JSS MAHAVIDYAPEETHA

JSS SCIENCE AND TECHNOLOGY UNIVERSITY

(Formerly Sri Jayachamarajendra College of Engineering)

Digital Compression Technique (EC821)

Event - 4

Report for Long Term Event on

“Image Compression Using PCA"

Submitted by

Sl. No.	USN	Name
1	01JST16EC019	Anup Desai

Submitted to

Professor D R Pavithra
Assistant Professor
Department of Electronics and Communication
JSS Science and Technology University
Mysuru 570006

DEPARTMENT OF ELECTRONICS AND COMMUNICATION
JSS SCIENCE AND TECHNOLOGY UNIVERSITY
JSS TECHNICAL INSTITUTIONS CAMPUS
MYSURU 570006
(2019-2020)

ABSTRACT

High-resolution image is referred as high-dimensional data space as each image data is organized into two-dimensional pixel values in which each pixel consists of its respective RGB bits value. The representation of image data poses a challenge to sharing image files over Internet. The lengthy image uploading and downloading time has always been a ma-jor issue for Internet users. Apart from data transmission problem, high-resolution image consumes greater storage space. Principal Component Analysis (PCA) is a mathemati-cal technique to reduce the dimensionality of data. It works on the principal of factoring matrices to extract the principal pattern of a linear system. This project aims to evalu-ate the application of PCA on digital image feature reduction and compare the quality of the feature reduced images with difference variance values. As a result of summarizing the preliminary literature, dimension reduction process by PCA generally consists of four major steps: (1) normalize image data (2) calculate covariance matrix from the image data (3)perform Single Value Decomposition (SVD) (4) find the projection of image data to the new basis with reduced features.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies successful completion of any task would be imcomplete without the mention of people who made it possible. We would like to express our deep sense of gratitude and indebtedness to the following people. We would like to thank Assistant Professor D R Pavithra for giving us an opportunity to do this project. We would also like to thank our Principal and Head of Dept. of Electronics and Communication Engineering for the opportunity.

Contents

1 Introduction 1

2 Literature Survey 3

3 Motivation 4

4 Objective 4

5 Proposed System 5

5.1 Working Principle 5
5.2 Software Requirements 7
5.3 Flow diagram 8

6 Implementation 9

7 Results 11

8 Advantages and Disadvantages 12

8.1 Advantages 12
8.2 Disadvantages 12

9 Conclusion 13

9.1 Future Scope 13

References 14

List of Figures

1 PCA of a multivariate Gaussian distribution centered at (1,3) with a standard deviation of 3 in roughly the (0.866, 0.5) direction and of 1 in the orthogonal direction. 1
2 Feature Scaling and Mean Normalisation. 5
3 Formulation of PCA. 5
4 2D to 1D 6
5 3D to 2D. 6
6 Flow diagram 8
7 Octave Code 9
8 Gray-scale Image 11
9 Recovered Image 11
10 Editor image 11
11 Recovered Image 12

1 Introduction

Principal component analysis (PCA) is mostly used as a tool in exploratory data analysis and for making predictive models. It is often used to visualize genetic distance and relatedness between populations. PCA can be done by eigenvalue decomposition of a data covariance (or correlation) matrix or singular value decomposition of a data matrix, usually after a normalization step of the initial data. The normalization of each attribute consists of mean centering – subtracting each data value from its variable's measured mean so that its empirical mean (average) is zero. Some fields, in addition to normaliz-ing the mean, do so for each variable's variance (to make it equal to 1). The results of a PCA are usually discussed in terms of component scores, sometimes called factor scores (the transformed variable values corresponding to a particular data point), and loadings (the weight by which each standardized original variable should be multiplied to get the component score). If component scores are standardized to unit variance, loadings must contain the data variance in them (and that is the magnitude of eigenvalues). If compo-nent scores are not standardized (therefore they contain the data variance) then loadings must be unit-scaled, ("normalized") and these weights are called eigenvectors; they are the cosines of orthogonal rotation of variables into principal components or back.

Given a collection of points in two, three, or higher dimensional space, a "best fitting" line can be defined as one that minimizes the average squared distance from a point to the line. The next best-fitting line can be similarly chosen from directions per-pendicular to the first. Repeating this process yields an orthogonal basis in which different individual dimensions of the data are uncorrelated. These basis vectors are called principal components, and several related procedures principal component analysis (PCA).

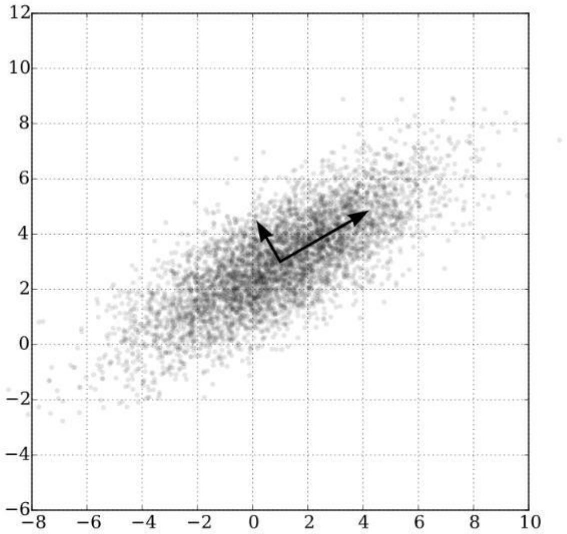


Figure 1: PCA of a multivariate Gaussian distribution centered at (1,3) with a standard deviation of 3 in roughly the (0.866, 0.5) direction and of 1 in the orthogonal direction.

PCA is the simplest of the true eigenvector-based multivariate analyses. Often, its operation can be thought of as revealing the internal structure of the data in a way that best explains the variance in the data. If a multivariate dataset is visualised as a set of coordinates in a high-dimensional data space (1 axis per variable), PCA can supply the user with a lower-dimensional picture, a projection of this object when viewed from its most informative viewpoint.[citation needed] This is done by using only the first few principal components so that the dimensionality of the transformed data is reduced.

PCA is closely related to factor analysis. Factor analysis typically incorporates more domain specific assumptions about the underlying structure and solves eigenvectors of a slightly different matrix.

PCA is also related to canonical correlation analysis (CCA). CCA defines coor-dinate systems that optimally describe the cross-covariance between two datasets while PCA defines a new orthogonal coordinate system that optimally describes variance in a single dataset.

2 Literature Survey

1. 1D-PCA, 2D-PCA to nD-PCA by Hongchuan Yu and Mohammed Bennamoun School of Computer Science and Software Engineering, University of Western Australia, Perth, WA6009, Australia

In this paper, we first briefly reintroduce the 1D and 2D formsoftheclassicalPrincipalComponentAnalysis(PCA). Then, the PCA technique is further developed and extended to an arbitrary n-dimensional space. Analogous to 1D- and 2D-PCA, the new nD-PCA is applied directly to n-order tensors (n 3) rather than 1-order tensors (1D vectors) and 2-order ten-sors (2D matrices). In order to avoid the difficulties faced by tensors computations (such as the multiplication, general transpose and Hermitian symmetry of tensors), our pro-posed nD-PCA algorithm has to exploit a newly proposed Higher-Order Singular Value Decomposition (HO-SVD). To evaluate the validity and performance of nD-PCA, a series of experiments are performed on the FRGC 3D scan facial database.

2. Color image compression using PCA and backpropagation learning by Clifford Clausen and Harry Wechsler. Department of Computer Science, George Mason University, Fairfax, VA 22030, USA

The RGB components of a color image contain redundant information that can be reduced using a new hybrid neural-network model based upon Sanger’s algorithm for representing an image in terms of principal components and a backpropagation algorithm for restor-ing the original representation. The PCA method produces a black and white image with the same number of pixels as the original color image, but with each pixel represented by a scalar value instead of a three-dimensional vector of RGB components. Experimental results show that as our hybrid learning method adapts to local (spatial) image character-istics it outperforms the YIQ and YUV standard compression methods. Our experiments also show that it is feasible to apply training results from one image to previously unseen images

3 Motivation

Computer images consist of large data and hence require more space to store in the memory. The compressed image requires less storing space of memory and less time to transmit. Principal component analysis of daylight illumination shows that 99% of the variance can be accounted for with only three principal components. Furthermore, 85% of variance can be represented with only two color channels. Today, color images are rendered on computer monitors using only three primary colors, usually red, green and blue (RGB). Therefore, a straightforward way to compress a color image is to compress each of the red, green and blue gray-scale images that compose the image.

The RGB components of a color image contain redundant information that can be reduced using a new hybrid neural-network model based upon Sanger's algorithm for representing an image in terms of principal components and a backpropagation algorithm for restoring the original representation. The PCA method produces a black and white image with the same number of pixels as the original color image, but with each pixel represented by a scalar value instead of a three-dimensional vector of RGB components. With the above as motivation we implement the project.

4 Objective

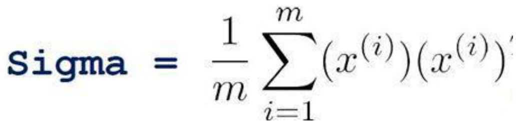
The main objectives of this project is to implement image compression appro-priately using PCA (Principal Components Algorithm) to appreciable compression ratio, using GNU Octave.

5 Proposed System

5.1 Working Principle

The compression used here is based on Principal Component Algorithm [2]: Principal Component Algorithm(PCA):

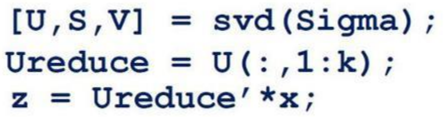
Any real matrix can be decomposed uniquely. An image is actually a matrix of numbers whose elements are the intensity value of corresponding pixels of the image. PCA is defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.[3].Data Pre-processing is done by computing Feature Scaling and Mean Normalisation and the formula is given by.



$$\text{Sigma} = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})'$$

Figure 2: Feature Scaling and Mean Normalisation.

We know need to reduce data from n-dimensions to k-dimensions thus after computing the above "covariance matrix" we compute eigen vectors of the above ma-trix ’Sigma’ by applying Singular Value Decomposition.There we obtain U matrix which contains ’n’ training examples.But, we require only ’K’ dimensional matrix,as such we consider only K examples obtained from the "U" matrix which is labeled as U-reduce ma-trix.Lastly we obtain our required matrix that is "Z" matrix by element wise multiplication of the transpose of U-reduce matrix and our training set matrix "X".This is the basic work-ing principle behind the working of PCA.



```
[U,S,V] = svd(Sigma);
Ureduce = U(:,1:k);
z = Ureduce'*x;
```

Figure 3: Formulation of PCA.

For simple example these below images portray the conversion of 2D data to 1D and 3D data to 2D respectively.

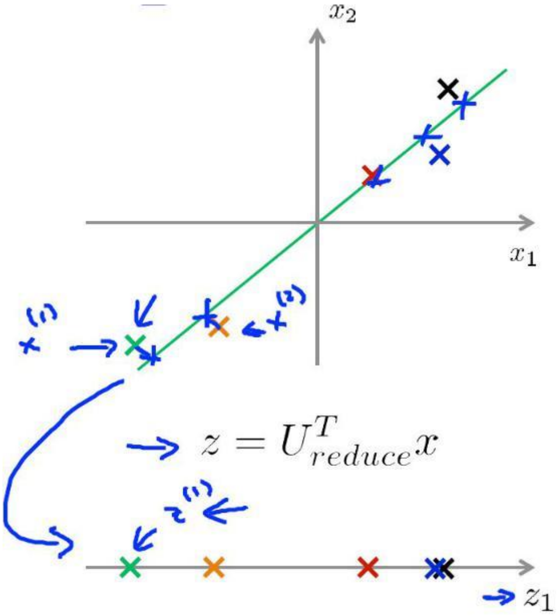


Figure 4: 2D to 1D .

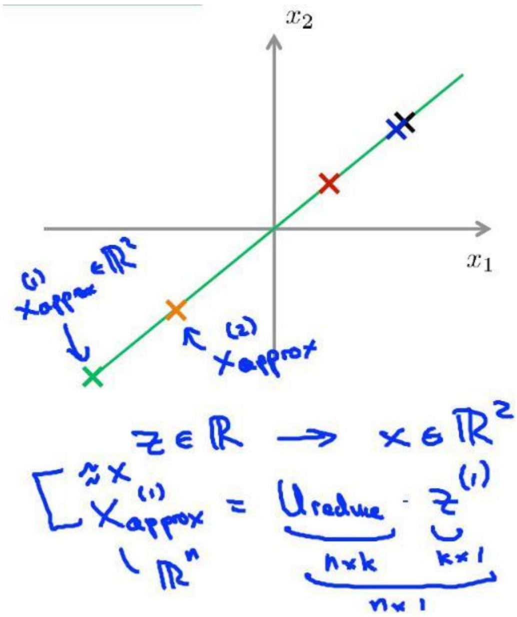

x_2
x_1
$x_{approx}^{(1)} \in \mathbb{R}^2$
$x_{approx}^{(2)}$
$z \in \mathbb{R} \rightarrow x \in \mathbb{R}^2$
$\approx x$
$x_{approx}^{(1)} = \underbrace{U_{reduce}}_{n \times k} \cdot \underbrace{z^{(1)}}_{k \times 1}$
$\mathbb{R}^n$
$n \times 1$

Figure 5: 3D to 2D.

The same technique is applied to the image in this project.The data set here is the RGB pixels of the image which are obtained in the size of (length*breadth*height,3).In Machine learning it is always converted into a matrix of (length*breadth*height*3,1).Then PCA is applied onto the data set and a compressed image is obtained which has the same characteristics of the original,but is less in size.

5.2 Software Requirements

It is said that the best way to learn is by trying it yourself. In this project we have used Octave for implementation.

GNU Octave is a high-level language, primarily intended for numerical computations. It provides a convenient command line interface for solving linear and nonlinear problems numerically, and for performing other numerical experiments using a language that is mostly compatible with Matlab. It may also be used as a batch-oriented language.

Octave was originally conceived (in about 1988) to be companion software for an undergraduate-level textbook on chemical reactor design being written by James B. Rawlings of the University of Wisconsin-Madison and John G. Ekerdt of the University of Texas. We originally envisioned some very specialized tools for the solution of chemical reactor design problems. Later, after seeing the limitations of that approach, we opted to attempt to build a much more flexible tool.

Octave has extensive tools for solving common numerical linear algebra prob-lems, finding the roots of nonlinear equations, integrating ordinary functions, manipulat-ing polynomials, and integrating ordinary differential and differential-algebraic equations. It is easily extensible and customizable via user-defined functions written in Octave’s own language, or using dynamically loaded modules written in C++, C, Fortran, or other lan-guages.

GNU Octave is also freely redistributable software. You may redistribute it and/or modify it under the terms of the GNU General Public License (GPL) as published by the Free Software Foundation.

Octave was written by John W. Eaton and many others. Because Octave is free software you are encouraged to help make Octave more useful by writing and contributing additional functions for it, and by reporting any problems you may have.

5.3 Flow diagram

Flow diagram for the compression process including preprocessing is given in Figure 6.

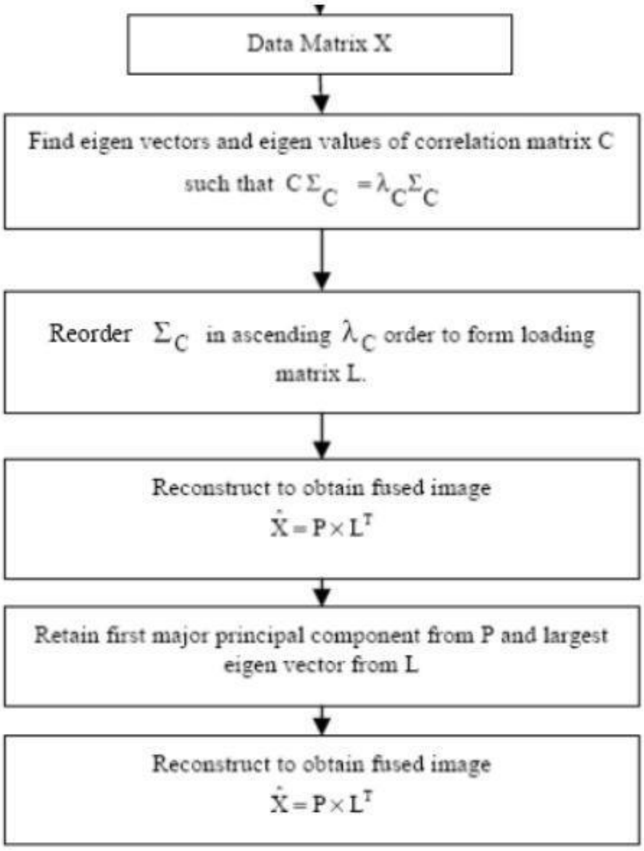
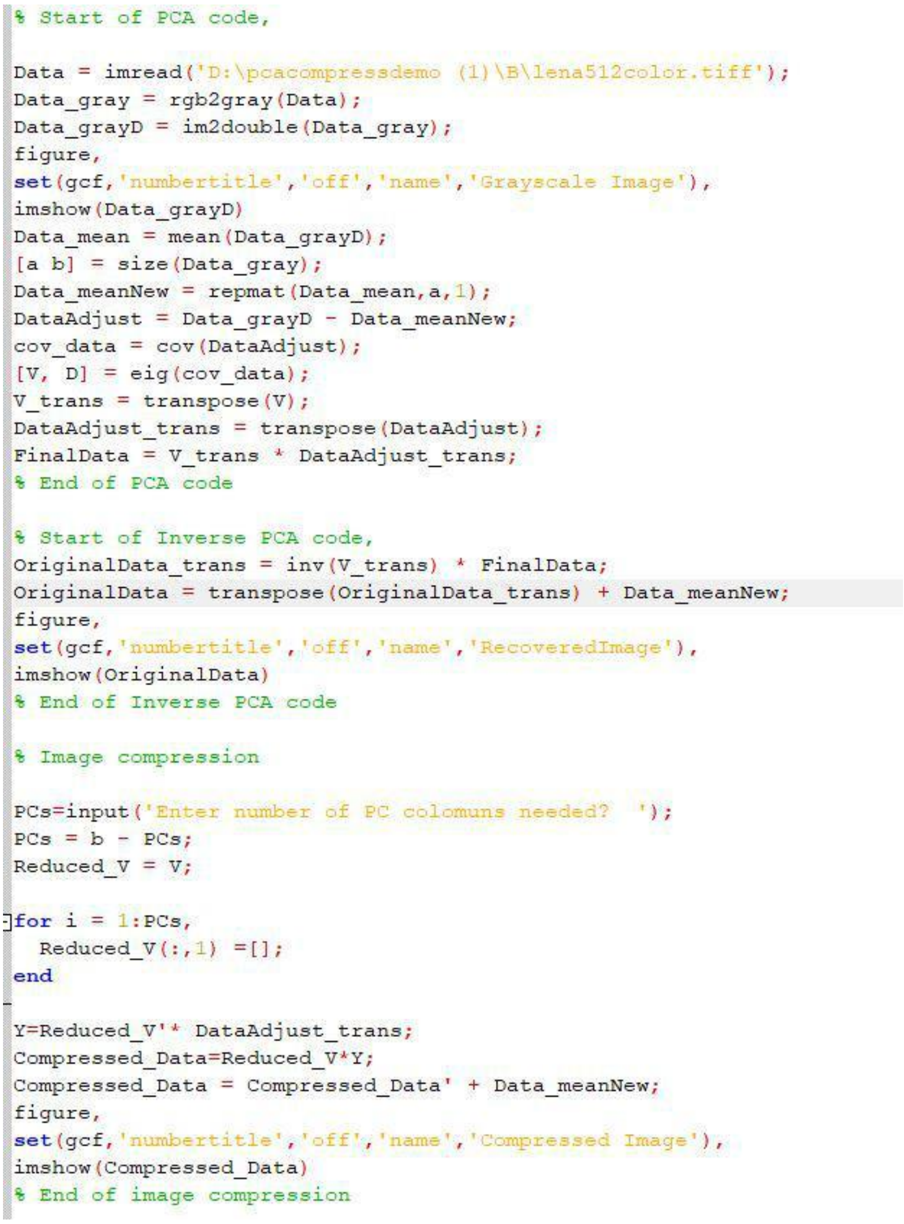

Data Matrix X
Find eigen vectors and eigen values of correlation matrix C such that $C\Sigma_C = \lambda_C \Sigma_C$
Reorder Σ_C in ascending λ_C order to form loading matrix L.
Reconstruct to obtain fused image $\dot{X} = P \times L^T$
Retain first major principal component from P and largest eigen vector from L
Reconstruct to obtain fused image $\dot{X} = P \times L^T$

Figure 6: Flow diagram

6 Implementation

Data compression plays a crucial role in various modern digital photography and video applications. Its main goal is to reduce irrelevant and/or redundant image data while keeping relevant information intact as much as possible. There are numerous inter-national as well as industry standards available that are based on either lossless or lossy compression of images. While a lossy compression standard such as JPEG offers a tradeoff between the image quality and the file size, in lossless compression methods – run length encoding, LZW, Chain codes and others – no information reduction takes place, but the image file size may be prohibitively large.

We have used Octave GNU and CLI to implement the Image Compression.The following code shiows the Implementation of the PCA.



```
% Start of PCA code,

Data = imread('D:\pcacompressdemo (1)\B\lena512color.tiff');
Data_gray = rgb2gray(Data);
Data_grayD = im2double(Data_gray);
figure,
set(gcf,'numbertitle','off','name','Grayscale Image'),
imshow(Data_grayD)
Data_mean = mean(Data_grayD);
[a b] = size(Data_gray);
Data_meanNew = repmat(Data_mean,a,1);
DataAdjust = Data_grayD - Data_meanNew;
cov_data = cov(DataAdjust);
[V, D] = eig(cov_data);
V_trans = transpose(V);
DataAdjust_trans = transpose(DataAdjust);
FinalData = V_trans * DataAdjust_trans;
% End of PCA code

% Start of Inverse PCA code,
OriginalData_trans = inv(V_trans) * FinalData;
OriginalData = transpose(OriginalData_trans) + Data_meanNew;
figure,
set(gcf,'numbertitle','off','name','RecoveredImage'),
imshow(OriginalData)
% End of Inverse PCA code

% Image compression

PCs=input('Enter number of PC colomuns needed?  ');
PCs = b - PCs;
Reduced_V = V;

for i = 1:PCs,
  Reduced_V(:,1) =[];
end

Y=Reduced_V'* DataAdjust_trans;
Compressed_Data=Reduced_V*Y;
Compressed_Data = Compressed_Data' + Data_meanNew;
figure,
set(gcf,'numbertitle','off','name','Compressed Image'),
imshow(Compressed_Data)
% End of image compression
```

Figure 7: Octave Code

We have reduced data from n-dimensions to k-dimensions computing thecovariance¨ matrix" we compute eigen vectors of the above matrix ’Sigma’ by applying Singular Value Decomposition.There we obtain U matrix which contains ’n’ training examples.But, we require only ’K’ dimensional matrix,as such we consider only K examples obtained from the "U" matrix which is labeled as U-reduce matrix.Lastly we obtain our required matrix that is "Z" matrix by element wise multiplication of the transpose of U-reduce matrix and our training set matrix "X".

Principal components analysis (PCA) is an established multivariate statistical tool that linearly transforms a number of possibly correlated variables into a smaller number of new variables, known as principal components. Since a digital image can be regarded as a two – or more – dimensional function of pixel values and represented as a 2D (grayscale image) or 3D (color image) data array, PCA can be performed on such an m x n matrix. The presentation has demonstrated how to implement PCA for image compression on different digital images and, in particular, how the choice of the number of extracted PCs affects the image compression ratio and consequently the image quality.

7 Results

Image compression using PCA was done on the image in Figure 8. The image is initially RBG coloured but it is converted to gray-scale for compression purposes.

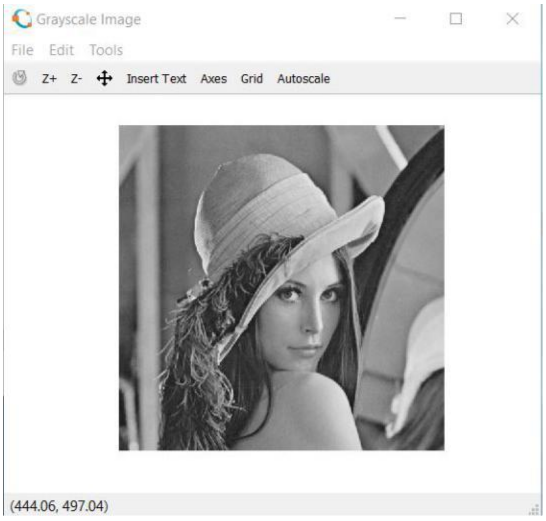


Figure 8: Gray-scale Image

The gray-scale image from Figure 8 is then made to go through compression and the image that is recovered is shown in Figure 9.

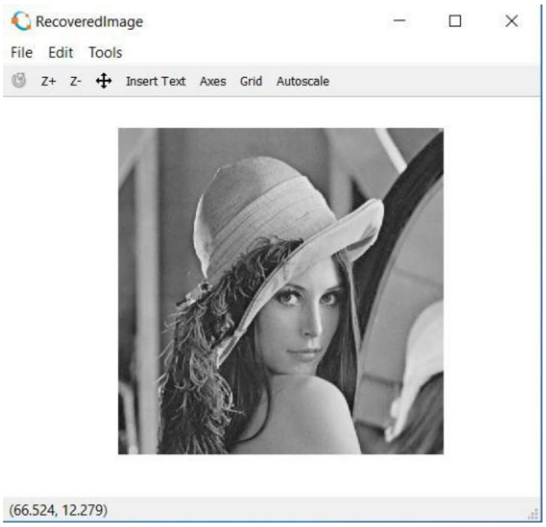
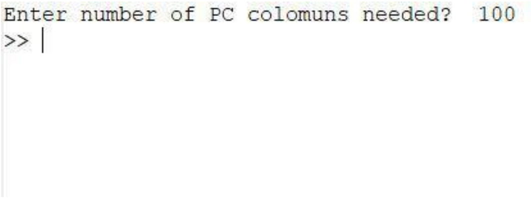


Figure 9: Recovered Image

After the grayscale process, the number of columns needed for pca is selected. 100 columns are selected in this case.



```
Enter number of PC colomuns needed?  100
>> |
```

Figure 10: Editor image

Once the number of coulmns is selected, the compression of image using PCA takes place. The final compressed image is recovered and is shown in figure 11. The size of the recoverd image is lesser than the original image and yet for the human eye there is not much difference in the original and recovered image.

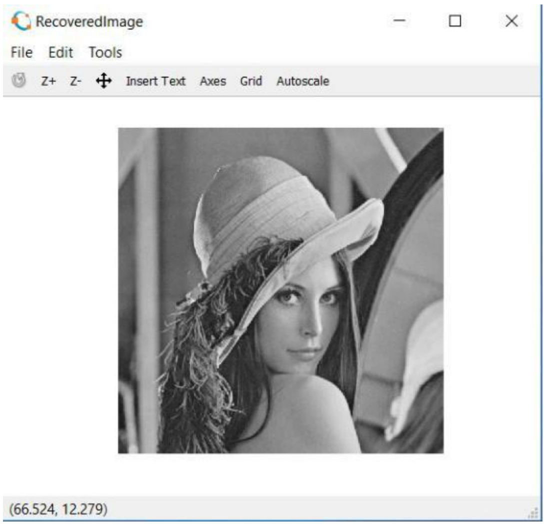


Figure 11: Recovered Image

8 Advantages and Disadvantages

8.1 Advantages

The advantages of this compression are:

1. Removes Correlated Features.
2. Improves Algorithm Performance.
3. Reduces Overfitting.
4. Improves Visualization.

8.2 Disadvantages

The disadvantages of this compression are:

1. Independent variables become less interpret able.
2. Data standardization is must before PCA.

9 Conclusion

The main idea of principal component analysis (PCA) is to reduce the dimension-ality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. The same is done by transforming the variables to a new set of variables, which are known as the principal components (or simply, the PCs) and are orthogonal, ordered such that the retention of variation present in the original variables decreases as we move down in the order. So, in this way, the 1st principal component retains maximum variation that was present in the original components. The principal components are the eigenvectors of a covariance matrix, and hence they are orthogonal. Importantly, the dataset on which PCA technique is to be used must be scaled. The results are also sensitive to the relative scaling. As a layman, it is a method of summarizing data. Imagine some wine bottles on a dining table. Each wine is described by its attributes like colour, strength, age, etc. But redundancy will arise because many of them will measure related properties. So what PCA will do in this case is summarize each wine in the stock with less characteristics. Intuitively, Principal Component Analysis can supply the user with a lower-dimensional picture, a projection or "shadow" of this object when viewed from its most informative viewpoint.

9.1 Future Scope

PCA using different PCA processing parameters using an unsupervised non-targeted approach as well as a knowledge-based targeted approach. Furthermore, different normalisation and scaling algorithms have been applied to the PCA dataset. The scope and limitation of the various PCA parameters are discussed with respect to the ability to differentiate between samples of different groups,or different processing parameters and with respect to the information content of the PCA analysis on a molecular level. We could show that while distinction between different groups of samples can be successfully carried out independent of PCA parameters employed, identifying molecular markers rationalising differentiation between sample groups varies significantly between PCA parameters and requires careful choice as well as critical evaluation.

References

[1] Color image compression using PCA and backpropagation learning Cliford Clausen, Harry Wechsler

[2] 1D-PCA, 2D-PCA to nD-PCA Hongchuan Yu and Mohammed Bennamoun School of Computer Science and Software Engineering, University of Western Australia, Perth, WA6009, Australia

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

[4] Image Compression using PCA and Improved Technique with MLP Neural Network Vilas Gaidhane , Vijander Singh , Mahendra Kumar Department of Electronics Communication, G.L.A Institute of Technology and Management, Mathura, U.P, India