PH6130 Data Science Analysis
Project: Principal Component Analysis

Karthik and Tarun
EE17BTECH11016 and EE17BTECH11042

Abstract

For this project, we have described the use of a statistical tool : Principal Component Analysis or simply PCA, for the recognition of patterns and compression and applied these concepts to an image. The description of Principal Component Analysis is made by means of the explanation of eigen values and eigen vectors of a matrix. The image is compressed by using different principal components, and concepts such as image dimension reduction and image reconstruction quality are explored. A qualitative comparative analysis of the compressed image using sixteen, thirty two and sixty four principal components is carried out.

Contents

1 Introduction

1.1 Definition

Principal Component Analysis or simply PCA is also known as the Hotelling transform or Karhunen-Loeve transform. It was proposed by Karl Pearson (1901) as part of factorial analysis. PCA helps in reducing a complex data set to a lower dimension, to reveal hidden, simplified structures that often underlie in the data. Dimensionality reduction simply means reducing the dimension of the feature space.

PCA is a technique for feature extraction which means, it combines the input variables of the data and drops the least important variables, while still retaining the most valuable parts of all of the variables. Each of the new variables after applyng PCA are all independent of one another based on the choosen PCA model.

In other words, it refers to the explanation of the structure of variances and covariances through a few linear or non-linear combination of the original variables, without losing a significant part of the original information.

1.2 Objective

The main objective pursued by the analysis of principal components is the representation of the numerical measurements of several variables in a space of few dimensions, where our senses can perceive relationships that would otherwise remain hidden in higher dimensions. When discarding higher dimensions, the loss of information must be minimal.

For example, Earth is a three-dimensional object, but for navigational purposes, we consider it as flat two-dimensional Map. A certain amount of information such as the continuity of the landmass is lost (as maps are discontinuous at edges) when carrying out this reduction in dimensionality. However, due to the simplification made, the loss of information is largely compensated, as many relationships, such as the directions, distance between places, are more evident when they are drawn on a plane than when done in three-dimensions.

1.3 Uses of PCA

PCA is primarily used in the following cases:

1. To reduce the number of variables, when it is difficult to identify the variables to be removed completely from consideration.
2. To ensure your variables are independent of one another.
3. When the independent variables can be made less interpretable.

1.4 Outline of the Report

The next few sections describe provide an intuitive explanation of the goal of PCA.

- The Theoretical Background section shows how PCA is related to singular value decomposition (SVD). This will help us in understanding how to apply PCA in the real world. PCA provides a foundation for many various fields of machine learning and dimensional reduction.

- We have implemented PCA for a 512px x 512px image of Lena which can be found in the Procedure section.

- The Analysis section provides the output of our image compression using PCA.

- The Conclusion section summarizes our findings after the implementation of our algorithm on the image of Lena.

- The Extensions to PCA section provides a brief explanation of other techniques which are built on PCA.

- The Code section provides the python algorithm implemented by us and gives a brief explanation of the functions used and about the code flow.

- References including the algorithm code are mentioned at the end of the report.

2 Theoretical Background

In this section, we explore the theory related to Principal Component Analysis and its application in image compression. We start by discussing some preliminaries which include Covariance matrix, Eigen values and Eigen vectors, Principal Components. The concepts which will be useful in our analysis but not mentioned in preliminaries, will be explained as and where mentioned. We then move on to describing the algorithm of PCA and its application in image compression.

2.1 Preliminaries

2.1.1 Covariance Matrix

If $X = [X_1 X_2 ... X_n]^t$ be a random column vector of dimension n, with entries as random entries. Let the mean of X_i be μ_i and variance be σ_{ii}. The covariance between two random entries, X_i and X_j is defined as,

$$Cov[X_i, X_j] = E[(X_i - \mu_i)(X_j - \mu_j)^T] = \sigma_{ij} \tag{1}$$

$$\Sigma = E[(X - \mu)(X - \mu)^T] \tag{2}$$

Σ is the covariance matrix of X, whose entries are the covariances between different entries of X, where μ is the mean matrix of X.

Consider X to be a matrix with random values as entries. The sample covariance matrix, S is given as,

$$S = \frac{n}{n-1}(X - \bar{X})(X - \bar{X})^T \tag{3}$$

where $\bar{X}$ is the sample mean column vector of X. The generalized sample variance is given by determinant of S.

2.1.2 Eigen values and Eigen vectors

Let A be a square matrix of dimension n. Let v be a vector is same dimension as A. The vector v is called an eigen vector of A, when the direction of v doesn't change when A is operated on v. The magnitude of v scales by λ, which is called the eigen value associated with the eigen vector v, but the direction remains same.

$$Av = \lambda v \tag{4}$$

From equation (4), it can be deduced that, A is an operator and when applied to v, in some sense it only changes the direction of v and scales it. The solution to the above equation would mean to find all possible λ values. This can be achieved by solving,

$$|A - \lambda I| = 0 \tag{5}$$

The concept of eigen values and eigen vectors are very useful when dealing with symmetric matrices. Let A be a symmetric matrix.

- All the eigen values are real.
- If all eigen values are positive, A is positive definite.
- The corresponding eigen vectors are orthogonal to each other, and thus the eigen vectors form an orthogonal basis for the matrix A.

2.1.3 Principal Components

Principal components are linear combinations of the random variables $X_1, X_2..., X_n$. These linear combinations represent a new coordinate system that is obtained by rotating the original reference system that has $X_1, X_2..., X_n$ as coordinate axes. The new axes represent the directions with maximum variability and provide a simple description of the structure of the covariance. Principal components depend only on Σ and the random entries can come from any distribution, not compulsorily gaussian distribution. Let $\lambda_1 > \lambda_2 > ... > \lambda_n$ be the eigen values of X in decreasing order. The linear combinations are given as

$$\begin{aligned} P_1 &= l_1^t X = l_{11}X_1 + ... + l_{1n}X_n \\ \mathrm{P}_2 &= l_2^t X = l_{21}X_1 + ... + l_{2n}X_n \\ &\vdots \\ \mathrm{P}_n &= l_n^t X = l_{n1}X_1 + ... + l_{nn}X_n \end{aligned}$$

Covariance of the linear combination vectors is given as,

$$Cov[P_i, P_j] = l_i^t \Sigma l_j^t \tag{6}$$

The linear combinations which are uncorrelated and have the greatest possible variance are the Principal components. Hence the first principal component is the linear combination with the greatest variance. The second principal component is the linear combination that maximizes the variance and is uncorrelated with the first one. Assuming normality in the data is not possible.

The principal components represent an orthogonal transformation whose coordinate axes are the axes of a multidimensional ellipsoid with axes lengths proportional to the square root of eigen values. The principal components can be seen as a translation of the origin and a rotation of the axes until they pass through the directions with greater variability.

2.1.4 Criteria for Dimensionality Reduction

The essence of the analysis of principal components are the eigenvalues and eigenvectors of the covariance matrix, as they identify the directions of maximum variability and determine the variances. Most of the variance can be explained with less than a few variables, if those few eigenvalues are much larger than the rest.

Dimensionality reduction must be applied carefully so as not to remove any useful component, but has good empirical results. A common rules states that, retain those components with variances and eigen values above a certain threshold. If we consider a case where the average value of the eigen values is one, the criterion is to keep the components associated with eigen values greater than unity and discard the rest.

2.2 Assumptions made for implementing PCA

The following assumptions are important for the simplification of our analysis. They provide a basis for us to start on solving and implementing the principal component analysis and are basic in the sense of their definitions.

- Linearity : This ensures that the problem is framed as a change of basis.
- Most important variables have high variances.
- The principal components are orthogonal

The above assumptions are accurate in terms of analysis and they do not impart any additional errors and disturbances to our implementation.

2.3 Singular Value Decomposition (SVD)

X is an arbitrary nxm matrix with rank r. Then X can be written as,

$$X = U\Sigma V^t \tag{7}$$

where V is the matrix obtained by the orthonormal eigen vectors of X^tX, U is a matrix with entries as $\frac{1}{\sigma_i}Xv_i$ and Σ is an all zero matrix with the first n diagonal matrices replaced by σ_i. Here, σ_i corresponds to the root of i^{th} eigen value arranged in decreasing order.

2.4 Principal Component Analysis and SVD

Principal Component Analysis and SVD are closely related. Let X be a nxm matrix. Let Y be an nm matrix, which is given by,

$$Y = \frac{1}{\sqrt{n}}X^t \tag{8}$$

where each column of Y has zero mean. The covariance matrix of X is given by,

$$Y^tY = \frac{1}{n}XX^t = C_x \tag{9}$$

From the previous sections, the principal components of X are given by the eigen vectors of C_x. The SVD of Y contains the matrix V, whose columns contain the eigen vectors of C_x. Hence, the columns of V are the principal components of X. This also means that the V spans the row-space of Y and hence spans the column space of X. The conclusion of the section is that, finding the principal components is equivalent to finding an orthonormal basis which spans the column space of X.

3 Procedure

3.1 Image of Lena

Image of Lena is a very famous image and is most often used by many engineers and researchers for experiments in image compression. Using the python matplotlib.pyplot library, this image, which is taken as a .jpg file, is converted to a matrix.

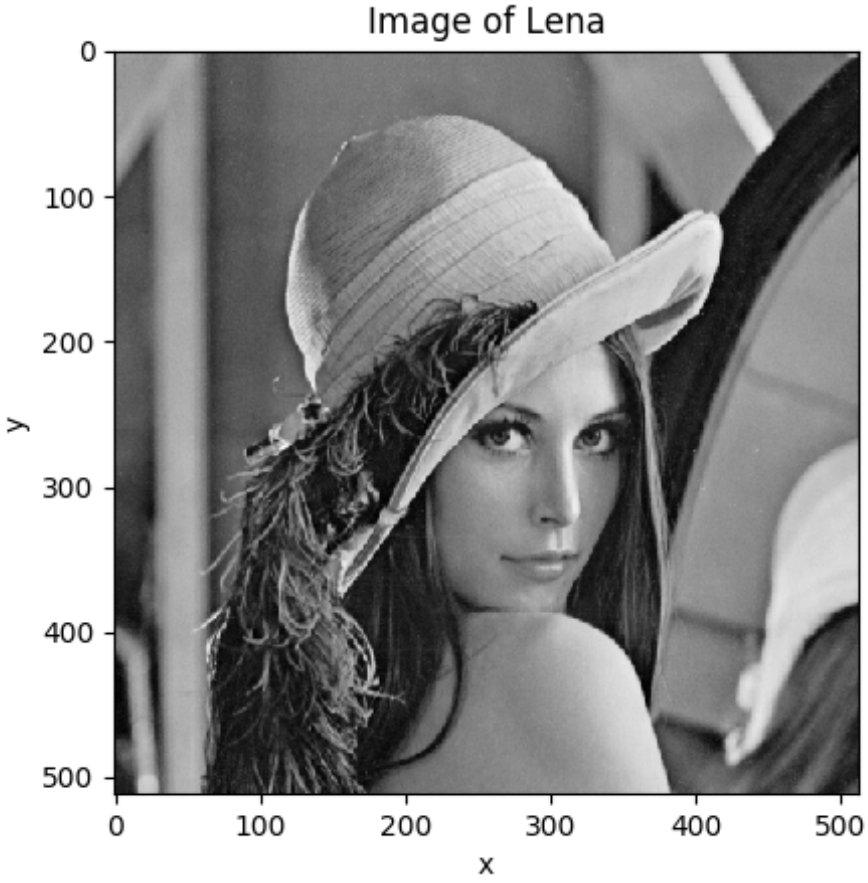


Figure 1: Image of Lena

The dimensions of the obtained matrix is determined by the pixel ratio of the image. Since the image has 512 x 512 pixels, the dimensions of the obtained matrix is 512 x 512. For further analysis, we refer this input matrix as Input Image matrix or X.

3.2 Covariance Matrix

The main step in dimensionality reducing is creating a new orthogonal basis for the image. Hence, we find the covariance matrix of our Input Image matrix which can be used to create the required orthogonal basis. From equation (1), we find the covariance matrix by subtracting the mean vector matrix (each row is the mean of the corresponding row of X) to make the mean of each row of the Input Image matrix 0. The covariance matrix is given by,

$$S = \frac{1}{n-1}(X - \bar{X})^T.(X - \bar{X}) \tag{10}$$

where, X is the Input Image matrix, and $\bar{X}$ is the mean vector matrix of the Input Image matrix. The components of the mean vector are the mean of each row of Input Image matrix. The dimension of S is 512 x 512.

3.3 Eigen values and Eigen vectors

The 512 pairs of eigen values and the corresponding eigen vectors of S are found using the numpy module of python. The eigen values are denoted by λ_i and the corresponding eigen vector is $\hat{e}_i$. These eigen value and eigen vector pairs are ordered accordingly in decreasing order of the magnitude of eigen values to ensure that we take the components of highest variability while reducing the dimension.

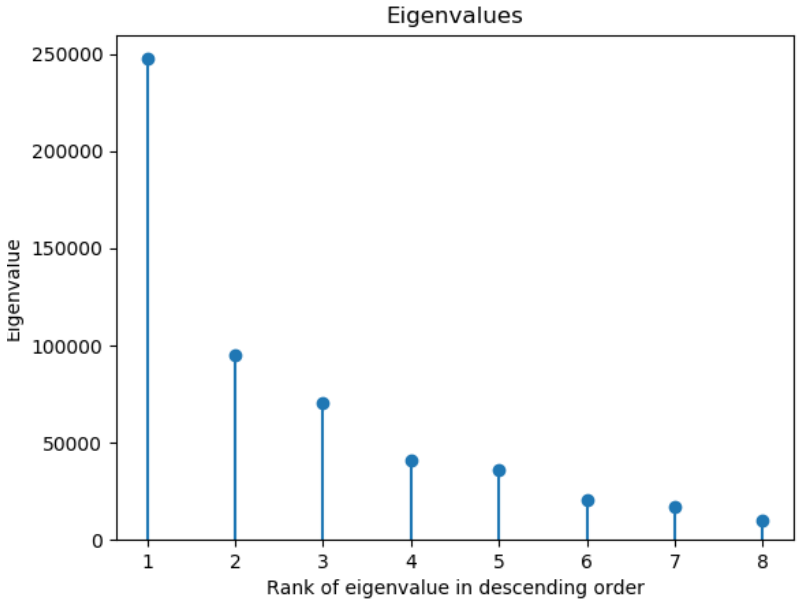


Figure 2: Eigen values

From figure (2), the first eigen value is nearly triple the second eigen value after the rearrangement. This shows that the most of the data is concentrated in the eigen vector direction corresponding to this eigen value. Hence, the first principal component contains maximum variability after the rearrangement.

3.4 Building of principal components

From the theory explained in section **2.1.3**, the principal components are thus given by,

$$\hat{y}_j = \hat{e}_{1j}x_1 + ... + \hat{e}_{512j}x_n \tag{11}$$

With this, we have built the new orthogonal basis of dimension 512.

3.5 Change of Basis

The basis of the Input Image matrix is taken as B. Also, the new orthogonal basis of the eigen vectors $\{\hat{e}_1,...,\hat{e}_{512}\}$ be B'.

Therefore the transformed matrix Y is obtained by,

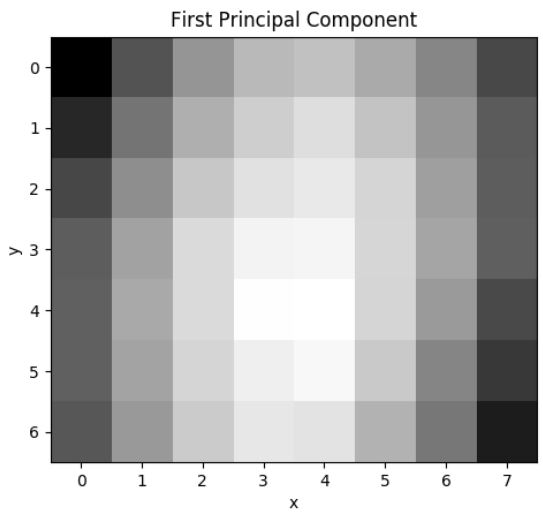
$$Y = X * [\hat{e}_1^t, ..., \hat{e}_{512}^t] \tag{12}$$

where, X is the Input Image matrix. The final step of image compression is to choose only first few vectors of B' which contribute to the variability.

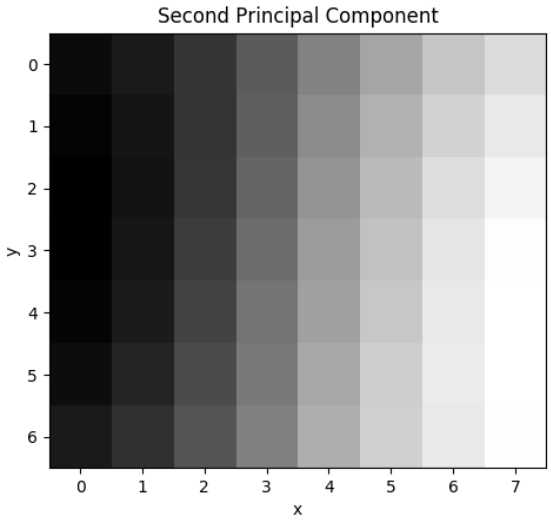
3.6 Principal Components

Here, we show how the principal components actually contain the variability in the image. We take an orthogonal basis of R^{64} and plot the images of the first three principal components as shown in figure (3).

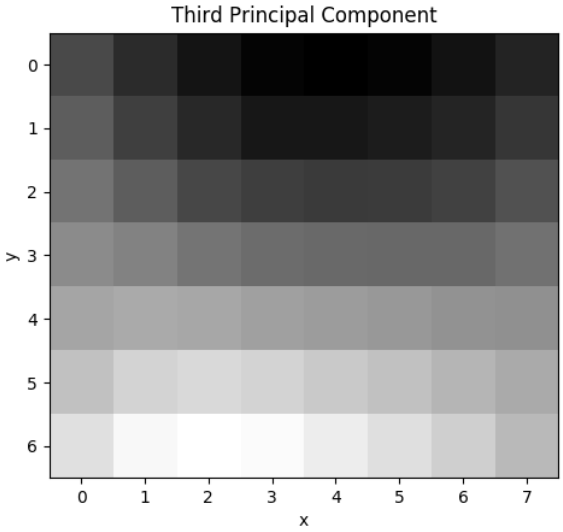
The reason we took 64 dimensions in our basis and not 512 is that, plotting the principal component of 512 basis can't give us a square image. Hence, comparing different principal components won't be as trivial as when compared to the 64 dimension basis.



(a) First Component



(b) Second Component



(c) Third Component

Figure 3: Principal components

3.7 Compressing the Image

After constructing the orthogonal basis, for compressing the image we need to take the first few principal components. This is done by creating a subset of the eigen vectors. The subset must be taken only from the beginning of the eigen vector matrix. Here, the orthogonal basis containing the eigen vectors was sorted based on the eigen values.

The dimension of the subset will determine the number of principal components taken to compress the image. The subset can be taken by using range function in python.

$$Compressed\_Image = (X - \bar{X}) * subset(S) \tag{13}$$

For obtaining the compressed image matrix, we multiply the subset of the eigen vector matrix to the covariance matrix S obtained in section **3.2**. We have taken 16, 32 and 64 dimension subsets to compress the image, and the corresponding compressed images are presented in section **4**.

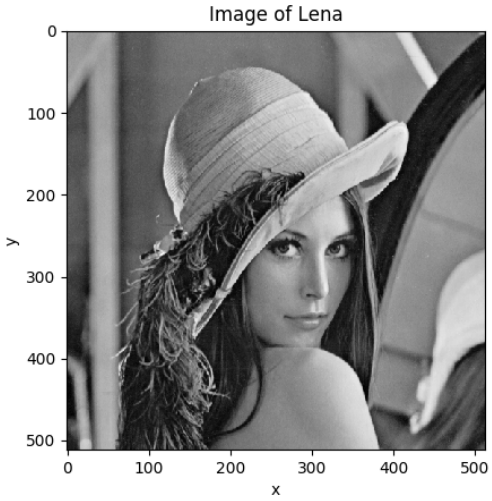
3.8 Reconstruction of Compressed Image

The compressed image matrix obtained in equation (13) is then plotted using imshow module of the matplotlib.pyplot library of python. Since we have taken a image which contains only one color component, the obtained image is grayscale in color.

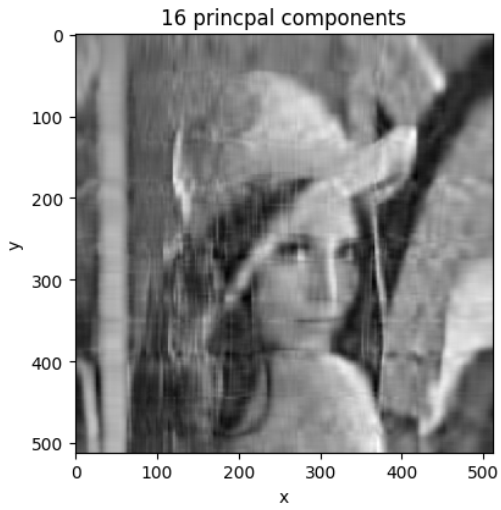
We plot three different images, which are obtained by taking the first 16, 32 and 64 principal components respectively. The change in the sharpness of the image for different principal components is clearly visible.

4 Analysis

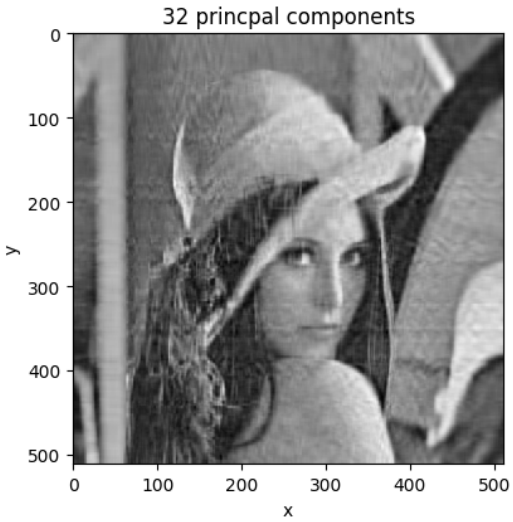
We have seen that, there are total of 512 principal components in the original image. This is due to the fact that the covariance matrix is 512 x 512. After performing PCA on the image of lena by taking the first sixteen, thirty two and sixty four principal components respectively, we obtained the following compressed images.



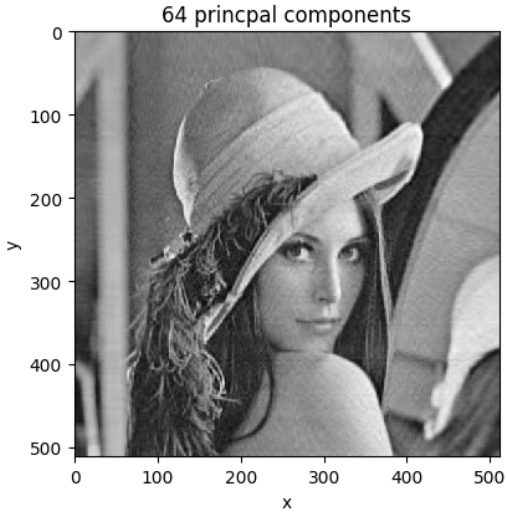
(a) Original Image



(b) Compressed image with 16 Components



(c) Compressed image with 32 Components



(d) Compressed image with 64 Components

Figure 4: Compressed Image

- As we can see from the above figure, the percentage of the variability increases by increasing the number of principal components.

- This increase in percentage is very small when the number of principal components are more, but the nuances added to the image are sufficiently remarkable.

- These nuances can be realised in the form of,
  - sharpness of the image.
  - smoothness of the contours.
  - marking of the tones.

- The proportion of variance retained is the sum of the eigen values of the features kept, divided by the sum of the eigen values of all features. We notice that just by taking the first 32 of 512 the principal components in decreasing magnitudes of eigen values, we obtain 95% of the original image information.

$$\frac{\sum_{i=1}^{16} \lambda_i}{\sum_{i=1}^{512} \lambda_i} = 63\% \tag{14}$$

$$\frac{\sum_{i=1}^{32} \lambda_i}{\sum_{i=1}^{512} \lambda_i} = 95\% \tag{15}$$

$$\frac{\sum_{i=1}^{64} \lambda_i}{\sum_{i=1}^{512} \lambda_i} = 99\% \tag{16}$$

- It is easy to operate on the compressed image as its variance is concentrated only in few directions and hence it reduces computation load by many factors. Therefore compressed image can be used as a substitute for any future analysis on the image with little error.

5 Conclusions

- We have given a basic introduction on the statistical technique of Principal Component Analysis and have shown its application in image compression.

- We started by discussing about covariance matrix, eigen values and eigen vectors, Principal components and SVD and have shown how these techniques are incorporated in applying PCA for image compression. We proceeded to describing the assumptions made in our analysis.

- Principal components are linear combinations of random variables, but with a constraint that they represent a new coordinate system that is obtained by rotating the original reference system. A simple description of the structure of the covariance is provided by the new axes. They represent the directions with maximum variance or variability.

- We have shown that the Principle Components can be used as a measure of how each variable is associated with one another, i.e. the directions of highest variances and the importance of these directions.

- We have applied the Principal Component Analysis for image compression and have shown our analysis on image of lena. The image was compressed by using different number of principal components and the proportion of variance or the percentage of similarity between the image and compressed image were reported.

- We have shown that by considering more principal components, the error in the compressed image is almost negligible.

- Principal Component Analysis can be applied to any data set and hence, is completely non-parametric. It requires no parameters to tweak and no past information on how the data was recorded.

- However, PCA fails when there is bias in the input data. As PCA works on the principal of taking dimensions of highest variance, hence if most of the data samples are from one part of the population, PCA will result in a component that mostly spans this population. For example, the data obtained by tracking the pendulum can be rendered as a single variable data θ, the angle between the axis of pendulum and the pendulum. PCA would fail in this case.

6 Extensions to PCA

6.1 Principal Component Regression

One major extension to Principal Component Analysis is Principal Component Regression. In this method, we take the matrix Y which is described in section **3.5** and regress it on a matrix Z without dropping any of its vectors, where Z is given by,

$$Z = \begin{bmatrix} I_{M,M} & O_{M,dim-M} \\ O_{dim-M,dim} & O_{dim-M,dim-M} \end{bmatrix} \tag{17}$$

where M is the number of principal components, dim is the dimension of the covariance matrix, I is the identity matrix and O is the zero matrix.

6.2 Kernel Principal Component Analysis

Another extension to PCA is Kernel Principal Component Analysis. In this method,by using a kernel, the original linear operations of PCA are performed in a reproducing kernel Hilbert space. Kernals such has gaussian kernal, polynomial kernal and hyperbolic tangent kernal are used in Kernal PCA.

PCA was done assuming the linearity of the principal components. Kernal PCA considers the non-linearity of the principal components and extracts them based on their mapped hilbert space.

7 Code

This code is also available in References [1].

7.1 Python Script

```
import matplotlib.pyplot as plt
import numpy as np

img = plt.imread("img.jpg")
img = np.array(img)

def pcaComp(img, NumPC):
        cov = img - np.mean(img , axis = 1)
        eVal, eVec = np.linalg.eigh(np.cov(cov))
        l = eVec.shape[1]
        idx = np.argsort(eVal)
        idx = idx[::-1]
        eVec = eVec[:,idx]
        eVal = eVal[idx]
        pc= NumPC
        if pc<l or pc>0:
                eVec = eVec[:, range(pc)]
        out = np.dot(eVec.T, cov)
        out = np.dot(eVec, out) + np.mean(img, axis = 1).T
        out = np.uint8(np.absolute(out))
        return out, eVal

out1, eVal = pcaComp(img, 16)
out2, eVal = pcaComp(img,32)
out3, eVal = pcaComp(img,64)

plt.figure(1)
x = np.linspace(1,8,8)
(markers, stemlines, baseline) = plt.stem(x,eVal[0:8])
plt.setp(baseline, visible=False)
plt.title('Eigenvalues')
plt.ylabel('Eigenvalue')
plt.xlabel('Rank of eigenvalue in descending order')
plt.ylim(0,260000)
plt.show()
plt.subplot(2,2,1)
plt.imshow(img, cmap='gray')
plt.title('Original Image')
plt.xlabel('x')
```

```
plt.ylabel('y')
plt.subplot(2,2,2)
plt.imshow(out1, cmap='gray')
plt.title('16 Princpal Pomponents')
plt.xlabel('x')
plt.ylabel('y')
plt.subplot(2,2,3)
plt.imshow(out2, cmap='gray')
plt.title('32 Princpal Components')
plt.xlabel('x')
plt.ylabel('y')
plt.subplot(2,2,4)
plt.imshow(out3, cmap='gray')
plt.title('64 Princpal Components')
plt.show()
plt.xlabel('x')
plt.ylabel('y')
```

7.2 Explanation

- In the code, we first read the image and store it in a 2-dimensional array. We then define a function, which takes in 2 parameters, the input array, and the number of principal components to consider while reconstructing the image, and the function returns a 2-dimensional array of the reduced image.

- Inside the function, we first calculate the covariance matrix of the image matrix, using numpy. We then calculate the eigen values and eigen vectors of the covariance matrix. We sort the eigen vectors based on decreasing order of eigenvalues.

- Next, we truncate the eigen vectors matrix to take only the first few. The number of eigen vectors we take is equal to the number of principal components we want to use. Finally, we compute the output image array by multiplying the reduced eigen vectors matrix with the covariance matrix as described in the procedure section.

- Finally, we computed the output image three times, taking the first 16 Principal Components, first 32 Principal Components and the first 64 Principal Components respectively. These were displayed along with the original image, and the difference in quality of the picture can be seen. As number of Principal Components increases, the output gets closer to the original picture.

References

[1] https://github.com/taruntejreddych/DSA-Project-Code

[2] A Tutorial on Principal Component Analysis, by Jonathon Shlens.

[3] https://arxiv.org/pdf/1403.2877.pdf

[4] https://www.intechopen.com/books/statistics-growing-data-sets-and-growing-demand-for-statistics/application-of-principal-component-analysis-to-image-compression

[5] https://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.html

[6] https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial

[7] https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c

[8] https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202

[9] https://learnche.org/pid/latent-variable-modelling/principal-components-regression

[10] http://www.cs.haifa.ac.il/ rita/uml_course/lectures/KPCA.pdf