Programming Assignment - 3
Principal Components Analysis
EE5180
Release Date : 11 April, 2023
Submission Date: On or before 11:59:59 PM on 19 April, 2023

Notes:

1. Please use moodle discussion threads to post your doubts and check it before posting if the same question has been asked earlier.
2. **What to Submit? :** Submit a single zip file in the moodle named as PA3_Rollno.zip containing reports named "Rollno_Report.pdf" and folders containing corresponding codes. Report should contain all the plots and figures asked in the tasks.
3. Read the problem fully to understand the whole procedure.
4. Inbuilt functions for PCA must not be used. Inbuilt functions for SVD or Eigen decomposition can be used.
5. Any plagiarism/cheating will be dealt very, very strictly. You may end up with U-grade. All your reports and codes will be matched through Turnitin with each other and all previous years submissions.
6. You should thoroughly know what you are doing, and it will be asked in your viva.
7. Late submissions will be evaluated for reduced marks, and for each day after the deadline, we will reduce the weightage by 10%.

1 Problem Statement

In this assignment, you are required to implement Principal Components Analysis for face images. You're given a train and test set as .mat files. You can use the Scipy function **loadmat()** to read the .mat files. Each .mat file contains the data matrix and the labels. For the train set, the data matrix is of shape 320×10304, while for the test set, it is of shape 80×10304. Each row of the data matrix is an image of shape 112×92 flattened into a vector of length 10304. There are 40 different labels or identities. For each identity, the training set contains 8 images, while the test set contains 2 images. The data matrices in the train and test sets are not centered.
Dataset can be found at:
`https://tinyurl.com/eig5180`

2 Tasks

1. **Visualization and Pre-processing**.

   - Visualize one image for each label/identity in the training and test set. You can choose any one image for each label. Since the images have been flattened, you need to reshape them into a 112×92 shaped matrix before displaying. Display them in a 5×8 grid of subplots. For displaying images, you can use the Matplotlib function **imshow()** in Python.
   - Find the mean image for the training set. Visualize it in the same way by reshaping. Center both the train and test set using this mean image. Visualize the centered train and test set the same way as described in the previous point.

2. **Find Eigenfaces**.

   - For the centered train set, find the eigenvectors of the covariance matrix. These eigenvectors are the principal components or eigenfaces. Display the first 25 eigenfaces in a 5×5 grid of subplots. Since the eigenvectors will be 1D vectors of length 10304, you'll need to reshape them into 112×92 matrices before displaying. What can you say about the eigenfaces? What do they reveal?
   - Sort the variance along each principal component in descending order and plot them. How many principal components do you need to capture 95% of the total variance? Let's call this number 'k'. If the variance along $i-th$ principal component is given by λ_i, then k is the smallest number for which $\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq 0.95$. Here d is the maximum number of principal components. What is the value of d for this train set?

3. **Face Recognition**.

   - Using the 'k' obtained in the previous task, find the k-dimensional representation along the principal components for each train and test sample. If x_i is a column vector representing a single centered data point and $\{v_j\}$ for $j = 1$ to k are the top-k eigenvectors of the data covariance matrix, then the k-dimensional representation for x_i is

   $$z_i = (x_i^T v_1, x_i^T v_2, x_i^T v_3, ..., x_i^T v_k) \tag{1}$$

   - Using the above low-dimensional representation, for each test sample, find the sample in the train set that is closest to it in the k-dimensional subspace. If z_T is the k-dimensional representation of a centered test sample x_T, and z_i is the k-dimensional representation of the $i-th$ centered train sample x_i, then x_i is the train sample closest to x_T in the k-dimensional space if $||z_i - z_T||^2$ is the minimum across all train samples. Display the test sample (x_T) and its closest train sample (x_i). Do the labels for the test and its closest train sample match? How accurately do they match for the whole test set? Does this accuracy change if you increase or decrease k?