB values of the  $k$  colors present in the image. We may achieve this by the following steps.

- Step 1 Download **leslie.zip** and unpack its contents into your Matlab working directory. There are two photos in this package. The one named **leslie-big.png** contains  $718 \times 408$  pixels, while they are only  $144 \times 82$  pixels in the image named "**leslie-small.png**". We will run  $k$ -means on the  $144 \times 82$  image "**leslie-small.png**".
- Step 2 In Matlab, load the small image into your program with the following command:
- ```
A = double(imread('leslie-small.png'));
```
- This creates a three-dimensional matrix  $A$  whose first two indices identify the pixel position and whose last index represents red, green, or blue. For example,  $A(50, 33, 3)$  gives you the blue intensity of the pixel at position  $y = 50, x = 33$ .
- Step 3 To initialize, pick  $k$  colors randomly from the small image. These are the  $k$ -means in the beginning.
- Step 4 Compute  $k$  cluster centroids from this image, with each centroid being a vector of length three that holds a set of RGB values.
- Step 5 After  $k$ -means has converged, load the large image into your program and replace each of its pixels with the nearest of the centroid color you found from the small image. You can set the maximum of the number of iterations to be 100.

For  $k = 4, 8, 16$ , repeat the above steps and draw the picture respectively. Compare your picture with **leslie-big.png**.

3. In **yalefaces.zip**, there are 10 images of dimensions  $61 \times 80$  and a Matlab file **Q2.m**. In the Matlab file, we rescale each pixel value to the interval  $[0,1]$  and reshape the image to a  $(61 \times 80) \times 1$  vector. First 9 images are stored in the matrix  $X$ . Do the following in **Q2.m**:
- Following the lecture notes, perform *principle component analysis* on  $X$ .
  - Find the four largest eigenvalues and show the corresponding eigenvectors (reshape it into the dimensions of the original image).
  - Compute the relative error for data compression using only the eigenvectors in part (b).
  - Use the eigenvectors in part (b), express the 10<sup>th</sup> image (**10.gif**). Explain why the representation has bad quality.