**Summary of Experiments**

Name: Shubham Jain

ID: AU1940315

Below is a brief description of the experiments conducted and their associated observations and learnings:

1. Finding Rank of the Matrix
   1. Experiment -
      1. This is a very simple experiment performed in Python using numpy libraries.
      2. I took a matrix in python and used numpy’s in-built method from linalg class to find out the rank of my inputted matrix.
   2. Significance of Rank of a Matrix -
      1. Rank of a matrix can be defined as the number of linearly independent columns the matrix has.
      2. The rank of the matrix gives us a fair idea about the dimension of the column space of the matrix. In simple, Laymen terms, it describes what is the minimum amount of data (those linearly independent vectors of the matrix) required to represent the entire matrix.
      3. Once we know those linearly independent vectors, we won’t require the entire matrix for operations like linear transformation as the rest of the columns can be fairly described as the linear combination of the independent column vectors.
   3. Observations –
      1. In the 3x3 matrix I took, I found the rank to be 2. This would mean that I only require to take only 2 columns and I can let go my 3rd dependent column in the matrix.
   4. Learnings –
      1. With the knowledge of the rank of the matrix, we can significantly reduce our data and the space to store the data.
      2. It will let us deal with only the most important data in the matrix.
2. Finding eigenfaces and rank of an image of a simple shape
   1. Experiment
      1. I took a simple cuboid image, and extracted the pixel matrix (each cell of the matrix contains 4 values – RGBA) from the image using in-built libraries.
      2. One can visualize the pixel matrix as a layer of matrices (RGBA), one stacked on top of the other. Each layer matrix is of size 578x900.
      3. For simplicity, I took the matrix associated with the first layer. I found its rank and first 50 eigenfaces using PCA technique.
      4. I have implemented the eigen\_and\_rank() function in my code, that takes the matrix, finds its rank and corresponding U,S and V matrices through SVD decomposition of the inputted matrix.
      5. With the help of U and V, I am generating my first 50 eigenfaces for the inputted matrix.
   2. Observations
      1. The rank of the matrix turns out to be 97. So, out of 900 column vectors, I am only interested in those 97 linearly independent vectors.
      2. Out of the 50 eigenfaces generated, only few of the eigenfaces represent most of the information and are the most important ones.
   3. Learnings
      1. The knowledge about rank has helped in reducing the dimensionality of the column space significantly.
      2. Due to this dimensionality reduction, I am able to represent my 578x900 image with just 578x97 vectors. This will be really beneficial for data compression, with minimal loss of data.
      3. Only some of the 50 eigenfaces represent useful information. The rest of the eigenfaces represent repetitive information. So, with this knowledge, I will retain only the most important eigenfaces that can help me gain most of the information about the original image.
3. Transformations on the image
   1. Rotation
      1. Experiment
         1. I rotated the image by different angles anti-clockwise.
         2. I recalculated the rank and eigenfaces of the transformed image.
      2. Observation
         1. The rank of the matrix increased to a large extent. From 97 rank in the original image, it jumped to 573 rank in the rotated image for 45 degree rotation.
         2. However, with changing the angle of rotation, the rank of the matrix also kept changing. With small angles of rotations, the rank of the matrix jumped to higher values like around 400 or 500.
         3. Also, I was able to see that now a greater number of eigenfaces are required for representing and reproducing the original image. The amount of information in each of the eigenfaces also became quite visible.
      3. Learnings
         1. With small rotations, the number of eigenfaces and hence, the data space also increased. Now, it would require a larger number of data points to represent the information.
   2. Reflection
      1. Experiment
         1. In this, I reflected the image horizontally and vertically one-by-one separately.
         2. I repeated the same procedure of finding the rank and eigenfaces of the transformed image.
      2. Observation
         1. No change in the rank of the matrix is observed.
         2. No change in the eigenfaces is observed.
      3. Learning
         1. Inference that can be derived from this experiment is that for symmetrical images, very less changes are observed in rank and eigenfaces of their reflection. This is because their reflections will only produce the very same, original image.
   3. Scaling
      1. Experiment
         1. I scaled up and scaled down my original image one-by-one.
         2. In doing so, I observed the changes in the rank of the transformed image and its eigenfaces.
      2. Observation
         1. For large amounts of scaling up and scaling down, I observed the rank of the image to decrease.
         2. With large scaling up, the clarity of the eigenfaces improved.
         3. With large scaling down, the clarity seemed to fade away.
      3. Learning
         1. Large amounts of Scaling up or scaling down leads to large decrements in rank, without affecting number of significant eigenfaces.
         2. With scaling up, each eigenface has more information to represent (as they were more clearer). With large scaling up, there can be a general increase in the number of significant eigenfaces.
         3. With scaling down, each eigenface has less information to represent. With large scaling down, there can be a general loss of number of significant eigenfaces.
   4. Random Changes to the Image (Adding noise manually)
      1. Experiment
         1. Here, I randomly chose some percent of the pixels in the first layer of the image.
         2. I added some random noise to these pixels, i.e., I changed their values to any random value between 0 and 255.
         3. Now, I observed the changes in the rank and the eigenfaces of the transformed image.
      2. Observation
         1. Firstly, the rank of the matrix increased to a large amount. But in any such random case run, the rank always used to become maximum (i.e., 578)
         2. However, the number of important eigenfaces reduced to large amounts.
      3. Learnings
         1. With very less noise added to the image, the rank and the eigenfaces of the image are mostly undisturbed. But with large amount of noise (possibly when more than 10^(-6) % pixels are affected), the eigenfaces tend to get affected, with most of them representing repetitive information.