

Principal Component Analysis

Priyansh R. Dalsania

September 29, 2018

1 Overview

PCA is a powerful statistical technique to find directions of maximum difference among data points, in order to recognize patterns existing in the sample data.

The report presents the basic steps used in computing principal components of the sample data-set comprising 520 images.

2 Implementation

2.1 Feature representation

Images received are of RGB format, 256×256 pixels. This is down-scaled to give effective size of 64×64 pixels. This 3-channel array is flattened out to a 1-D vector of length $d = 64 \times 64 \times 3 = 12288$ that will represent the image.

2.2 Computing Principal Components

From X , the $n \times d$ data matrix, we compute the centred data matrix,

$$Z = X - \mu(X)$$

The no. of samples, $n = 520$, is far less than the no. of features, d . This makes the $n \times n$ matrix ZZ^T computationally very economical as compared to the $d \times d$ matrix $Z^T Z$, the general co-variance matrix, also giving the same eigenvalues, while the eigenvectors of $Z^T Z$ are simply $Z^T u$, where u is an eigenvector of ZZ^T , computable much faster, as given in $O(n^3)$ vs $O(d^3)$ time complexity.

All the eigenvalues do not contribute considerably to data scatter. Hence, we can handpick the k ($= 32$ in our case) most significant ones magnitude-wise, that contribute to the variations in data.

This finally gives us the d length eigenvectors corresponding to the k largest eigenvalues.

2.3 Dimensionality Reduction

The quality of images created with k eigenvectors definitely increases with k , but not in a linear fashion, as each eigenvalue doesn't contribute equally. To calculate the mean square error (or the statistically equivalent Frobenius norm)

of the data matrix generated by k eigenvectors, we take the running sum of components of each eigenvector, in descending order of their magnitude.

Analytically, the image vector, x , can be expressed in terms of its eigenvector basis given by PCA, as

$$x = c_1 v_1 + c_2 v_2 + \cdots + c_n v_n$$

Taking dot product with i^{th} eigenvector v_i ,

$$\begin{aligned} v_i^T x &= c_1 v_1^T v_i + c_2 v_2^T v_i + \cdots + c_n v_n^T v_i \\ &= c_i, \quad \text{as } v_i^T v_j = \delta_{ij}. \end{aligned}$$

This gives us the coefficient of x in the direction of v_i . Using k such coefficients, one for each direction, we can effectively represent a lossy image in k -features, from the original d .

Therefore, the component of x in the direction of v_i is given by :

$$c_i v_i = v_i v_i^T x$$

The coefficients in first 3 directions cumulatively depict the scatter of data in 1-D, 2-D, 3-D.

Iterating up-to k eigenvectors, we get the compressed t :

$$t = \sum_{i=1}^k c_i v_i = \sum_{i=1}^k v_i v_i^T x$$


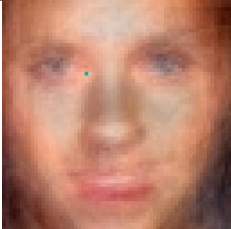



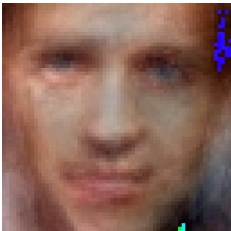

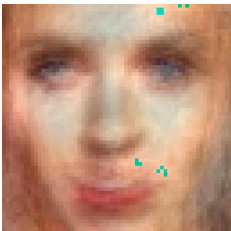

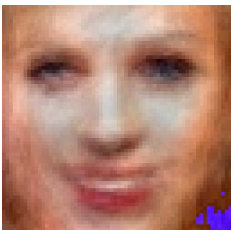
These are the images we finally obtain.




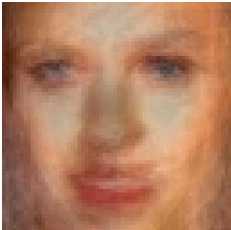

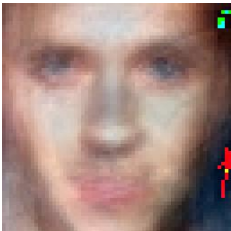



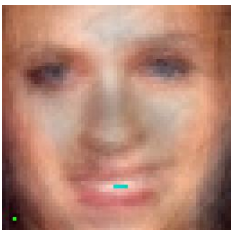
To obtain the difference from the original images, we take the Frobenius norm, equal to the L2 norm of flattened matrix vector, of the difference of the compressed and the original data matrix, in each iteration.

3 Inference

3.1 Reconstructed images

Few of the original and reconstructed counterparts, are shown below, for comparison :

<i>Original Image</i>	<i>Reconstructed Image</i>
	
	
	
	
	

<i>Original Image</i>	<i>Reconstructed Image</i>
	
	
	
	
	

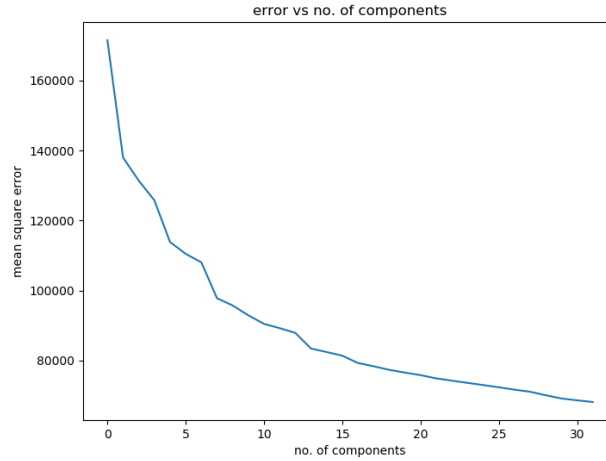
We observe that, (apart from some deviations) the images are more centred, in the sense that they more closely resemble the average of the population, since many of the eigenvectors are thrown away, that gave the subtle differences.

What remains is basic texture of face in the centre, and contour lines and expressions. Facial features are blurred out.

However, from spatial point of view, the features of image vector corresponding to each compressed image preserves the patterns among the classes, that differentiate them from samples of other classes.

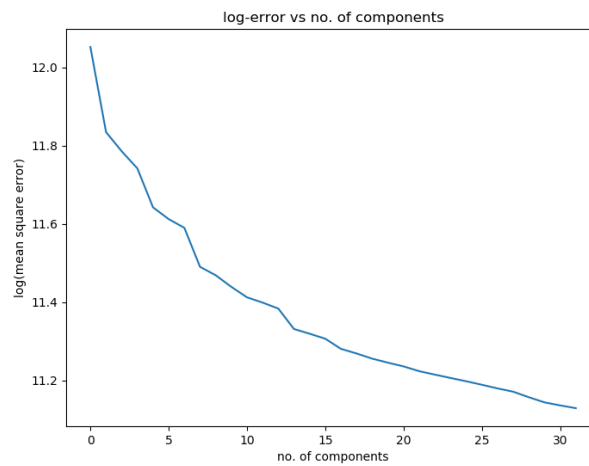
3.2 No. of eigenvectors vs. MSE

The plot for No. of eigenvectors vs. MSE :



This shows that with just a few eigenvectors, the mean error starts decaying near exponentially. A few reasonable no. of eigenvectors can provide the optimum balance between data compression factor and data quality.

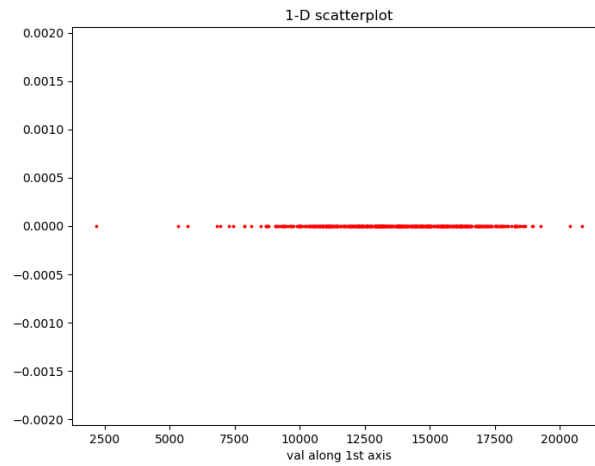
For further empirical analysis, the semi-log variant of the above plot :



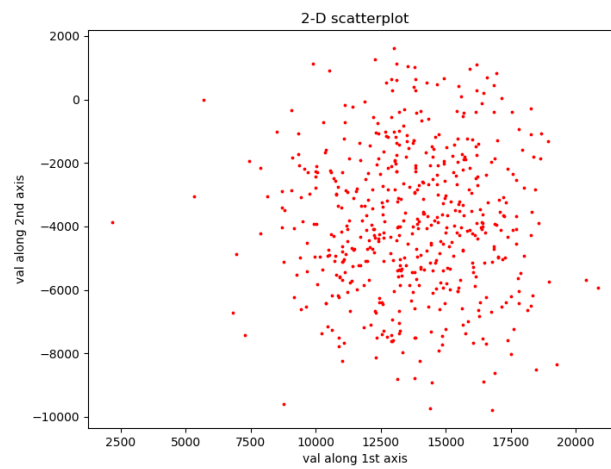
This shows a flatter decay on logarithmic scale.

3.3 Scatter-plots in various dimensions

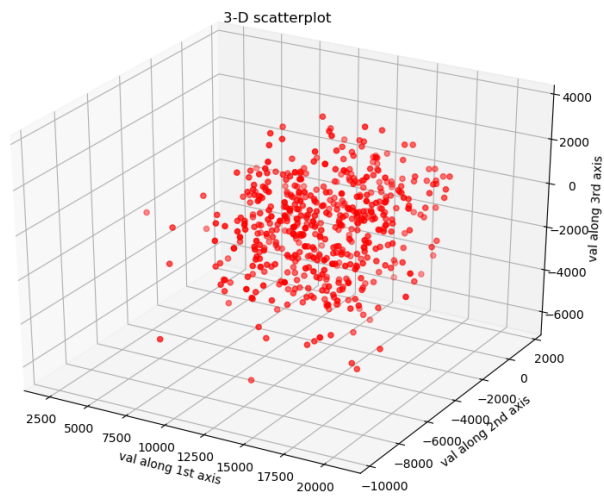
The scatter-plot obtained in 1-D :



The scatter-plot obtained in 2-D :



The scatter-plot obtained in 3-D :



These are the dimensions corresponding to the largest eigenvalues, and hence highest contribution to the variation of patterns in data. We see that the values are spread across the dimensions, with some clustering around the mean values.