Principal Component Analysis based on Fuzzy Objective Functions

Tai-Ning Yang and Sheng-De Wang
Department of Electrical Engineering, National Taiwan University
tnyang/sdwang@hpc.ee.ntu.edu.tw

Abstract

In this paper, we propose a robust principal component analysis algorithm based on a fuzzy objective function. By defining a fuzzy objective function for considering the outliers in principal component analysis, we derive an on-line robust algorithm that can extract the appropriate principal components from the spoiled data set. An artificially generated data set is used to evaluate the performance of the proposed algorithm.

keywords: *neural network, fuzzy theory, principal component extraction, outlier, robust statistics*

1 Introduction

Principal component analysis is an important and essential technique for data reduction, image compression, and feature extraction. It has been widely used in many fields including communication, pattern recognition, and image processing. Oja [1] found that a simple linear neuron model with a constrained Hebbian learning rule could extract the principal components of a stationary data set. Thus, the self-organizing learning rule for computing weights of the hidden nodes in the neural network is associated with PCA techniques. Since then, many other neural network based PCA techniques are proposed. Sanger [2] extended Oja's method and designed an algorithm for extracting the first k principal components. Foldiak [3] and Kung et al. [4] developed other similar algorithms based on anti-Hebbian learning rules. Unlike the traditional eigenvector analysis algorithms, these approaches do not require the computation of the input data covariance which may increase significantly with the dimensionality of the training data. Furthermore, there is no need to evaluate all the eigenvalues and eigenvectors if only the eigenvector corresponding to the most significant eigenvalue is required.

Since PCA algorithms process information from the real world, it should have the ability to cope with the noise or outliers. The batch-type robust PCA has been studied for a long time [5]. However, almost all of the existing on-line PCA algorithms assume the training data set contains no outlier. Xu and Yuille [6] proposed one robust algorithm but their algorithm needs a careful parameter setting procedure. Robustness theory is concerned about solving problems subject to model perturbation or added noise. According to Huber [7], a robust algorithm not only performs well under the assumed model, but also produces a satisfactory result under some deviation of the assumed model. Moreover, it will not deteriorate drastically due to the noise or outliers. To establish the robust PCA algorithms, we define a fuzzy objective function and derive an algorithm that could update the weight according to the membership.

The remaining parts of this paper are organized as follows. In section 2, we introduces our algorithm. The following section illustrates the simulation results. Finally, section 4 contains the conclusion.

2 PCA based on fuzzy objective function

Let $X = \{x_1, x_2, ..., x_n\}$ denote the data set with zero mean. The first principal component is defined to be the dot product $w^t x$ if w is the vector that maximizes the variance of the transformed data set $X = \{w^t x_1, w^t x_2, ..., w^t x_n\}$ with the constraint $w^t w = 1$. The traditional eigenvalue based approach to compute w is composed of two steps:

step 1. Compute the data covariance matrix $\Sigma = E\{xx^t\}$. E represents the expectation operation.

step 2. Apply some kind of numerical procedure to extract the eigenvalues and the eigenvectors of the covariance matrix Σ. The eigenvector w corresponding to the largest eigenvalue is just the vector which maximizes the variance of the transformed data set$\{w^t x_1, w^t x_2, w^t x_3, ..., w^t x_n\}$.

Since the above approach processes the input data in a batch way, several algorithms [8] [9] suitable for on-line training are proposed without the necessity to compute Σ. For simplicity, we only use the following algorithm in this paper.

PCA algorithm:

step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$ and the initial weight w.

step 2. While t is less than T, do steps 3-8.

step 3. Compute $\alpha_t = \alpha_0(1-t/T)$ and set $i = 1$.

step 4. While i is less than n, do steps 5-7.

step 5. Compute $y = w^t x_i$.

step 6. Update the weight:

$$w = w + \alpha_t(x_i y - \frac{w}{w^t w}y^2). \quad (1)$$

step 7. Add 1 to i.

step 8. Add 1 to t.

Equation (1) is the gradient rule for minimizing E with respect to w.

$$E = \sum_{i=1}^{n} e(x_i) \quad (2)$$

, where $e(x_i) = \|x_i\|^2 - \frac{\|w^t x_i\|^2}{\|w\|^2} = x_i^t x_i - \frac{w^t x_i x_i^t w}{w^t w}$.

The detail of the derivation could be found in Xu and Yuille [6].

We propose a fuzzy objective function:

$$FE = \sum_{i=1}^{n}(u_i)^m e(x_i) + \eta \sum_{i=1}^{n}(1-u_i)^m, \quad (3)$$

subject to $u_i \in [0,1]$ and $m \in [1,\infty)$. u_i is the membership of x_i belonging to the data cluster and $(1-u_i)$ is the membership of x_i belonging to the noise cluster. m is the weighting exponent. $e(x_i)$ measures the error between x_i and class center.

The concept is to add a noise cluster in which the data has a constant influence η. The idea comes from Noise clustering designed by Davé [10]and Fuzzy C-means algorithm by Bezdek [11]. Let us discuss this function from a clustering viewpoint. u_i is the membership of x_i in the data cluster; while $(1-u_i)$ is the membership of x_i in the noise cluster. The fuzziness variable, m, determines the influence of small u_i compared to large u_i. Following the fuzzy clustering approach, this is an appropriate formulation when only one data cluster exists. This function measures the weighted sum of distances between data and cluster center which is zero in the data set.

Let us derive our algorithm with gradient descent approach. First, we compute the gradient of FE with respect to u_i. By setting $\frac{\partial FE}{\partial u_i} = 0$, we get

$$u_i = \frac{1}{1+(\frac{e(x_i)}{\eta})^{1/(m-1)}}. \quad (4)$$

Substitute this membership back and after simplification, we get

$$FE = \sum_{i=1}^{n}\left(\frac{1}{1+(\frac{e(x_i)}{\eta})^{1/(m-1)}}\right)^{(m-1)} e(x_i) \quad (5)$$

Following the multidimensional chain rule, the gradient of FE with respect to w is:

$$\frac{\partial FE}{\partial w} = \left(\frac{\partial FE}{\partial e(x_i)}\right)\left(\frac{\partial e(x_i)}{\partial w}\right) \quad (6)$$

$$= \left(\frac{1}{1+(\frac{e(x_i)}{\eta})^{1/(m-1)}}\right)^m \left(\frac{\partial e(x_i)}{\partial w}\right) \quad (7)$$

We use $\beta(x_i)$ to denote $\left(\frac{1}{1+(\frac{e(x_i)}{\eta})^{1/(m-1)}}\right)^m$. m is called fuzziness variable in the literature of fuzzy clustering. If $m = 1$, the fuzzy membership, formula (4), reduces to the hard membership and could be determined by the following rule.

$$u_i = 1, \text{if } e(x_i) < \eta,$$
$$= o, \text{otherwise.}$$

η plays the role of hard threshold in this situation.

If $m \to \infty$, the maximum fuzziness is achieved.

$$u_i = \frac{1}{2}, \textit{ for all } x_i. \quad (8)$$

We show the membership relative to some other values of m in Fig. 1 . An interesting observation shows η is a soft threshold that determines where the membership becomes 0.5. Since 0.5 is the average value in the membership domain $[0,1]$, a reasonable choice for η is the average distance, $\frac{\sum_{i=1}^{n} e(x_i)}{n}$. There is no general rule for the setting of m, most papers set $m = 2$ since it leads to a simpler modification rule. The derived algorithm is as follows.

Fuzzy PCA algorithm:

step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0,1]$, soft threshold η and the initial weight w.

step 2. While t is less than T, do steps 3-9.

step 3. Compute $\alpha_t = \alpha_0(1-t/T)$, set $i = 1$ and $\sigma = 0$.

step 4. While i is less than n, do steps 5-8.

step 5. Compute $y = w^t x_i$.

step 6. Update the weight:

$$w = w + \alpha_t \beta(x_i)(x_i y - \frac{w}{w^t w}y^2). \quad (9)$$

step 7. Update the temporary count: $\sigma = \sigma + e(x_i)$

step 8. Add 1 to i.

step 9. Compute $\eta = (\sigma/n)$ and add 1 to t.

In some applications, it is necessary to compute the first k principal components. Two strategies could be used to generalize the above algorithm for this purpose. One is the Gram-Schmidt orthogonalization method based strategy. Using this approach, we derive the following algorithm.

Fuzzy k-PCA algorithm:

step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$, initial weight w and soft threshold η_j, $j = 1...k$.

step 2. While t is less than T, do steps 3-9.

step 3. Compute $\alpha_t = \alpha_0(t/T)$, set $i = 1$ and $\sigma_j = 0$, $j = 1...k$.

step 4. While i is less than n, do steps 5-7.

step 5. $x_i(1) = x_i$ and $y(1) = w^t x_i(1)$. For $j = 2...k$, $x_i(j) = x_i(j-1) - w^t(j-1)x_i(j-1)w(j-1)$, and $y(j) = w^t x_i(j)$.

step 6. For $j = 1...k$, update the weight and the temporary count:

$$\beta = \Big(\frac{1}{1 + (\frac{e(x_i(j))}{\eta_j})^{1/(m-1)}}\Big)^m, \qquad (10)$$

$$w = w + \alpha_t \beta (x_i(j)y(j) - \frac{w}{w^t w} y(j)^2), \quad (11)$$

$$\sigma_j = \sigma_j + \left\| x_i(j) - w^t(j)x_i(j)w(j) \right\|^2. \quad (12)$$

step 7. Add 1 to i.

step 8. For $j = 1...k$, compute $\eta_j = (\sigma_j/n)$.

step 9. Add 1 to t.

Another strategy called the asymmetrical lateral anti-Hebbian learning method uses an additional set of weights to connect the output nodes laterally. These weights are modified by the anti-Hebbian learning rule. An example of this strategy could be found in Rubner and Schulten [12].

3 Simulations

A three dimensional data set with 100 elements and zero mean is generated. As shown in Figs. 2-4, there are 10 outliers in the set. We set $T = 80$ and $\alpha_0 = 0.1$. The comparative result is shown in Fig. 5. The error is defined to be the Euclidean distance between the principal component of the unrobust data set and the currently estimated principal component. It is clear that the proposed algorithm could produce better result than the traditional PCA in this experiment.

4 Conclusions

With consideration of outliers, we derive a robust principal component extraction algorithms by introducing the fuzzy concept into the objective function. The derived algorithm adapts the estimated principal component according to the current membership of the input data. Thus the influence of outlizers is alleviated. There exists other form of fuzzy PCA algorithms. One simple modification is to change the learning law to batch mode or using a momentum updating law. These alterations may be better than the original algorithm if the input presentation order is biased.

References

[1] E. Oja (1982). A simplified neuron model as a principal component analyzer, *J. Math. Biol.*, pp.267–273.

[2] T. D. Sanger (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Network*, pp.459–473.

[3] P. Foldiak (1989). Adaptive network for optimal linear feature extraction. in *Int. Joint Conf. Neural Networks,* Washington. DC, pp.I401–I406.

[4] S. Y. Kung and K. I. Diamantaras (1990). A neural network learning algorithm for adaptive principal extraction. in *Proc. ICASSP,* Albuquerque. pp861–864.

[5] F. M. Hampel, E. M. Ponchotti, P.J. Rousseeuw, and W. A. Stahel (1986). Robust statistics: The approach base on influence functions. Wiley, New York.

[6] Lei. Xu and A. L. Yuille (1995). Robust principal component analysis by self-organizing rules based on statistical physics approach. IEEE Trans. Neural Net., vol. 6, no. 1. pp.131–143.

[7] P. J. Huber (1981). Robust statistics. Wiley, New York.

[8] E. Oja and J. Karhunen (1985). On stochastic approximation of eigenvectors and eigenvalues of the expectation of a random matrix. in *J. Math. Ana. Appl.* pp.69–84.

[9] Lei. Xu (1993). Least mean square error reconstruction for self-organization neural nets. *Neural networks*. vol 6. pp.627–648.

[10] R. N. Davé (1991) Characterization and dection of noise in clustering. Pattern Recognition Lett. vol. 12, no. 11. pp.345–355.

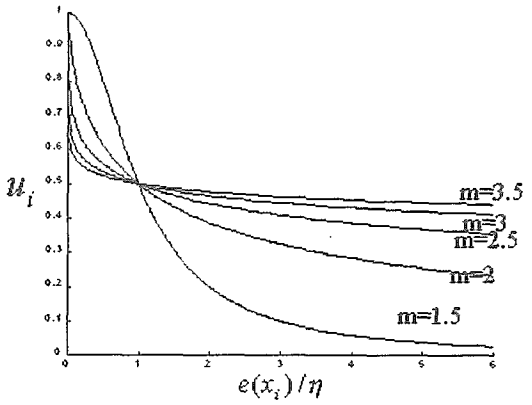

Figure 1: Plot of the membership generated with different m.

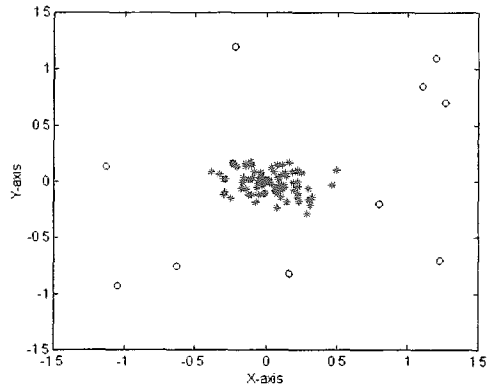

Figure 2: The projection of the testing data on the x-y plane. "*" represents the normal data and "o" represents the outlier.

[11] J. C. Bezdek (1981). Pattern recognition with fuzzy objective functions. Plenum, New York.

[12] J. Rubner and K. Schulten (1990). Development of feature detectors by self- organization. *Biol. Cybern.* vol. 62. pp.193-199.

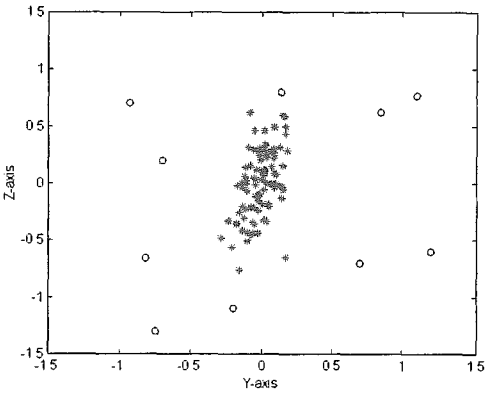

Figure 3: The projection of the testing data on the y-z plane.

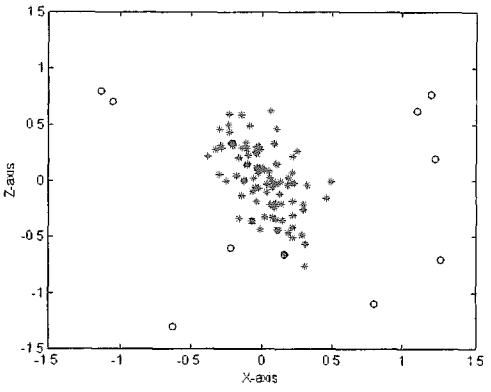

Figure 4: The projection of the testing data on the x-z plane.

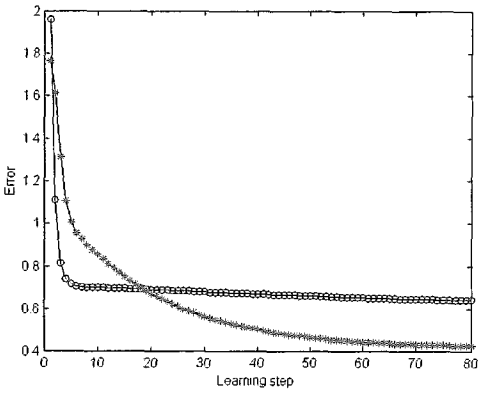

Figure 5: This curve shows the result of the proposed algorithm represented by "*" is better than the traditional PCA represented by "o" when outliers exist.