ELSEVIER

Pattern Recognition Letters 20 (1999) 927–933

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

Robust algorithms for principal component analysis

Tai-Ning Yang *, Sheng-De Wang

Department of Electrical Engineering, National Taiwan University, EE Building, Room 441, 1 Roosevelt Road, Sec. 4, Taipei 106, Taiwan

Received 1 December 1997; received in revised form 3 May 1999

Abstract

In this paper, we address the issues related to the design of fuzzy robust principal component analysis (FRPCA) algorithms. The design of robust principal component analysis has been studied in the literature of statistics for over two decades. More recently Xu and Yuille proposed a family of online robust principal component analysis based on statistical physics approach. We extend Xu and Yuille's objective function by using fuzzy membership and derive improved algorithms that can extract the appropriate principal components from the spoiled data set. The difficulty of selecting an appropriate hard threshold in Xu and Yuille's approach is alleviated by replacing the threshold by an automatically selected soft threshold in FRPCA. Artificially generated data sets are used to evaluate the performance of various PCA algorithms. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Principal component analysis; Robust algorithm; Noise clustering; Neural networks; Fuzzy theory

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 data communication, pattern recognition, and image processing. Since PCA algorithms have to process information from the real world, it should have the ability to cope with the noise or outliers.

Robustness theory is concerned about solving problems subject to model perturbation or added noise. According to Huber (1981), a robust algorithm not only performs well under the assumed model, but also produces a satisfactory result under the deviation of the assumed model. Moreover, it will not deteriorate drastically due to the noise or outliers. Much effort has been done in the investigation of the robust principal component analysis algorithm especially in the literature of statistics. Several strategies have been used to deal with the problem of outliers in PCA. One is to robustify the existing algorithms by applying some kind of robust estimate of the covariance matrix. Several such estimates are reviewed in (Huber, 1981). Ruymgaart (1981) proposed another robust PCA based on robust estimates for dispersion in the univariate case along with a certain linearization of the bivariate structure. Critchley (1985) designed another robust PCA which produces the diagnostic statistics based on the influence function. Most of the above algorithms from statistics field are operated in a batch way.

In the neural network literature, Oja (1982) found that a simple linear neuron model with a

* Corresponding author. Tel.: +886-2-23635251; fax: +886-2-23671909; e-mail: sdwang@cc.ee.ntu.edu.tw

PII: S0167-8655(99)00060-4

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 a neural network can be associated with PCA techniques. Since then, many other neural network based PCA techniques are proposed. Sanger (1989) extended Oja's method and designed an algorithm for extracting the first k principal components. Foldiak (1989) and Kung and Diamantaras (1990) 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, it is not necessary to evaluate all the eigenvalues and eigenvectors if only the eigenvector corresponding to the most significant eigenvalue is required. To robustify the existing methods, Xu and Yuille (1995) first related the PCA learning rules to energy functions and proposed an objective function with the consideration of outliers. Based on statistical physics approach, robust PCA algorithms are derived.

This paper attempts to develop a family of robust PCA algorithms without the difficulty of choosing a hard threshold in Xu and Yuille's approach. First we define a fuzzy objective function which includes Xu and Yuille's as a crisp special case. Using gradient descent optimization, we propose the robust algorithms called FRPCA. Only one parameter, the fuzziness variable, needs presetting and affects the influences of outliers.

The remaining parts of this paper are organized as follows. In Section 2, we review Xu and Yuille's PCA and introduce our algorithm called fuzzy robust principal component analysis (FRPCA). In Section 3, artificially generated data sets are used to illustrate the performance of various PCA algorithms. We demonstrate the difficulty of parameters setting in Xu and Yuille's PCA. The effects of various fuzziness values on FRPCA are also indicated. Finally, Section 4 contains the summary and conclusion.

2. Robust principal component analysis algorithms

For deriving robust PCA algorithms, Xu and Yuille (1995) proposed an optimization function, Eq. (1), subject to $u_i \in \{0,1\}$:

$$E(U,w) = \sum_{i=1}^{n} u_i e(x_i) + \eta \sum_{i=1}^{n} (1-u_i), \tag{1}$$

where $X = \{x_1, x_2, \ldots, x_n\}$ is the data set and $U = \{u_i \,|\, i = 1, \ldots, n\}$ is the membership set. η is the threshold. Now we briefly review their method. The goal is to minimize Eq. (1) with respect to u_i and w simultaneously. Since u_i is a binary variable and w is a continuous variable, it is a mixture of discrete and continuous optimization and is hard to solve with the gradient descent approach. To overcome the problem, they transformed the goal from the minimization of Eq. (1) to the maximization of the following Gibbs distribution:

$$P(U,w) = \frac{\exp(-\gamma E(U,w))}{Z}, \tag{2}$$

where Z is the partition function that ensures $\Sigma_U \int_w P(U,w) = 1$. Using the same procedure for computing the mean field approximation to the statistical physics system by the saddle point method in (Parisi, 1988), they computed the marginal distribution $P_{\text{marginal}}(w)$ for approximating the maximization of $P(U,w)$. $P_{\text{marginal}}(w)$ is calculated by averaging the variables in $\{u_i\}$. The measure $e(x_i)$ could be one of the following functions:

$$e_1(x_i) = \left\| x_i - w^{\mathrm{T}} x_i w \right\|^2, \tag{3}$$

$$e_2(x_i) = \|x_i\|^2 - \frac{\|w^{\mathrm{T}} x_i\|^2}{\|w\|^2} = x_i^{\mathrm{T}} x_i - \frac{w^{\mathrm{T}} x_i x_i^{\mathrm{T}} w}{w^t w}. \tag{4}$$

The gradient descent rules for minimizing $E_1 = \sum_{i=1}^{n} e_1(x_i)$ and $E_2 = \sum_{i=1}^{n} e_2(x_i)$ are

$$w^{\text{new}} = w^{\text{old}} + \alpha_t [y(x_i - u) + (y - v)x_i], \tag{5}$$

$$w^{\text{new}} = w^{\text{old}} + \alpha_t \left(x_i y - \frac{w}{w^{\mathrm{T}} w} y^2 \right). \tag{6}$$

α_t is the learning rate. Under the following conditions:

$$\lim_{t\to\infty}\ \alpha_t = 0, \quad \sum_t \alpha_t = \infty, \quad \sum_t \alpha_t^k < \infty, \quad \text{for some } k > 1, \tag{7}$$

the weight w in the updating rules, converges to the principal component vector almost surely (Oja, 1982; Oja and Karhunen, 1985).

Setting $e = e_1$ or $e = e_2$, Xu and Yuille derived the following on-line algorithms.

Xu and Yuille's PCA1 algorithm.
Step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$, the initial weight w and the threshold η.
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 $y = w^{\mathrm{T}}x_i$, $u = yw$ and $v = w^{\mathrm{T}}u$.
Step 6. Update the weight:

$$w^{\text{new}} = w^{\text{old}} + \alpha_t \frac{1}{1 + \exp(\gamma e_1(x_i) - \eta)} \times [y(x_i - u) + (y - v)x_i]. \tag{8}$$

Step 7. Add 1 to i.
Step 8. Add 1 to t.

Xu and Yuille's PCA2 algorithm. The same as Xu and Yuille's PCA1 except step 6.
Step 6. Update the weight:

$$w^{\text{new}} = w^{\text{old}} + \alpha_t \frac{1}{1 + \exp(\gamma e_2(x_i) - \eta)} \times \left(x_i y - \frac{w}{w^{\mathrm{T}}w} y^2\right). \tag{9}$$

There is another weight updating rule called one-unit Oja's algorithm:

$$w^{\text{new}} = w^{\text{old}} + \alpha_t\left(x_i y - w y^2\right). \tag{10}$$

Although one-unit Oja's algorithm is not a gradient rule of any kind of objective function as pointed by Xu and Yuille (1991), Xu (1993) proved the following results:

1. Only one local (also global) minimum exists for E_1 and E_2, and all the other critical points are saddle points.
2. $E(x_i y - w y^2)^{\mathrm{T}} E[y(x_i - u) + (y - v)x_i] \geqslant 0$, E represents the expectation operation.
3. $(x_i y - w y^2)^{\mathrm{T}}(x_i y - (w/(w^{\mathrm{T}}w))y^2) \geqslant 0$ and $E(x_i y - w y^2)^{\mathrm{T}} E(x_i y - (w/(w^{\mathrm{T}}w))y^2) \geqslant 0$.

So Eq. (10) minimizes E_1 in the average sense and minimizes E_2 in both the on-line sense and the average sense. Since there is only one minimum for E_1 and E_2, the three rules will finally produce the same solution, the principal component. Based on the above relationship, it is reasonable to propose the following algorithm. $e(x_i)$ could be set as $e_1(x_i)$ or $e_2(x_i)$.

Xu and Yuille's PCA3 algorithm. The same as Xu and Yuille's PCA1 except step 6.
Step 6. Update the weight:

$$w^{\text{new}} = w^{\text{old}} + \alpha_t \frac{1}{1 + \exp(\gamma e(x_i) - \eta)} \times \left(x_i y - w y^2\right). \tag{11}$$

After the training, the membership is decided by the following rule:

$$u_i = \begin{cases} 1 & \text{if } e(x_i) < \sqrt{\eta}, \\ 0 & \text{otherwise.} \end{cases}$$

γ and η are two parameters in this algorithm. Xu and Yuille suggest setting a small γ at first then track the minimum of the objective function as γ increases to infinity. The hard threshold η would be determined before the training process. We expect to find another algorithm that could set the threshold automatically.

We propose an objective function:

$$RE = \sum_{i=1}^{n} (u_i)^m e(x_i) + \eta \sum_{i=1}^{n} (1 - u_i)^m, \tag{12}$$

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 the 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 design by (Dave, 1991) and fuzzy C-means algorithm by Bezdek (1981). 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 the data and the cluster center which is zero in the data set.

Since u_i is a continuous variable in our objective function (12), we do not encounter the difficulty caused by the mixture of discrete and continuous optimization. Let us derive our algorithm with the gradient descent approach. First, we compute the gradient of RE with respect to u_i. By setting $(\partial RE)/(\partial u_i) = 0$, we get

$$u_i = \frac{1}{1 + (e(x_i)/\eta)^{1/(m-1)}}. \tag{13}$$

Substituting this membership back and after simplification, we get

$$RE = \sum_{i=1}^{n} \left(\frac{1}{1 + (e(x_i)/\eta)^{1/(m-1)}} \right)^{(m-1)} e(x_i). \tag{14}$$

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

$$\frac{\partial RE}{\partial w} = \left(\frac{\partial RE}{\partial e(x_i)} \right) \left(\frac{\partial e(x_i)}{\partial w} \right) = \left(\frac{1}{1 + (e(x_i)/\eta)^{1/(m-1)}} \right)^{m} \left(\frac{\partial e(x_i)}{\partial w} \right). \tag{15}$$

Let $\beta(x_i)$ denote

$$\left(\frac{1}{1 + (e(x_i)/\eta)^{1/(m-1)}} \right)^{m}.$$

m is called a fuzziness variable in the literature of fuzzy clustering. If $m = 1$, the fuzzy membership, Eq. (13), reduces to the hard membership and could be determined by the following rule:

$$u_i = \begin{cases} 1 & \text{if } e(x_i) < \eta, \\ 0 & \text{otherwise.} \end{cases}$$

η plays the role of hard thresholding in this situation.

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

$$u_i = \frac{1}{2} \quad \text{for all } x_i. \tag{16}$$

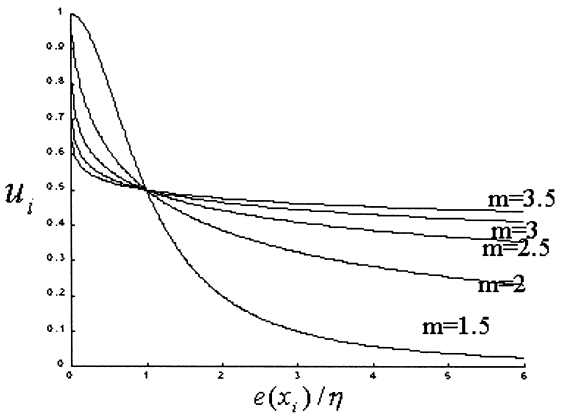

Fig. 1. Plot of the membership generated with different m.

We show the membership relative to some other values of m in Fig. 1. An interesting observation shows η is not a hard threshold any more but 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, $(\sum_{i=1}^{n} e(x_i))/n$. There is no general rule for the setting of m, most papers set $m = 2$ since it leads to a simpler modification rule. Replacing $e(x_i)$ with $e_1(x_i)$ or $e_2(x_i)$, FRPCA1 and FRPCA2 algorithms are derived.

FRPCA1 algorithm.

Step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$, soft threshold η to a small positive value and randomly initialize the 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 $y = w^{\mathrm{T}}x_i$, $u = yw$ and $v = w^{\mathrm{T}}u$.

Step 6. Update the weight:

$$w^{\text{new}} = w^{\text{old}} + \alpha_T \beta(x_i)[y(x_i - u) + (y - v)x_i]. \tag{17}$$

Step 7. Update the temporary count: $\sigma = \sigma + e_1(x_i)$.

Step 8. Add 1 to i.

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

FRPCA2 algorithm. The same as FRPCA1 except steps 6–7.

Step 6. Update the weight:

$$w^{\text{new}} = w^{\text{old}} + \alpha_T \beta(x_i)\left(x_i y - \frac{w}{w^{\mathrm{T}} w} y^2\right). \tag{18}$$

Step 7. Update the temporary count: $\sigma = \sigma + e_2(x_i)$.

Based on the same reason of Xu and Yuille's PCA3, we propose FRPCA3 as follows.

FRPCA3 algorithm. The same as FRPCA1 except steps 6–7.
Step 6. Update the weight:

$$w^{\text{new}} = w^{\text{old}} + \alpha_t \beta(x_i)\left(x_i y - w y^2\right). \tag{19}$$

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

Both Xu and Yuille's PCA and FRPCA belong to the group of algorithms called M-estimator. The theoretical maximum breakdown point for M-estimator could be found in (Huber, 1981; Hampel et al., 1986). The limit that is a function of the input dimension is higher than the limit of the traditional approach.

In some applications, it is necessary to compute the first k principal components. We can also modify those algorithms for the first k principal components in (Xu and Yuille, 1995) in a similar way.

3. Simulations

In the first of this section, we introduce some results obtained from comparative experiments on the unrobust PCA and FRPCA. The unrobust PCA algorithms using weight updating rules (5), (6) and (10) are called PCA1, PCA2 and PCA3, respectively. Fig. 2 is a set of two-dimensional training data with 100 elements and zero mean. There are 5 outliers. We set $T = 40$ and $\alpha_0 = 1$. That is, the final learning rate is 0.025 and each input data is processed 40 times. Fig. 2 shows the results in PCA1, PCA2 and PCA3 are affected by these outliers significantly. The artificially generated data set is also used to train FRPCA1, FRPCA2 and FRPCA3. With the same setting as the former simulation and $m = 2$, the result shown in Fig. 3 indicates FRPCA-type algorithms are robust to these outliers. The weight is initialized with random value and is almost unchanged in the first iteration, since a very small value, 10^{-6}, is assigned to the initial value of the soft threshold, η. In FRPCA3, $e(x_i)$ is replaced by $e_1(x_i)$ or $e_2(x_i)$ separately, so there are four overlapped lines in Fig. 3. Since the learning rate is changed from α_t to $\alpha_t \beta(x_i)$, we find the iterations required by FRPCA is less than PCA. Fig. 4 shows the results of FRPCA when T reduces to 5 and the number of outliers increases to 10.

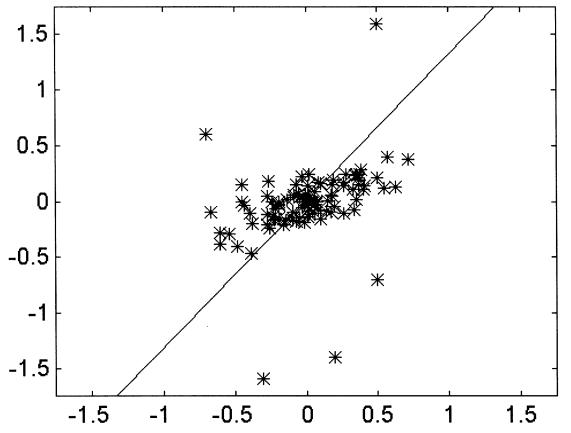

Fig. 2. Testing results of PCA1, PCA2 and PCA3 on the spoiled data set.

To show the experimental differences between Xu and Yuille's PCA and FRPCA, we use the

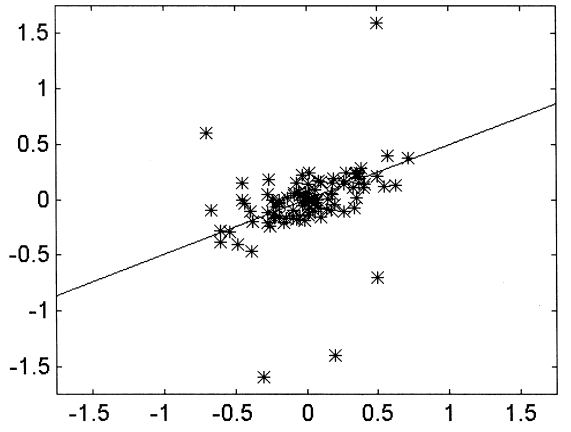

Fig. 3. Testing results of FRPCA1, FRPCA2 and FRPCA3 on the spoiled data set.

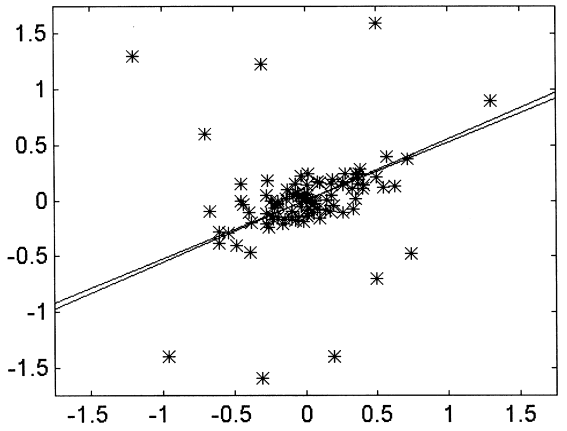

Fig. 4. Testing results of FRPCA1, FRPCA2 and FRPCA3 on another spoiled data set.

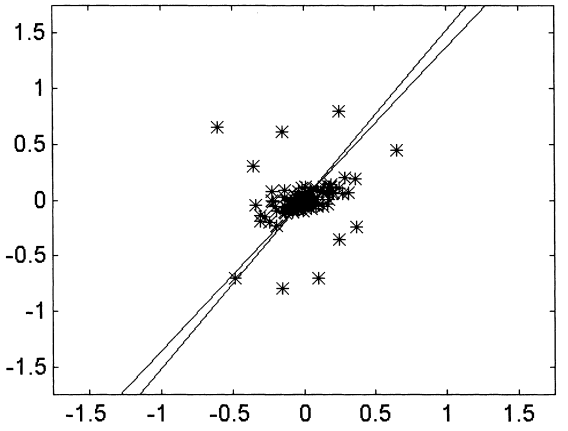

Fig. 6. Testing results of Xu and Yuille's PCA on the scale-down data set.

same data set and a transformed data set in which x and y coordinates of the data point are scaled down by half as shown in Fig. 6. In the following experiments, we set $T = 40$ and $\alpha_0 = 1$. Note that there are four overlapped extracted principal axes in each illustrated line. Sorted by the distance between the origin and the y-intercept of the principal axis, the parameters setting are $\{\gamma = 30, \eta = 0.6\}$ and $\{\gamma = 0.5, \eta = 4\}$ in Figs. 5 and 6. Since the parameters used are $\gamma = 0.5$ and $\eta = 4$ in (Xu and Yuille, 1995), we start from this setting and get the unrobust result. After experiments of various parameter setting, Xu and Yuille's PCA produces a robust result when $\gamma = 30$ and $\eta = 0.6$ as shown in Fig. 5. Unfortunately, these two parameters need to be reset even when the data set is scaled down. As shown in Fig. 6, Xu and Yuille's PCA produces an unrobust result when $\gamma = 30$ and $\eta = 0.6$ on a scale-down data set. Setting γ and η properly could be even more difficult in the case of computing not only the first but also the first k principal components.

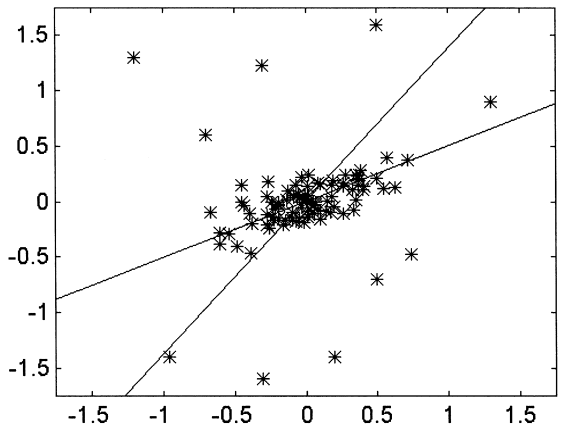

Fig. 5. Testing results of Xu and Yuille's PCA on the spoiled data set.

Although when $m = 1$ and u_i belongs to $\{0, 1\}$, FRPCA's objective function, Eq. (12), reduces to Xu and Yuille's objective function, Eq. (1), Xu and Yuille's PCA is not a special case of FRPCA because different optimization approaches are used. Any very small value could be used to initialize the soft threshold, η. In the following, we want to find the influences of various m values in FRPCA on the performance. Before doing the experiments, we may predict FRPCA will be more like an unrobust PCA as m increases because m raises the membership of the outlier as indicated in Fig. 1. Sorted by the distance between the origin and the y-intercept of the principal axis, the fuzzy variable m are set as 1.5, 2.5, 3.5, 4.5 and 5.5 in Fig. 7. The results that may be regarded as some kind of interpolations between results of noise-filtering PCA and unrobust PCA correspond to our prediction. Using the same m, FRPCA produces the similar results on the scale-down data set as shown in Fig. 8.

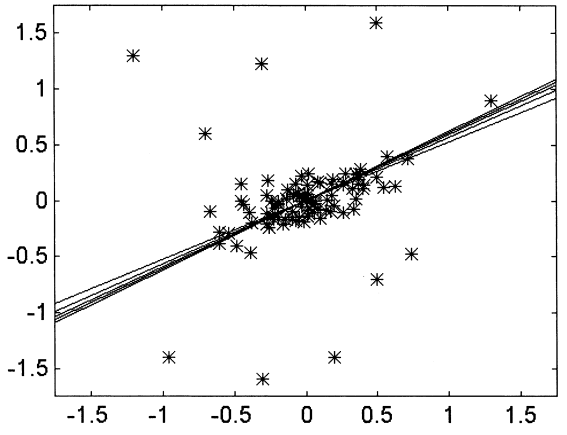

Fig. 7. Testing results of FRPCA on the spoiled data set with different m values.

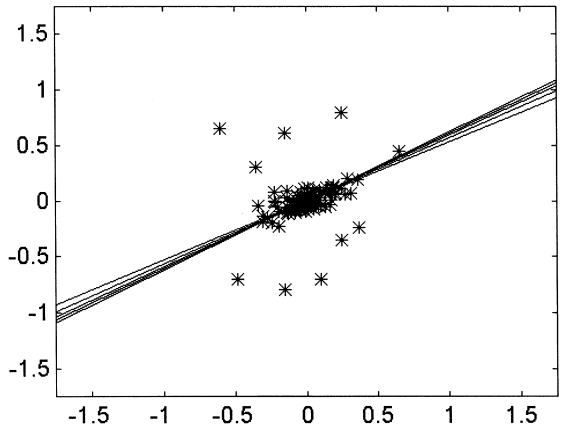

Fig. 8. Testing results of FRPCA on the scale-down data set with different m values.

4. Conclusions

Stemming from the work of Xu and Yuille and the concept of noise clusters, we derive a family of robust principal component extraction algorithms by a fuzzy objective function. The main characteristics of the proposed algorithm are as follows:

- In comparison with the traditional robust PCA, the proposed FRPCA is more robust when outliers exist.
- FRPCA uses a soft threshold that is automatically determined in the algorithm.
- As demonstrated by the simulations, the initial value to the the soft threshold can easily be set to any very small value.

There exist other forms of FRPCA-like 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

Bezdek, J.C., 1981. Pattern Recognition with Fuzzy Objective Functions. Plenum Press, New York.

Critchley, F., 1985. Influence in principal component analysis. Biometrica 72 (3), 627–636.

Dave, R.N., 1991. Characterization and detection of noise in clustering. Pattern Recognition Letters 12 (11), 657–664.

Foldiak, P., 1989. Adaptive network for optimal linear feature extraction. In: Internat. Joint Conf. Neural Networks, Washington, DC, pp. I401–I406.

Hampel, F.M., Ponchotti, E.M., Rousseeuw, P.J., Stahel, W.A., 1986. Robust Statistics: The Approach Base on Influence Functions. Wiley, New York.

Huber, P.J., 1981. Robust Statistics. Wiley, New York.

Kung, S.Y., Diamantaras, K.I., 1990. A neural network learning algorithm for adaptive principal extraction. In: Proc. ICASSP, Albuquerque, pp. 861–864.

Oja, E., 1982. A simplified neuron model as a principal component analyzer. J. Math. Biol., 267–273.

Oja, E., Karhunen, J., 1985. On stochastic approximation of eigenvectors and eigenvalues of the expectation of a random matrix. J. Math. Anal. Appl., pp. 69–84.

Parisi, G., 1988. Statistical Field Theory. Addison-Wesley, Reading, MA.

Ruymgaart, F.H., 1981. A robust principal analysis. J. Multivariate Anal. 11, 485–497.

Sanger, T.D., 1989. Optimal unsupervised learning in a single-layer linear feedforward neural network. Neural Network, 459–473.

Xu, L., 1993. Least mean square error reconstruction for self-organization neural nets. Neural networks 6, 627–648.

Xu, L., Yuille, A.L., 1991. Back-propagation and unsupervised learning in linear networks, in: Chauvin, Y., Rumelhart, E.E. (Eds.), Back-Propagation Theory, Architecture, and Application, Hillsdale, Erlbaum, Hillsdale, NJ.

Xu, L., Yuille, A.L., 1995. Robust principal component analysis by self-organizing rules based on statistical physics approach. IEEE Trans. Neural Net. 6 (1), 131–143.