SMAI - Mini Project -1 - Report
Rollno: 20161163

METHOD OF PCA:

```
# PCA Algo

def pca(imlist):
     X = find_X(imlist)
     no_img, dim = X.shape
     mean_X = X.mean(axis = 0) #Mean Image
     X = X - mean_X
     Xtr = np.transpose(X)
     if dim > no_img:
     M  = np.dot(X,Xtr)
     e, EV = np.linalg.eigh(M)
     EV = np.transpose(EV)
     tmp = np.transpose(np.dot(Xtr, EV))
     V = tmp[::-1]
     e = e[::-1]
     for i in range(V.shape[1]):
          V[:,i] /= np.linalg.norm(V[:,i])
     else:
     U,S,V = np.linalg.svd(X)
     V = V[:num_data]
     return X, V,e,mean_X
```

The above code snippet gives an idea of the PCA algorithm used
Here as the number of features are more than dimensions, the trick of inverting matrices has been used.

TOP 3 eugen features:

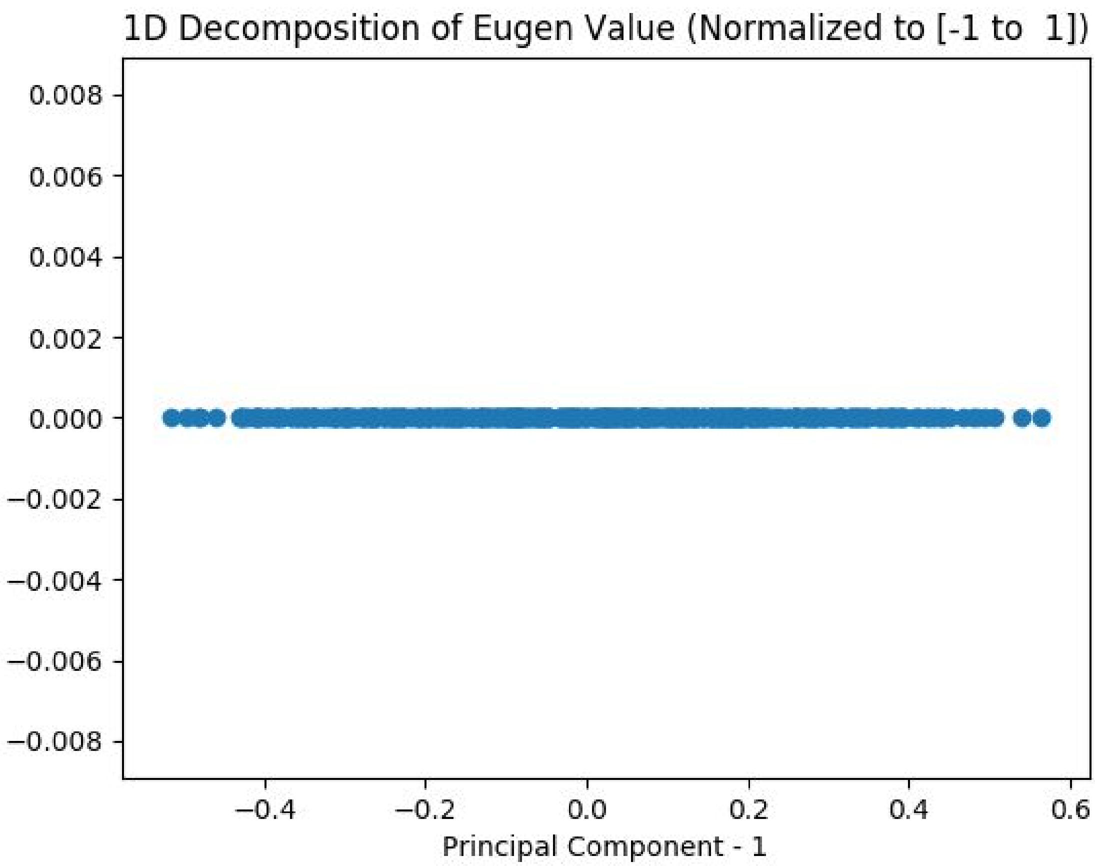
-- Mean Image -- -- Eigen Face -1 --

-- Eigen Face 2 -- -- Eigen Face 3 --

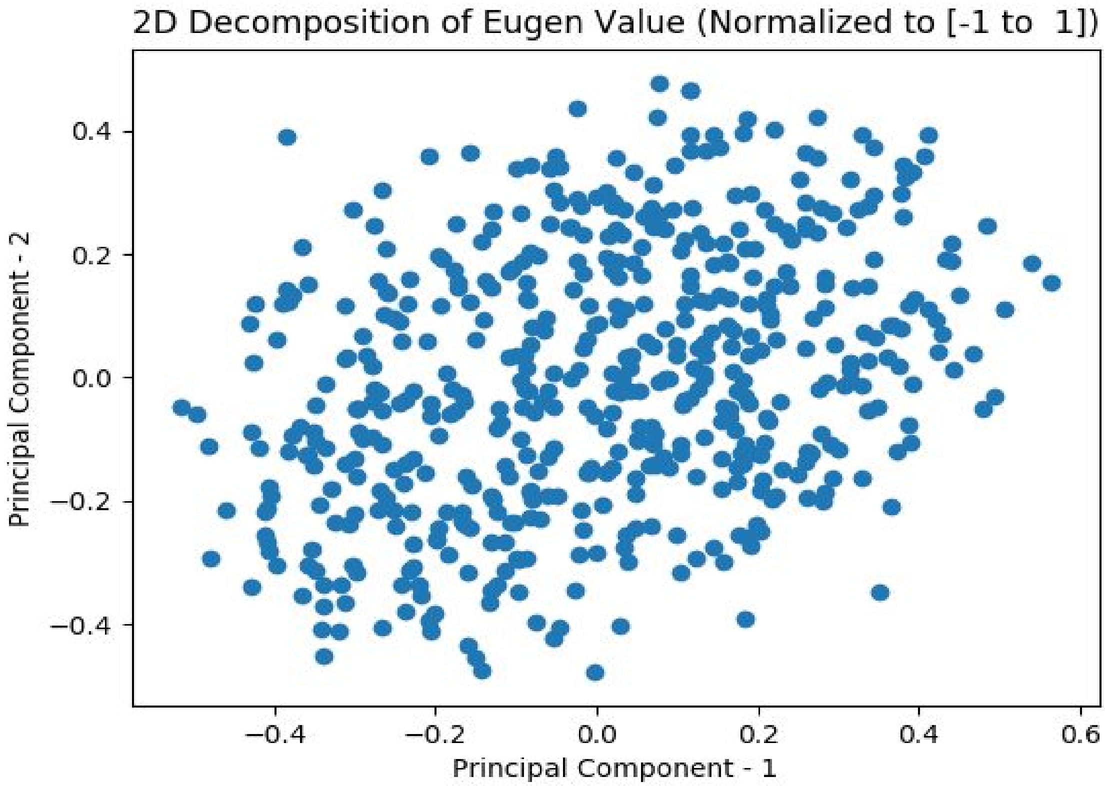
The mean image and the eigen features have been observed as above

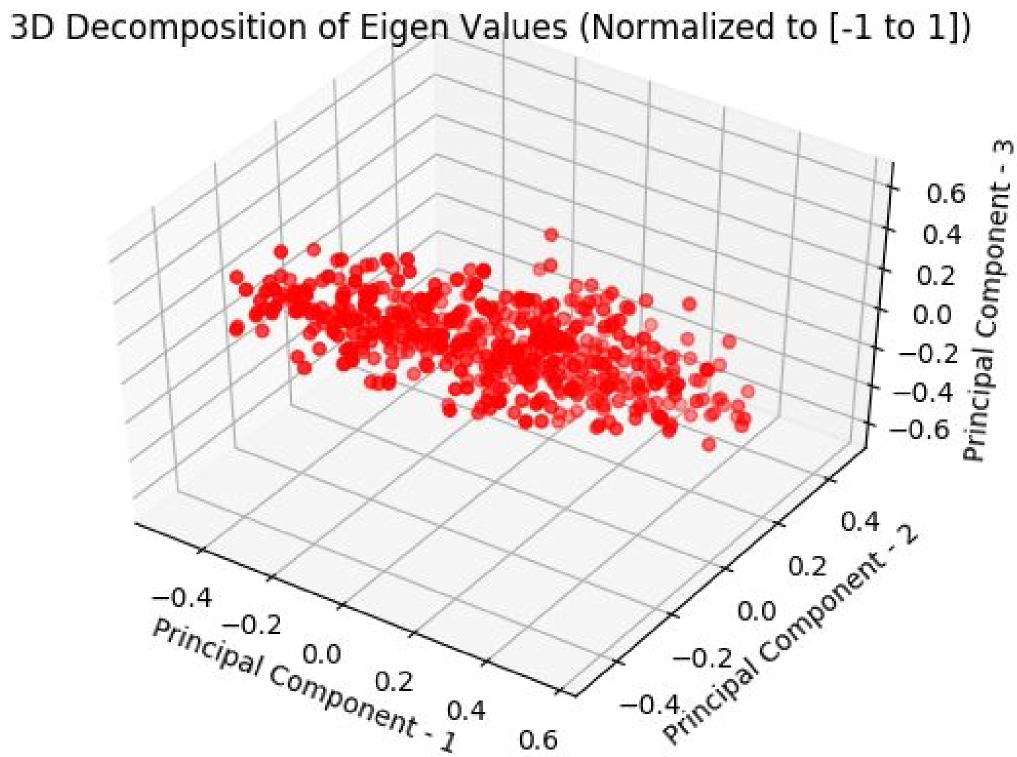
Plots:

1. **1D Decomposition**



2. 2D Decomposition



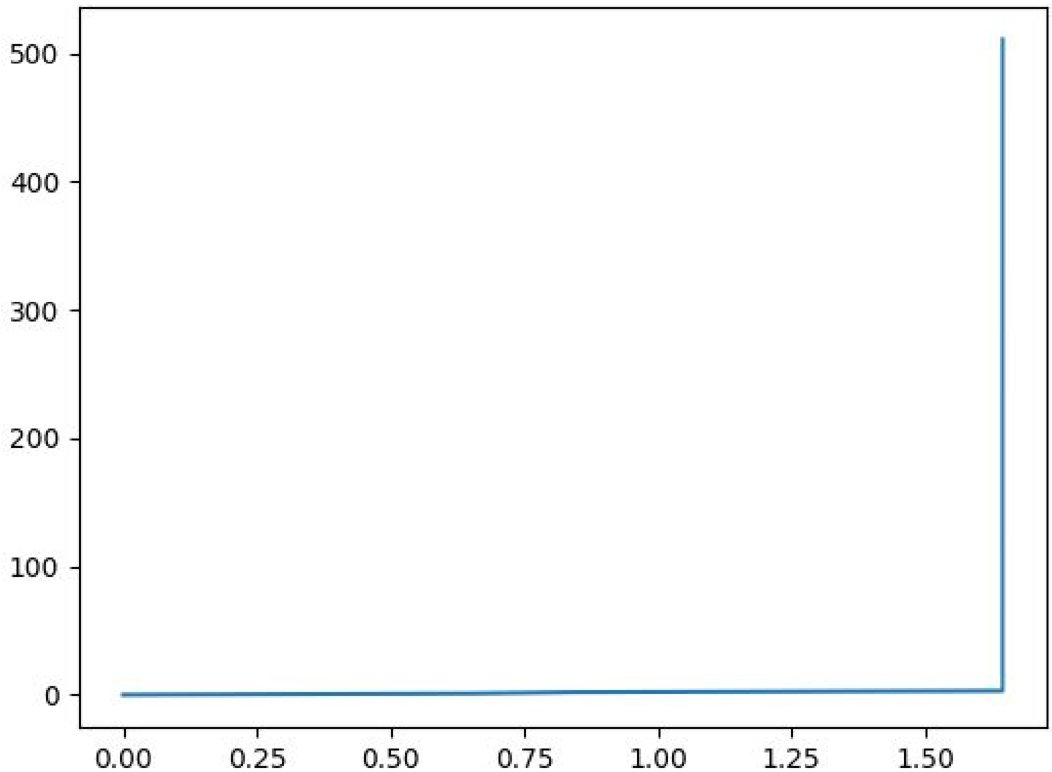


3D Decomposition

It can be observed that the feature points are becoming more widely spread

Variance versus number of principal components:

Due to the variances being of the order of 10^10. The graph has been scaled on the Y-axis.
The change in the variance is negligible when compared to the magntiude of variance itself and hence the graph shows only an appreciable amount of increase


500
400
300
200
100
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50