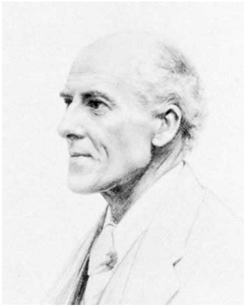
Lecture 14

Principal Component Analysis

Introduction

Karl Pearson (1857-1936)

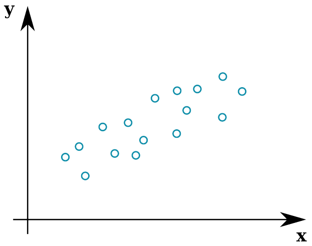


Introduction

What is PCA?

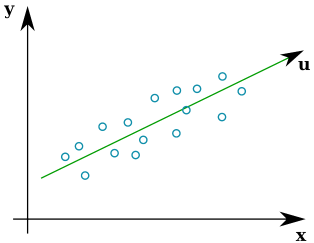
Principal Component Analysis, or simply PCA, is a multivariate statistical procedure concerned with representing the covariance structure of a set of variables through a small number of linear combinations of these variables.

PCA in Outline



For a given data set

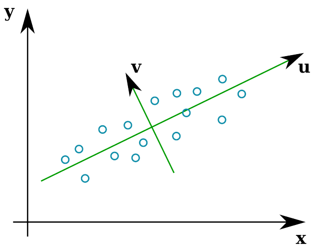
PCA in Outline



For a given data set

We determine the direction where the variation is the greatest

PCA in Outline

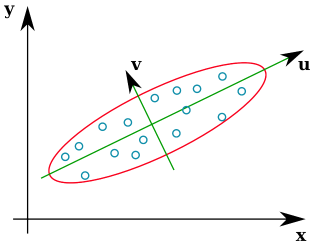


For a given data set

We determine the direction where the variation is the greatest

Then we find the direction where the remaining variation is the greatest

PCA in Outline



For a given data set

We determine the direction where the variation is the greatest

Then we find the direction where the remaining variation is the greatest

We continue the process and thus find the coordinate system that most compactly represents the data.

PCA in Outline

In algebraic terms, principal components are particular linear combinations of the original variables that form a projection that best represents the data in a least-square sense.

Geometrically, these linear combinations represent a new coordinate system obtained by translating and rotating the original one. The new axes can be used to determine the directions with maximum variability of the sample data.

PCA in Outline

In computational terms the principal components are found by calculating the eigenvectors and eigenvalues of the data covariance matrix.

The eigenvector with the largest eigenvalue is the direction of greatest variation.

&c.

A brief review of Eigenvectors

Let $\boldsymbol{A}$ be an $n \times n$ matrix. The eigenvalues of $\boldsymbol{A}$ are defined as the roots of:

$$determinant(\boldsymbol{A} - \lambda \boldsymbol{I}) = |(\boldsymbol{A} - \lambda \boldsymbol{I})| = 0$$

where $\boldsymbol{I}$ is the $n \times n$ identity matrix. This equation is called the characteristic equation and has n roots.

A brief review of Eigenvectors

Let λ be an eigenvalue of $\boldsymbol{A}$. Then there exists a vector $\boldsymbol{x}$ such that

$$\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{x}$$

The vector $\boldsymbol{x}$ is called an eigenvector of $\boldsymbol{A}$ associated with the eigenvalue λ. Ordinarily we normalise $\boldsymbol{x}$ so that it has length one, that is,

$$\boldsymbol{x} \cdot \boldsymbol{x}^T = 1$$

Eigenvectors and Diagonalisation

Suppose we have a 3×3 matrix A with eigenvectors $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_3$, and eigenvalues λ_1, λ_2, λ_3 so:

$$\boldsymbol{A}\boldsymbol{x}_1 = \lambda_1 \boldsymbol{x}_1 \qquad \boldsymbol{A}\boldsymbol{x}_2 = \lambda_2 \boldsymbol{x}_2 \qquad \boldsymbol{A}\boldsymbol{x}_3 = \lambda_3 \boldsymbol{x}_3$$

Putting the eigenvectors into a column matrix gives:

$$A \left[\begin{array}{ccc} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_3 \end{array} \right] = \left[\begin{array}{ccc} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_3 \end{array} \right] \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

Eigenvectors and Diagonalisation

Writing:

$$\Phi = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_3 \end{bmatrix} \qquad \Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

we get:

$$\boldsymbol{A}\Phi = \Phi\Lambda$$

and since we normalised the eigenvectors we know that:

$$\Phi\Phi^T = \Phi^T\Phi = \boldsymbol{I}$$

thus:

$$\Phi^T\boldsymbol{A}\Phi = \Lambda$$

$$\boldsymbol{A} = \Phi\Lambda\Phi^T$$

Eigenvectors of Covariance Matrices

Let Σ be an $n \times n$ covariance matrix. There is an orthogonal $n \times n$ matrix Φ whose columns are eigenvectors of Σ and a diagonal matrix Λ whose diagonal elements are the eigenvalues of Σ, such that:

$$\Phi^T \Sigma \Phi = \Lambda$$

The linear transformation Φ takes our variables to a new coordiante system where they are uncorrelated. The correlation matrix of our data in the new axis system is Λ which has only diagonal elements.

PCA in Practice

Given our data set in the axis system $X = [x_1, x_2 \cdots x_n]$:

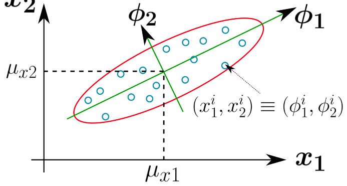
1. Calculate the means $(\mu_{x1}, \mu_{x2} \cdots)$ and the covariance Σ.

2. Calculate the eigenvectors of Σ: $\Phi = [\phi_1,\ \phi_2 \cdots]$.

3. Any point $\boldsymbol{p_x}$ in the $[\boldsymbol{x}_1, \boldsymbol{x}_2 \cdots]$ axis system is transformed to the point $\boldsymbol{p}_\phi$ in the $[\phi_1, \phi_2, \cdots]$ system with the equation:

$$\boldsymbol{p}_\phi = (\boldsymbol{p_x} - \boldsymbol{\mu_x}) \cdot \Phi$$

where $\boldsymbol{\mu_x} = (\mu_{x1}, \mu_{x2} \cdots)$.



Exemplar: Face recognition

The objectives of using PCA in face reconition are:

1. Data Reduction
2. Feature Selection

Exemplar: Face recognition

In image recognition an input image with n pixels can be treated as a point in an n-dimensional space called the image space. The coordinates of this point represent the values of each pixel of the image and form a vector

$$\boldsymbol{p_x} = (i_1, i_2, i_3, ...i_n)$$

obtained by concatenating the rows (or columns) of the image matrix.

Converting an Image to a vector

Given a greyscale image of, for example, 128 by 128 pixels:

$$= \begin{bmatrix} 150 & 152 & \cdot & 151 \\ 131 & 133 & \cdot & 72 \\ \cdot & \cdot & \cdot & \cdot \\ 144 & 171 & \cdot & 67 \end{bmatrix} \; 128 \times 128$$

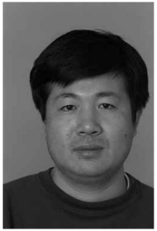
We concatinate each row to make a 16384 vector

$$[150, 152, \cdots 151, 131, 133, \cdots 72, \cdots 144, 171, \cdots 67]_{16K}$$

Redundant Information

Face images are highly redundant:

- All the background pixels are the same
- Each subject has the same facial features



Dimension Reduction

In the data space each pixel is a variable, so the dimension of the space is very high (min 16K). Dimension reduction is achieved by PCA. Let an $N \times n$ data matrix D be composed of N input face images with n pixels. Each row is one image of our data set.

$$\boldsymbol{D} = \begin{bmatrix} 150 & 152 & \cdots & 254 & 255 & \cdots & 252 \\ 131 & 133 & \cdots & 221 & 223 & \cdots & 241 \\ \cdot & \cdot & & \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot & \cdot & & \cdot \\ 144 & 171 & \cdots & 244 & 245 & \cdots & 223 \end{bmatrix}_{N \times n}$$

Mean Centring the data

Suppose the mean of the columns of $\boldsymbol{D}$ (the average image) is:

$$[120\ \ 140\ \ \cdots\ \ 230\ 230\ \ \cdots\ \ 240]$$

The origin is moved to the mean of the data by subtracting this average image from each row. This creates the mean centred data matrix:

$$\boldsymbol{U} = \begin{bmatrix} 30 & 12 & \cdots & 24 & 25 & \cdots & 12 \\ 11 & -7 & \cdots & -9 & -7 & \cdots & 1 \\ . & . & & . & . & & . \\ 24 & 31 & \cdots & 14 & 15 & \cdots & -17 \end{bmatrix} \quad N \times n$$

Calculating the covariance matrix

The covariance matrix $\boldsymbol{\Sigma}$ can be calculated easily from the mean centered data matrix:

$$\Sigma = \boldsymbol{U}^T \boldsymbol{U} / (N-1)$$

N is the number of data points (images) and the covariance matrix has dimension $n \times n$.

Finding the Eigenvectors and Eigenvalues

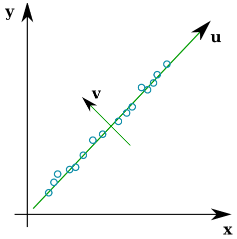
We use the techniques covered earlier in this lecture to find Φ and Λ that satisfy:

$$\Sigma = \Phi \Lambda \Phi^T$$

As before, we normalise the eigenvectors ϕ_i so that they form an orthonormal basis:

$$\forall \phi_i \; \phi_j \in \Phi, \; \phi_i \cdot \phi_j = \begin{cases} 1 & \mathit{if} \; i = j \\ 0 & \mathit{if} \; i \neq j \end{cases}$$

Data Reduction



Eigenvectors with low eigenvalues contribute little information in the data representation. Data reduction is achieved by ignoring the eigenvectors with low eigenvalues.

The set of $m(m < n)$ eigenvectors of Σ which have the m largest eigenvalues, minimises the mean square reconstruction error over all choices of orthonormal basis of size m.

Data Reduction

Although n variables are required to reproduce an original sample X exactly, much of the significant variability in the data can be accounted for by a smaller number m of principal components.

Thus, the original data set consisting of N examples on n variables can be reduced to a data set consisting of N examples on m principal components (eigenvectors).

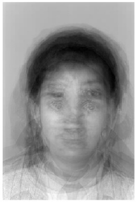
In face recognition the eigenvectors are often called eigenfaces.

Practical Face recognition

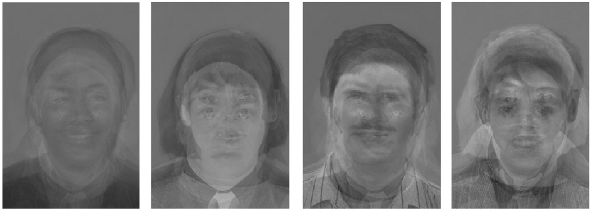
As an example we will find the eigenface basis of a set of fourteen faces images of resolution 384×256. This initial data set D is sometimes called the training data set.

What do the eigenfaces look like?

Mean:

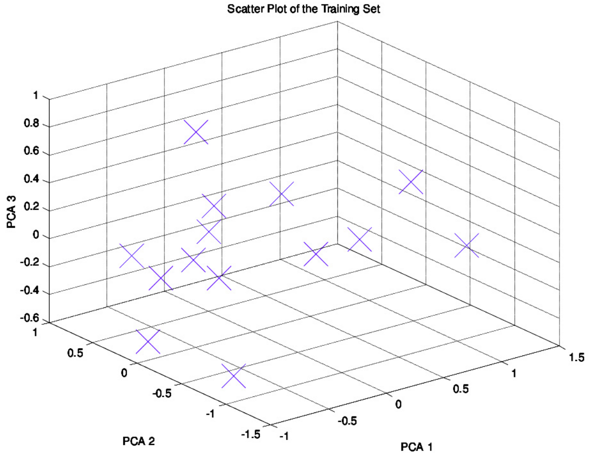


The four eigenfaces with the largest eigenvalues:



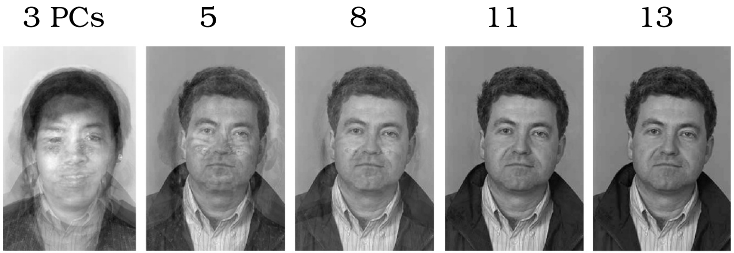
How different are the face images?

This is a plot of the fourteen face images on the first three principal components:



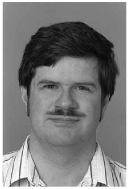
Reconstructing the face images: example 1

Original:

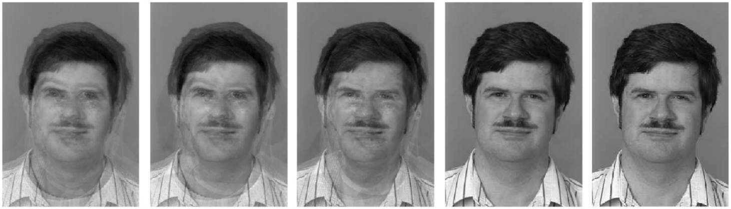


Reconstructing the face images: example 2

Original:



3 PCs 5 8 11 13



How many principal components do we need?

There is no definitive answer to this question. We want to minimise m (the number retained) but maximise the accuracy of the data representation.

One approach is to see how much of the variance each principal component accounts for. We can express the percentage of the total variance accounted for by the i^{th} eigenvector as:

$$r_i = 100 \times \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

One heuristic method would be to discard components where r_i falls below a threshold, say 1%.

Few Samples and many Variables

Face recognition is an example of a small sample size problem. The number of variables (pixels) n is very large, but the number of samples (pictures in the data base) N is much smaller.

Since the covariance matrix Σ is very large ($n \times n$) the computation of the eigenvectors is difficult.

However, since there are only N samples the rank of the covariance matrix is at most $N - 1$ and therefore there are at most $N - 1$ eigenvectors with non-zero eigenvalues.

Karhunen Loeve Transform

Karhunen and Loeve came up with an ingenious way of computing principal components efficiently.

Note that the matrix $\boldsymbol{U}\boldsymbol{U}^T$ is much smaller than $\boldsymbol{U}^T\boldsymbol{U}$, and we can calculate its eigenvectors easily. They satisfy:

$$\boldsymbol{U}\boldsymbol{U}^T\Phi' = \Phi'\Lambda$$

Multiplying both sides by $\boldsymbol{U}^T$ gives:

$$\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{U}^T\Phi' = \boldsymbol{U}^T\Phi'\Lambda$$

Adding brackets clarifies that $\boldsymbol{U}^T\Phi'$ are the eigenvectors of $\boldsymbol{U}^T\boldsymbol{U}$:

$$\boldsymbol{U}^T\boldsymbol{U}(\boldsymbol{U}^T\Phi') = (\boldsymbol{U}^T\Phi')\Lambda$$

Karhunen Loeve Transform

In summary we find the eigenvectors Φ' of the small matrix $\boldsymbol{U}\boldsymbol{U}^T$, then multiply them by $\boldsymbol{U}^T$ to find the eigenvectors of $\boldsymbol{U}^T\boldsymbol{U}$.

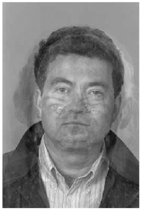
The resulting eigenvectors are not orthonormal. To complete the process we must normalise them. This is done by dividing each column of $\boldsymbol{U}^T\Phi'$ by its magnitude $\sqrt{(N-1)\lambda_i}$ for the scatter matrix or $\sqrt{\lambda_i}$ for the covariance matrix.

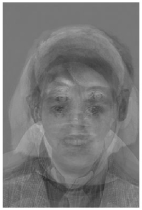
There are at most $N-1$ non-zero eigenvalues in $\boldsymbol{U}\boldsymbol{U}^T$, but this is not a restriction since, as noted before, the rank of $\boldsymbol{U}^T\boldsymbol{U}$ is at most $N-1$.

Correspondence in PCA

In 2D face images the pixels are not usually in correspondence. That is to say a given pixel $[x_i, y_i]$ may be part of the cheek in one image, part of the hair in another and so on.

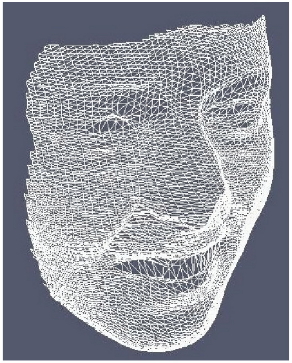
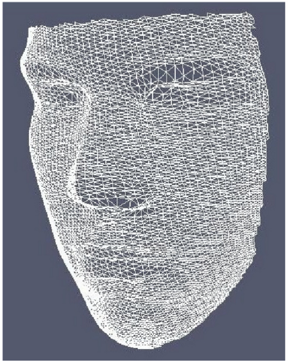
This means that any linear combination of eigenvectors does not represent a true face but a composition of face parts.





Correspondence in 3D face models

If we represent a face in 3D, then it is possible to establish a correspondence between each point on the surface map. Here are two different faces that are in correspondence.

PCA on 3D face surface models

We can apply PCA to the 3D surface representation.

The variables are the individual co-ordinates of the surface points.

A data set is of the form:

$$[x_1, y_1, z_1, x_2, y_2, z_2, x_3, \cdots x_N, y_N, z_N]$$

The face maps on the previous slide have about 5,000 surface points, hence the number of variables is 15,000.

Active Shape Models

If the points of each subject are in correspondence, and the different subjects are aligned as closely as possible in 3D before calculating the PCA, then any reasonable combination of the eigenvectors will represent a valid face.

Faces can be created that are not real, but are a linear combination of those in the data set.

The set of eigenvectors representing surface data in this manner is called an active shape model

Active Appearance Models

Texture can be mapped onto a 3D surface map to give it a realistic face appearance. The texture values (like pixel values in 2D) can also be included as variables in PCA.

Models of this sort are called active appearance models.