PROJECT

ECE 6337 — FALL 2024

Instructions:

- **Academic Honesty**: By submitting your assignment, you understand that collaboration is not permitted in this assignment. You may discuss the project problems, concepts and libraries you would like to use to solve these problems. However, you may not collaborate on the report and the code - the code, results and the report you generate must be entirely your work. The University of Houston has an approved academic honesty policy that applies to all students. It is contained in the UH Student handbook available on the web. You are expected to be familiar with these policies and ensure that you comply with them.
- You may use either Python or Matlab for these problems.
- This assignment has **3 problems**
- Write your solutions in a professionally formatted, typed report that clearly explains your results and provides a discussion on the observations etc. **Convert the report to a pdf file before uploading**.
- Submit your assignment online (on Teams) as a professionally typed report, and in addition to the report pdf file, **include all source code and results/figures/discussion** used to generate results in your report.
- Submissions are due Friday, December 9, 2024 by 3pm.

Problem 1 (10 points). In this task, we will generate random vectors and perform some operations related to concepts we learned in class.

Use a function to generate multi-variate Gaussian data given the parameters of the Gaussian, such as the Matlab function *mvnrnd*. The function requires you to provide the mean vector and covariance matrix and the number of points you would like to sample from the distribution.

(a) Generate 1000 samples from a Gaussian distribution with the following parameters: $\mu = [0, 0, 0]$, and $\Sigma = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}$.

(b) Generate 1000 samples from a Gaussian distribution with the following parameters: $\mu = [0, 0, 0]$, and $\Sigma = \begin{bmatrix} 2 & 0.35 & 0.62 \\ 0.35 & 0.5 & 0.33 \\ 0.62 & 0.33 & 1 \end{bmatrix}$.

Plot the resulting scatter plots of the data you generate in parts (a) and (b) using the scatter3() function. What can you conclude about the shape of the data point-cloud as it relates to the parameters of the distribution?

Problem 2 (10 points). You will now implement your own function to carry out projection of random vectors onto principle components using concepts you learned in class. Assume that the data are zero-mean. Implement the following in your custom function (call it ProjectData.m to avoid conflicts with built-in functions):

- Assume that the input to the function is a data matrix, X, of size $d \times n$, comprising of n data samples, each a point (vector) in the $d-$ dimensional Euclidean space, $\mathbb{R}^d$. The output of the function will be a data matrix Y of size $d_2 \times n$, where $d_2 << d$.
- Estimate the Covariance Matrix of the Data (you may use the built-in Matlab function, *cov* for this).
- Perform the eigen-decomposition of the covariance matrix (you may use the built-in Matlab function, *eig* for this). The eigenvector matrix you will get out of this is of size $d \times d$.
- Ensure your eigenvectors and eigenvalues are sorted in descending order of eigenvalues - i.e., the first eigenvector corresponds to the largest eigenvalue, and so on...
- Your projection (transformation) matrix T is then obtained by selecting the first d_2 columns of the eigenvector matrix, resulting in a transformation matrix T of size $d \times d_2$
- The transformation $T : \mathbb{R}^d \rightarrow \mathbb{R}^{d_2}$ can be used to project data in the d-dimensional Euclidean space to a (lower) d_2 dimensional subspace by the simple matrix operation $Y = T'X$.

Apply your transformation function to the synthetic Gaussian data generated in problem (1a) above, projecting the data from a 3-dimensional space to a 2-dimensional space. Estimate and report the covariance matrix of the projected data Y. Also visualize the 2-dimensional scatter plot of Y and compare it to the 3-D scatter plot obtained in (1a). What observations can you make about the original data X and the projected data Y by comparing the covariance matrices and scatter plots? Explain in detail.
Repeat the exercise above with the data generated in (1b).

Problem 3 (10 points). Histogram Equalization: In image analysis/image processing, histogram equalization (and a related concept, histogram matching) is a common tool used to enhance the contrast in images with poor contrast. Consider a gray-scale image. If the intensity of the pixels in the image is treated as a random variable (say X), the histogram of intensity values would be an approximation of the pdf of the intensity R.V. In histogram equalization, the goal is often to modify the random variable (pixel intensities in this case) through a transformation, say $Y = g(X)$, such that after the transformation, the new random variable (Y) represents intensity values of a new (transformed) image whose resulting histogram (think pdf) is flat (uniform). This results in areas of the image with poor contrast to result in greater contrast. A modification of this approach, called histogram matching, instead finds a mapping $Y = g(X)$ such that the pdf/histogram of the modified data Y matches any desired "target" pdf/histogram (not just a uniform pdf). This concept is commonly used in many photo-editing apps, where such a transformation is used to modify the histogram to a desired "target" histogram.

In this problem, you are required to understand and utilize histogram equalization on the gray-scale images provided to you. Specifically,

- Review the concept of histogram equalization from any resource of your choice. Here are some suggested references (additionally, this topic would be covered in any graduate text-book on image processing):
  `http://www.sci.utah.edu/~acoste/uou/Image/project1/Arthur_COSTE_Project_1_report.html#equalization`
  `https://en.wikipedia.org/wiki/Histogram_equalization`
  `https://www.mathworks.com/help/images/histogram-equalization.html`
  `https://docs.opencv.org/4.x/d5/daf/tutorial_py_histogram_equalization.html`

- Apply the concept of histogram equalization by transforming the gray-scale images provided to you, such that the new (transformed) image has a histogram that is uniform (i.e., has been "equalized"). For this problem, you do not have to write your own histogram equalization function - you can use built-in functions in Matlab (e.g., there is a function that does just this, called *histeq*). In your report, explain what you learned about histogram equalization (how it works, particularly in the context of a function of a RV), show your results with the images provided (original image and image enhanced through histogram equalization), and interpret the results.