# Robust fitting by nonlinear neural units

Erkki Oja and Liuyue Wang

Helsinki University of Technology

Laboratory of Computer and Information Sciences

Rakentajanaukio 2 C, FI-02150 Espoo, Finland

## Abstract

A central problem in statistics is fitting a model which is linear in the parameters to a set of observation points. Examples are regression, curve fitting, time series modelling, digital filtering, system theory, and automatic control. The usual approaches are least squares (LS) or total least squares (TLS) regression. The TLS criterion is equivalent to finding the minor component of the input points, based on the eigenvector of the input covariance matrix corresponding to the smallest eigenvalue. In impulsive and colored noise environments, or in the presence of outliers, these methods are not optimal, however. Then robust fitting, based on a non-quadratic criterion, may give better results than the usual TLS.

Neural network approaches to robust regression are reviewed and a new learning algorithm is introduced, based on a robust TLS criterion involving a nonlinear function. The algorithm has the form of a nonlinear constrained anti-Hebbian learning rule. Instead of extracting the exact minor component, the algorithm extracts the robust minor component. The results of computer simulations are provided to illustrate that this neural fitting method outperforms the commonly used LS and TLS fitting methods in resisting both Gaussian noise and outliers.

# 1 Introduction

In many data analysis applications, either principal components or minor components of data vectors are used. They are projections of the data vectors on the eigenvectors of the data covariance matrix, corresponding to the largest or the smallest eigenvalues. Most of the neural Principal Component Analysis (PCA) learning algorithms are based on the one-unit constrained Hebbian rule introduced by one of the authors in [5]. With this learning rule, the unit can extract the first principal component of the stationary input vector stream $\mathbf{x}_t$, in the sense that the weight vector $\mathbf{w}_t$ of the unit tends asymptotically to the principal eigenvector of the input data correlation matrix $\mathbf{C} = E\{\mathbf{x}_t\mathbf{x}_t^T\}$.

In some applications, minor components are needed instead of principal components. It was first shown by Thompson [9], how the minor components can be used to estimate frequencies of sinusoidal signals buried in white noise. Karhunen [4] and recently Joutsensalo [3] studied in detail the related learning algorithms. Xu, Oja and Suen [12] considered the line or surface fitting problem in the Total Least Square (TLS) sense [10] and proposed a neural learning algorithm for extracting the minor component from the observation data set. The TLS criterion was shown to outperform the usual least squares (LS) criterion.

When the observation data are corrupted by colored or impulsive noise or outliers, the minimum Total Least Square criterion is no longer optimal. The exact minor component does not give the best fitting parameters anymore, and the linear neural unit converging to the exact minor component will not be able to solve the optimal fitting problem. In this case, *robust fitting methods*, which are able to reject outliers automatically are needed.

The main objection to the use of robust fitting in practice has been a computational one: while the TLS criterion can be solved in closed form and the minor eigenvector can be computed with standard numerical techniques like the singular value decomposition (SVD), this is no longer true for more complicated criterion functions. An iterative gradient descent algorithm is necessary. Neural networks can be an advantage here.

In this paper, we will solve the robust fitting problem having a general non-quadratic criterion. The resulting algorithm is a nonlinear constrained anti-Hebbian learning rule that has a natural implementation in an adaptive artificial neuron. The activation function of the neuron will be the derivative of the robust criterion function. A standard sigmoid function can be used, but also other activation functions are possible. Due to the nonlinearity, quite high outlier resistance is achieved,

Section 2 reviews the optimal fitting problem and introduces the robust TLS concept. In Section 3, a constrained gradient descent algorithm for a nonlinear neural unit is proposed to solve the robust MCA problem. Section 4 gives simulation results on line fitting and nonlinear surface fitting, demonstrating that the nonlinear neural unit approach is better than the classical LS and TLS methods in resisting both Gaussian noise and outliers. Some conclusions are summarized in section 5. In the Appendix, the nonlinear rule is analyzed mathematically.

# 2 Optimal fitting criteria

For simplicity, let us consider the multidimensional hyperplane fitting problem. The hyperplane model is expressed as

$$w_1x_1 + w_2x_2 + ... + w_nx_n + w_0 = 0, \tag{1}$$

where $x_i, i = 1, ...n$, are the variables and $w_1, w_2, ..., w_n, w_0$ are the parameters to be estimated. Define the 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 model that is linear in parameters. It covers e.g. polynomials or other similar parametric curves as well as linear models in application fields like regression analysis, estimation, digital filtering, system theory, and automatic control.

In the following, it is convenient to normalize the variables $x_i$ to zero means. Taking averages in the linear model with respect to $\mathbf{x}$ gives $\mathbf{w}^T E\{\mathbf{x}\} + w_0 = 0$, giving $\mathbf{w}^T(\mathbf{x} - E\{\mathbf{x}\}) = 0$. We now redefine $\mathbf{x}$ to mean the zero-mean input vector for which the hyperplane model is

$$\mathbf{w}^T\mathbf{x} = 0. \tag{2}$$

In this model, the hyperplane contains the origin, and the distance $d$ of a point $\mathbf{x}$ from the hyperplane is the projection of $\mathbf{x}$ on the direction of $\mathbf{w}$, i.e.,

$$d = \frac{|\mathbf{w}^T\mathbf{x}|}{\sqrt{\mathbf{w}^T\mathbf{w}}}. \tag{3}$$

The optimal fitting problem for $N$ points $\mathbf{x}^i, i = 1, ..., N$ in the total least square (TLS) sense [1] is to find the parameter vector $\mathbf{w}$ which will minimize the mean of the squared orthogonal distances $d_i$ of the points $\mathbf{x}^i$ from the hyperplane:

$$J(\mathbf{w}) = 1/N \sum_{i=1}^{N} d_i^2, \qquad d_i = \frac{|\mathbf{w}^T\mathbf{x}^i|}{\sqrt{\mathbf{w}^T\mathbf{w}}} \tag{4}$$

This is illustrated for the line fitting problem in Fig.1. When only the $x_2^i$ measurement contains observation errors, the least squares (LS) method gives the optimal solution, which minimizes the sum of the squares of the vertical distances $e_i$. However, when both $x_1^i$ and $x_2^i$ measurements contain errors, the total least squares method minimizing the sum of the squares of the perpendicular distances $d_i$ is the optimal solution.

Denoting the sample covariance matrix of the (zero mean) input points by $\hat{\mathbf{C}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}^i\mathbf{x}^{iT})$, $J(\mathbf{w})$ can be expressed as

$$J(\mathbf{w}) = 1/N \sum_{i=1}^{N} \frac{(\mathbf{w}^T\mathbf{x}^i)^2}{\mathbf{w}^T\mathbf{w}} = \frac{\mathbf{w}^T\hat{\mathbf{C}}\mathbf{w}}{\mathbf{w}^T\mathbf{w}}. \tag{5}$$

It is obvious that $J(\mathbf{w})$ and $\mathbf{w}$ should be the smallest eigenvalue and the corresponding eigenvector of the matrix $\hat{\mathbf{C}}$. We call this eigenvector the minor component for convenience. Thus the problem of optimal fitting in the TLS sense is identical to extracting the minor component [12].

If the noise in the elements of $\mathbf{x}^i$ is uncorrelated and the variances are equal, then it is known [10] that the TLS criterion will produce the true parameters as the number of observation points grows large. However, when the observation errors are impulsive noise or colored noise with unknown covariance, or some strong outliers exist in the observation data, a more robust criterion is needed. Instead of using the TLS criterion, one may then use an alternative criterion by minimizing the sum of certain functions of variable $d_i$ instead of squares:

$$J_f(\mathbf{w}) = 1/N \sum_{i=1}^{N} f(d_i). \tag{6}$$

Generally, function $f$ would be a monotonically increasing function of its non-negative argument $d_i$. For mathematical convenience, in the following we will use instead of $d_i$ the term $\frac{\mathbf{w}^T\mathbf{x}^i}{\sqrt{\mathbf{w}^T\mathbf{w}}}$ that can have both positive and negative values, and choose $f$ to be a non-negative even function. Function $f$ can be freely designed for specific applications. When the quadratic function $f(d_i) = d_i^2$ is used, eq.(6) is the same as eq.(4), which minimizes TLS. In general, a meaningful choice of $f(d_i)$ would be an even function, increasing slower than $d_i^2$. This will decrease the effect of strong outliers on the solution. Such alternatives are considered in Section 3.3.

# 3 Neural learning algorithms for MCA and robust MCA

First, notice that if the norm of $\mathbf{w}$ is equal to one, then according to eq. (4), the orthogonal distance from the $i$-th sample point to the current fitted hyperplane is just the absolute value of $\mathbf{w}^T\mathbf{x}^i$. This has the form of the output of a linear neuron with input vector $\mathbf{x}^i$ and weight vector $\mathbf{w}$. This is the starting point of deriving neural learning algorithms for Minor Component analysis.

In [6], one of the authors proposed the following gradient descent algorithm for minimizing the quadratic TLS criterion (4):

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

The sequence $\{\mathbf{x}_k\}$ consists of the available observation points $\mathbf{x}^i, i = 1, ..., N$ in some order; usually, the points are used as inputs repeatedly in cyclical order, or the current input $\mathbf{x}_k$ at each iteration step is randomly picked from the set of observation points. The term $y_k = \mathbf{w}_k^T\mathbf{x}_k$ is the distance, assuming $\|\mathbf{w}_k\| = 1$. This will be true asymptotically, because algorithm (7) tends to normalize the weight vector to unit length. The step sizes $\alpha_k$ form a suitably chosen decreasing sequence.

Under certain conditions [6], the weight vector $\mathbf{w}_k$ will tend to the unit-norm minor eigenvector[1] of $\hat{\mathbf{C}}$, and the output of the neuron $y$ will thus be the minor component of the input vector $\mathbf{x}$. In [11], the authors extended this result on a heuristic basis to several nonlinear neural network learning algorithms to solve the TLS fitting problem or the robust version of it. In the following, these algorithms will be derived rigorously as instantaneous gradient descent algorithms minimizing a Lagrangian cost function.

## 3.1 Constrained optimization via a Lagrange multiplier

Replacing the finite sum in eq. (6) by the theoretical expectation of $f(\mathbf{w}^T\mathbf{x})$ and using a Lagrange multiplier with constraint $\mathbf{w}^T\mathbf{w} = 1$ gives the following cost function:

$$J_L(\mathbf{w}, \lambda) = E\{f(\mathbf{w}^T\mathbf{x})\} + 1/2\lambda(1 - \mathbf{w}^T\mathbf{w}). \tag{8}$$

The expectation is with respect to the density of $\mathbf{x}$. Thus $J_L$ is a function of $\mathbf{w}$ and $\lambda$ only. Minimizing $J_L(\mathbf{w}, \lambda)$ means minimizing the average "robust distance" from the observation points to the fitted hyperplane. In the following, an equation for the extremal points is derived and analyzed.

[1] One of the two possible choices with opposite signs

Differentation of $J_L(\mathbf{w}, \lambda)$ with respect to the Lagrange multiplier $\lambda$ yields the constraint equation:

$$\mathbf{w}^T\mathbf{w} = 1. \tag{9}$$

Denote now the derivative of $f(y)$ by $g(y)$:

$$g(y) = \frac{d}{dy}f(y). \tag{10}$$

Then the gradient of $J_L(\mathbf{w}, \lambda)$ of eq.(8) with respect to the weight vector $\mathbf{w}$ will be

$$\frac{\partial J_L(\mathbf{w}, \lambda)}{\partial \mathbf{w}} = E\{g(\mathbf{w}^T\mathbf{x})\mathbf{x}\} - \lambda\mathbf{w}. \tag{11}$$

At the minimum, the gradient must be equal to zero, giving

$$E\{g(\mathbf{w}^T\mathbf{x})\mathbf{x}\} - \lambda\mathbf{w} = 0. \tag{12}$$

This together with $\mathbf{w}^T\mathbf{w} = 1$ are necessary conditions for the optimum. Multiplying eq. (12) by $\mathbf{w}^T$ from the left and substituting $\mathbf{w}^T\mathbf{w} = 1$ yields

$$\lambda = E\{g(\mathbf{w}^T\mathbf{x})\mathbf{w}^T\mathbf{x}\}, \tag{13}$$

finally giving the necessary condition for the optimum point $\mathbf{w}$:

$$E\{g(\mathbf{w}^T\mathbf{x})\mathbf{x}\} - E\{g(\mathbf{w}^T\mathbf{x})\mathbf{w}^T\mathbf{x}\}\mathbf{w} = 0. \tag{14}$$

Changing the term $E\{g(\mathbf{w}^T\mathbf{x})\mathbf{w}^T\mathbf{x}\}\mathbf{w}$ to the form $\mathbf{w}^T E\{g(\mathbf{w}^T\mathbf{x})\mathbf{x}\}\mathbf{w}$ and denoting

$$\mathbf{g}(\mathbf{w}) = E\{g(\mathbf{w}^T\mathbf{x})\mathbf{x}\} \tag{15}$$

we obtain from (14):

$$\mathbf{g}(\mathbf{w}) = (\mathbf{w}^T\mathbf{g}(\mathbf{w}))\mathbf{w}. \tag{16}$$

This is the equation for the extremal points $\mathbf{w}$ of the constrained minimization problem.

In the linear case $g(y) = y$, the vector function $\mathbf{g}(\mathbf{w})$ becomes $\mathbf{C}\mathbf{w}$ and (16) becomes the eigenvector - eigenvalue equation for covariance matrix $\mathbf{C}$. For nonlinear

$g(y)$, (16) is a generalization of the eigenvector - eigenvalue equation. The right hand side is still proportional to vector $\mathbf{w}$, with the scalar multiplier a function of $\mathbf{w}$. The left hand side is the nonlinear vector function $\mathbf{g}(\mathbf{w})$. In analogy with the eigenvectors of linear operators, we are thus looking for vectors $\mathbf{w}$ whose orientation in the $n$-space coincides with the orientation of the vector function $\mathbf{g}(\mathbf{w})$.

This function depends on higher order moments of $\mathbf{x}$, but usually only on the even order moments. This can be seen as follows: in the original robust TLS criterion (6), it can be assumed that $f$ is an even function. Then its derivative $g$ will be odd, having zero value at the origin. Assuming that $g(y)$ has a Taylor series, the even terms will be zero. Then the left hand side of eq. (14) becomes

$$\mathbf{g}(\mathbf{w}) = E\{g(\mathbf{w}^T\mathbf{x})\mathbf{x}\} = E\{g'(0)(\mathbf{w}^T\mathbf{x})\mathbf{x}\} + 1/6E\{g'''(0)(\mathbf{w}^T\mathbf{x})^3\mathbf{x}\} + ... \tag{17}$$

The first term on the right is $g'(0)\mathbf{C}\mathbf{w}$. All higher statistics of even order are included. Thus the optimal weight vector, which must be among the roots of eq. (16), is usually not equal to the minor eigenvector of the data correlation matrix $C$. We may call the solution *robust MCA*.

## 3.2 Gradient descent algorithm as a neural learning rule

To proceed to a stable learning algorithm, we can use a more complicated form for the parameter $\lambda$ instead of eq. (13):

$$\lambda = E\{g(\mathbf{w}^T\mathbf{x})\mathbf{w}^T\mathbf{x}\} + 1 - \mathbf{w}^T\mathbf{w}. \tag{18}$$

This is equal to (13) at the minimum point, because of the constraint $\mathbf{w}^T\mathbf{w} = 1$. Substituting this into eq. (11) yields

$$\frac{\partial J_L(\mathbf{w}, \lambda)}{\partial \mathbf{w}} = E\{g(\mathbf{w}^T\mathbf{x})\mathbf{x}\} - (E\{g(\mathbf{w}^T\mathbf{x})\mathbf{w}^T\mathbf{x}\} + 1 - \mathbf{w}^T\mathbf{w})\mathbf{w}. \tag{19}$$

This is the starting point for a stable gradient descent algorithm.

The expression for the gradient in eq. (19) can be used for searching the minimum of the cost function $J_L(\mathbf{w}, \lambda)$ in a stochastic approximation type gradient algorithm in which the expectations are replaced by the instantaneous values of the variables. Denoting again by $k$ the discrete time and by $y_k$ the term $\mathbf{w}_k^T \mathbf{x}_k$ yields

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_k [g(y_k)\mathbf{x}_k - (g(y_k)y_k + 1 - \mathbf{w}_k^T \mathbf{w}_k)\mathbf{w}_k] \tag{20}$$

where $\alpha_k$ is a positive learning rate.

To conclude, we have thus derived (20) as a stochastic gradient algorithm for finding the minimum of the objective function $E\{f(\mathbf{w}^T \mathbf{x})\}$ under the constraint $\mathbf{w}^T \mathbf{w} = 1$. The algorithm is somewhat approximative in the sense that the expression of $\lambda$ is determined from the optimal solution and used then everywhere. The stability and the convergence properties of this algorithm are considered in Section 3.4. and in the Appendix.

The algorithm (20) has a straightforward implementation as the learning rule of an artificial neuron with output activation function $g(y)$. Several popular activation functions are now shown to correspond to certain forms of the minimization problem.

## 3.3 Choice of the nonlinearity

Depending on the choice of the function $f(y)$, several different algorithms are obtained from the general algorithm of eq.(20). 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 [8].

1. When the quadratic function $f(y) = \frac{1}{2}y^2$ is used, eq.(20) becomes

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_k [y_k \mathbf{x}_k - (y_k^2 + 1 - \mathbf{w}_k^T \mathbf{w}_k)\mathbf{w}_k]. \tag{21}$$

   This is exactly the linear MCA algorithm already given in eq. (7), proposed by one of the authors in [6].

2. When the function $f(y) = |y|$ is used, then $g(y_k) = sign(y_k)$, and eq.(20) becomes

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_k [sign(y_k)\mathbf{x}_k - (sign(y_k)y_k + 1 - \mathbf{w}_k^T \mathbf{w}_k)\mathbf{w}_k]. \tag{22}$$

Now the neuron is a basic hard-limiting Perceptron.

3. When the function $f(y) = \frac{1}{\beta} lncosh(\beta y)$ is used, then $g(y_k) = tanh(\beta y_k)$, and eq.(20) becomes

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_k[tanh(\beta y_k)\mathbf{x}_k - (tanh(\beta y_k)y_k + 1 - \mathbf{w}_k^T\mathbf{w}_k)\mathbf{w}_k]. \tag{23}$$

Algorithm (23) was first introduced in [11] on heuristic grounds. It was shown that such a nonlinear algorithm has good performance in a non-Gaussian noise environment, and it can resist outliers automatically. The motivation for choosing a cost function of the form $lncosh(y)$ is that it behaves at small values of $y$ roughly quadratically, while for larger values of $|y|$ it holds $lncosh(y) \approx |y| - ln2$. This function is satisfactory in many respects. While the function $f(y) = |y|$ is used in minimizing the mean absolute error and $f(y) = \frac{1}{2}y^2$ is used in minimizing the mean square error, the function $f(y) = lncosh(y)$ can be used for minimizing another criterion between these two. In addition, the derivative of this function is the sigmoid $tanh(y)$ widely used as an activation function in neural networks. The same function has been proposed for the robust least square problem in [2].

## 3.4 Convergence of the learning rule

Some results on the convergence of the general learning rule are given here, with more detailed analysis in the Appendix. First, the discrete-time learning rule (20) is replaced by a continuous-time differential equation, whose right hand side is obtained by taking averages of $(\mathbf{w}_{k+1} - \mathbf{w}_k)/\alpha_k$ with respect to the density of $\mathbf{x}_k$ [7]. Now the differential equation obtained from (20) is

$$d\mathbf{z}/dt = -\mathbf{g}(\mathbf{z}) + (\mathbf{z}^T\mathbf{g}(\mathbf{z}) + 1 - \mathbf{z}^T\mathbf{z})\mathbf{z} \tag{24}$$

where, in accordance with eq. (16),

$$\mathbf{g}(\mathbf{z}) = E\{g(\mathbf{z}^T\mathbf{x})\mathbf{x}\}. \tag{25}$$

The relation of the differential equation to the original learning rule is such that the possible limits of convergence of the learning rule are among the asymptotically stable stationary points of eq. (24). The conditions require that the learning rate sequence $\{\alpha_k\}$ must tend to zero with a specified rate, and all variances of the components of $\mathbf{x}$ must be nonzero. For details on the exact conditions, see [7].

It is possible to get a very general stability result for certain stationary points of (24) in a special case: when the elements of input vector $\mathbf{x}$ are statistically independent. Let us assume this. The assumption may not be unrealistic in view of the general model fitting problem of Chapter 2. Note that this does not mean in any way that there would not be outliers, because we are not otherwise restricting the form of the density of $\mathbf{x}$.

The general result is that unit vectors in the directions of the coordinate axes in the input space are stationary points (although not necessarily the only ones) of the differential equation, and in a sense the coordinate with the smallest variance, weighted with the nonlinear function $g(y)$, is asymptotically stable. Thus the hyperplane perpendicular to the stable point will be the one spanned by the largest variances. This is a natural result in view of the minimization problem of Chapter 2. More rigorously, we will prove:

*Theorem 1.* Assume the elements of vector $\mathbf{x}$ zero-mean and statistically independent. Then a unit vector $\mathbf{e}_i$ in the direction of the $i-th$ coordinate axis in the input space, with elements $e_{ij} = \delta_{ij}$, will be an asymptotically stable stationary point of eq. (24) if it holds:

$$E\{g(x_i)x_i\} < 1, \tag{26}$$

and

$$E\{g(x_i)x_i\}/E\{g'(x_i)\} < E\{x_j^2\}] \tag{27}$$

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

The proof will be given in the Appendix, section A3.

It may be instructive to check the conditions (26), (27) for the two cases already considered above in a more general setting: the case of a linear function $g(y) = y$ and the case of the sigmoidal nonlinearity $g(y) = tanh(\beta y)$. The hard sigmoid $g(y) = sign(y)$ is obtained from the latter one when $\beta \to \infty$.

In the linear case, the conditions become

$$E\{x_i^2\} < min[1, E\{x_j^2\}] \tag{28}$$

which clearly gives the direction in which the variance (eigenvalue) is the smallest, as long as it is also smaller than 1. So, the minor eigenvector is the solution. There is no tolerance to outliers.

Consider then the case of a steep sigmoid, $g(y) \approx sign(y)$, and assume that the density of a certain element $x_i$ consists of a central part and outlying modes. Then the denominator $E\{g'(x_i)\}$ in (27) is determined by the central part of the density only, because the derivative $g'(x_i)$ will be practically zero outside this range. This term is therefore insensitive to the outliers. The numerator is close to $E\{sign(x_i)x_i\} = E\{|x_i|\}$ which is sensitive to the outliers but less so than the variance $E\{x_i^2\}$. Comparing the sigmoidal case to the linear case, the left hand side of (27) grows more slowly in the sigmoidal case, when the probability of the outliers is increased. The solution of the learning rule will be the robust minor component.

This can be illustrated by a simple 2-dimensional numerical example, in which $x_1$ is uniformly distributed over [-1,1] and $x_2$ has a central uniform mode over [-0.5, 0.5] but also another uniform mode with width, say, 0.1 at a distance $r$ from the origin. When the distance $r$ is small, the central mode will determine the variances and both the linear and the sigmoidal neuron will choose $x_2$, with smaller variance, as the stable direction. When $r$ grows, the variance of $x_2$ starts to grow and at a certain point ($r \approx 1.2$) becomes larger than the variance of $x_1$. The linear neuron will switch at this point and choose $x_1$ as the stable direction. The steep sigmoidal neuron will still have $x_2$ as the stable direction until about $r \approx 2$, when this direction loses stability and this neuron, too, will choose $x_1$.

# 4 Experimental results

In this Section, the robust algorithm (23) with sigmoidal activation function $g(y) = tanh(\beta y)$, corresponding to the robust weighting function $f(y) = lncosh(\beta y)$, will be compared to the usual LS method and the TLS method. This will be illustrated in line and surface fitting experiments. Both the line fitting and the surface fitting experimental setups are the same as those used by Xu, Oja and Suen in [12] for convenience of comparison.

On the pure (noise-free) data sets, the LS, TLS, as well as the nonlinear algorithm (23) can give the perfect solution. We only consider the case when Gaussian noise or non-Gaussian impulse noise is added to the observation data set.

Computer simulations are conducted on the noisy set, using the usual LS method, the linear MCA (LMCA) algorithm (21), and the nonlinear MCA (NMCA) algorithm (23). The steepness of the sigmoid was always chosen as $\beta = 1$. In order to compare the results, the exact minor component of the input correlation matrix is also computed using the standard singular value decomposition (SVD). The absolute error between the true parameter vector $\mathbf{w}$ and the estimated $\hat{\mathbf{w}}$, defined as $\varepsilon_w = ||\hat{\mathbf{w}} - \mathbf{w}||$ is used to compare the accuracy.

## 4.1 Line Fitting

In this simulation, a line model $w_1 x + w_2 y = 1$ is fitted to an observation data set by estimating the parameters $w_1$, $w_2$. First, points $(x, y)$ were taken from the line segment $0.25x + 0.5y = 1$, $0 < x < 5$ by uniform sampling. Then, a noisy observation data set $D_x = \{(x_i, y_i), i = 1, ..., 500)\}$ was generated by adding noise $n_x$ and $n_y$ to $x$

and $y$, respectively. The $n_x$ and $n_y$ are Gaussian observation errors with zero means and variance $\sigma^2$. Fig. 2 shows the line segment and the noise-disturbed data set $D_x$ with noise variance $\sigma^2 = 0.5$. Another observation data set with all points taken from the line segment except two strong outliers was also used in this experiment.

For all the neural fitting methods, the simulations have been implemented in the following steps:

1. Compute the means $E\{x_i\}$, $E\{y_i\}$.

2. Subtract the mean vector from each data vector to get a zero-mean data set.

3. Initialize the weight vector $\mathbf{w} = [w_1, w_2]^T$ by random elements from a uniform distribution on [0,1] $\times$ [0,1].

4. Use the learning algorithm linear MCA (21) or nonlinear MCA (23), respectively, to adaptively adjust $\mathbf{w}_k$. Choose the input vector $\mathbf{x}_k$ at each step by picking one of the sample points, with equal probabilities.

5. After convergence, renormalize the two parameters to the case of nonzero means for the data points.

For simplicity, here we let the learning rate $\alpha_k$ start at 0.01 and linearly reduce it to 0.0025 at the first 500 learning steps, and then keep it constant.

The experimental results for the Gaussian noise case are given in Table 1 and those for the case of two outliers in Table 2.

The results show that the NMCA algorithm (23) gives in both cases the best result and the usual LS method gives the worst. The nonlinear algorithm also outperforms the linear MCA method corresponding to total least squares minimization. The LMCA algorithm (21) gives the same result after convergence as the SVD method, as predicted by the theory.

## 4.2 Nonlinear Surface Fitting

Many nonlinear curves and hypersurfaces can be expressed by

$$w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + ... + w_n f_n(\mathbf{x}) + w_0 = 0, \tag{29}$$

where $\mathbf{x}$ is the original coordinate point and $f_i(\mathbf{x})$ are some basis functions, i.e. polynomials of the elements of $\mathbf{x}$. The neural learning approach with nonlinear anti-Hebbian learning can still be used to estimate the parameters $[w_1, w_2, ..., w_0]^T$.

Different from line fitting, an observation data transformation is needed. Now instead of using $\mathbf{x}$ directly, the vector $\mathbf{f} = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_n(\mathbf{x})]^T$ is used as input data to extract the minor component. So, even if the observation data $\mathbf{x}$ has Gaussian noise, the noise in the transformed input $\mathbf{f}$ is non-Gaussian. In this case, the experimental results show that the exact minor component does not give the best result. Again, nonlinear learning is the best approach to solve this problem.

In this simulation, a data set $D_x$ is used

$$D_x = \{(x_i, y_i, z_i), i = 1, ..., 993)\}$$

which comes from an ellipsoid

$$0.04x^2 + 0.0625y^2 + 0.1111z^2 = 1,$$

shown in Figure 3. Gaussian noise or six strong outliers are added to the sample points. Similar to the line fitting simulation, the problem is to fit a parameterized model $w_1x^2 + w_2y^2 + w_3z^2 = 1$ to the point set $D_x$ by estimating the parameters $w_1$, $w_2$, $w_3$.

We first calculate $E(\mathbf{f})$ and then use the zero-mean data set to train the neural unit according to the LMCA or NMCA algorithm. The experimental results for Gaussian noise are given in Table 3 and those for six strong outliers in Table 4. It can be seen that the linear MCA algorithm (8) gives the minor component as the SVD method does, but no longer the best results on fitting. The nonlinear algorithm NMCA gives a robust minor component, which gives better results on fitting parameters. In all the cases, the usual LS gives the worst results.

# 5 Conclusions

We have shown in this paper that nonlinear constrained anti-Hebbian learning algorithms can be effectively used to solve the optimal fitting problem in the robust total least squares (TLS) sense. The algorithms extract the robust minor component of the input data. Due to the nonlinearity, these neural fitting methods clearly outperform the LS method and other linear methods, including TLS, in impulse noise and colored noise environments or in the presence of outliers. The results of computer simulations demonstrated that our method can provide a new tool for adaptively solving the classical optimal fitting or modelling problem.

# References


[1] G. Golub and C. Van Loan. *Matrix Computations*. Baltimore, MD: Johns Hopkins University Press, 1983.

[2] A. Cichocki and R. Unbehauen. *Neural Networks for Optimization and Signal Processing*. Chichester, UK: John Wiley & Sons, 1993.

[3] J. Joutsensalo. "Conventional and neural methods for estimating principal component type subspaces with application to sinusoidal frequency estimation". *Dr. Tech. Thesis, Helsinki University of Technology*, 1994.

[4] J. Karhunen. "Recursive estimation of eigenvectors of correlation type matrices for signal processing applications ". *Dr. Tech. Thesis, Helsinki University of Technology*, 1984.

[5] E. Oja. “A simplified neuron model as a principal component analyzer”. *Journal of Mathematical Biology*, 16: 267-273, 1982.

[6] E. Oja. “Principal Components, Minor Components, and Linear Neural Networks”. *Neural Networks*, 5: 927 – 935, 1992.

[7] E. Oja and J. Karhunen. ”On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix ”. *J. Math. Anal. Appl.* , 106: 69 - 84, 1985.

[8] E. Oja, H. Ogawa, and J. Wangviwattana. ”Learning in nonlinear constrained Hebbian networks”. *Proc. Int. Conf. on Artificial Neural Networks*, Espoo, Finland, 737 - 745, 1991.

[9] P. Thompson. “An adaptive spectral analysis technique for unbiased frequency estimation in the presence of white noise” *Proc. 13th Asilomar Conf. on CSC*, Pacific Grove, CA, 529-533, 1979.

[10] S. Van Huffel and J. Vandewalle. ”The total least squares technique: computation, properties and applications ”. In E. F. Deprettere (Ed.), *SVD and Signal processing - Algorithms, Applications and Architectures*. Amsterdam: North Holland, 1988, 189 - 207.

[11] ”Robust fitting by nonlinear neural units ”. *Proc. 8th Scand. Conf. on Image Analysis*, Tromsø, Norway, 531 - 537, 1993.

[12] L. Xu, E. Oja, and C. Y. Suen. “Modified Hebbian Learning for Curve and Surface Fitting”. *Neural Networks*, 5: 441–457, 1992.

## Appendix

In this appendix, the convergence of the learning rule (20) is considered in a more general case than in Section 3.4. Thus, we do not assume here that the elements of input vector $\mathbf{x}$ are statistically independent.

**A1. The norm of the weight vector**. We start with a lemma concerning the norm of the solution $\mathbf{z}(t)$ of the averaged differential equation (24) of Section 3.4.

*Lemma 1.* Let the density of input vectors $\mathbf{x}$ and the function $g(y)$ be such that

$$\|\mathbf{z}\|^2 - E\{(\mathbf{z}^T\mathbf{x})g(\mathbf{z}^T\mathbf{x})\} > 0 \tag{30}$$

for all $\mathbf{z}$ in the neighborhood N of the unit sphere $\|\mathbf{z}\| = 1$.

Then $\|\mathbf{z}(t)\|$ will tend to 1 asymptotically in eq. (24) from any initial point in N.

*Proof.* Multiplying eq. (24) by $\mathbf{z}^T$ from the left yields

$$\mathbf{z}^T d\mathbf{z}/dt = 1/2d/dt\|\mathbf{z}\|^2 = (1 - \|\mathbf{z}\|^2)(\|\mathbf{z}\|^2 - \mathbf{z}^T\mathbf{g}(\mathbf{z}))$$

Due to condition (30), the second factor on the right hand side is positive, and it follows that $\|\mathbf{z}\|$ will tend to 1 from any initial point in N.

The condition (30) is fairly general and its validity can be checked for various forms of function $g(y)$. Let us consider a special case covering the linear and smooth sigmoidal functions, and give sufficient conditions for the density of $\mathbf{x}$ in these cases.

*Corollary 2.* Assume $g(y) \leq y$ for all $y$. Assume that $\lambda_1$, the largest eigenvalue of the data correlation matrix $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$ satisfies

$$\lambda_1 < 1.$$

Then (30) holds for $\mathcal{N} = \{z \| \|\mathbf{z}\| > 0\}$.

*Proof.* Now

$$\begin{aligned} \|\mathbf{z}\|^2 - E\{(\mathbf{z}^T\mathbf{x})g(\mathbf{z}^T\mathbf{x})\} &> \|\mathbf{z}\|^2 - E\{(\mathbf{z}^T\mathbf{x})^2\} \\ &= \|\mathbf{z}\|^2 - \mathbf{z}^T\mathbf{C}\mathbf{z} \\ &> \|\mathbf{z}\|^2 - \lambda_1\mathbf{z}^T\mathbf{z} \\ &= \|\mathbf{z}\|^2(1-\lambda_1) \\ &> 0 \end{aligned}$$

for any $\mathbf{z}$ with $\|\mathbf{z}\| > 0$.

Note that the condition of Corollary 2 covers sigmoidal functions like $g(y) = tanh(y)$. For steeper sigmoids, which are still bounded by a linear function, this corollary has a straightforward extension.

**A2. Stationary and stable points**. In the following, it will be assumed that a stationary solution of (24) will have unit norm. If $\mathbf{z}_0$ is such a solution, it holds

$$-\mathbf{g}(\mathbf{z}_0) + (\mathbf{z}_0^T\mathbf{g}(\mathbf{z}_0))\mathbf{z}_0 = 0 \tag{31}$$

with

$$\|\mathbf{z}_0\| = 1. \tag{32}$$

To check asymptotic stability of a stationary point, let $\mathbf{z}(t) = \mathbf{z}_0 + \mathbf{e}(t)$ with $\|\mathbf{e}(t)\|$ small. Then

$$d\mathbf{e}/dt = -\mathbf{g}(\mathbf{z}_0+\mathbf{e}) + [(\mathbf{z}_0+\mathbf{e})^T\mathbf{g}(\mathbf{z}_0+\mathbf{e}) + 1 - \|\mathbf{z}_0+\mathbf{e}\|^2](\mathbf{z}_0+\mathbf{e}). \tag{33}$$

Here, assuming $g$ differentiable, it holds

$$\begin{aligned} \mathbf{g}(\mathbf{z}_0+\mathbf{e}) &= E\{g(\mathbf{z}_0^T\mathbf{x} + \mathbf{e}^T\mathbf{x})\mathbf{x}\} \\ &= E\{g(\mathbf{z}_0^T\mathbf{x})\mathbf{x}\} + E\{g'(\mathbf{z}_0^T\mathbf{x})\mathbf{x}\mathbf{x}^T\}\mathbf{e} + O(\|\mathbf{e}\|^2). \end{aligned}$$

Let us denote the Hessian matrix above by

$$\mathbf{H}(\mathbf{z}_0) = E\{g'(\mathbf{z}_0^T\mathbf{x})\mathbf{x}\mathbf{x}^T\}. \tag{34}$$

After some manipulation we get

$$d\mathbf{e}/dt = [(\mathbf{z}_0\mathbf{z}_0^T - I)\mathbf{H}(\mathbf{z}_0) - 2\mathbf{z}_0\mathbf{z}_0^T + \mathbf{z}_0\mathbf{g}(\mathbf{z}_0)^T + \mathbf{z}_0^T\mathbf{g}(\mathbf{z}_0)\mathbf{I}]\mathbf{e} + O(\|\mathbf{e}\|^2). \tag{35}$$

The asymptotic stability of a stationary point $\mathbf{z}_0$ requires that the eigenvalues of the coefficient matrix on the right hand side of (35),

$$\mathbf{K}(\mathbf{z}_0) = (\mathbf{z}_0\mathbf{z}_0^T - I)\mathbf{H}(\mathbf{z}_0) - 2\mathbf{z}_0\mathbf{z}_0^T + \mathbf{z}_0\mathbf{g}(\mathbf{z}_0)^T + \mathbf{z}_0^T\mathbf{g}(\mathbf{z}_0)\mathbf{I} \tag{36}$$

are all strictly negative. Due to the nonlinearity of $g(y)$, a general solution is difficult. Let us again check some special cases only.

**A3. Stability in the linear case**. If $g(y) = y$, then $\mathbf{g}(\mathbf{z}_0) = \mathbf{C}\mathbf{z}_0$ with $\mathbf{C}$ the covariance matrix, and $\mathbf{H}(\mathbf{z}_0) = \mathbf{C}$. Furthermore, assume that the condition of Corollary 2 holds in order to ensure that the stationary points will have unit norm. The stationary points from eq. (31) are then the unit-norm eigenvectors of $\mathbf{C}$. Consider the stability of an eigenvector $\mathbf{z}_0 = \mathbf{c}_i$, assuming that the eigenvectors are ordered according to decreasing eigenvalues. Now the matrix $\mathbf{K}(\mathbf{z}_0)$ of (36) becomes

$$\begin{aligned} \mathbf{K}(\mathbf{c}_i) &= (\mathbf{c}_i\mathbf{c}_i^T - I)\mathbf{C} - 2\mathbf{c}_i\mathbf{c}_i^T + \mathbf{c}_i\mathbf{c}_i^T C + \mathbf{c}_i^T\mathbf{C}\mathbf{c}_i\mathbf{I} &(37)\\ &= 2(\lambda_i - 1)\mathbf{c}_i\mathbf{c}_i^T - \mathbf{C} + \lambda_i\mathbf{I}. &(38)\end{aligned}$$

The eigenvectors of this matrix are the eigenvectors $\mathbf{c}_1, ..., \mathbf{c}_n$, and the corresponding eigenvalues, say $\mu_j$, can be solved from the equation $\mathbf{K}(\mathbf{c}_i)\mathbf{c}_j = \mu_j\mathbf{c}_j$. If $j \neq i$, we obtain

$$\mathbf{K}(\mathbf{c}_i)\mathbf{c}_j = (\lambda_i - \lambda_j)\mathbf{c}_j. \tag{39}$$

Assuming all the eigenvalues $\lambda_j$ distinct, it follows that the eigenvectors $c_1, ..., c_{n-1}$ cannot be stable points because for all of them the matrix $\mathbf{K}(\mathbf{c}_i)$ has at least the one positive eigenvalue $\lambda_i - \lambda_n$. For the minor eigenvector $\mathbf{c}_n$, the eigenvalues of $\mathbf{K}(\mathbf{c}_n)$ are negative, because both the $n-1$ eigenvalues $\lambda_n - \lambda_j, j = 1, ..., n-1$ determined by (39) and also the final one determined by

$$\mathbf{K}(\mathbf{c}_n)\mathbf{c}_n = 2(\lambda_n - 1) - \lambda_n + \lambda_n = 2(\lambda_n - 1) \tag{40}$$

are negative; note that by the condition of Corollary 2 it holds $\lambda_n < 1$.

**A4. Proof of the general stability condition of eigenvectors with an independence assumption**. The proof of Theorem 1 in Section 3.4 is given here. Two auxiliary results are useful in establishing the main result:

*Lemma 1.* Consider the vector function $\mathbf{g}(\mathbf{z}) = E\{g(\mathbf{z}^T\mathbf{x})\mathbf{x}\}$ of eq. (25). Assuming the elements of $\mathbf{x}$ statistically independent and zero mean, it holds

$$\mathbf{g}(\mathbf{e}_i) = \mu_i \mathbf{e}_i \tag{41}$$

with

$$\mu_i = E\{g(x_i)x_i\}. \tag{42}$$

*Proof.* Substituting $\mathbf{e}_i$ in $\mathbf{g}$ gives

$$\mathbf{g}(\mathbf{e}_i) = E\{g(\mathbf{e}_i^T\mathbf{x})\mathbf{x}\} = E\{g(x_i)\mathbf{x}\}, \tag{43}$$

whose elements are zero because of the independence and zero mean of $x_i$, except the $i-th$ element that is $E\{g(x_i)x_i\}$.

*Lemma 2.* Consider the matrix $\mathbf{H}(\mathbf{z}_0) = E\{g'(\mathbf{z}_0^T\mathbf{x})\mathbf{x}\mathbf{x}^T\}$ of eq.(34). Assume the elements of $\mathbf{x}$ statistically independent and zero mean. The eigenvectors of matrix $\mathbf{H}(\mathbf{e}_i)$ are the vectors $\mathbf{e}_1, ..., \mathbf{e}_n$, and the eigenvalues defined by

$$\mathbf{H}(\mathbf{e}_i)\mathbf{e}_j = \nu_{ij}\mathbf{e}_j \tag{44}$$

are

$$\nu_{ij} = E\{g'(x_i)x_j^2\}. \tag{45}$$

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

Now it is easy to find the eigenvalues of matrix $\mathbf{K}(\mathbf{e}_i)$, which will determine the asymptotic stability of $\mathbf{e}_i$ as a stationary solution to the differential equation (24). Matrix $\mathbf{K}(\mathbf{e}_i)$ is now

$$\begin{aligned} \mathbf{K}(\mathbf{e}_i) &= (\mathbf{e}_i\mathbf{e}_i^T - \mathbf{I})\mathbf{H}(\mathbf{e}_i) - 2\mathbf{e}_i\mathbf{e}_i^T + \mu_i\mathbf{e}_i\mathbf{e}_i^T + \mu_i\mathbf{I} && (46) \\ &= (\nu_{ii} - 2 + \mu_i)\mathbf{e}_i\mathbf{e}_i^T - \mathbf{H}(\mathbf{e}_i) + \mu_i\mathbf{I} && (47) \end{aligned}$$

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

$$\mathbf{K}(\mathbf{e}_i)\mathbf{e}_j = (\nu_{ii} - 2 + \mu_i)\delta_{ij}\mathbf{e}_j - \nu_{ij}\mathbf{e}_j + \mu_i\mathbf{e}_j. \tag{48}$$

The case $i \neq j$ gives the eigenvalue $-\nu_{ij} + \mu_i$. The case $i = j$ gives the eigenvalue $\nu_{ii} - 2 + \mu_i - \nu_{ii} + \mu_i = 2(\mu_i - 1)$. All of these must be negative in order for $\mathbf{e}_i$ to be a stable stationary solution. Condition (26) of Theorem 1 is equivalent to $\mu_i < 1$ and (27) is equivalent to $\mu_i < \nu_{ij}$. This completes the proof of Theorem 1.

Table 1: *Line fitting. The solutions obtained by different methods with Gaussian noise ($\sigma^2 = 0.5$) present in the data set. The average values of $w_1$ and $w_2$ at the learning steps between 4500 and 5000 are given.*

| | $w_1$ | $w_2$ | $\varepsilon_w$ |
|---|---|---|---|
| Original | 0.25 | 0.5 | |
| LS | 0.233 | 0.604 | 0.105 |
| SVD | 0.246 | 0.511 | 0.011 |
| LMCA | 0.246 | 0.510 | 0.012 |
| NMCA | 0.248 | 0.506 | 0.006 |

Table 2: *Line fitting. The solutions obtained by different methods when two strong outliers are present in the data set. The average values of $w_1$ and $w_2$ at the learning steps between 4500 and 5000 are given.*

| | $w_1$ | $w_2$ | $\varepsilon_w$ |
|---|---|---|---|
| Original | 0.25 | 0.5 | |
| LS | 0.216 | 0.5827 | 0.089 |
| SVD | 0.232 | 0.5322 | 0.036 |
| LMCA | 0.232 | 0.5323 | 0.036 |
| NMCA | 0.244 | 0.492 | 0.010 |

FIGURE CAPTIONS

**Figure 1:**

*To fit a set of data points by a line, LS minimizes the sum of the squared lengths of the vertical distances $e_i$, whereas TLS minimizes the sum of the squared lengths of the distances $d_i$ perpendicular to the estimated line.*

Table 3: *Surface fitting. The solutions obtained by different methods with Gaussian noise ($\sigma^2 = 0.3$) present in the data set. The average values of $w_1$ , $w_2$, and $w_3$ at the learning steps between 4500 and 5000 are given.*

| | $w_1$ | $w_2$ | $w_3$ | $\varepsilon_w$ |
|---|---|---|---|---|
| Original | 0.0400 | 0.0625 | 0.1111 | |
| LS | 0.043 | 0.052 | 0.089 | 0.022 |
| SVD | 0.042 | 0.070 | 0.098 | 0.015 |
| LMCA | 0.042 | 0.070 | 0.098 | 0.015 |
| NMCA | 0.035 | 0.063 | 0.103 | 0.006 |

Table 4: *Surface fitting. The solutions obtained by different methods with six strong outliers present in the data set. The average values of $w_1$ , $w_2$, and $w_3$ at the learning steps between 4500 and 5000 are given.*

| | $w_1$ | $w_2$ | $w_3$ | $\varepsilon_w$ |
|---|---|---|---|---|
| Original | 0.0400 | 0.0625 | 0.1111 | |
| LS | 0.046 | 0.070 | 0.081 | 0.032 |
| SVD | 0.044 | 0.078 | 0.089 | 0.027 |
| LMCA | 0.042 | 0.073 | 0.093 | 0.022 |
| NMCA | 0.040 | 0.062 | 0.109 | 0.002 |

**Figure 2:**

*The data set used in line fitting simulations. The 500 noise-disturbed sample points are obtained by adding Gaussian noise ($\sigma^2 = 0.5$) to the pure samples on the line segment.*

**Figure 3:**

*The data set used in surface fitting simulations. 933 data points of our data set are obtained by sampling the ellipsoid $0.04x^2 + 0.0625y^2 + 0.1111z^2 = 1$ in such a way that the sampling intervals on x-y plane are uniform. Thus each knot on the ellipsoid is a sample point. (a) Pure sampling data. (b) Gaussian noise $\sigma^2 = 0.3$ is added to z-axis observation data. (c) Six randomly generated outliers are presented in sampling data.*