

## 3.1 Overview of Eigenfaces.

A general digital image is stored in a computer (or display on a computer screen) as a rectangular array or matrix where the size of the array determines the resolution of the image. For instance one may be dealing with a 100x100 pixel, 8-bit grey scale image. This means that any of the  $100 \times 100 = 10,000$  pixels takes on any of  $2^8 = 256$  values. Normally it is arranged that the values of the pixels of a 8-bit image vary between 0 and 255, representing all grey scales from black to white. Clearly one has to determine 10 000 values for each of the 10 000 pixels of a 100x100 image. Mathematically one can think about all 100x100 images as a 10,000 dimensional linear vector space. Of course the dimension increases for higher resolution images.

Thus, whenever one is dealing with a general 100x100 pixel image, 10,000 pixel values need to be specified; the true dimension of our vector space is 10,000. Of course we don't always need to save this amount of data. There exist powerful compression algorithms that take regional correlations in an image into account, leading to substantial storage savings. However, random images cannot be compressed and the true dimension of the vector space in general is 10,000. In our case we know in advance that we are only interested in facial images. How can one exploit this fact? Facial images form a subset of all images that can be represented on say, a 100x100 grid. This opens the possibility that one might be able to find a basis for the subspace of our 10,000 vector space, containing only, and all facial images. The rest of this section describes how we set about finding a basis for the subspace containing all facial images.

First we need a collection of facial images that are truly representative of all faces that one might encounter. It is no trivial matter to collect this training set of facial images, but let assume for the moment that this has been done and that our training set consists of  $M$  faces. Let us also assume that our facial images are represented on a  $N \times N$  pixel grid ( $M$  is a large number but very much smaller than  $N \times N$ , the dimension of our image vector space). Instead of working with the images as a

2-dimensional array, we concatenate the rows (or columns) to form a 1-dimensional column vector of dimension  $N \times N$ . Thus the faces in our training set is given by  $M$ ,  $N \times N$  dimensional vectors,  $\mathbf{I}_j, j = 1, \dots, M$ . In practice is very important that the images in the training set are standardized with respect to size, orientation and intensity. This means that we have to ensure that all the faces are of the same size, at the same angle (upright is good) and has the same lighting intensity. These are nontrivial images processing tasks that we will taken for granted that it can be done.

Figure 3.1 shows two normalized images in our training set

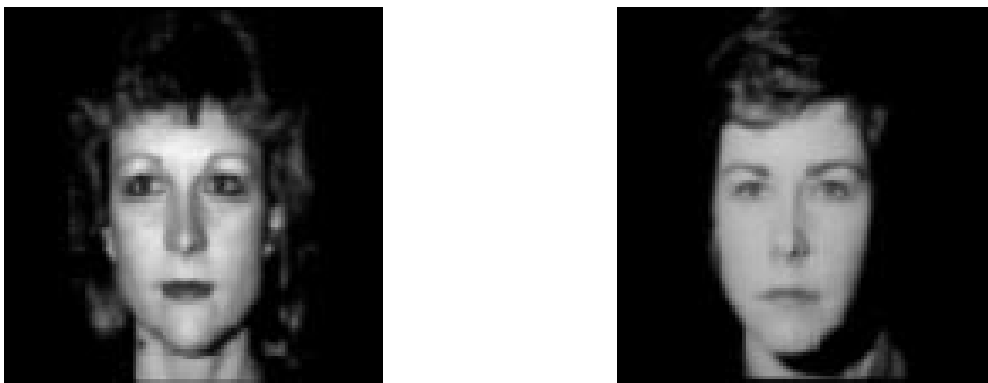


Figure 3.1: Two normalized faces in the training set.

The next step is to calculate an average face. The idea is to capture as much as possible of typical facial features in the average face; hopefully any deviations from the average face will not be too large, leading a more efficient representation. The average face is defined in the standard way as,

$$\mathbf{A} = \frac{1}{M} \sum_{j=1}^M \mathbf{I}_j \quad (3.1)$$

and the normalized deviations from the average face is given by

$$\mathbf{X}_j = \frac{\mathbf{I}_j - \mathbf{A}}{\|\mathbf{I}_j - \mathbf{A}\|}. \quad (3.2)$$

*We are ignoring many technicalities in the construction of the average face. The main idea is to capture all the essential ingredients of a face in the average face and then work with the deviations from the average, all in an attempt to arrive at the most efficient representation of a face. It should be clear that if the average face is to have this property, the faces in the training set should all be well normalized with respect to size, orientation (upright is good!) and even lighting. For a small training set the normalization can be done manually. However, for training sets of realistic size manual manipulation is not feasible and one has to develop an automated procedure, which will in any case be important when we want to compare faces in the future. This entails nontrivial image processing which we will not discuss in any further detail.*

Figure 3.2 shows the average face calculated for for training set according to (1.1).



Figure 3.2: The average face for our training set.

The nonnormalized versions of (3.2) are sometimes referred to as the caricatures of the faces and denoted by

$$\mathbf{C}_j = \mathbf{I}_j - \mathbf{A}. \tag{3.3}$$

The caricatures for the two faces shown in Figure 1.1 are shown in Figure 3.3. Note how the individual characteristics are retained, even enhanced, by the caricatures.



Figure 3.3: The caricatures of the faces of Figure 3.1

We now need to find a basis for the space spanned by the  $\mathbf{X}$ 's. These will become our basic building blocks for reconstructing any face in future, whether it is in the training set or not. One idea is to try and determine the deviation between the faces in the training set. For example we may look for the direction of the maximum average deviation,  $\mathbf{u}_1$ , defined by

$$\lambda_1 = \max_{\|\mathbf{u}_1\|=1} \frac{1}{M} \sum_{n=1}^M (\mathbf{u}_1^T \mathbf{X}_n)^2, \quad (3.4)$$

where  $\lambda_1$  provides a measure of the amount of deviation in the direction of  $\mathbf{u}_1$ . Introducing the covariance matrix,

$$C = \frac{1}{M} \sum_{j=1}^M \mathbf{X}_j \mathbf{X}_j^T \quad (3.5)$$

we rewrite (3.4) as

$$\lambda_1 = \max_{\|\mathbf{u}_1\|=1} \mathbf{u}_1^T C \mathbf{u}_1 \quad (3.6)$$

Note that  $C$  is a  $N^2 \times N^2$  symmetric matrix. In addition it is straightforward to

show that it is positive semi-definite, i.e. all its eigenvalues are nonnegative. In fact a large number of its eigenvalues will turn out to be very small, or zero.

We now look for the maximum deviation orthogonal to  $\mathbf{u}_1$ . Call this direction  $\mathbf{u}_2$  and let  $\lambda_2$  denote the amount of deviation in this direction. Similarly we define the pairs  $(\mathbf{u}_3, \lambda_3)$ , etc. It is not difficult to show using Lagrange multipliers that we need that these quantities are exactly the eigenvalues and eigenvectors of the covariance matrix  $C$ . Thus we need to solve the eigenvalue problem,

$$C\mathbf{u} = \lambda\mathbf{u} \tag{3.7}$$

Before we go any further let us look at the eigenvalues of the covariance matrix constructed from the faces in our training set. Figure 3.4 shows the first, third, twentieth and fortieth eigenvector displayed as a two dimensional image.

For obvious reasons these are called eigenfaces and the complete set become the basic building blocks from which any arbitrary face will be constructed in future. In that sense the eigenfaces are something like a mathematical equivalent of the identikit used law enforcement agencies to reconstruct facial images.

The covariance matrix is very large,  $N^2 \times N^2$  for an original  $N \times N$  image. It has therefore  $N^2$  eigenvectors, or eigenfaces. At this point it is important to recall that the eigenvectors of the covariance matrix,  $C$ , describe the directions of maximum deviation between the faces in the training set and the eigenvalues denote the amount of deviation in these directions. Therefore, directions for which the eigenvalues are negligibly small, carry no information about the faces. Fortunately this true of the majority of the eigenvalues, and those can be safely discarded. This is demonstrated by Figure 3.5 which shows the magnitude of the eigenvalues, listed in order of magnitude.

The most relevant feature of this Figure is the fact that there is a rapid drop in the magnitude of the eigenvalues, becoming negligible after about 50 eigenvalues. Keeping in mind that the eigenvalues measure the deviation in the direction of its



Figure 3.4: The first, the third, the twentieth and the fortieth eigenface (left to right, top to bottom).

associated eigenvector, it should be clear that the eigenvectors (eigenfaces) associated with the higher (essentially zero) eigenvalues, do not carry any significant information and can be discarded. Thus we find that all the faces in the training set can be represented by about 50 eigenfaces. The next section explains how we set out to do this. If the training set used for these experiments were really representative, the results summarized in Figure 1.5 would indicate that all faces can be accurately represented by about 50 eigenfaces. However, this is not the case; our training set is not really representative but more extensive experiments with a heterogeneous population as found for instance in South Africa, indicates that a number of eigenfaces slightly in excess of 100 should suffice. There is no doubt that

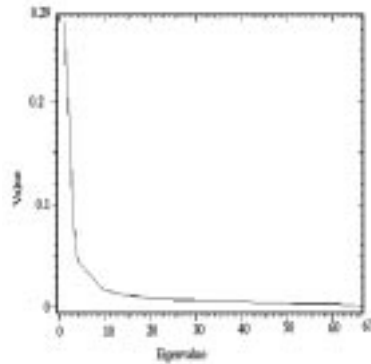


Figure 3.5: The magnitude of the eigenvalues, listed in the order of magnitude.

a eigenface representation of facial images is highly effective. We have shown that the dimension of the facial subspace is in the order of 100, rather than  $N^2$  as for a general  $N \times N$  image.

Two basic question still need to be addressed:

1. How do we compute the eigenvalues and eigenvectors of the covariance matrix  $C$ , recalling that  $C$  is very large.?
2. How do we represent a given face in terms of the eigenfaces?

The second of these questions is the easier to answer and will be dealt with first in the next section.