Dimensionality Reduction and Visualization

Exercise pack 4 Solutions

Md. Abdullah-Al Mamun

Part B: Linear Dimensionality Reduction, Continued

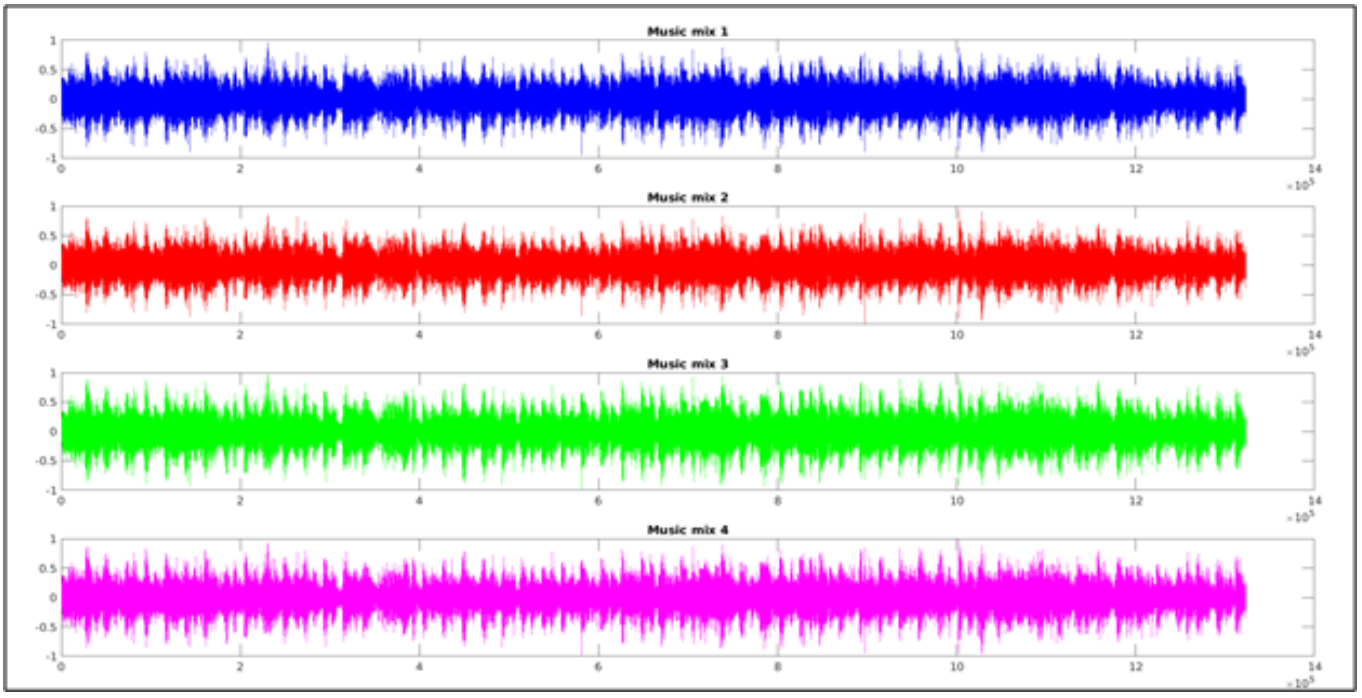
Problem B5: Independent Component Analysis for Separating Audio Mixtures

First we have read the sound file.Matrix has been created.and created the empty data matrix and colleted the data files.Afterward, plotted the signals.Tried to find out the independant components using the software package "fastica", plotted the Plotted the IC's. Then, writing the IC's as sound files. We have Separated the speech mixes and performed ICA. Here we have done independent component anaysis using the “fastica” function.

In [ ]:

```
[originalvector,originalfrequency]=audioread("musicmix01.wav");
info=audioinfo("musicmix01.wav");
originalbits=info.BitsPerSample;
sz=size(originalvector);

% empty data matrix
Data=zeros(sz(1),4);
Data(:,1)=originalvector;

%Collecting the  files
parfor i=2:4
    FILENAME=sprintf('musicmix0%d.wav',i);
    [originalvector,originalfrequency]=audioread(FILENAME);
    Data(:,i)=originalvector;
end

% plotting
fig=figure('Position', get(0, 'Screensize'));

subplot(4,1,1);
plot(Data(:,1),'b');
title('Music mix 1');
subplot(4,1,2);
plot(Data(:,2),'r');
title('Music mix 2');
subplot(4,1,3);
plot(Data(:,3),'g');
title('Music mix 3');
subplot(4,1,4);
plot(Data(:,4),'m');
title('Music mix 4');

saveas(fig,'Problem_B5musicmix.png');

%Finding the independant components using the software package
[ics]=fastica(Data);


# plotting
fig=figure('Position', get(0, 'Screensize'));

subplot(4,1,1);
plot(ics(1,:)','b');
title('Estimated Independant Component 1');
subplot(4,1,2);
plot(ics(2,:)','r');
title('Estimated Independant Component 2');
subplot(4,1,3);
plot(ics(3,:)','g');
title('Estimated Independant Component 3');
```

```
subplot(4,1,4);
plot(ics(4,:)','m');
title('Estimated Independant Component 4');

saveas(fig,'Problem_B5ICs.png');

%Writing the IC's
parfor i=1:4
    FN=sprintf('IC_%d.wav',i);
    audiowrite(FN, ics(i,:)', originalfrequency, 'BitsPerSample',originalbits);
end


%Separating the speech mixes, performing ICA
[originalvector,originalfrequency]=audioread("speechmix01.wav");
info=audioinfo("speechmix01.wav");
originalbits=info.BitsPerSample;
sz=size(originalvector);

%creating the empty data matrix
Data_S=zeros(sz(1),4);
Data_S(:,1)=originalvector;

%Collecting the  files
parfor i=2:4
    FILENAME=sprintf('speechmix0%d.wav',i);
    [originalvector,originalfrequency]=audioread(FILENAME);
    Data_S(:,i)=originalvector;
end


%ICs
[ics_S]=fastica(Data_S);

% Writing the ICA
parfor i=1:4
    FN=sprintf('Speech_%d.wav',i);
    audiowrite(FN, ics_S(i,:)', originalfrequency, 'BitsPerSample',originalbits);
end
```

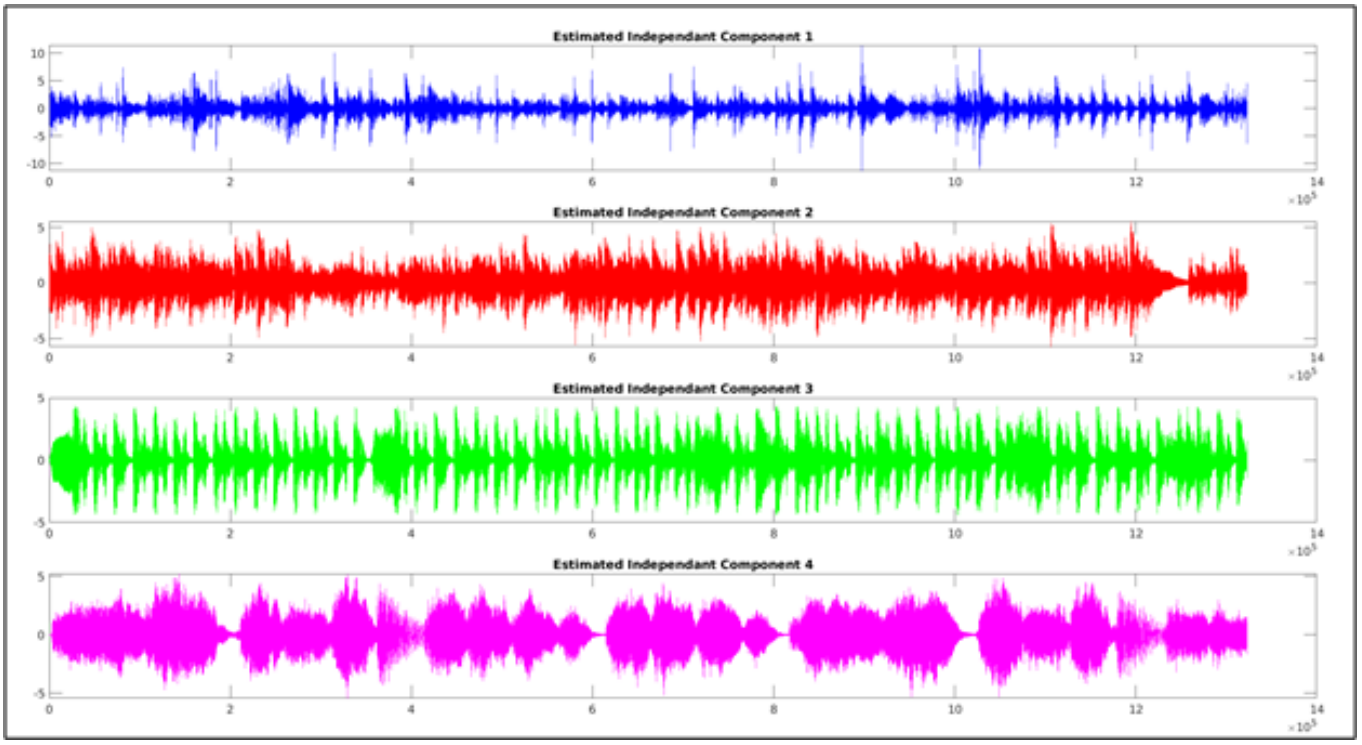


Here are the song info we have defined:

- IC : Water Music from the Handel Show by The United States Army Old Guard Fife and Drum Corps.
- IC : Variatio 1 a 1 Clav by Kimiko Ishizaka.
- IC : Mad Hatter Tea Party by John Bartmann.
- IC :X. La grande porte de Kiev Allegro alla breve, Maestoso, Con gran by Skidmore College Orchestra.

Speech seperation were done and got the following result:

- Speech1: "A Tale of Two Cities" by Charles Dickens
- Speech2: "Pride and Prejudice" by Jane Austen
- Speech3: "The Adventures of Sherlock Holmes" by Arthur Conan Doyle
- Speech4: "Moby Dick; Or, The Whale" by Herman Melville



Problem B6: Independent Component Analysis for Compressing Images

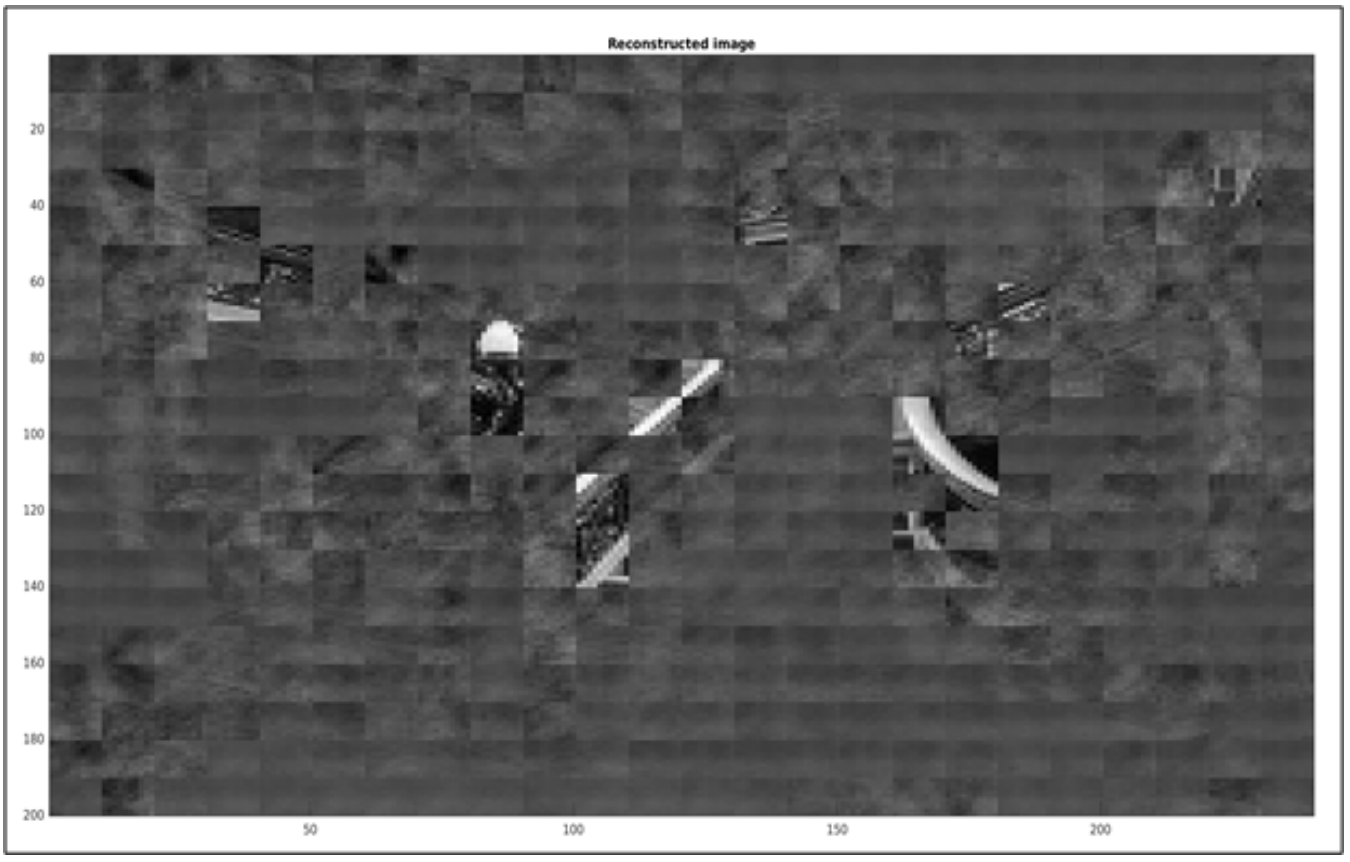
The mean was calculated and substructed from feature data. We have used “fastica” for independent component analysis.

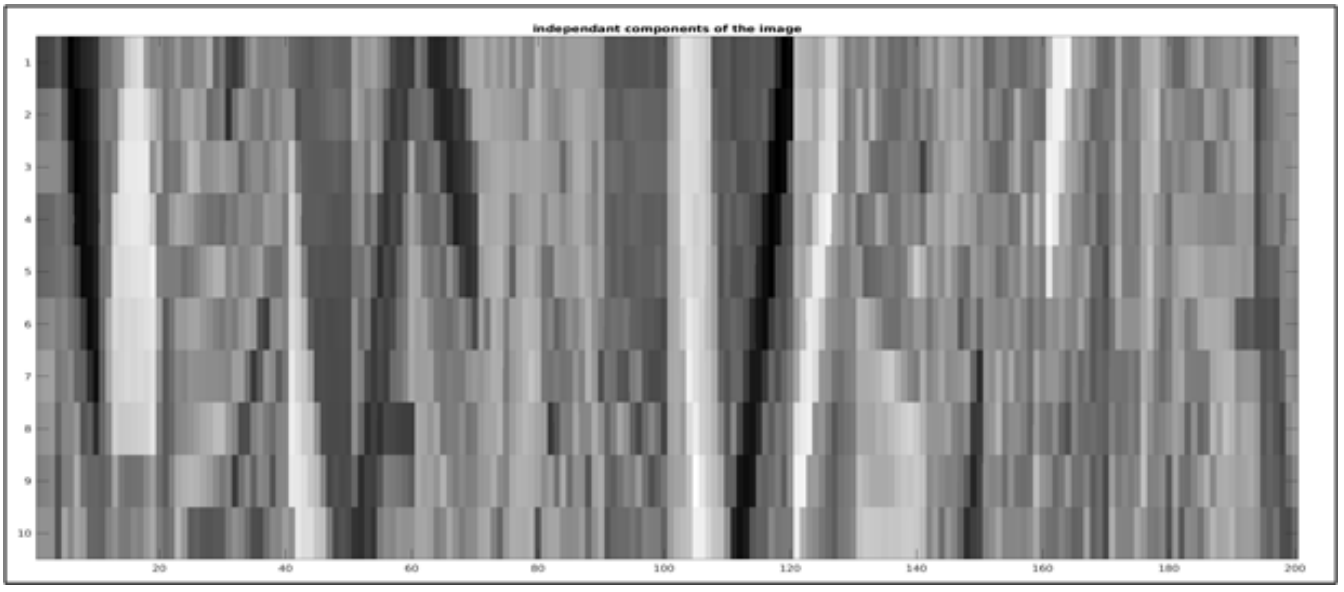
In [ ]:

```
fprintf(1,'Reading in the image\n');
imagematrix=imread('staircase.png');

fprintf(1,'Reducing the image to grayscale\n');
imagematrix=squeeze(mean(imagematrix,3));

npixelrows=size(imagematrix,1);
npixelcolumns=size(imagematrix,2);

% Drawing the original image
fprintf(1,'Drawing the image\n');
figure;
imagesc(imagematrix);
colormap(gray);
title('Original image');

% Dividing the image into 10x10 pixel
blocksize_pixelrows=10;
blocksize_pixelcolumns=10;

% Fitting inde image
nblockrows=floor(npixelrows/blocksize_pixelrows);
nblockcolumns=floor(npixelcolumns/blocksize_pixelcolumns);


fprintf(1,'Creating the featuredata matrix\n');
featuredata=zeros(nblockcolumns*nblockrows,blocksize_pixelcolumns*blocksize_pixelro
ws);

% Collect the pixel values in all blocks
featurerow_index=0;
for blockcolumn_index=1:nblockcolumns,
  for blockrow_index=1:nblockrows,
    featurerow_index=featurerow_index+1;
    first_pixelcolumn = (blockcolumn_index-1)*blocksize_pixelcolumns + 1;
    last_pixelcolumn = (blockcolumn_index-1)*blocksize_pixelcolumns + blocksize_pix
elcolumns;
    first_pixelrow = (blockrow_index-1)*blocksize_pixelrows + 1;
    last_pixelrow = (blockrow_index-1)*blocksize_pixelrows + blocksize_pixelrows;
    blockpixels=imagematrix(first_pixelrow:last_pixelrow,first_pixelcolumn:last_pix
elcolumn);
    featuredata(featurerow_index,:)=blockpixels(:)';
  end;
end;

% Finding the ICs
m=mean(featuredata,1)
Data=featuredata-m

[icasig, A, W]=fastica(Data', 'numOfIC',20);
```

```
%Reconstructing
independantcomponents=A';
reconstructed_featuredata=(A*icasig)'+m;

% Creating a reconstructed image
fprintf(1,'Creating the reconstructed image\n');
reconstructed_imagematrix = zeros(npixelrows,npixelcolumns);

featurerow_index=0;
for blockcolumn_index=1:nblockcolumns,
  for blockrow_index=1:nblockrows,
    featurerow_index=featurerow_index+1;
    blockpixels=zeros(blocksize_pixelrows, blocksize_pixelcolumns);
    blockpixels(:) = reconstructed_featuredata(featurerow_index,:);
    first_pixelcolumn = (blockcolumn_index-1)*blocksize_pixelcolumns + 1;
    last_pixelcolumn = (blockcolumn_index-1)*blocksize_pixelcolumns + blocksize_pix
elcolumns;
    first_pixelrow = (blockrow_index-1)*blocksize_pixelrows + 1;
    last_pixelrow = (blockrow_index-1)*blocksize_pixelrows + blocksize_pixelrows;
    reconstructed_imagematrix(first_pixelrow:last_pixelrow,first_pixelcolumn:last_p
ixelcolumn) = blockpixels;
  end;
end;

% printing the reconstructed image
fprintf(1,'Drawing the reconstructed image\n');
figure;
imagesc(reconstructed_imagematrix);
colormap(gray);
title('Reconstructed image');

fprintf(1,'Press the Enter key to continue\n');
pause;

% Creating an image of the independant component projection directions
fprintf(1,'Creating the independant components image\n');
ncomponents=size(independantcomponents,1);
independantcomponents_imagematrix = zeros(blocksize_pixelrows,ncomponents*blocksize
_pixelcolumns);

for component_index=1:ncomponents,
  blockpixels=zeros(blocksize_pixelrows, blocksize_pixelcolumns);
  blockpixels(:) = independantcomponents(component_index,:);
  first_pixelcolumn = (component_index-1)*blocksize_pixelcolumns + 1;
  last_pixelcolumn = (component_index-1)*blocksize_pixelcolumns + blocksize_pixelco
lumns;
  first_pixelrow = 1;
  last_pixelrow = blocksize_pixelrows;
  independantcomponents_imagematrix(first_pixelrow:last_pixelrow,first_pixelcolumn:
last_pixelcolumn) = blockpixels;
end;
```

```
% printing the reconstructed image
fprintf(1,'Drawing the independant components image\n');
imagesc(independantcomponents_imagematrix);
colormap(gray);
title('Independant components of the image');
```





Finally we can ssay that there is a difference between PCA and reconstructed image using ICA.PCA gives better image compared to ICA. ICA did not give more information about pixels.

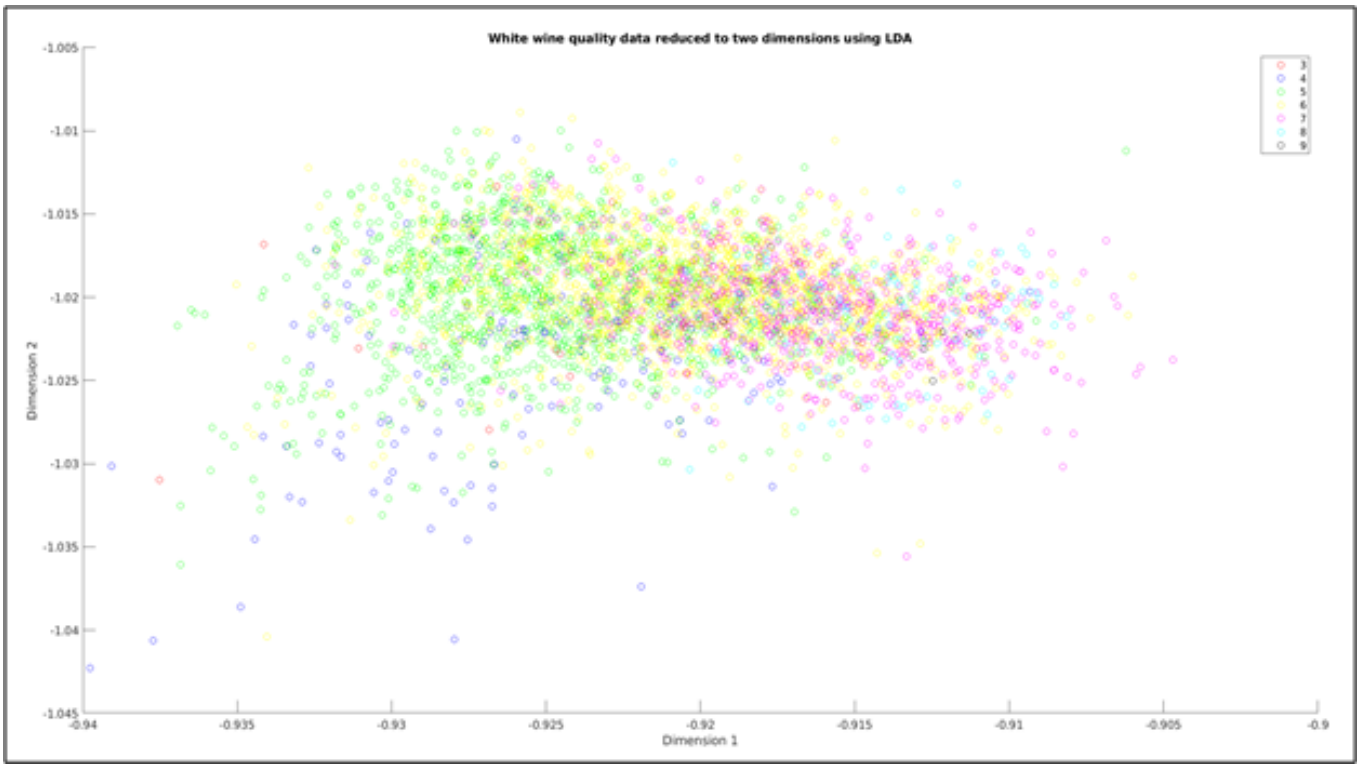
Problem B7: Linear Discriminant Analysis

- Package downloaded from https://se.mathworks.com/matlabcentral/fileexchange/45006-fda-lda-multiclass?s_tid=FX_rc3_behav (https://se.mathworks.com/matlabcentral/fileexchange/45006-fda-lda-multiclass?s_tid=FX_rc3_behav)
- Finding indices for different classes
- plotting

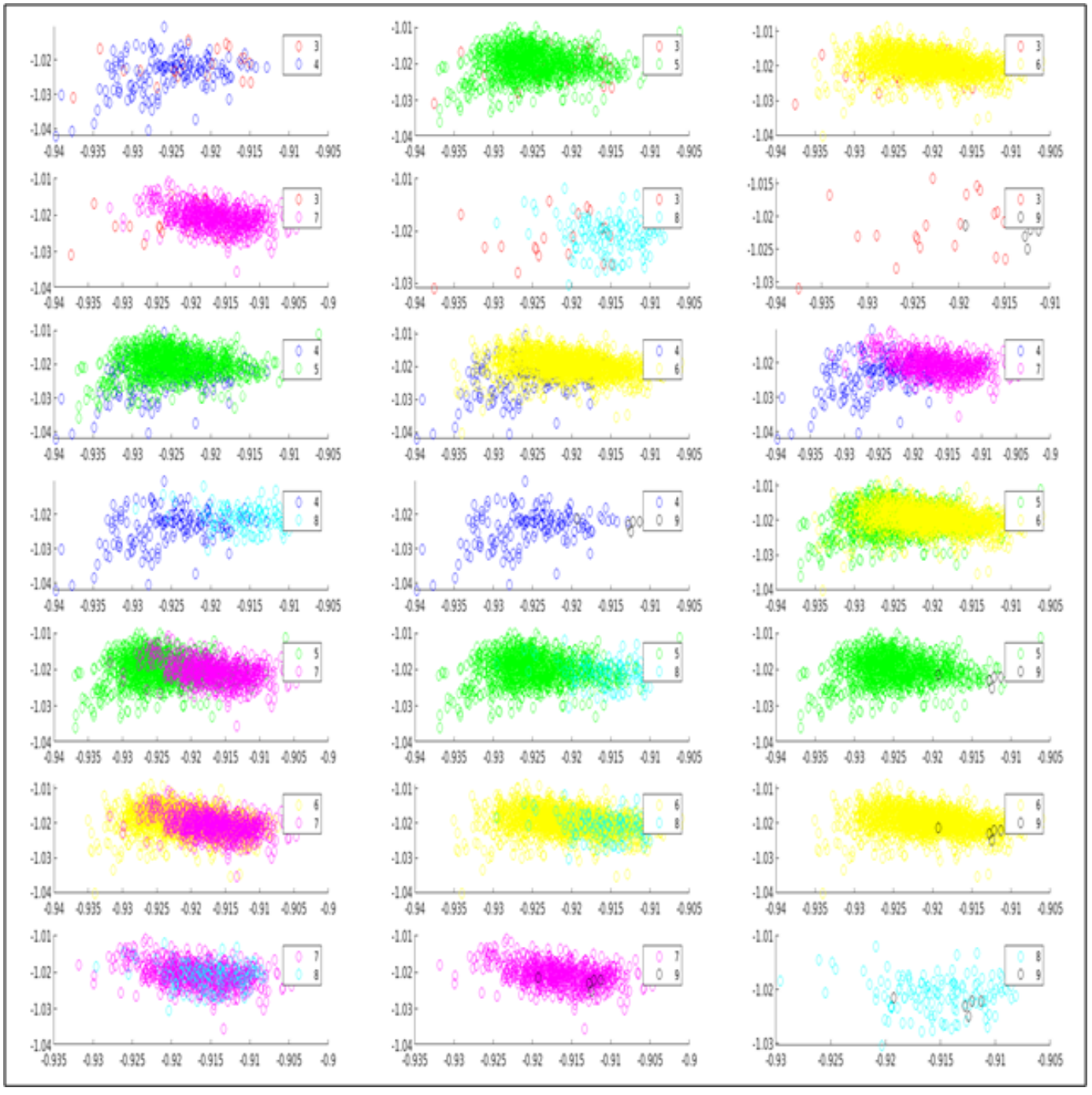
In [ ]:

```
Data=winequalitywhite(:,1:11);
Y=winequalitywhite(:,end);

[Z,W]=FDA(Data',Y,2);
data_r=Z';

%Finding indices
idx_3=find(Y==3);
idx_4=find(Y==4);
idx_5=find(Y==5);
idx_6=find(Y==6);
idx_7=find(Y==7);
idx_8=find(Y==8);
idx_9=find(Y==9);

combs=combnk([3 4 5 6 7 8 9], 2);
colors=['r','b','g','y','m','c','k']
fig=figure('Position', get(0, 'Screensize'));
for i=1:21
    class_1=combs(i,1)
    class_2=combs(i,2)
    c1=colors(class_1-2)
    c2=colors(class_2-2)
    subplot(7,3,i);
    idx_c1=find(Y==class_1)
    idx_c2=find(Y==class_2)
    scatter(data_r(idx_c1,1),data_r(idx_c1,2), c1);
    hold on;
    scatter(data_r(idx_c2,1),data_r(idx_c2,2), c2);
    legend([sprintf('%d',class_1)],[sprintf('%d',class_2)])
end
saveas(fig,'Problem_B7matrix.png');

%Plotting
fig=figure('Position', get(0, 'Screensize'));
scatter(data_r(idx_3,1),data_r(idx_3,2), 'r');
hold on;
scatter(data_r(idx_4,1),data_r(idx_4,2), 'b');
hold on;
scatter(data_r(idx_5,1),data_r(idx_5,2), 'g');
hold on;
scatter(data_r(idx_6,1),data_r(idx_6,2), 'y');
hold on;
scatter(data_r(idx_7,1),data_r(idx_7,2), 'm');
hold on;
scatter(data_r(idx_8,1),data_r(idx_8,2), 'c');
hold on;
scatter(data_r(idx_9,1),data_r(idx_9,2), 'k');
xlabel('Dimension 1');
ylabel('Dimension 2');
```

```
legend(['3'],['4'],['5'],['6'],['7'],['8'],['9']);
title('White wine quality data reduced to two dimensions using LDA');
saveas(fig,'Problem_B712.png');
```



- According to PCA result obtained, we can see that data has been seperated nicely using LDA. However, wecan see the overlapping classes as well.
- According to projection matrix, 2 and 8 feature influed a lot on projeciton coordinates. first rejeted feature is 5 and 3 is the second.
- another matrix plot were created to compare two possible classes given below:



In [ ]:

```
```