PCA without eigenvalue calculations: a case study on face recognition*

E. Kokiopoulou and Y. Saad
Comp. Sci. & Eng. Dept
University of Minnesota
{kokiopou,saad}@cs.umn.edu

Abstract

Principal component analysis (PCA) is an extensively used dimensionality reduction technique, with important applications in many fields such as pattern recognition, computer vision and statistics. It employs the eigenvectors of the covariance matrix of the data to project it on a lower dimensional subspace.

However, the requirement of PCA eigenvectors is a computational bottleneck which poses serious challenges and limits the applicability of PCA-based methods, especially for real-time computations. This paper proposes an alternative framework, relying on polynomial filtering which enables efficient implementations of PCA. We showcase the applicability of the proposed scheme on face recognition. In particular, we consider the eigenfaces methods which employ PCA. The numerical experiments reported indicate that the proposed technique competes with the PCA-based method in terms of recognition rate, while being much more efficient in terms of computational and storage cost.

Keywords Principal Component Analysis, Polynomial Filtering, Face Recognition.

1 Introduction

Principal component analysis (PCA) [5] is one of the most popular dimensionality reduction techniques. It has numerous applications in many areas such as pattern recognition, computer vision, statistics and data analysis. PCA has been successfully applied in automated face recognition [14], resulting in the so called method of *eigenfaces* introduced by Kirby and Sirovich [6], Sirovich and Kirby [12] and Turk and Pentland [10], [13]. The eigenfaces method is one of the most popular appearance-based holistic approaches (see e.g., [1], [13]) which employs PCA on the covariance matrix C, constructed by the training data.

Typical implementations of the eigenfaces method rely upon eigendecomposition of the covariance matrix. However, when the datasets are dynamic and of large scale, the applicability of the above methods is limited due to their high computational cost (which is $O(n^3)$ for dense matrices). This is even more evident in the case of real-time and adaptive algorithms (see e.g. [9]). In these cases, the eigendecomposition must be updated frequently and the time constraints are very strict. To that end, a lot of research efforts have been devoted to efficient eigenspace update schemes such as the one proposed in [4].

In this paper we propose an alternative implementation scheme which approximates directly the similarity score without computing the eigendecomposition of C or any other matrix decomposition. Denoting by A the data matrix in the input space, the new method relies on polynomial filtering, where a well defined polynomial ψ of the matrix $AA^\top$ or $A^\top A$ is applied on the new face image and yields an approximation to the similarity score that is very close to the one obtained using eigendecomposition. The polynomial ψ is chosen appropriately such that it is a good approximation of the step function.

The polynomial filtering framework was applied successfully in [7] for dimensionality reduction in information retrieval. In this paper we showcase the applicability of this technique in a different context, that of face recognition. We claim that the proposed framework can be applied in any method employing PCA to estimate similarities among data vectors. Numerical experiments indicate that the proposed framework is quite close to the PCA methods in terms of recognition rate without suffering from their computational and storage limitations.

The remaining sections of this paper are organized as follows: Section 2 provides an overview of the eigenfaces method using eigenvalue decomposition. In Section 3 the eigenfaces method is interpreted in terms of

*Work supported by the Minnesota Supercomputing Institute

Singular Value Decomposition (SVD). Next, in Section 4 the implementation of face recognition using eigenfaces, via polynomial filtering is described. Finally, Section 5 provides a series of numerical results verifying the practical advantages of the proposed scheme.

2 The method of eigenfaces

2.1 Construction of the face space Suppose that a face image consists of N pixels, so it can be represented lexicographically by a vector x of dimension N. Let $\{x_i | i = 1, \ldots, M\}$ be the training set of face images. The mean face is given by

$$\mu = \frac{1}{M} \sum_{i=1}^{M} x_i. \tag{2.1}$$

The covariance matrix of the translated training data is

$$C = \frac{1}{M} A A^\top \in R^{N \times N}, \tag{2.2}$$

where $A = [\tilde{x}_1, ..., \tilde{x}_M] \in R^{N \times M}$ is the matrix of the translated data points

$$\tilde{x}_i = x_i - \mu, \ i = 1, \ldots, M. \tag{2.3}$$

The eigenvectors u_l, $l = 1, \ldots, M$ of the covariance matrix C are usually called "eigenfaces", since they resemble faces when reshaped and illustrated in a pictorial fashion. In practice only a small number, say k, of eigenvectors corresponding to the largest eigenvalues are computed and then used for performing Principal Component Analysis (PCA) for face identification. The subspace spanned by the eigenfaces is called face space.

2.2 Face recognition using eigenfaces The face recognition procedure consists of two stages; the training stage and the recognition stage. In the training stage each face image x_i of the known individuals is projected on the face space and a k-dimensional vector P_i is obtained

$$P_i = U_k^\top (x_i - \mu), \ i = 1, \ldots, M, \tag{2.4}$$

where $U_k = [u_1, \ldots, u_k]$ is the matrix with orthonormal columns, which are the eigenvectors associated with the k largest eigenvalues.

In the recognition stage, the new image $x \in R^N$ to be processed, is translated and then projected into the face space to obtain the vector

$$P_x = U_k^\top (x - \mu). \tag{2.5}$$

The distance between P_x and each face image is defined by

$$\begin{aligned} d_i^2 &= \|P_x - P_i\|_2^2 \\ &= \|P_x\|_2^2 + \|P_i\|_2^2 - 2P_x^\top P_i, \ i = 1, \ldots, M, \end{aligned} \tag{2.6}$$

where $\|.\|_2$ is the Euclidean norm. Furthermore, in order to discriminate between face images and non-face images, the distance ϵ between the original image x and its reconstructed image from the face space, $x_f = U_k P_x + \mu$, is also computed:

$$\epsilon = \|x - x_f\|_2. \tag{2.7}$$

Note in passing that

$$\begin{aligned} \epsilon &= \|x - \mu - U_k P_x\|_2 \\ &= \|(x - \mu) - U_k U_k^\top (x - \mu)\|_2, \end{aligned}$$

and therefore ϵ represents simply the distance between $x - \mu$ and its orthogonal projection onto span$\{\mathrm{U_k}\}$, i.e.,

$$\epsilon^2 = \|(I - U_k U_k^\top)(x - \mu)\|_2^2 \tag{2.8}$$

$$= \|x - \mu\|_2^2 - \|P_x\|_2^2. \tag{2.9}$$

This metric is used to decide whether or not a given image is a face.

3 Eigenfaces in terms of the SVD

In this section we interpret the above training and recognition stages in terms of the truncated singular value decomposition of A. The SVD [3] of a rectangular $N \times M$ matrix A of rank r, is defined as

$$A = U \Sigma V^\top, \tag{3.10}$$

$$U^\top U = I_N \in R^{N \times N}, \tag{3.11}$$

$$V^\top V = I_M \in R^{M \times M}, \tag{3.12}$$

where $U = [u_1, \ldots, u_N]$ and $V = [v_1, \ldots, v_M]$ are unitary matrices and $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_M)$, $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > \sigma_{r+1} = \ldots = \sigma_M = 0$. The σ_i's are the *singular values* of A and the u_i's and v_i's are respectively *the left and right singular vectors* associated with σ_i, $i = 1, \ldots, r$. We define the i-th singular triplet of A as $\{u_i, \sigma_i, v_i\}$. It follows from the SVD that the matrix A can be expressed as a sum of r rank-one matrices,

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^\top \ .$$

Additionally, it is well known that

$$\min_{\mathrm{rank}(B) \leq k} \|A - B\|_F = \|A - A_k\|_F$$

where $A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^\top$ and $\|.\|_F$ is the Frobenius norm. It is helpful for what follows to rewrite the matrix A_k as

$$A_k = U_k \Sigma_k V_k^\top, \tag{3.13}$$

where U_k (resp. V_k), consists of the first k columns of U (resp. V), and Σ_k is a diagonal matrix of size $k \times k$.

Thus, if we truncate the SVD to keep only the k largest singular triplets we obtain the closest (in a least-squares sense) approximation to A.

Observe that the matrix U_k containing the k largest left singular vectors of $\tilde{A} = \frac{1}{\sqrt{M}}A$, is exactly the matrix computed by PCA containing the largest eigenvectors of the covariance matrix. This follows from the fact that

$$C = \tilde{A}\tilde{A}^\top = U\Sigma V^\top V\Sigma^\top U^\top = U\Sigma\Sigma^\top U^\top,$$

is the eigendecomposition of the covariance matrix. Using this observation, equation (2.4) can be written in the form

$$\begin{aligned} P_i &= U_k^\top \tilde{x}_i = U_k^\top \tilde{A} e_i \\ &= U_k^\top [U_k\ U_{N-k}] \begin{bmatrix} \Sigma_k & 0 \\ 0 & \Sigma_{M-k} \end{bmatrix} \begin{bmatrix} V_k^\top \\ V_{M-k}^\top \end{bmatrix} e_i \\ &= [I_k\ 0] \begin{bmatrix} \Sigma_k V_k^\top \\ \Sigma_{M-k} V_{M-k}^\top \end{bmatrix} e_i \\ &= \Sigma_k V_k^\top e_i,\ i = 1, \ldots, M. \end{aligned}$$

Denote by $P = \Sigma_k V_k^\top$ the matrix whose columns are the projections $P_i,\ i = 1, \ldots, M$, of every known face image to the face space. Assuming that all vectors are normalized, the similarity measurement (2.6) among the new image x and all known images, can be equivalently computed by the similarity vector s_k,

$$\begin{aligned} s_k &= P^\top P_x = V_k \Sigma_k^\top U_k^\top (x - \mu) \\ &= \tilde{A}_k^\top (x - \mu), \end{aligned} \tag{3.14}$$

containing a similarity score between the new face image and each of the known images. Thus, the computation of the similarity vector s_k employs a rank k approximation of the translated matrix A. We discuss the assumption of normalized projected vectors in the following section.

Note also that using the SVD, equation (2.8) expresses the metric ϵ as the distance from $x - \mu$ to the space span$\{\mathrm{U_k}\}$ of the dominant left singular space. In the sequel, we show how to approximate the similarity vector s_k in (3.14), as well as the distance ϵ in (2.8) without using eigendecompositions. The proposed scheme relies on polynomial filtering.

4 Eigenfaces using polynomial filtering

Polynomial filtering allows to closely approximate the effect of reduced rank approximation used in PCA models. Denote by $\psi(A)$ a matrix polynomial of degree d on the matrix A, i.e.,

$$\psi(A) = \xi_d A^d + \xi_{d-1} A^{d-1} + \ldots + \xi_1 A + \xi_0 I.$$

Assuming that A is normal (i.e., $A^\top A = AA^\top$) and letting $A = Q\Lambda Q^\top$ be its eigendecomposition, observe that $\psi(A) = \psi(Q\Lambda Q^\top) = Q\psi(\Lambda)Q^\top$. Therefore, the polynomial on A is translated to a polynomial on its eigenvalues. We are now ready to describe how one can use polynomial filtering to approximate the similarity vector directly, avoiding completely eigenvalue computations.

Let $\tilde{x} = x - \mu$ be the translated new image. In order to estimate the similarity measurement, we use a polynomial ψ of $\tilde{A}^\top\tilde{A}$ such that

$$\begin{aligned} s &= \psi(\tilde{A}^\top\tilde{A})\tilde{A}^\top\tilde{x} \\ &= \psi(V\Sigma^\top\Sigma V^\top)V\Sigma^\top U^\top\tilde{x} \\ &= V\psi(\Sigma^\top\Sigma)V^\top V\Sigma^\top U^\top\tilde{x} \\ &= V\psi(\Sigma^\top\Sigma)\Sigma^\top U^\top\tilde{x}. \end{aligned} \tag{4.15}$$

Compare the last expression above with (3.14). Choosing the polynomial $\psi(t)$ appropriately will allow us to interpretate this approach as a compromise between the correlation [2] and the PCA approaches. Assume now that ψ is not restricted to being a polynomial but can be any function (even discontinuous). When $\psi(t) = 1\ \forall x$, then $\psi(\Sigma^\top\Sigma)$ becomes the identity operator and the above scheme would be equivalent to the correlation method. On the other hand, taking ψ to be the step function

$$\psi(t) = \begin{cases} 0, & 0 \le t \le \sigma_k^2 \\ 1, & \sigma_k^2 \le t \le \sigma_1^2 \end{cases} \tag{4.16}$$

results in $\psi(\Sigma^\top\Sigma) = \begin{bmatrix} I_k & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$ where I_k is the identity matrix of size k and $\mathbf{0}$ is a zero matrix of an appropriate size. Then, equation (4.15) may be re-written as:

$$\begin{aligned} s &= V\psi(\Sigma^\top\Sigma)\Sigma^\top U^\top\tilde{x} \\ &= \begin{bmatrix} V_k & V_{n-k} \end{bmatrix} \begin{bmatrix} \Sigma_k^\top & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} U_k^\top \\ U_{m-k}^\top \end{bmatrix} \tilde{x} \\ &= \begin{bmatrix} V_k\Sigma_k^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} U_k^\top \\ U_{m-k}^\top \end{bmatrix} \tilde{x} \\ &= V_k\Sigma_k^\top U_k^\top\tilde{x} \\ &= \tilde{A}_k^\top\tilde{x} \end{aligned} \tag{4.17}$$

which is precisely the rank-k approximation provided in equation (3.14).

Using polynomial filtering we can also approximate the "faceness" (i.e., whether or not a given image contains a face) of an image as it is expressed by equation (2.8). Using the SVD, observe that

$$\begin{aligned} \psi(C)(x - \mu) &= \psi(\tilde{A}\tilde{A}^\top)(x - \mu) \\ &= \psi(U\Sigma V^\top V\Sigma^\top U^\top)(x - \mu) \\ &= U\psi(\Sigma\Sigma^\top)U^\top(x - \mu). \end{aligned} \tag{4.18}$$

Note that if ψ is exactly the step function (4.16), then $\|\psi(C)(x-\mu)\|_2 = \|U_k U_k^\top (x-\mu)\|_2 = \|P_x\|_2$ which would

allow to obtain ϵ from (2.8). If the polynomial ψ is an approximation of the step function, this will provide an estimate of the distance metric ϵ, needed to decide on the faceness of an image, without the availability of U or U_k.

Therefore, the approach of polynomial filtering in PCA models can give virtually the same result as eigendecomposition, without resorting to the costly eigenvalue decomposition or any other matrix decomposition. Furthermore, the need to store additional (dense or sparse) matrices as is the case in PCA, is completely avoided as is the need to update these matrices, when the subspace used for learning changes dynamically. The selection of the cut-off point is somewhat similar to the issue of choosing the parameter k in the PCA method. However, there is a salient difference between the two: choosing a large k in PCA may render the method much more expensive, while selecting a high cut-off in polynomial filtering does not affect cost significantly.

Recall that in the computation of the similarity vector we assumed that the projected vectors P_i have unity norm. Here are two solutions to overcome this problem. Before applying the proposed scheme we normalize all input data vectors x_i. Next, we compute the similarity score and sort the samples in descending order. Then we have two options. Using the first $k \ll M$ samples, either we can employ PCA or we can use k-nearest neighbor classification. Observe that since $k \ll M$, the cost of exact PCA will be very limited, and certainly orders of magnitude smaller than PCA on the original data matrix. Similarly, applying k-nearest neighbor classification on a very small set of data points will have very limited cost. We observed empirically that the first option yields slightly better results and this is the option that we included in our experiments (Section 5) with $k = 30$.

5 Numerical results

All experiments are implemented in `MATLAB` 6.5 on a Xeon@2.4GHz. We use three datasets that are publically available: YALE, ORL and a subset of AR. The YALE database [1] contains 165 images of 15 individuals that include variation in both facial expression and lighting. In the preprocessing phase, each face image is closely cropped, and the size of images after the cropping phase is decreased to 112×92. The ORL (formerly Olivetti) database [11] contains 40 individuals and 10 different images for each individual. In this case no preprocessing is done. Finally, the AR face database [8] contains 126 subjects with 4 different facial expressions for each individual.

In what follows, error rates are estimated using a cross validation "leave-one-out" strategy. In order to compute the error rate with respect to a certain facial expression, the image associated with it is used as a test image. In order to recognize the test image, all images, excluding the test one, are projected to the reduced subspace. Then, the test image is projected as well and recognition is performed using a nearest neighbor rule. Denote by e_i as the number of misses counted across the subjects for a given facial expression i. Denote also by N_f the number of different facial expressions/poses associated with each individual in the database. Define $e = \frac{1}{N_f} \sum_{i=1}^{N_f} e_i,\ i = 1, ..., N_f$. Thus, e is the mean error rate averaged across all different facial expressions. In what follows, denote by PCA the "eigenfaces" method and by PPF the polynomial filtering method.

$k = 40$	ORL (%)	YALE (%)	AR (%)
γ=2	2.5	26.06	8.33
γ=3	3.5	25.45	8.53
γ=4	2.75	26.06	7.14
γ=5	3	26.06	6.15

Table 1: Error rates of the PPF method for various values of γ, on all face databases.

Example 1 In the first example we investigate the behavior of the PPF method with respect to the degree of the polynomial ψ. Table 1 illustrates the error rate of PPF with respect to γ. The parameter γ affects the degree of the polynomial approximation to the step function. The higher the value of γ the higher the degree of the polynomial. Observe that in most cases the value $\gamma = 4$ seems to give the most satisfactory results. To that end, in what follows, we use $\gamma = 4$ for PPF.

Example 2 We now investigate the effect of the dimension k of the reduced space on the recognition performance of the methods. We use `MATLAB`'s `svd` builtin function since the matrix is dense and this way we avoid the explicit use of the matrices $AA^\top$ or $A^\top A$. We experiment with $k = 20 : 20 : 100$ (in `MATLAB` notation) and measure the error rate (%) for all face databases.

Table 2 illustrates the error rate e versus the dimension k measured on the ORL, YALE and AR datasets respectively. All tables contain the corresponding time measurements t (in sec) for each method. The timings for PCA methods measure the time needed to construct the subspace (i.e., computing the eigenvectors) and perform the recognition of the test image (i.e., one step of "leave-one-out" cross validation). The timings for PPF methods measure the time needed to recognize the test data point via polynomial filtering.

Concerning the ORL database, observe that PPF

ORL	PCA		PPF	
	e	t	e	t
k=20	3.5	32.74	3	2.52
k=40	2.75	30.68	2.75	2.49
k=60	3.25	30.93	3.25	2.48
k=80	3.25	32.96	3	2.52
k=100	3	32.03	3	2.49
YALE	PCA		PPF	
	e	t	e	t
k=20	29.70	5.93	25.45	1.15
k=40	27.88	6.02	26.06	1.16
k=60	27.27	6.10	25.45	1.14
k=80	27.27	6.22	25.45	1.16
k=100	26.06	6.33	25.45	1.15
AR	PCA		PPF	
	e	t	e	t
k=20	8.34	82.02	6.35	5.71
k=40	6.75	82.02	7.34	5.71
k=60	6.15	83.12	7.14	5.71
k=80	6.15	83.67	6.75	5.70
k=100	5.75	83.64	6.35	5.71

Table 2: Error rates e (%) and timings t (in sec) of both methods for various values of k, on all the face databases.

competes with PCA in terms of error rate. Furthermore, the PPF method is much more efficient achieving significant speedups over its PCA counterpart. On the YALE dataset, the results are quite similar with PPF outperforming PCA not only in timings but in error rate as well. Finally, on the AR dataset, the results are similar to ORL, with the PPF methods being quite close to PCA in terms of error rate and being much more efficient in terms of computational cost.

6 Conclusion

We have described an alternative framework for implementing PCA without eigenvalue calculations. The proposed framework relies on polynomial filtering, in order to render the same effect as PCA, for dimensionality reduction. We illustrated the applicability of the proposed technique in the eigenfaces method for face recognition. The numerical experiments indicated that the new scheme has very close performance to the PCA method, while being much more efficient in terms of computational cost and storage.

7 Acknowledgments

We would like to thank prof. N. Papanikolopoulos for his valuable advice concerning face recognition issues.

References

[1] P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *IEEE Trans. Pattern Analysis and Machine Intelligence, Special Issue on Face Recognition*, 19(7):711—20, July 1997.

[2] R. Brunelli and T. Poggio. Face recognition: Features vs Templates. *IEEE Trans. Patt. Anal. Mach. Intell.*, 15(10):1042–1053, 1993.

[3] G. H. Golub and C. Van Loan. *Matrix Computations, 3rd edn.* The John Hopkins University Press, Baltimore, 1996.

[4] L. Hoegaerts, L. De Lathauwer, J.A.K. Suykens, and J. Vanderwalle. Efficiently updating and tracking the dominant kernel eigenspace. *16th International Symposium on Mathematical Theory of Networks and Systems*, July 5-9 2004. Belgium.

[5] I.T. Jolliffe. *Principal Component Analysis.* Springer Verlag, New York, 1986.

[6] M. Kirby and L. Sirovich. Application of the Karhunen-Loeve Procedure for the Characterization of Human Faces. *IEEE Trans. Patt. Anal. Mach. Intell.*, 12, 1990.

[7] E. Kokiopoulou and Y. Saad. Polynomial Filtering in Latent Semantic Indexing for Information Retrieval. In *ACM-SIGIR Conference on research and development in information retrieval*, Sheffield, UK, July 25th-29th 2004.

[8] A.M. Martinez and R. Benavente. The AR Face Database. Technical report, CVC no. 24, 1998.

[9] S.K. Nayar, S.A. Nene, and H. Murase. Subspace Methods for Robot Vision. *IEEE Transactions on Robotics and Automation*, 12(5):750–758, October 1996.

[10] A. Pentland, B. Moghaddam, and T. Starner. View-based and Modular Eigenspaces for Face Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[11] F. Samaria and A. Harter. Parameterisation of a Stochastic Model for Human Face Identification. In *2nd IEEE Workshop on Applications of Computer Vision*, Sarasota FL, December 1994.

[12] L. Sirovich and M. Kirby. Low-dimensional Procedure for the Characterization of Human Faces. *J. Optical Soc. Am. A (JOSA A)*, 4(3):519–524, March 1987.

[13] M. Turk and A. Pentland. Face Recognition using Eigenfaces. In *Int. Conf. on Patt. Recog.*, pages 586–591, 1991.

[14] W. Zhao, R. Chellapa, P. Phillips, and A. Rosenfeld. Face Recognition: a Literature Survey. *ACM Computing Surveys*, 35(4):399–458, December 2003.