A Comparative Study of Principal Component Analysis Techniques

Rafael A. Calvo* Matthew Partridge
Marwan A. Jabri
SEDAL
Department of Electrical Engineering
University of Sydney
NSW, 2006, Australia
{rafa,mp,marwan}@sedal.usyd.edu.au

Abstract

Principal Component Analysis (PCA) is a useful technique for reducing the dimensionality of datasets for compression or recognition purposes. Many different methods have been proposed for performing PCA. This study aims to compare these methods by analysing the solutions which these methods find. We have estimated the correlation between these solutions and produced the errors using bootstrap resampling.

KEYWORDS: Principal component analysis, bootstrapping, Image processing.

1 Introduction

High dimensional problems are becoming increasingly common. With high dimensional data, it is difficult to understand the underlying structure: it is difficult to "see the wood for the trees". Additionally, the storage, transmission and processing of high dimensional data places great demands on systems. All these are aspects of one of the most interesting computational and data analysis problems. A first solution is to reduce the dimensionality of the data, whilst maintaining as much of its original structure. The second solution is to increase the computational power by means of parallel implementations. The compression process also has large time and memory requirements, parallel implementations of the compression algorithms have been developed and are described in an accompanying paper [9].

*Permanent address: Instituto de Fisica Rosario, Bvd. 27 de Febrero 210 Bis, 2000 Rosario, Argentina.

A second statistical problem is that the compression parameters are *estimated* for a particular sample. Their deviation depends on the amount of data used, but usually large databases are not available or are expensive to obtain. The analytical estimation of the errors is not possible. In order to solve this problem resampling techniques are used.

Since the beginning of this century, several researchers have worked on the first problem (for example, [2], [7], [11] and [6]), of developing dimensionality reduction techniques; principal component analysis (PCA) is one of these techniques. In mathematical terms, n correlated random variables are transformed into a set of $d \leqslant n$ uncorrelated variables. These uncorrelated variables are linear combinations of the original variables and can be used to express the data in a reduced form. Data modelling and pattern recognition are better able to work on this reduced form, and the form is efficient for storage and transmission. PCA is also sometimes used as a data visualization technique since high dimensional datasets can be reduced to a low dimension and then plotted. Recently, adaptive methods such as Hebbian Neural Networks [4] and Simple PCA [10] have been proposed to handle situations in which the dimensionality of the data is high, or there is a time constraint for example in real-time systems.

The solutions obtained by these methods might be somewhat different due to the iterative and the generalization properties of each. It is of interest to perform a comparison of the solutions obtained, and to consider the generalisation capabilities of the methods for different samples of the same distribution. Several resampling techniques have resently been developed [5], and can be used to obtain more information out of the available data sets.

Section 2 gives an overview of work on matrix and data oriented techniques for PCA. Section 3 describes the methods which we use to compare the different techniques and in Section 4 we show the comparative results of the different methods on a real world dataset. Finally, Section 5 concludes.

2 Dimensionality Reduction Methods

Two types of methods have been used for PCA. Firstly, there are the more conventional *matrix methods*, in which all the data are used to estimate the variance-covariance structure and express it in a matrix. In practice this usually means that the matrix is diagonalized using some numerical technique such as singular value decomposition ([1] and [12]), or Hotelling's power method.

We call the second type *data methods* since they work directly with the data. They might be implemented adaptively so the directions of the PCs are adjusted after a new datum is received, without the need for reusing all the data. This approach is suitable for real-time applications or for very high dimensional problems where the computational expense is an important consideration. Neural networks with Hebbian learning have been proposed for adaptive PCA [4]. Simple PCA [10] which is a faster method that does not require learning parameters has also been developed.

2.1 Matrix Methods

Most multivariate analysis textbooks (for example [1], [3], [8] and [13]) describe matrix methods for performing PCA. The goal is to find the eigenvectors of the covariance matrix. These eigenvectors correspond to the directions of the principal components of the original data, their statistical significance is given by their corresponding eigenvalues. In more detail, these techniques can be structured as:

1. Collect $\mathbf{x}_i$ of an n dimensional data set $\mathbf{X}$, $i = 1, 2, \ldots, m$.

2. Mean correct all the points: calculate the mean $\overline{\mathbf{x}}$ and substract it from each data point $\mathbf{x}_i - \overline{\mathbf{x}}$.

3. Calculate the variance-covariance matrix $\mathbf{C}$.

$$C_{ij} = (\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})$$

4. Determine eigenvalues and eigenvectors of the matrix. $\mathbf{C}$ is a real symmetric matrix so a positive real number λ and a nonzero vector $\boldsymbol{\alpha}$ can be found such that

$$\mathbf{C}\boldsymbol{\alpha} = \lambda\boldsymbol{\alpha}$$

where λ is called an eigenvalue and $\boldsymbol{\alpha}$ is an eigenvector of $\mathbf{C}$. To find a nonzero $\boldsymbol{\alpha}$ the characteristic equation $|\mathbf{C} - \lambda\mathbf{I}| = 0$ must be solved. If $\mathbf{C}$ is a $n \times n$ matrix of full rank, n eigenvalues can be found $\lambda_1, \lambda_2, \ldots, \lambda_n$. Using $(\mathbf{C} - \lambda I)\boldsymbol{\alpha} = 0$ all the corresponding eigenvectors can be found.

5. Sort the eigenvalues (and corresponding eigenvectors) so that $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n$.

6. Select the first $d \leqslant n$ eigenvectors and generate the data set in the new representation.

2.1.1 Finding Eigenvalues and Eigenvectors

The determination of eigenvalues and eigenvectors in item 4 above can be performed using any diagonalization routine. If the matrix $\mathbf{A} = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_n)$ contains the eigenvectors of a symmetric matrix $\mathbf{C}$, then $\mathbf{A}$ is orthogonal, and $\mathbf{C}$ can be decomposed as

$$\mathbf{C} = \mathbf{A}\mathbf{D}\mathbf{A}^T$$

where $\mathbf{D}$ is a diagonal matrix of the eigenvalues, and A is a matrix of the eigenvectors. Singular value decomposition (SVD) is widely used for this diagonalisation because of its numerical stability.

We ultimately want $d \leqslant n$ of the eigenvalues and eigenvectors. If $d \ll n$ then it is wasteful to calculate all n eignevalues and eigenvectors, only then to discard $n-d$ of these. Hotelling's power method [7] is an iterative technique which can be used to find just the largest $d \leqslant n$ of the eigenvalues and eigenvectors. As it finds the eigenvalues and eigenvectors in order, it is unnecessary to sort them.

2.2 Data Methods

2.2.1 Hebbian Networks

Neural network models that perform PCA have recently been developed [4], most of them using Hebbian learning rules. In general these neural networks will contain processing units with forward connections given by matrix $\mathbf{A}$, where the column vectors $\mathbf{a}_i$ are the connections between the inputs $\mathbf{x}_k$ and the output y_i. The i^{th} weight, $\mathbf{a}_i$ is used to approximate the i^{th} eigenvector, $\boldsymbol{\alpha}_i$. The output units have a value given by:

$$y_i = \mathbf{a}_i^T \mathbf{x}_k$$

In some of Hebbian architectures there are also lateral connections between outputs, such that these outputs effect each other (see [4] for a full description).

In the k^{th} iteration, Hebbian learning rules update the weights using:

$$\mathbf{a}_i^{k+1} = \mathbf{a}_i^k + \Phi(y_i, \mathbf{x}_k) \tag{1}$$

where Φ is some function of the inputs and outputs.

2.2.2 Simple PCA

Simple PCA (SPCA) [10] is not strictly speaking a Hebbian algorithm, although it does have similarities, it is iterative, local and gets the principal components in order (largest eigenvalues first). Most importantly, it does not necessarily add a Hebbian term. Instead, SPCA considers other forms of Φ. In particular, [10] considers two different functions Φ_1 and Φ_2.

$$\Phi_1(y_i, \mathbf{x}_k) = \begin{cases} \mathbf{x}_k & \text{if } y_i \geq 0 \\ 0 & \text{otherwise} \end{cases}$$
$$\Phi_2(y_i, \mathbf{x}_k) = y_i \mathbf{x}_k$$

Each principal component direction found is normalised. Either of these algorithms may be implemented in an iterative or a batch mode. Initially $\mathbf{a}^0$ can be set to any vector. In a batch mode, the sum of $\Phi(y_i, \mathbf{x}_j)$ over all samples $\mathbf{x}_j$, is calculated where y_i is calculated relative to the old estimate $\mathbf{a}^k$. (For Φ_1 this is equivalent to summing those vectors positively correlated with $\mathbf{a}^k$.) This sum is then normalised to give the new estimate (equation 2).

$$\mathbf{a}^{k+1} = \frac{\sum_j \Phi(y_i, \mathbf{x}_j)}{\left\| \sum_j \Phi(y_i, \mathbf{x}_j) \right\|} \; ; \quad y_i = \left(\mathbf{a}^k\right)^T \mathbf{x}_j \tag{2}$$

In an iterative mode, updating is performed according to equation 1, and the estimate is normalised once every pass.

In order to find further principal components, the dataset may be deflated, removing the effect of the principal components already found (see [10], [4]).

3 Bootstrapped correlation

The estimation of the standard errors for most statistics on non-normal distributions can not be done analitically. In our application, the distribution of the correlation between two eigenvectors is non-normal. But in the last 20 years several computer-based methods have been proposed for producing standard errors and other measures of statistical accurracy of estimates. We chose bootstrapping since it gives good estimations of the standard error, has nice statistical properties and is easy to implement [5]. In essence, bootstraping is a *data-based* simulation, where the simulated data is produced by data-based estimates of the population.

The basic algorithm is:

1. Select B independent bootstrap samples, $x^{\star 1}, x^{\star 2}, ...x^{\star B}$, each consisting of n data points drawn with replacement from the original x.

2. Evaluate the bootstrap replication corresponding to each bootstrap sample. This means evaluating by whichever method we are analysing the eigenvalues and eigenvectors.

3. Estimate the standard error by the sample standard deviation of the B replications.

4 Experiments and Results

Two sources of errors are found in the estimation of the correlation. First, since we are using a finite sample (a fraction of the total population) the eigenvalues and eigenvectors produced are estimates, and as such they will vary for different samples (e.g. different sets of handwritten digits). Secondly, despite PCA techniques are due to give the same estimates for an infinite number of iterations, this is usually not the case for finite training times, the differences might be due to the randomness in the iterative/learning process, in the learning parameters or in the sample used.

The correlations of the solutions (eigenvectors) and their means and standard deviations can be studied using bootstraping. Our intention with these experiments is to show a general method for comparing the solutions of different techniques for PCA.

The goal of PCA is to explain as much variance as possible with the smallest number of variables (PCs). All the methods which we examined have similar compression rates, that is when they have converged they all explain approximately the same variance using the same number of principal components, see [10] for more details. But the eigenvectors obtained might be different, in particular when two eigenvalues are similar (the data is spherically distributed) the corresponding eigenvectors are usually not well determined.

4.1 Dataset

The handwritten digits dataset compiled by AT&T [14] consists of 11716 samples in 256-dimensional greyscale images (16 × 16) classified in 10 classes, one for each digit: most are real handwritten; some are artificially generated. Most of the classes contain more than 1000 samples. Examples of these images are given in Figure 1.

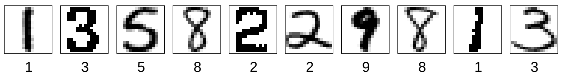

Figure 1: Sample images of handwritten digits.

4.2 Methods

The matrix methods which we examined were singular value decomposition (Numerical Recipes' implementation [12]) and Hotelling's power method (from [4]). In the tables which follow, these are given the labels "SVD" and "PM" respectively. 20 iterations were performed for the power method.

The data oriented methods described here are $SPCA_1$ and $SPCA_2$. In all the following experiments involving SPCA we applied one iterative pass through the data using Φ_1, this was followed by 20 batch iterations using Φ_1 for $SPCA_1$ and Φ_2 for $SPCA_2$.

The explained variance obtained for these and other methods is described in [10].

4.3 Correlation results

For each bootstrap sample the correlation between the first eigenvectors of each pair of methods is produced. Table 1 shows the mean and the standard deviation over 50 bootstap samples, performed on digit 1. This standard deviation is equal to the standard error as defined in [5].

The same number of bootstrap samples is taken for each digit and the mean and standard deviation of these are shown in Table 2.

The mean correlation is averaged over the different subsets, so the result is independent of the differences in their ditributions. This sample means have a normal distribution and the standard deviation describes the differences between the different subsets.

In the schematic of figure 2 we show the relationship between the methods according to the results shown in table 2.

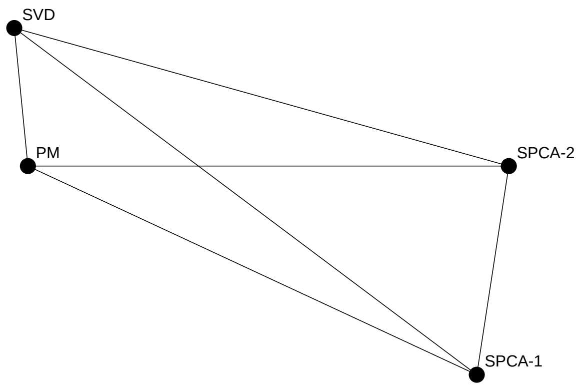

Figure 2: Schematic relation of the correlation between methods.

5 Conclusion

We compare the results obtained through different techniques for principal component analysis. The mean correlations of these solutions are tabulated together with their standard errors.

Looking at the correlations table for a particular digit (Table 1) we can see that all the correlations are very high, this is a logical result since all of the have converged and are close to their limit value.

From the results in Table 2 it can be seen that the matrix methods give very similar solutions (high correlation) and the same is true for the data oriented methods, and this correlation is very precise. The same happens with the two SPCA methods analysed. These results are quantifying the similarities due to the nature of techniques. SPCA and SVD show a bigger difference, the difference is bigger than the one between SPCA2 and any of the matrix methods.

Although these differences do not appear as big, the fact that they exist, and the relations between them, makes bootstraping particularly usefull to study new methods whose statistical properties (e.g. robustness, generalisation,...), are not well known and usefull to see the relationship with other methods.

Method	SVD	PM	SPCA$_1$	SPCA$_2$
SVD	1.000 ± 0.000	0.9910 ± 0.0468	0.9890 ± 0.0042	0.9819 ± 0.0299
PM		1.000 ± 0.000	0.9754 ± 0.0126	0.9886 ± 0.0073
SPCA$_1$			1.000 ± 0.000	0.9952 ± 0.0018
SPCA$_2$				1.000 ± 0.000

Table 1: Correlation between 1st eigenvectors obtained through different methods. +/- values correspond to standard errors (1 standard deviation). Digit 1.

Method	SVD	PM	SPCA$_1$	SPCA$_2$
SVD	1.000 ± 0.000	0.997 ± 0.000	0.941+0.007	0.959 ± 0.006
PM		1.000 ± 0.000	0.952 ± 0.007	0.964 ± 0.007
SPCA			1.000 ± 0.000	0.993 ± 0.000
SPCA2				1.000 ± 0.000

Table 2: Mean over all the digits of correlation between 1st eigenvectors values correspond to standard errors (1 standard deviation).

Acknowledgements

Matthew Partridge was supported by an Australian Postgraduate Award and a Norman I Price Scholarship. Rafael Calvo was supported by a scholarship from Universidad Nacional de Rosario and a Norman I Price Scholarship.

References

[1] A. Basilevsky. *Statistical factor analysis and related methods: theory and applications*. Wiley, New York, 1994.

[2] A. Bravais. Analyse mathematique sur les probabilite's des erreurs de situation d'un point. *Sci. Math Phys.*, 9:255–332, 1846.

[3] C. Chatfield. *Introduction to multivariate analysis*. Chapman and Hall, London ; New York, 1980.

[4] K. Diamantaras and Kung. *Principal component neural networks : theory and applications*. Wiley, New York, 1996.

[5] B. Efron and R. Tibshirani. *An introduction to the bootstrap*. Chapman and Hall, New York, 1993.

[6] R. Frisch. Correlation and scatter in statistical variables. *Nordic Stat. J*, 8:36–102, 1929.

[7] H. Hotelling. Analysis of a complex of statistical variables into principal components. *J. Educ. Psych.*, 24:417–441, 498–520, 1933.

[8] I. T. Jolliffe. *Principal component analysis*. Springer-Verlag, New York, 1986.

[9] I. Z. Milosavljević, M. G. Partridge, R. A. Calvo, and M. A. Jabri. High-performance principal component analysis with paral. *ACNN98*, 1998.

[10] M. Partridge and R. A. Calvo. Fast dimensionality reduction and simple pca. *Intelligent Data Analysis*, 2(3), 1998 (to appear).

[11] K. Pearson. On lines and planes of closest fit to a system of points in space. *Philosophical Magazine*, 2:79–156, 1901.

[12] W. Press and [et al.]. *Numerical recipes in C : the art of scientific computing*. Cambridge University Press, Cambridge, 1988.

[13] A. C. Rencher. *Methods of multivariate analysis*. Wiley, New York, 1995.

[14] P. Simard, Y Le Cun, and J. Denker. Efficient pattern recognition using a new transformation distance. In *NIPS*, volume 5, pages 50–58, 1993.