

3.5 Updating the training set.

The range of faces which can be described by our eigenfaces is determined by the training set. Only if our training set is sufficiently representative will we be able to represent any face well. If a particular image is poorly represented, we will have to modify our training set and thus our eigenfaces to describe this new image. Naturally, we would like to do this as efficiently as possible.

In figure 3.14, we show the original image of a face that is poorly represented by our set of eigenfaces.



Figure 3.14: Original image, poorly described by training set.

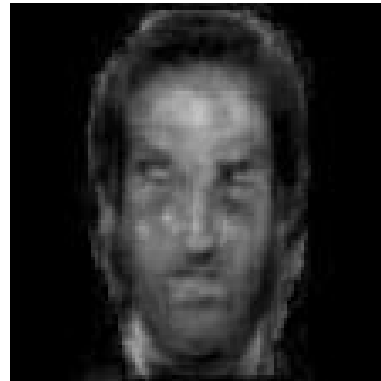


Figure 3.15: Reconstruction of image poorly represented by training set.

The reconstruction is shown in figure 3.19. This is clearly unacceptable and we need to modify our set of eigenfaces to describe this new face. Figure ?? shows the error in using this representation. Note that the information needed to recognize the individual is lost in this reconstruction.

We obviously need to recalculate the eigenfaces. This could easily be done by simply adding the new face to our training set and recalculate all the eigenfaces. However, this seems very wasteful because the eigenfaces should not really change all that much because of the addition of a single face to the training set. Instead of the lengthy recalculation, we now describe a more efficient method of updating the eigenfaces.

3.5.1 Perturbation Analysis

Assuming that the eigenvalues and eigenvectors are not going to change too much by the addition of the new face, one might contemplate using perturbation analysis to calculate the effect of the addition. The main idea is to consider the addition of the new face to the training set as a small perturbation of the covariance matrix. The question then becomes how much does a small perturbation of a symmetric matrix, affect the eigenvalues and eigenvectors. Accordingly, we consider the perturbed eigenvalue problem

$$(C + \delta C)(\mathbf{u}_l + \delta \mathbf{u}_l) = (\lambda_l + \delta \lambda_l)(\mathbf{u}_l + \delta \mathbf{u}_l) \quad (3.18)$$

where δC is the known change in C , $\delta \lambda_l$ the unknown change in λ_l and $\delta \mathbf{u}_l$ the unknown change in \mathbf{u}_l .

Expanding (3.18) and keeping the first order terms leads to

$$(C - \lambda_l I) \delta \mathbf{u}_l = (-\delta C + \delta \lambda_l I) \mathbf{u}_l. \quad (3.19)$$

For a solution to exist, the right hand side must be an element of the column space of $C - \lambda_l I$. Thus, by the Fredholm alternative, it is orthogonal to the null-space of $C - \lambda_l I$. But the null-space of $C - \lambda_l I$ is the eigenvector \mathbf{u}_l . Since the eigenvectors are orthogonal, it follows that

$$\mathbf{u}_l^T (-\delta C + \delta \lambda_l I) \mathbf{u}_l = 0$$

or

$$\delta \lambda_l = \mathbf{u}_l^T \delta C \mathbf{u}_l. \quad (3.20)$$

This is an expression for the change in the eigenvalue. Substituting back into (3.20), one can use the pseudoinverse to calculate the change in the eigenvectors. Let us investigate how this can be done in practice.

3.5.2 Reduction

As before C is a $N^2 \times N^2$ matrix, for an $N \times N$ image, and again we need an efficient procedure to deal with matrices this big. We did manage before when we calculated the eigenfaces in the first place. Those ideas will prove useful once more.

First note that our assumption that P eigenfaces describe every face in the original training set accurately is equivalent to assuming that

$$C \approx \lambda_1 \mathbf{u}_1 \mathbf{u}_1^T + \lambda_2 \mathbf{u}_2 \mathbf{u}_2^T + \cdots + \lambda_P \mathbf{u}_P \mathbf{u}_P^T. \quad (3.21)$$

Let us also assume that the perturbation δC can be written as,

$$\delta C \approx \boldsymbol{\phi} \boldsymbol{\phi}^T \quad (3.22)$$

for some unknown $\boldsymbol{\phi}$, determined by the additional face.

Assuming that the new eigenvectors are some combination of the old eigenvectors and of $\boldsymbol{\phi}$, we write

$$\tilde{\mathbf{u}}_l = \sum_{j=1}^P v_{jl} \sqrt{\lambda_l} \mathbf{u}_j + v_{(P+1)l} \boldsymbol{\phi}$$

where $\tilde{\mathbf{u}}_l = \mathbf{u}_l + \delta \mathbf{u}_l$ is the l 'th eigenvector of the new system.

We substitute this into our perturbed eigenvector equation (3.18) and use (3.21) and (3.22) to obtain

$$\begin{aligned} & \left(\lambda_l \mathbf{u}_l \mathbf{u}_l^T + \cdots + \lambda_p \mathbf{u}_p \mathbf{u}_p^T + \boldsymbol{\phi} \boldsymbol{\phi}^T \right) \left(v_{1l} \sqrt{\lambda_l} \mathbf{u}_1 + \cdots + v_{(P+1)l} \boldsymbol{\phi} \right) \\ & = (\lambda_l + \delta \lambda_l) \left(v_{1l} \sqrt{\lambda_l} \mathbf{u}_1 + \cdots + v_{(P+1)l} \boldsymbol{\phi} \right). \end{aligned}$$

Using similar arguments as in section 3.4, it follows that $\mathbf{v}_l = \begin{bmatrix} v_{1l} & \cdots & v_{(P+1)l} \end{bmatrix}$ must satisfy the eigenvalue equation

$$K^{new} \mathbf{v}_l = (\lambda_l + \delta \lambda_l) \mathbf{v}_l \quad (3.23)$$

where

$$K^{new} = \begin{bmatrix} \lambda_1 & \cdots & 0 & \sqrt{\lambda_1} \mathbf{u}_1^T \boldsymbol{\phi} \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \lambda_P & \sqrt{\lambda_P} \mathbf{u}_P^T \boldsymbol{\phi} \\ \sqrt{\lambda_1} \mathbf{u}_1^T \boldsymbol{\phi} & \cdots & \sqrt{\lambda_P} \mathbf{u}_P^T \boldsymbol{\phi} & \boldsymbol{\phi}^T \boldsymbol{\phi} \end{bmatrix}. \quad (3.24)$$

This matrix is only $(P + 1) \times (P + 1)$, which represents a considerable reduction in the size of the problem. It also has a sparse structure that can be exploited during the calculation of the new eigenvectors and eigenvalues.

It is not easy to find an expression for $\boldsymbol{\phi}$. However, heuristic arguments as well as numerical experiments indicate that the following choice is not unreasonable

$$\boldsymbol{\phi} = \frac{1}{\sqrt{M+1}} \widetilde{\mathbf{X}}_{M+1}. \quad (3.25)$$

All that remains to be done is to update the eigenfaces.

3.5.3 Efficient Recalculation of the Eigenfaces

Equation (3.25) allows us to approximate δC and perturbation analysis related this to a change in the eigenvalues through (3.20). Thus we find that

$$\delta \lambda_j \approx \mathbf{u}_j^T (\delta C) \mathbf{u}_j = \frac{1}{M+1} (\mathbf{u}_j^T \widetilde{\mathbf{X}}_{M+1})^2.$$

Making use of this, we obtain a good approximation of the eigenvalues of the eigenvalue problem, (3.23). Thus the standard inverse power method should rapidly yield the eigenvectors ($1 \leq j \leq P$) of the sparse matrix (3.24), see for example Golub and Van Loan [??] or Trefethen and Bau [??].

The special sparse structure of K^{new} in 3.24) is easily exploited for an efficient implementation of the inverse power method. The easiest way of seeing this is perhaps by noting that the LU decomposition of K^{new} inherits its sparse structure and is explicitly given by,

$$L = \begin{bmatrix} 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \\ \frac{1}{\lambda_1} \sqrt{\frac{\lambda_1}{M+1}} \mathbf{u}_1^T \widetilde{\mathbf{X}}_{M+1} & \cdots & \frac{1}{\lambda_P} \sqrt{\frac{\lambda_P}{M+1}} \mathbf{u}_P^T \widetilde{\mathbf{X}}_{M+1} & 1 \end{bmatrix}$$

and

$$U = \begin{bmatrix} \lambda_1 & \cdots & 0 & \sqrt{\frac{\lambda_1}{M+1}} \mathbf{u}_1^T \widetilde{\mathbf{X}}_{M+1} \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \lambda_P & \sqrt{\frac{\lambda_P}{M+1}} \mathbf{u}_P^T \widetilde{\mathbf{X}}_{M+1} \\ 0 & \cdots & 0 & Z(U) \end{bmatrix}$$

where

$$Z(U) = \frac{1}{M+1} \left(1 - \left(\mathbf{u}_1^T \widetilde{\mathbf{X}}_{M+1} \right)^2 - \dots - \left(\mathbf{u}_P^T \widetilde{\mathbf{X}}_{M+1} \right)^2 \right).$$

Both these matrixes are sparse and operations involving these can be done efficiently. Of course, implementing the inverse power method actually involves the decomposition of the matrix $K^{new} - \widehat{\lambda}_l I$, where $\widehat{\lambda}_l$ is our approximation to the l 'th eigenvalue and its LU decomposition is left as an exercise (the structure remains the same!).

We do not discuss the slightly more complicated situation when we have degenerate eigenvalues.

However, the above approximation of the changes in the eigenvalues only gives us P eigenvalues and eigenvectors and we need to find $P+1$ eigenvectors. In order to find the final eigenvalue, we note that, the eigenvalues of $C + \delta C$ and K^{new} are the same. Thus

$$\text{Tr}(C + \delta C) = \text{Tr} K^{new}$$

which implies that

$$(\lambda_1 + \delta\lambda_1) + \cdots + (\lambda_P + \delta\lambda_P) + \lambda_{P+1} = \lambda_1 + \cdots + \lambda_P + \frac{1}{M+1}$$

and

$$\frac{1}{M+1} - \delta\lambda_1 - \dots - \delta\lambda_P = \lambda_{P+1}.$$

This allows us to calculate the last eigenvalue, from which we can easily find the last eigenvector, which is the final eigenface.

We end this section with a numerical example of the updating procedure. Returning to the image shown in Figure 3.14, we calculated its eigenface representation, by using two different sets of eigenfaces. The first set of eigenfaces was obtained by adding the face of Figure 3.14 to the training set and doing a full recalculation of all the eigenfaces. In this case the representation should be exact (except for negligible errors due to discarding the higher order eigenfaces), and the result is shown in Figure 3.16.



Figure 3.16: Representation of 3.18 using a fully recalculated eigenfaces.

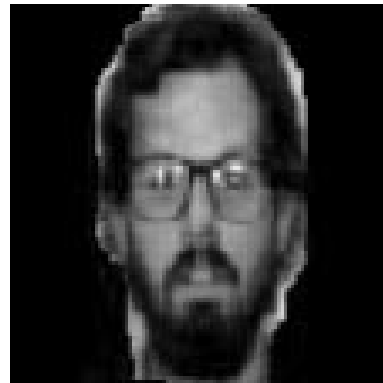


Figure 3.17: Representation of 3.14 using the updating procedure of this section.

The second set of eigenfaces was obtained by the updating procedure described in this section and the result is shown in Figure 3.17. There is no discernible difference between the two representations.