Neural Fitting: Robustness by Anti-Hebbian Learning

Erkki Oja and Liuyue Wang

Helsinki University of Technology
Laboratory of Computer and Information Science
Rakentajanaukio 2 C, FI-02150 Espoo, Finland

Abstract

A nonlinear neuron, learning by anti-Hebbian learning rules, can perform robust fitting of parametric models to data. Robustness means that the solution is tolerant to non-Gaussian noise and resistant to outliers. The convergence of nonlinear anti-Hebbian learning rules, called here nonlinear Minor Component Analysis algorithms, is analyzed using a simple outlier data model. Both rigorous mathematical results and computer simulations are given. The results show that the linear algorithm, minimizing a Total Least Squares criterion, always has a unique solution which is the direction of the smallest variance of the input data, and thus it is sensitive to outliers. Contrary to this, the nonlinear algorithms exhibit a richer behavior. They do not have a unique solution in general, but will converge to one of several stable fixed points. They are able to maintain their robustness when the effect of the outliers is increasing.

1 Introduction

The Principal Component Analysis (PCA) and Minor Component Analysis (MCA) expansions are standard tools of present-day signal processing methods. The i-th principal component is defined as the projection of the input vector to the i-th eigenvector of the input data covariance matrix, corresponding to the i-th largest eigenvalue. The minor component is here defined as the projection to the eigenvector corresponding to the smallest eigenvalue. To compute the components usually involves estimating the input covariance matrix and then solving the eigenvector-eigenvalue problem.

Another alternative is on-line learning, by which an artificial neuron or layer of neurons receive input vectors one at a time, and in the course of learning the neuron outputs develop into principal or minor components. Typically Hebbian type learning rules are used, which are based on the one-unit learning algorithm originally proposed by one of the authors in (Oja, 1982). Many different versions and extensions of this basic algorithm have been proposed during the recent years; for reviews and introductions, see e.g. (Oja, 1992; Cichocki and Unbehauen, 1993; Haykin, 1994). PCA and MCA networks are useful in signal characterization, optimal feature extraction, signal separation, and data compression.

Recently, there has been growing interest in nonlinear generalizations of PCA criteria, learning algorithms, and networks, mainly because the nonlinear algorithms depend on higher-order statistics compared to the second-order that is implicit in PCA. In general, research and applications of higher-order statistical methods are currently increasing rapidly in all fields of signal processing. Oja et al, 1991, first generalized the one-unit and multi-unit PCA learning rules to nonlinear versions. Other directions and theoretical analysis have been taken by several authors, e.g. (Shapiro and Prugel-Bennett, 1993; Sudjianto and Hassoun, 1994); for more references and details, see (Oja and Karhunen, 1993).

These neural learning algorithms belong to the general class of stochastic approximation, and the usual analysis method is to derive a corresponding deterministic differential equation (see e.g. Oja and Karhunen, 1985; Hornik and Kuan, 1992), and study its asymptotically stable solutions. While some closed form solutions are possible for the linear PCA networks (Yan et al, 1994), the analysis of nonlinear networks turns out to be much harder. Recently, the authors considered a class of MCA algorithms in the context of robust regression and gave some results on the asymptotic behavior of the algorithms. The purpose of this study is to give more detailed results on the asymptotically stable limits of nonlinear MCA algorithms.

The robust regression problem is defined in Section 2 together with the associated neural learning algorithms. Section 3 gives some asymptotic results, and Section 4 demonstrates the behaviour of the learning rules in the case of line fitting under outlying noise.

2 Nonlinear MCA and robust regression

Let us consider the general multidimensional fitting problem. Following (Xu et al, 1992), a parametrically linear regression model is expressed as

$$w_1 x_1 + w_2 x_2 + ... + w_n x_n + w_0 = 0, \tag{1}$$

where $x_i, i = 1, ...n$, are the variables or values of basis functions and $w_1, ..., w_{n+1}$ are the parameters. Define the corresponding vectors $\mathbf{w} = [w_1, w_2, ..., w_n]^T$, $\mathbf{x} = [x_1, x_2, ..., x_n]^T$, giving the equivalent model $\mathbf{w}^T\mathbf{x} + w_0 = 0$. This is a general formulation of a parametrically linear model, covering e.g. polynomials or other similar curves as well as linear models in application fields like regression analysis, estimation, digital filtering, system theory, and automatic control. Typically, the x_i will be functions of some basic variables ξ_j such that $x_i = \phi_i(\xi_1, ..., \xi_m)$, where ϕ_i are some predetermined nonlinear functions. Depending on the ϕ_i functions, the noise in the original variables ξ_j, even if Gaussian, will be transformed to non-Gaussian noise and any model fitting methods based on second order statistics will be sub-optimal. In the following, we will call the x_i input data to a neural unit, but it is understood throughout that in practice another preprocessing layer using the fixed ϕ_i functions may precede the computations analyzed here.

It is convenient to normalize the variables x_i to zero means. Taking averages gives $\mathbf{w}^T E\{\mathbf{x}\} + w_0 = 0$, giving $\mathbf{w}^T(\mathbf{x} - E\{\mathbf{x}\}) = 0$. We now redefine the $\mathbf{x}$ to mean the original values from which the mean has been subtracted. The optimal fitting problem for N points $\mathbf{x}^i, i = 1, ..., N$ in the Total Least Square (TLS) sense is to find the parameter vector $\mathbf{w}$ which will minimize

$$J(\mathbf{w}) = 1/N \sum_{i=1}^{N} r_i^2, \qquad r_i = \frac{\mathbf{w}^T \mathbf{x}^i}{\|\mathbf{w}\|} \tag{2}$$

where r_i is the orthogonal distance of the i-th point $\mathbf{x}^i$ from the hyperplane.

Denoting the sample correlation matrix of the input points by $\hat{\mathbf{C}}$, $J(\mathbf{w})$ can be expressed as

$$J(\mathbf{w}) = 1/N \sum_{i=1}^{N} \frac{(\mathbf{w}^T \mathbf{x}^i)^2}{\mathbf{w}^T \mathbf{w}} = \frac{\mathbf{w}^T \hat{\mathbf{C}} \mathbf{w}}{\mathbf{w}^T \mathbf{w}}. \tag{3}$$

It is obvious that $J(\mathbf{w})$ and $\mathbf{w}$ should be the smallest eigenvalue and the corresponding eigenvector of the correlation matrix $\hat{\mathbf{C}}$. We call this eigenvector the minor component. Thus the problem of optimal fitting in the TLS sense is identical to extracting the minor component.

In many cases, it is desirable to have robustness in fitting when the observation errors are impulsive noise, or some strong outliers exist in the observation data. Instead of using the TLS criterion, one may then use an alternative robust criterion by minimizing the sum of certain functions of variables r_i instead of squares:

$$J_f(\mathbf{w}) = 1/N \sum_{i=1}^{N} f(r_i). \tag{4}$$

Note that if $\mathbf{w}^T\mathbf{w} = 1$, then $r_i = \mathbf{w}^T \mathbf{x}^i$ in eq. (2), and (4) is a finite - sample estimate of $E\{f(\mathbf{w}^T\mathbf{x})\}$. Based on this, the authors introduced the following Lagrangian cost function in (Oja and Wang, 1994):

$$J_L(\mathbf{w}) = E\{f(\mathbf{w}^T\mathbf{x})\} + \lambda(1 - \mathbf{w}^T\mathbf{w}) \tag{5}$$

Minimizing $J_L(\mathbf{w})$ means minimizing the average robust distance from the observation points to the fitted hyperplane.

The corresponding gradient algorithm will be

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \gamma_k[g(y_k)\mathbf{x}_k - (g(y_k)y_k + 1 - \mathbf{w}_k^T\mathbf{w}_k)\mathbf{w}_k] \tag{6}$$

where $y_k = \mathbf{w}_k^T\mathbf{x}_k$, $g(y) = f'(y)$, and γ_k is a positive learning rate. This is the Robust MCA learning algorithm. It could be implemented in a Perceptron - like artificial neuron, whose inputs are the x_i, the corresponding weights the w_i, and the nonlinear activation function $g(y) = g(\mathbf{w}^T\mathbf{x})$. On the right hand side of the learning rule (6), the first term can then be interpreted as the product of the inputs and the output. Because of the negative sign, this is an anti-Hebbian learning rule. As usual in neural network learning, the input vectors $\mathbf{x}_k$ should be picked from the training set randomly or cyclically over a number of epochs, and the learning rate should decrease slowly. Convergence is analyzed in Section 3.

Depending on the choice of the function $f(y)$, several different algorithms are obtained from the general algorithm of eq.(6). Basically, many functions used for robust fitting in statistical methods are suitable. We list three of them in the following. For more such functions, see (Oja et al, 1991).

When the quadratic function $f(y) = \frac{1}{2}y^2$ is used, (6) becomes

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \gamma_k[y_k\mathbf{x}_k - (y_k^2 + 1 - \mathbf{w}_k^T\mathbf{w}_k)\mathbf{w}_k] \tag{7}$$

This is the linear MCA algorithm proposed earlier by (Oja, 1992). When the function $f(y) = |y|$ is used, then $g(y_k) = sign(y_k)$, and we have

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \gamma_k[sign(y_k)\mathbf{x}_k - (sign(y_k)y_k + 1 - \mathbf{w}_k^T\mathbf{w}_k)\mathbf{w}_k]. \tag{8}$$

When the function $f(y) = (1/\beta)lncosh(\beta y)$ is used, then $g(y_k) = tanh(\beta y_k)$, yielding

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \gamma_k[tanh(\beta y_k)\mathbf{x}_k - (tanh(\beta y_k)y_k + 1 - \mathbf{w}_k^T\mathbf{w}_k)\mathbf{w}_k]. \tag{9}$$

Note that $lncosh(\beta y)$ behaves at small values of y roughly quadratically, while for large values of $|y|$ it tends to $|y| - ln2$. This function therefore minimizes a criterion between mean square error and mean absolute error, which may be desirable in robust fitting where we want a good Total Least Squares fit for points close to the model, but the outlying points should have less weight. The derivative of this function, $tanh(\beta y)$, is the symmetrical sigmoid widely used as the activation function in artificial neural networks.

The nonlinear algorithms were first introduced in (Wang and Oja, 1993), where it was shown that they have good performance in a non-Gaussian noise environment, and they can resist outliers automatically. It was also shown experimentally that when the observation errors are Gaussian noise, the Robust MCA algorithm gives an almost identical result to the linear MCA algorithm.

Recently, Wang and Karhunen (1994) introduced another, simplified nonlinear MCA algorithm based on a bi-gradient optimization formulation of the robust TLS problem. Their algorithm is

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \gamma_k[g(y_k)\mathbf{x}_k - \beta(1 - \mathbf{w}_k^T\mathbf{w}_k)\mathbf{w}_k] \tag{10}$$

where β is a parameter controlling the relative effects of the Hebbian term and the constraint; usually it is chosen large. The parameter γ_k is again the learning rate. This algorithm, too, is of the anti-Hebbian type.

Algorithm (10) is slightly simpler than algorithm (6). However, their asymptotic properties are in general rather similar to each other. With a linear learning function $g(y)$, both algorithms converge to the minor component. There are some differences in the case of a nonlinear function $g(y)$, though, as will be illustrated in the two theorems of the next section.

3 Asymptotic analysis of the learning rules

Mathematical analysis of the convergence of the general algorithm (6) was given in (Oja and Wang, 1994). The associated differential equation is

$$d\mathbf{z}/dt = -\mathbf{g}(\mathbf{z}) + (\mathbf{z}^T\mathbf{g}(\mathbf{z}) + 1 - \mathbf{z}^T\mathbf{z})\mathbf{z} \tag{11}$$

where

$$\mathbf{g}(\mathbf{z}) = E\{g(\mathbf{z}^T\mathbf{x})\mathbf{x}\}. \tag{12}$$

By (11), the stationary points satisfy

$$\mathbf{g}(\mathbf{z}_0) = scalar * \mathbf{z}_0 \tag{13}$$

which superficially resembles an eigenvector-eigenvalue equation, and becomes one if g is linear. Then $\mathbf{g}(\mathbf{z}) = E\{\mathbf{x}\mathbf{x}^T\mathbf{z}\} = E\{\mathbf{x}\mathbf{x}^T\}\mathbf{z} = \mathbf{C}\mathbf{z}$, as seen from (12), and the eigenvectors of the covariance matrix $\mathbf{C}$ will be the only stationary points.

Eq. (11) was analyzed in (Oja and Wang, 1994) in the case where the elements of input vectors $\mathbf{x}$ are statistically independent; since the input vectors are the data points normalized to zero means, their deviation from zero is measurement noise and thus the independence assumption may not be unrealistic in this particular application. For nonlinear g, the general result is that eigenvectors of the covariance matrix will be stationary points, but there may be others, too. As for asymptotic stability, (Oja and Wang, 1994) gave a criterion that can be checked for any stationary point.

For example, we can analyze the asymptotic solutions in the simple case when $\mathbf{x}$ is two-dimensional and has a uniform density in the box $[-a, a] \times [-b, b]$, and the nonlinearity g is the *sign* function. Then the vector $\mathbf{z}_0 = [z_{01}\ z_{02}]^T = [1\ 0]^T$ is asymptotically stable if $b/a \geq \sqrt{3/2}$ and the unit vector $\mathbf{z}_0 = [z_{01}\ z_{02}]^T$ with $z_{02}/z_{01} = \sqrt{3(a/b)^2 - 2}$ is asymptotically stable if $1 \leq b/a \leq \sqrt{3/2}$. For $b/a \leq 1$, symmetrical relations hold. Geometrically, this result means that for an oblong uniform density, a vector in the direction of the shorter side is the asymptotically stable solution, until the ratio of the sides decreases to $\sqrt{3/2}$. Then the asymptotically stable vector begins to turn and reaches the diagonal orientation when the ratio of the sides is equal to 1. This result has been verified numerically, and for the case of the vector $[1\ 0]^T$ it also follows from the following basic result:

Theorem 1. Assume the elements of vector $\mathbf{x}$ zero-mean and statistically independent. Assume function $g(y)$ everywhere differentiable. Then a unit vector $\mathbf{e}_i$ in the direction of the $i-th$ coordinate axis in the input space $(i = 1, ..., n)$, with elements $e_{ij} = \delta_{ij}$, will be an asymptotically stable stationary point of eq. (11) if it holds:

$$E\{g(x_i)x_i\} < 1, \tag{14}$$

and for all $j = 1, ..., n, j \neq i$,

$$E\{g(x_i)x_i\}/E\{g'(x_i)\} < E\{x_j^2\}. \tag{15}$$

The proof was given in the Appendix of (Oja and Wang, 1994) and will not be repeated here.

We then turn to the analysis of the learning rule (10). The differential equation corresponding to (10) is given by

$$d\mathbf{z}/dt = -\mathbf{g}(\mathbf{z}) + \beta(1 - \mathbf{z}^T\mathbf{z})\mathbf{z} \tag{16}$$

where again

$$\mathbf{g}(\mathbf{z}) = E\{g(\mathbf{z}^T\mathbf{x})\mathbf{x}\}.$$

A result corresponding to Theorem 1 can be given for algorithm (16), too:

Theorem 2. Assume the elements of vector $\mathbf{x}$ zero-mean and statistically independent. Assume function $g(y)$ everywhere differentiable. Then a vector $\alpha\mathbf{e}_i$ in the direction of the $i-th$ coordinate axis in the input space ($i = 1, ..., n$), with elements $\alpha e_{ij} = \alpha\delta_{ij}$ where α is a scalar, will be an asymptotically stable stationary point of (16) if it holds:

$$E\{g(\alpha x_i)x_i\} = \beta(1 - \alpha^2)\alpha, \tag{17}$$

and

$$E\{g'(\alpha x_i)x_i^2\} + 3\alpha^2\beta > \beta, \tag{18}$$

and for all $j = 1, ..., n, j \neq i$,

$$E\{g'(\alpha x_i)\}E\{x_j^2\} > \beta(1 - \alpha^2). \tag{19}$$

The proof will be given in the Appendix.

The conditions of Theorem 1 will be tested in the next section using a simple example that allows an explicit evaluation of all of them. The extension to Theorem 2 would be straightforward.

4 An example: robust line fitting

Consider the line fitting problem given in Fig. 1. There is a central "cloud" of points, that would be best fitted by the line labelled x_1. However, there are also a number of outliers. When the relative number of the outliers and their distance from the central cloud are large enough, then the line labelled x_2 will be the best fitted line. The question is: what is the effect of the nonlinearity in the robust MCA algorithms on the choice of whether x_1 or x_2 gives a better fit? It is clear that the linear MCA algorithm will converge to the solution $\mathbf{w}$ of the TLS problem, which is the minor eigenvector of the data covariance matric. Then the line $\mathbf{w}^T\mathbf{x} = 0$ will give the direction of the principal axis of the data cloud. When outliers get stronger, there is a certain point at which the order of the eigenvalues will change and linear MCA will flip from line x_1 to line x_2. We could expect the nonlinearity to increase robustness so that the line x_1 will stay as the optimal solution even when the linear

Figure 1: *A line fitting problem in the presence of outliers.*

Figure 2: *The simplified data model used in the analysis: x_1 is uniformly distributed over [-1 , 1] and x_2 has a central uniform mode over [-0.5, 0.5] but also another symmetrical pair of uniform modes with width 0.1 at a distance r from the origin.*

algorithm has switched over to x_2. This question will now be solved both in theory and by simulations under a simple data model.

By suitable rotation, translation, and scaling, the data points in Fig. 1 can in fact be presented in a coordinate system given by the two axes x_1 and x_2. Then the density of x_1 is unimodal, while x_2 has a central mode and additional modes at a certain distance from the origin. For simplicity of analysis, let us assume the data model given in Fig. 2: x_1 is uniform over $[-1, 1]$, and x_2 has a central uniform mode over $[-0.5, 0.5]$ and additional symmetrical uniform modes of width 0.1 at a distance r from the origin. The probability of the central mode is $1 - p$ and that of the outlying modes is p. Since this is the outlier model, the probability p should be small, say less than 0.3.

While uniform densities may not be the most natural assumption, they have two advantages: first, we are interested in methods that apply to non-Gaussian situations, and second, it is now possible to evaluate the expectation integrals in Theorems 1 and 2 in closed form. The subsection 4.1. gives experimental results on the linear MCA compared to the nonlinear algorithms (6) and (10), and theoretical validation is given in subsection 4.2.

4.1 Experimental results

In our computer simulations, two dimensional data are generated according to the data model of Fig. 2. The experiments have been carried out by generating random input data with different values of the outlier distance r, and by comparing the asymptotic results obtained by the learning rules in the case of linear or hard-limiting nonlinear learning functions, respectively. In practice, the nonlinear function is $g(y) = tanh(30y)$ which is very close to $sign(y)$. Both the nonlinear algorithms (6) (Nonlinear algorithm 1) and (10) (Nonlinear algorithm 2) were tested together with the linear algorithm (7) producing the usual TLS solution.

The experimental results are shown in Table 1, where 20 percent outliers are present in x_2 data (percentage $p = 0.2$). When $r < 1.1$, both the linear and nonlinear algorithms converge from any initial point to $\mathbf{e}_2$, which is the direction with the smaller variance (minor component direction). Thus the direction of the fitted line will coincide with the x_1 axis, as it should. However, when $1.2 < r < 1.3$, the minor component direction has switched to $\mathbf{e}_1$, and the linear algorithm has switched the direction of convergence, too, as expected. The nonlinear algorithms with steep sigmoid still converge to the $\mathbf{e}_2$ direction. Thus they indeed give a robust fit by being able to resist the outliers. When the distance r continues to increase from

Figure 3: *When $r = 1.9$, $p = 0.2$, the nonlinear MCA algorithm converges to the minor component direction from the initial values [0.5 0.5].*

Figure 4: *When $r = 1.9$, $p = 0.2$, the nonlinear MCA algorithm converges to a point close to the principal component direction from the initial values [0.1 0.7].*

about 1.3, the nonlinear algorithms have a phase where two possible convergence points emerge, depending on the initial values. These points are close to either $\mathbf{e}_1$ or $\mathbf{e}_2$. From about $r > 2.3$, both algorithms again converge to $\mathbf{e}_1$ from any initial points.

Typical learning curves (the two vector components w_1 and w_2 as functions of time) for algorithm (6) are plotted in Fig. 3 and Fig. 4, where the distance $r = 1.9$ and the outlier possibility $p = 0.2$ are fixed. Taking the steep sigmoid as the learning function, and starting from different initial values, the nonlinear algorithm converges to different directions. Fig. 4 also shows that the limit can in fact be neither of the two eigenvectors $\mathbf{e}_1$, $\mathbf{e}_2$ although it is close to the principal component direction.

4.2 Theoretical validation

The conditions of Theorem 1 (Section 3) are evaluated here for the data model used in this section, and theoretical stability regions are found. We need to compute $E\{g(x_i)x_i\}, E\{g'(x_i)\}$, and $E\{x_i^2\}$ for x_1 and x_2.

When $g(y) = sign(y)$, it follows that $E\{g(x_i)x_i\} = E\{|x_i|\}$ and $E\{g'(x_i)\} = 2p_i(0)$, with $p_i(t)$ denoting the density function of x_i. This yields for x_1 that has a uniform density on $[-1, 1]$:

$$E\{|x_1|\} = 1/2, E\{x_1^2\} = 1/3, E\{g'(x_1)\} = 1. \tag{20}$$

For x_2, the shape of the density function $p_2(x)$ (see Fig. 2) is as follows: it has constant value $1 - p$ over $[-0.5, 0.5]$ and value $5p$ over both $[r, r + 0.1]$ and $[-r - 0.1, -r]$. Then the probability of x_2 being an outlier is p. Let us assume $p = 0.2$.

Table 1: *Comparing the linear and the two nonlinear algorithms with very steep sigmoidal nonlinearity in the case of 20 percent outliers in x_2 with distance r from the origin. The columns give the experimental direction of convergence.*

	$r < 1.1$	$1.2 < r < 1.3$	$1.3 < r < 2.2$	$r > 2.3$
Linear algorithm	$\mathbf{e}_2$	$\mathbf{e}_1$	$\mathbf{e}_1$	$\mathbf{e}_1$
Nonlinear alg. 1	$\mathbf{e}_2$	$\mathbf{e}_2$	close to $\mathbf{e}_2$ or $\mathbf{e}_1$ or zero	$\mathbf{e}_1$
Nonlinear alg. 2	$\mathbf{e}_2$	$\mathbf{e}_2$	close to $\mathbf{e}_2$ or $\mathbf{e}_1$	$\mathbf{e}_1$

Then, after some manipulation we have

$$E\{|x_2|\} = 0.21 + 0.2r, E\{x_2^2\} = 0.202/3 + 0.2r^2 + 0.02r, E\{g'(x_2)\} = 1.6. \quad (21)$$

Let us first take the simplest case when $g(y) = y$. Then condition (15) simply becomes $E\{x_i^2\} < E\{x_j^2\}$. Using the above expressions we see that $\mathbf{e}_1$ is asymptotically stable when $1/3 < 0.202/3 + 0.2r^2 + 0.02r$, with $r > 0$, which is equivalent to $r > 1.1043$. This, then, is the limiting distance below which $\mathbf{e}_2$ is the limit (and the fitted line is in the direction of the first coordinate x_1) and above which the opposite happens. This is in agreement with the first row of Table 1.

Consider then the case of steep nonlinearity in the nonlinear algorithms 1 and 2. The asymptotic stability for $\mathbf{e}_1$ requires that $E\{|x_1|\} < 1$ (eq. (14)) that is now satisfied. For (15) to hold, we must have $1/2 < 0.202/3 + 0.2r^2 + 0.02r$, for $r > 0$, which yields the condition $r > 1.4217$. If the outlier distance exceeds this limit, then the weight vector will converge to $\mathbf{e}_1$ from a suitably close initial value, and the fitted line direction will be x_2.

The asymptotic stability for $\mathbf{e}_2$ requires that $E\{|x_2|\} < 1$ (eq. (14)) that is now satisfied if $r < 3.95$. For (15) to hold, we must have $(0.21 + 0.2r)/1.6 < 1/3$, for $r > 0$, which yields the condition $r < 1.6167$. If the outlier distance is smaller than this limit, then the weight vector will converge to $\mathbf{e}_2$ from a suitably close initial value, and the fitted line direction will be x_1.

Note that there is a region for r around $r = 1.5$ when both $\mathbf{e}_1$ and $\mathbf{e}_2$ are asymptotically stable. It then depends on the initial values which one will be the limit. Also, it must be emphasized that Theorem 1 only concerns the vectors $\mathbf{e}_i$ as the possible limits; this does not mean that there could not be some other asymptotically stable limits in other directions, too. Looking at the experimental results of Table 1, the results agree with the theory.

5 Conclusion

The convergence of the nonlinear MCA algorithms has been analyzed in this report by rigorous mathematical analysis and simulations on a simple outlier data model. It is known that the weight vector of the linear MCA algorithm converges to the Minor Component direction. Thus the linear algorithm has a unique limit point (to be exact, the two possibilities given by the different signs). The nonlinear MCA algorithms have several asymptotically stable fixed points, however. When outliers are relatively small, both linear and nonlinear algorithms converge to the minor component direction. When the effect of the outliers continues to increase, the linear algorithm will always switch to the minor component direction, while the nonlinear MCA algorithms maintain a strong tendency to converge to the previous direction. This behavior of resisting the outliers offers an advantage over the linear algorithm. The point of convergence depends on the initial value of the weights, the outlier percentage, and the distance of the outlier distribution.

6 Appendix

The proof of Theorem 2 in Section 3.1 is given here. For the sake of completeness, let us repeat the equation (16) here:

$$d\mathbf{z}/dt = -\mathbf{g}(\mathbf{z}) + \beta(1 - \mathbf{z}^T\mathbf{z})\mathbf{z} \tag{22}$$

where

$$\mathbf{g}(\mathbf{z}) = E\{g(\mathbf{z}^T\mathbf{x})\mathbf{x}\}.$$

First, the stationary points $\mathbf{z}_0$ of eq. (22) are given by

$$\mathbf{g}(\mathbf{z}_0) = \beta(1 - \|\mathbf{z}_0\|^2)\mathbf{z}_0. \tag{23}$$

Substituting $\mathbf{z}_0 = \alpha\mathbf{e}_i$, with α any scalar, in the left hand side of (23) yields

$$\mathbf{g}(\alpha\mathbf{e}_i) = E\{g(\alpha\mathbf{e}_i^T\mathbf{x})\mathbf{x}\} = E\{g(\alpha x_i)\mathbf{x}\} = E\{g(\alpha x_i)x_i\}\mathbf{e}_i \tag{24}$$

because the elements x_i are zero-mean and independent. With the same substitution, the right hand side of (23) becomes $\beta(1-\alpha^2)\alpha\mathbf{e}_i$. If the first condition of Theorem 2, eq. (17) holds, then $\alpha\mathbf{e}_i$ is a stationary point.

As for asymptotic stability, let us linearize the right hand side of (22) around a stationary point $\mathbf{z}_0$. Let $\mathbf{z}(t) = \mathbf{z}_0 + \mathbf{d}(t)$, with $\mathbf{d}(t)$ small. Then

$$d\mathbf{d}/dt = -\mathbf{g}(\mathbf{z}_0 + \mathbf{d}) + \beta(1 - (\mathbf{z}_0 + \mathbf{d})^T(\mathbf{z}_0 + \mathbf{d}))(\mathbf{z}_0 + \mathbf{d}). \tag{25}$$

The first term on the right hand side gives $\mathbf{g}(\mathbf{z}_0 + \mathbf{d}) = E\{g(\mathbf{z}_0^T\mathbf{x} + \mathbf{d}^T\mathbf{x})\mathbf{x}\} = E\{g(\mathbf{z}_0^T\mathbf{x})\mathbf{x} + g'(\mathbf{z}_0^T\mathbf{x})(\mathbf{d}^T\mathbf{x})\mathbf{x} + O(\|\mathbf{d}\|^2)\}$, assuming g has a derivative. We thus obtain

$$\mathbf{g}(\mathbf{z}_0 + \mathbf{d}) = \mathbf{g}(\mathbf{z}_0) + E\{g'(\mathbf{z}_0^T\mathbf{x})\mathbf{x}\mathbf{x}^T\}\mathbf{d} + O(\|\mathbf{d}\|^2). \tag{26}$$

The second part in the right hand side of (25) becomes, after some manipulation, $\beta(1 - \|\mathbf{z}_0\|^2)\mathbf{z}_0 + \beta(1 - \|\mathbf{z}_0\|^2)\mathbf{d} - 2\beta(\mathbf{z}_0\mathbf{z}_0^T)\mathbf{d} + O(\|\mathbf{d}\|^2)$, and finally the linearized version reads

$$d\mathbf{d}/dt = \mathbf{K}(\mathbf{z}_0)\mathbf{d} + O(\|\mathbf{d}\|^2) \tag{27}$$

where

$$\mathbf{K}(\mathbf{z}_0) = -E\{g'(\mathbf{z}_0^T\mathbf{x})\mathbf{x}\mathbf{x}^T\} - 2\beta(\mathbf{z}_0\mathbf{z}_0^T) + \beta(1 - \|\mathbf{z}_0\|^2)\mathbf{I}. \tag{28}$$

In order to show the asymptotic stability of a vector $\alpha\mathbf{e}_i$ as a stationary point, we have to show that all the eigenvalues of matrix $\mathbf{K}(\alpha\mathbf{e}_i)$ are negative. This matrix is clearly symmetrical and hence it has a complete orhonormal set of eigenvectors. We now show that in fact the vectors $\mathbf{e}_j$, $j = 1, ..., n$, are the eigenvectors of $\mathbf{K}(\alpha\mathbf{e}_i)$.

An auxiliary result is useful in establishing this result:

Lemma A1. Consider the matrix $\mathbf{H}(\mathbf{z}_0) = E\{g'(\mathbf{z}_0^T\mathbf{x})\mathbf{x}\mathbf{x}^T\}$ in eq.(28). Assume the elements of $\mathbf{x}$ statistically independent and zero mean. The eigenvectors of matrix $\mathbf{H}(\alpha\mathbf{e}_i)$ are the vectors $\mathbf{e}_1, ..., \mathbf{e}_n$, and the eigenvalues defined by

$$\mathbf{H}(\alpha\mathbf{e}_i)\mathbf{e}_j = \nu_{ij}\mathbf{e}_j \tag{29}$$

are

$$\nu_{ij} = E\{g'(\alpha x_i) x_j^2\}. \tag{30}$$

Proof. We obtain $\mathbf{H}(\alpha \mathbf{e}_i)\mathbf{e}_j = E\{g'(\alpha \mathbf{e}_i^T \mathbf{x})\mathbf{x}\mathbf{x}^T \mathbf{e}_j\} = E\{g'(\alpha x_i)\mathbf{x} x_j\}$. The elements of this vector are $E\{g'(\alpha x_i) x_k x_j\}$. Because of the zero means and independence of the x_i, the elements will all be zero except the $j - th$ element, which is ν_{ij}.

Now it is easy to find the eigenvalues of matrix $\mathbf{K}(\alpha \mathbf{e}_i)$, which will determine the asymptotic stability of $\alpha \mathbf{e}_i$ as a stationary solution to the differential equation (22). Matrix $\mathbf{K}(\alpha \mathbf{e}_i)$ is now

$$\mathbf{K}(\alpha \mathbf{e}_i) = -\mathbf{H}(\alpha \mathbf{e}_i) - 2\beta\alpha^2 \mathbf{e}_i \mathbf{e}_i^T + \beta(1 - \alpha^2)\mathbf{I} \tag{31}$$

whose eigenvectors are the vectors $\mathbf{e}_j$, with eigenvalues determined from

$$\mathbf{K}(\alpha \mathbf{e}_i)\mathbf{e}_j = (-\nu_{ij} - 2\beta\alpha^2\delta_{ij} + \beta(1 - \alpha^2))\mathbf{e}_j. \tag{32}$$

The case $i = j$ gives the eigenvalue $-\nu_{ii} - 2\beta\alpha^2 + \beta(1 - \alpha^2)$, and the case $i \neq j$ gives the eigenvalue $-\nu_{ij} + \beta(1 - \alpha^2)$. All of the eigenvalues must be negative in order for $\alpha \mathbf{e}_i$ to be a stable stationary solution. Condition (18) of Theorem 2 is equivalent to the first condition, and (19) is equivalent to the second one. This completes the proof of Theorem 2.

7 References

A. Cichocki and R. Unbehauen (1993), *Neural Networks for Optimization and Signal Processing.* John Wiley, New York.

S. Haykin (1994), *Neural Networks - a Comprehensive Foundation.* IEEE Computer Society Press, New York.

K. Hornik and C.-M. Kuan (1992), “Convergence Analysis of Local Feature Extraction Algorithms”. *Neural Networks* **5**, 229-240.

E. Oja (1982), ”A simplified neuron model as a principal components analyzer”. *J. of Math. Biology* **15**, 267-273.

E. Oja and J. Karhunen (1985), ”On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix”. *J. of Math. Analysis and Applications* **106**, 69-84.

E. Oja, H. Ogawa, and J. Wangviwattana (1991), ”Learning in nonlinear constrained Hebbian networks”. In *Artificial Neural Networks (Proc. ICANN'91, Espoo, Finland)*, ed. T.Kohonen et al. North-Holland, Amsterdam, 385 - 390.

E. Oja (1992), ”Principal Components, Minor Components, and Linear Neural Networks”. *Neural Networks* **5**, 927-935.

E. Oja and J. Karhunen (1993), ”Nonlinear PCA: algorithms and applications”, Helsinki Univ. of Technology, Lab. of Computer and Inform. Science, Report A18.

E. Oja and L. Wang (1994), ”Robust fitting by nonlinear neural units ”. Submitted to a journal.

J. L. Shapiro, and A. Prugel-Bennett (1993), "Nonlinear Statistical Analysis and Self-Organizing Hebbian Networks". *NIPS:6*, 407–414.

A. Sudjianto and M. H. Hassoun (1994), "Nonlinear Hebbian Rule: A statistical Interpretation". *Proc. IEEE ICNN-94*, Orlando, Fla., July 1994, 1247–1252.

L. Wang and J. Karhunen (1994), "A unified neural bigradient algorithm for robust PCA and MCA". Submitted to a journal.

L. Wang and E. Oja (1993), "Robust fitting by nonlinear neural units". *Proc. 8th Scand. Conf. on Image Analysis*, Tromsa, Norway, May 1993, 531 - 537.

L. Xu, E. Oja and C.Y. Suen (1992), "Modified Hebbian learning for curve and surface fitting". *Neural Networks* **5**, 441 - 457.

W. Yan, U. Helmke, and J. Moore (1994), "Global analysis of Oja's flow for neural networks". *IEEE Trans. Neural Networks*, **5**, 674 - 683.