DATE: 2/1/2022

TO: Linear Algebra Teaching Staff

FROM: Students of EMPx17 Under Leadership of "Hisham Ahmed Abosena", 7616

Participants:

Ahmed Sobhy Al-Safy, 7377

Aly Gamal El-Din Fadel, 7454

Hisham Ahmed Abosena, 7616

Mahmoud Haytham Mahmoud, 7560

Mohamed Hussein Aly, 7498

Mustafa Abd-Al-Atty Ismaeil, Sultan 7900

Sohail Waleed El-Ganagy, 7372

RE: Project about using eigenvalues and eigenvectors of a matrix to create a simple face recognition algorithm.

Abstract: Linear algebra is used in a variety of fields in our current lives. We can use matrix operations, some matrix theories and Principal component analysis method in order to obtain a simple face recognition algorithm. This method was first proposed by “Matthew Turk” and “Alex Pentland” by 1991. Different fields of mathematics were utilized in this project like statistics and linear algebra. From the statistics department we utilized some general concepts like the mean and covariance. While from the linear algebra department, eigenvectors, eigenvalues and simple matrix operations were utilized. All of this was digitized using the MATLAB language on modern computers which increased the scope of the algorithm massively

1 INTRODUCTION

1.1 BACKGROUND

In our project, we are presenting a practical application of eigenvalues/vectors and matrix operations to apply Principal Component Analysis (PCA) to program a simple image recognition algorithm. Greyscale images of similar dimensions are used.

1.2 SCOPE

In our projects we apply a variety matrix operations and theories in order to produce the algorithm.

Namely, Matrix means, Matrix addition/subtraction, Eigenvectors and Eigenvalues, Matrix transpose.

In this project we implement the algorithm using the online octave website: https://octave-online.net/

2 DISCUSSION

2.1 STEPS

1. At the beginning we input our database of grey scale images with dimension m*n (ALL IMAGES MUST HAVE SAME DIMENSIONS) into octave online and reshape them as vectors. (Code 2.1.1)

```
for j=1:Database_Size
    image_read=imread(['person' num2str(j) '.pgm']);
    [m,n]=size(image_read);

P(:,j)=reshape(image_read,m*n,1
```

CODE 2.1.1: IMAGE INPUT

2. We compute the mean of the input images $X_{mean} = \frac{1}{N}\sum_{i=1}^{N} X_i$ using the following code. (Code 2.1.2)

```
mean_face=mean(P,2);
```

CODE 2.1.2: MEAN CALCULATION

3. Subtract the mean vector from each of the column vectors of our array of images and create a matrix “P” whose columns are the result of the mean subtraction. (Code 2.1.3)

```
P=double(P);
P=P-mean_face*ones(1,Database_Size);
```

CODE 2.1.3: MEAN SUBTRACTION

4. We use the multiplication of the transpose of matrix “P” (PPT) and itself respectively to solve the eigen value problem for the images. (Code 2.1.4)

5. We use the resulted eigenvalues to obtain the eigen vectors of the product of transpose of “P” and itself (PTP). Then we use multiply the matrix “P” and the result eigen vector to obtain the eigenvector of the product of matrix “P” and its transpose (PPT). (Code 2.1.4)

6. We compile the resulted eigenvectors into a set of components constituting of the largest covariance between the data. (Code 2.1.4)

```
[Vectors,Values]=eig(P.'*P);%% P 1000*1 Pt

EigenVectors=P*Vectors;

for j=2:Database_Size;

    if j==2


EigenFaces=reshape(EigenVectors(:,j)+mean_face
,m,n);

    else

    EigenFaces=[EigenFaces
reshape(EigenVectors(:,j)+mean_face,m,n)];

    end

end
```

CODE 2.1.4: EIGEN VECTOR CALCULATION

7.Using PCA, we identify the important information about the image and compare ONLY the most important information in order to implement the face recognition algorithm. (Code 2.1.5)

```
U=reshape(image_read,m*n,1);
NormsEigenVectors=diag(Products);
W=(EigenVectors'*(double(U)-mean_face));
W=W./NormsEigenVectors;
U_approx=EigenVectors*W+mean_face;
image_approx=uint8(reshape(U_approx,m,n));
figure;
```

CODE 2.1.5: FACE RECOGNITION ALGORITHM

2.2 DETAILS

1. We enter images of same dimensions (100*100 in our case) so that we can reshape them into dimensionally equal matrices which facilitates dealing with them. (Number of images in our case: N = 30)

2. We compute the mean by using matrix addition operation and then using matrix scalar multiplication to divide by their number.

3. We use matrix addition operations to subtract the resulted matrix mean from each of the image matrices in order to obtain the covariance of each matrix from the mean images. And then we use the resulted covariance matrices to form a matrix “P” of dimensions n*m x 30 (100^2 x 30 in our case).

4. PCA principal component analysis is used here to minimize the dimensions used in the evaluation and the data loss without affecting the accuracy of the program. It uses a covariance matrix descripted later to represent the face subspace entered by the images. Then it uses 2 orthogonal vectors to represent this eigenspace as they will give best results as shown in fig(2.2.2) and (2.2.3)

5. I-We use covariance matrix which is the product of PTP instead of PPT in order to facilitate dealing with the eigenvalue matrix as the dimensions of the resulting matrix of PTP is 30 x 30 while thedimensions of the product matrix of PPT is 100^2 x 100^2.

II- We used the fact that only at most N-1 eigenvalues of matrix PPT are non-zero as the covariance matrix equal the sum of all entered images minus (N * mean_face) so should equal (0 matrix) we are solving a homogenous system, a trivial zero solution is always present which is deducted from our set of eigenvalues.

III- The non-zero eigenvalues of PTP are the same of the non-zero eigenvalues of PPT (Shown in Fig. 2.2.1)

What is the relationship between u_i and v_i ?

$$A^T A\, v_i = \mu_i v_i$$

$$A\, A^T A\, v_i = \mu_i A v_i$$

$$C A\, v_i = \mu_i A v_i \quad [\text{since } C = A\, A^T\ ]$$

$$C\, u_i = \mu_i A v_i \qquad \text{where, } u_i = A\, v_i \text{ Thus,}$$

$C = A\, A^T$ and $L = A^T A$ have the same eigenvalues and their eigenvectors are related as follows: $u_i = A\, v_i$

Note 1: $C = A\, A^T$ can have upto N^2 eigenvalues and eigenvectors.

FIG 2.2.1 EIGENVALUE/VECTOR EQUALITY PROOF

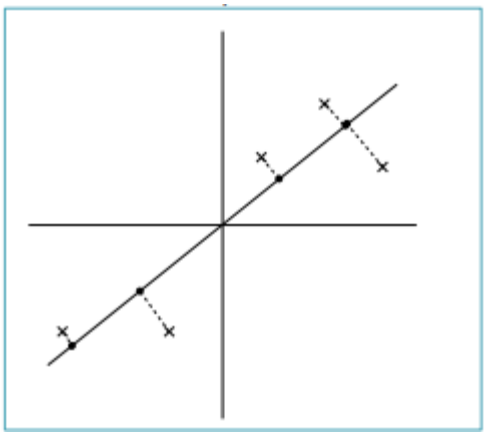


FIG 2.2.3 PCA to minimize the variance

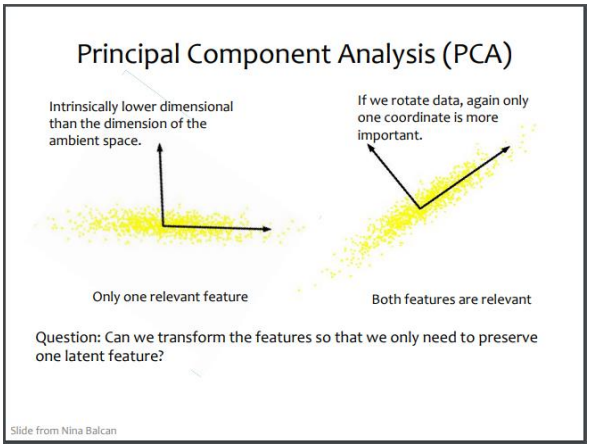


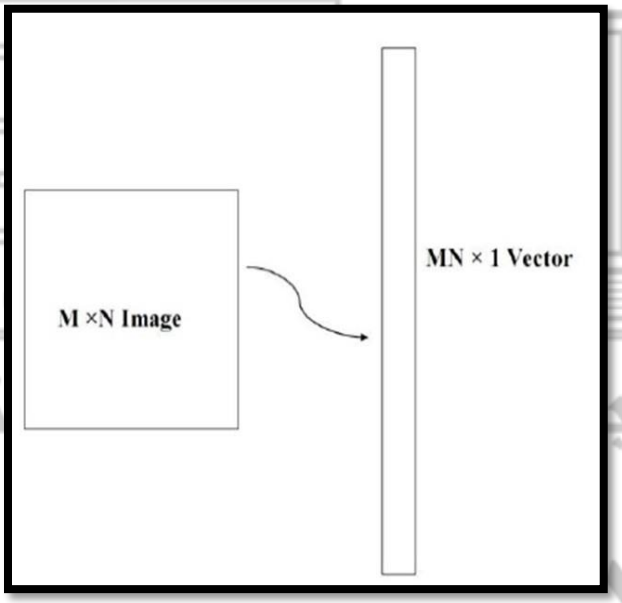
FIG 2.2.2 PCA used in getting the dimensions and orthonormal

6. The eigen vectors correspond to the set of components of largest data covariance between the database images.

7. We start by entering the image required to check(image_read), reshaping it into the form of a column vector and subtracting the result vector(mean_face) to be in the same data shape entered into the PCA algorithm. We need to compute the projection of this photo using the result eigenfaces stored in database. We construct the image through additionof scalar ($Wn = (image\_read)^T * Eigneface_n$) reshaping of every eigenface then adding the mean. It is saved in the (image_approx) variable and then reshaped and displayed alongside the entered image.

2.3 RESULTS

By compiling the code in the first (3) steps we get a variable “P” of the matrices of input images as column vectors shown in Codes (2.1.1, 2.1.2, 2.1.3) and calculating mean in Code (2.1.2) and display it as in Fig (2.3.2)



FFIG 2.3.1 IMAGE RESHAPING

FFIG 2.3.2 MEAN FACE

In Code (2.1.4) we calculate the eigen faces. We get the output as in Figs (2.3.3, 2.3.4) and it is displayed as in Fig (2.3.5).

```
line 0: warning: Terminal canvas area too small to
hold plot.
Check plot boundary and font sizes.
```

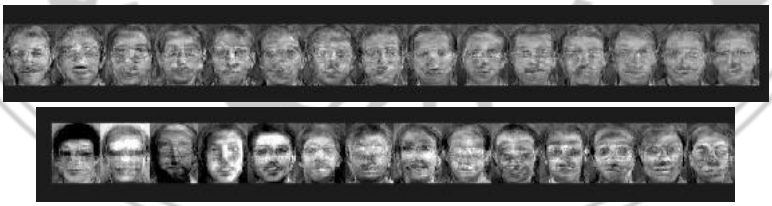
FFIG 2.3.2 CODE OUTPUT

```
Products =

 Columns 1 through 6:

   3.2141e-23  -3.3219e-09   3.4401e-09  -1.7900e-09
5.8917e-10  -1.0461e-09
  -3.3219e-09   3.3070e+06   5.0431e-10  -5.3120e-09
4.7476e-10   6.5629e-09
   3.4401e-09   5.0431e-10   3.4283e+06   8.3674e-10
5.7389e-10  -3.5652e-10
  -1.7900e-09  -5.3120e-09   8.3674e-10   3.5295e+06
2.9904e-09   7.8362e-09
   5.8917e-10   4.7476e-10   5.7389e-10   2.9904e-09
3.7663e+06  -1.8517e-09
  -1.0461e-09   6.5629e-09  -3.5652e-10   7.8362e-09
-1.8517e-09   4.1778e+06
   8.0474e-10  -6.0263e-09  -1.4461e-10  -7.0759e-10
-9.4587e-10   2.5832e-09
   8.8487e-10   5.6716e-09   2.7285e-10   5.5879e-09
```

FFIG 2.3.3 CODE OUTPUT



FFIG 2.3.4 EIGEN FACES

After entering the 2 altered images in code (2.1.5) and got results as shown in fig (2.3.5, 2.3.6).

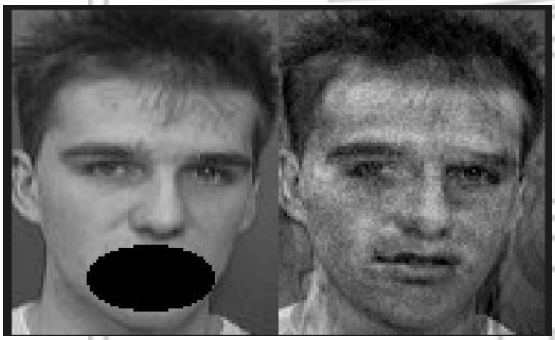


FIG 2.3.5 IMAGE RECOGNITION EX

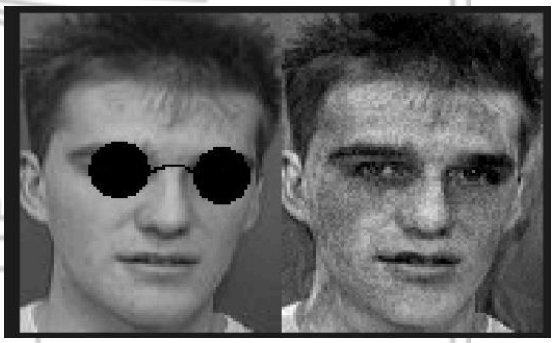


FIG 2.3.6 IMAGE RECOGNITION EX

3 CONCLUSION

As presented in this project, linear algebra has a great number of applications in our daily lives. We chose one of those daily life applications in the form of eigenvector/values to represent the images in the form of a long number instead of dealing with whole matrices. We also use eigenvectors to calculate an approximate image using the data provided in the program database input done in the first step.

This technology has important applications in different fields of industry as the power of the “PCA” (Principal Component Analysis) algorithm is the work horse behind a variety of modern techniques in data mining, including face recognition algorithms.

جامعة الإسكندرية
ALEXANDRIA
UNIVERSITY

EIGEN VECTORS: FACE RECOGNITION ALGORITHM

PSUEDOCODE:

1- Input images.

2- Reshape into matrices.

3- Calculate Mean of matrices.

4- Subtract mean from image matrices.

5- Use subtracted matrices as columns of formed matrix “P”.

6- Calculate eigenvalues and eigenvectors of matrix “PPT” to obtain eigen faces.

7- Input altered image.

8- Reshape altered image.

9- Compare reshaped image to eigen faces.

10- Obtain approximate image using addition of eigen faces.