ECE3093 Assignment 1 Part B

Written by Kun Zhang (22701478)

# Task 3: Age Estimation

## 1) Modified MATLAB code

clear all;close all;clc

% prepare the data for face recognition

train\_dir = 'F:\ECE3093 Optimisation Estimation and Numerical Methods\Assignments\Assignment 1\ages\Train'; % change this to wherever the unzipped Face Recognition\Train folder has been saved

img\_dir = dir(fullfile(train\_dir, '\*.jpg'));

%Extract age from filenames

for i=1:max(size(img\_dir));

img\_age(i)=str2num(img\_dir(i,1).name(5:6));

end

img\_num = length(img\_dir);

person\_num = img\_num; % each person has one image, so #persons = #images

traindata = zeros(img\_num, 69\*45); % #images = img\_num, image size = 69\*45.

% read all images into the matrix traindata:

for i=1:img\_num

img = imread(fullfile(train\_dir, img\_dir(i).name));

img = double(img)/255; % normalize to double in [0,1], 255 values for each pixel

traindata(i,:) = img(:);

end

% train eigenface:

[eigenV,SCORE,LATENT] = princomp(traindata); % alternatively could be done using the svd(traindata) function

% eigenV: eigen vectors.

% SCORE: projection vectors of training data on the eigen vectors.

% LATENT: eigenvalues

% test:

test\_dir = 'F:\ECE3093 Optimisation Estimation and Numerical Methods\Assignments\Assignment 1\ages\Test'; % change this to wherever the unzipped Face Recognition\Test folder has been saved

img\_dir = dir(fullfile(test\_dir, '\*.jpg'));

img\_num = length(img\_dir);

%Extract age from test file names

for i=1:max(size(img\_dir));

test\_age(i)=str2num(img\_dir(i,1).name(5:6));

end

testdata = zeros(img\_num, 69\*45); % # images = img\_num, image size = 69\*45.

for i=1:img\_num

img = imread(fullfile(test\_dir, img\_dir(i).name));

img = double(img)/255; % normalize to double in [0,1]

testdata(i,:) = img(:);

end

mae = zeros(1,max(size(eigenV))); %construct mae

for i=1:100

pc\_num = i; % we keep the first i Principal Components:

eigenfaces = eigenV(:, 1:pc\_num);

gallery = SCORE(:, 1:pc\_num);

%Modelling the relationship

W=[gallery.^2,gallery,ones(size(gallery,1),1)]\transpose(img\_age);

w1=W(1:pc\_num);

w2=W((pc\_num+1):pc\_num+pc\_num);

b=W(end);

% project onto the eigenfaces:

meanV = mean(traindata);

features = (testdata-repmat(meanV,size(testdata,1),1))\*eigenfaces;

features = transpose (features);

est\_age = transpose(w1)\*(features.^2)+transpose(w2)\*features+b;

%Mean Absolute Error

mae(i) = sum (abs(test\_age-est\_age))/max(size(test\_age));

%my face, rescale to 45 by 69

myface=double(imresize(rgb2gray(imread('F:\ECE3093 Optimisation Estimation and Numerical Methods\Assignments\Assignment 1\my\_face\_for\_math.png')),[45 69]));

%imshow(myface)

myface=myface/255; %normalise myface

myface\_data(1,:)=myface(:); %put the data in a row

myfeatures = (myface\_data-repmat(meanV,size(myface\_data,1),1))\*eigenfaces; %project the data on eigenvectors

myfeatures = transpose(myfeatures);

my\_est\_age = transpose(w1)\*(myfeatures.^2)+transpose(w2)\*myfeatures+b; %estimate age

end

plot(mae(1:100))

title('Mean average error vs. k value');

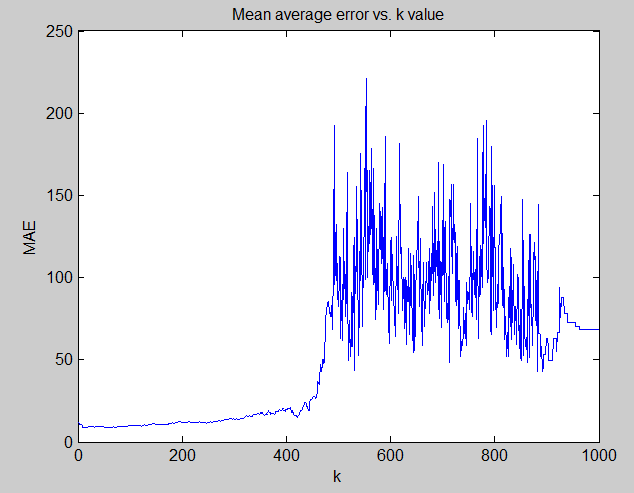
xlabel('k');

ylabel('MAE')

[minV minidx]=min(mae(1:100));%minV gives the lowest MAE which corresponds to the k value minidx

## 2) The best k value

The best k value is the one which gives the lowest mean absolute error (MAE). By iteratively incrementing the value of k starting from 1, we have the MAE versus k plot for the first 1000 k value as shown below.



Further investigation into MAE (using min()) gives us the lowest MAE which is 8.54 when k is 14. Therefore, the best k value is 14 and its corresponding MAE is 8.54.

## 3) My image and Estimate Age

* My processed image



* My estimate age is 18.77, which deviates from my actual age (23.5) by 4.73. The error is thus 4.73.

## 4) Discussion

One of the limitations is that the eigenanalysis of the covariance matrix is time consuming for large matrices. In addition, eigenface method is susceptible to factors that might change the images’ variability, including different lightings, expressions, etc. Possible ways of improving accuracy and usability includes: a) using higher order modeling methods; b) using more test data for modeling c) processing the test image with more advanced methods;

# Task 4: Image Compression

## Section (a)

### 1) MATLAB code

% main function

clc;clear;close all

origin = double(imread('zhang.png'));%read the image

origin\_R=origin(:,:,1);

origin\_G=origin(:,:,2);

origin\_B=origin(:,:,3);

keepV= 25; %the values that we keep

dec\_ALL(:,:,1)=decomp(keepV,origin\_R);

dec\_ALL(:,:,2)=decomp(keepV,origin\_G);

dec\_ALL(:,:,3)=decomp(keepV,origin\_B);

figure;

imshow(uint8((dec\_ALL))) %reconstruction

%function for decompostion, sugular value plotting and approximation

function dec\_X=decomp(keepV,data\_X)

% a way to set title name of the figures, not important

nhdle = inputname(2);

if(strcmp(nhdle,'origin\_R'))

name = 'Red';

else if (strcmp(nhdle,'origin\_G'))

name='Green';

else

name='Blue';

end

end

%SVD

[U,S,V] = svd(data\_X);

%sort the absolute values

sorted=sort(abs(S),'descend');

%excerpt the largest values

[sortedSV idx]=sort(sorted(1,:),'descend');

figure

plot(sortedSV);

title(sprintf('Singular Values for %s', name));

xlabel('Index');

ylabel('Singular values');

hold on

nidx=idx(keepV:end);%corresponding index of colums

for i=1:max(size(nidx))

S(:,nidx(i))=zeros(max(size(S)),1);%make the insignificant singularvalues zero

end

%approximated matrix

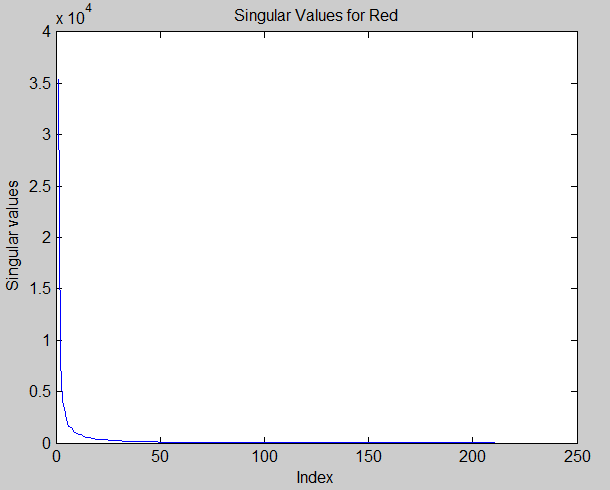
dec\_X=U\*S\*transpose(V);

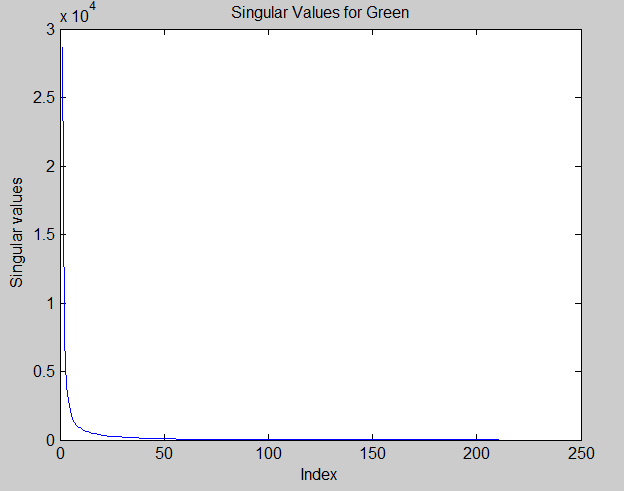
### 2) Images

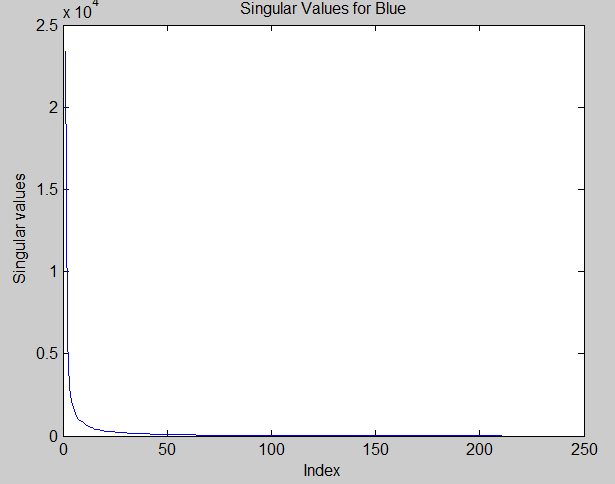
 

Original Image Approximate Image (k=70)

### 3) Plots of Singular Values for Red, Green and Blue matrices







## Section (b)

The dimension of the figure is 263\*211

We know is a column vector of 263 elements, is a row vector of 211 elements and is a coefficient, so the numbers need storing with k sets of data is:

The total numbers in the original image is

The compression rate R% is determined by

For a high compression rate of 95%, k is calculated to be 5.84≈6.

For a medium compression rate of 90%, k is calculated to be 11.68≈12.

For a low compression rate of 70%, k is calculated to be 35.04≈35

With the computed k values, the images produced are shown below

High compression rate Medium compression rate Low compression rate

(95%, k=6) (90%, k=12) (70%, k=35)



Original image