A GENERALIZED LEARNING ALGORITHM OF MINOR COMPONENT

Fa-Long Luo and Rolf Unbehauen

Lehrstuhl für Allgemeine und Theoretische Elektrotechnik
Universität Erlangen-Nürnberg, Cauerstr. 7, D-91058 Erlangen, Germany
luo@late.e-technik.uni-erlangen.de

ABSTRACT

This paper proposes a generalized nonlinear minor component analysis algorithm. First, we will prove that with appropriate nonlinear functions the proposed algorithm can extract adaptively the minor component. Then we will discuss how to choose the related nonlinear functions so as to guarantee the desired convergence. Furthermore, we will show that all the other available minor component analysis algorithms are special cases of this proposed generalized algorithm. Finally, the complex-valued version of the proposed algorithm will be given in this paper for wider applications. In addition, this proposed minor component analysis algorithm can also be used to extract the principal component by simply reversing the sign of the corresponding terms.

1. INTRODUCTION

The eigenvector corresponding to the smallest eigenvalue of the autocorrelation matrix of an input signal is referred to as the minor component. The adaptive extraction of the minor component is an essential problem in many diverse applications of signal processing [1]. Based on the unsupervised learning (self-organization) and parallel processing properties of neural networks, a number of algorithms for extracting adaptively the minor component have been proposed and analysed [2,3,4,5,6]. However, these available minor component analysis (MCA) algorithms suffer from at least one of the following problems: (1) the norm convergence can not be guaranteed unless the smallest eigenvalue is assumed less than unity; (2) the learning phase needs division computation; (3) it is necessary to estimate the trace of the autocorrelation matrix of the input signal before to start the learning phase. These shortcomings prevent the available algorithms from being widely used and more effective MCA algorithms are desirable.

To attack the above problems, this paper proposes a generalized nonlinear minor component analysis algorithm. First, we will prove analytically and illustrate by simulation results that with appropriate nonlinear functions the proposed algorithm can extract adaptively the minor component, however neither with division computation nor with any assumption on the eigenvalues. This proposed algorithm can also avoid the estimation of the trace of the autocorrelation matrix of the input signal before learning. Second, we will discuss how to choose the related nonlinear functions so as to guarantee the desired convergence. Third, we will show that all the other available MCA algorithms are special cases of this proposed generalized MCA algorithm. Fourth, the complex-valued version of the proposed algorithm will be given in this paper for wider applications. In addition, this proposed MCA algorithm can also be used to extract the principal component by simply reversing the sign of the corresponding terms. All these demonstrate that the proposed MCA algorithm is much more suitable and can provide more flexibility for practical applications than the other available MCA algorithms.

2. PROPOSED ALGORITHM AND ITS CONVERGENCE

Let us consider a bounded continuous-valued stationary ergodic data vector $\boldsymbol{X}(t) = [x_1(t), x_2(t), \ldots, x_N(t)]^T$ with finite second-order moments, whose autocorrelation matrix $\boldsymbol{R}$ is defined as

$$\boldsymbol{R} = E[\boldsymbol{X}(t)\boldsymbol{X}^T(t)] \tag{1}$$

If $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N \geq 0$ denote the eigenvalues and $\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_N$ denote the corresponding orthonormal eigenvectors of the matrix $\boldsymbol{R}$. Then, our task is to extract adaptively the eigenvector $\boldsymbol{S}_N$ directly from the input vector $\boldsymbol{X}(t)$.

The proposed generalized algorithm is written as

$$\boldsymbol{W}(t+1) = \boldsymbol{W}(t) - \gamma(t)(z(t)g(t)\boldsymbol{X}(t) - f(t)\boldsymbol{W}(t)) \tag{2}$$

where various quantities are described as follows: $\boldsymbol{W}(t) = [w_1(t), w_2(t), \ldots, w_N(t)]^T$ is an N-dimensional weight vector; $\gamma(t)$ is a step-length parameter; $g(t)$ and $f(t)$ are nonlinear scalar functions and

$$z(t) = \sum_{i=1}^{N} w_i(t)x_i(t) = \boldsymbol{W}^T(t)\boldsymbol{X}(t) \tag{3}$$

According to the stochastic approximation theory mentioned in Reference [2,3], it can be shown that if some

conditions are satisfied, then the asymptotic limit of the above discrete learning algorithm can be solved by applying the corresponding continuous-time differential equations

$$\frac{d\boldsymbol{W}(t)}{dt} = -g(t)z(t)\boldsymbol{X}(t) + f(t)\boldsymbol{W}(t) \tag{4}$$

and

$$\frac{d\boldsymbol{W}(t)}{dt} = -g(t)\boldsymbol{R}\boldsymbol{W}(t) + f(t)\boldsymbol{W}(t) \tag{5}$$

These conditions include mainly that:
(1) $\gamma(t)$ is a sequence of positive real numbers such that $\gamma(t) \to 0$, $\sum_t \gamma^p(t) < \infty$ for some p, and $\sum_t \gamma(t) = \infty$,
(2) $\boldsymbol{X}(t)$ is zero mean, stationary and bounded with unity probability,
(3) the right-hand side term of (4) is continuously differentiable in $\boldsymbol{W}(t)$ and $\boldsymbol{X}(t)$ and its derivative is bounded in time.

Now we give a theorem concerning the convergence of the algorithm (2).

Theorem

If $\boldsymbol{W}^T(0)\boldsymbol{S}_N \neq 0$ and $g(t)$ is a positive function, then it holds

$$\lim_{t\to\infty} \boldsymbol{W}(t) = \lim_{t\to\infty} y_N(t)\boldsymbol{S}_N \tag{6}$$

where we used the notation $y_i(t) = \boldsymbol{W}^T(t)\boldsymbol{S}_i$ (for $i = 1, 2, \cdots, N$).

The proof of this theorem reads:

Using $y_i(t) = \boldsymbol{W}^T(t)\boldsymbol{S}_i$ and (5), we get

$$\frac{dy_i(t)}{dt} = -\lambda_i y_i(t)g(t) + f(t)y_i(t) \tag{7}$$
$$(\ i = 1, 2, \cdots, N)$$

According to the assumption of the initial value $\boldsymbol{W}(0)$, we may define

$$c_i(t) = \frac{y_i(t)}{y_N(t)} \tag{8}$$
$$(\ i = 1, 2, \cdots, N-1)$$

and obtain

$$\frac{dc_i(t)}{dt} = \frac{y_N(t)\frac{dy_i(t)}{dt} - y_i(t)\frac{dy_N(t)}{dt}}{(y_N(t))^2} \tag{9}$$

Combining (7) and (9) gives

$$\begin{aligned}\frac{dc_i(t)}{dt} &= \frac{1}{y_N^2(t)}((-\lambda_i y_i(t)g(t) + f(t)y_i(t))y_N(t) \\ &\quad -(-\lambda_N y_N(t)g(t) + f(t)y_N(t))y_i(t)) \\ &= (\lambda_N - \lambda_i)g(t)c_i(t) \end{aligned} \tag{10}$$
$$(\ i = 1, 2, \cdots, N-1)$$

and

$$c_i(t) = K_{iN} exp((\lambda_N - \lambda_i)\int_0^t g(\tau)d\tau) \tag{11}$$
$$(\ i = 1, 2, \cdots, N-1)$$

where K_{iN} is a constant depending on the initial values and the eigenvalues of the matrix $\boldsymbol{R}$.

Using (8) yields

$$y_i(t) = K_{iN} y_N(t) exp((\lambda_N - \lambda_i)\int_0^t g(\tau)d\tau) \tag{12}$$
$$(\ i = 1, 2, \cdots, N-1)$$

If the smallest eigenvalue λ_N is single, that is, $\lambda_N < \lambda_i$ (for $i = 1, 2, \cdots, N-1$), then we have from (12),

$$\lim_{t\to\infty} y_i(t) = 0 \tag{13}$$
$$(\ i = 1, 2, \ldots, N-1)$$

and

$$\lim_{t\to\infty} \boldsymbol{W}(t) = \lim_{t\to\infty} y_N(t)\boldsymbol{S}_N \tag{14}$$

that is (6).

In the case that the smallest eigenvalue λ_N is multiple, that is, $\lambda_N = \lambda_{N-1} = \ldots = \lambda_K = \lambda$, (12) still holds but (14) becomes

$$\lim_{t\to\infty} \boldsymbol{W}(t) = \sum_{i=K}^{N} \lim_{t\to\infty} y_i(t)\boldsymbol{S}_i. \tag{15}$$

Multiplying the above equation by $\boldsymbol{R}$ on the left yields

$$\begin{aligned}\lim_{t\to\infty} \boldsymbol{R}\boldsymbol{W}(t) &= \sum_{i=K}^{N} \lim_{t\to\infty} y_i(t)\boldsymbol{R}\boldsymbol{S}_i \\ &= \sum_{i=K}^{N} \lim_{t\to\infty} \lambda_i y_i(t)\boldsymbol{S}_i \\ &= \lambda \sum_{i=K}^{N} \lim_{t\to\infty} y_i(t)\boldsymbol{S}_i \\ &= \lambda \lim_{t\to\infty} \boldsymbol{W}(t) \end{aligned} \tag{16}$$

which means that $\lim_{t\to\infty} \boldsymbol{W}(t)$ is in the direction of the eigenvector corresponding to the multiple eigenvalue λ of the matrix $\boldsymbol{R}$. This fact and (14) show that (6) holds and conclude the proof of this theorem.

In addition, it is easy to see from the above proof that the direction of the weight vector of the algorithm (2) will converge to the direction of the second minor component $\boldsymbol{S}_{N-1}$ if $y_N(0) = \boldsymbol{W}^T(0)\boldsymbol{S}_N = 0$ and $y_{N-1}(0) = \boldsymbol{W}^T(0)\boldsymbol{S}_{N-1} \neq 0$, that is, $\lim_{t\to\infty} \boldsymbol{W}(t) = \lim_{t\to\infty} y_{N-1}(t)\boldsymbol{S}_{N-1}$. We can also come to the conclusion that the direction of the weight vector $\boldsymbol{W}(t)$ will converge to the direction of the eigenvector $\boldsymbol{S}_{i-1}$ corresponding to the eigenvalue λ_{i-1} if $y_N(0) = y_{N-1}(0) = \cdots = y_i(0) = 0$

and $y_{i-1}(0) \neq 0$, that is, $\lim_{t\to\infty} \boldsymbol{W}(t) = \lim_{t\to\infty} y_{i-1}(t)\, \boldsymbol{S}_{i-1}$.

In fact, the convergence of $\boldsymbol{W}(t)$ to $\boldsymbol{S}_N$ can be divided into two parts, that is, the direction convergence and the norm convergence. Because (12) is independent of the nonlinear function $f(t)$, the speed of direction convergence depends mainly on the positive nonlinear function $g(t)$. However, the norm convergence depends on not only $g(t)$ but also $f(t)$. The above theorem guarantees only the direction convergence but not the norm convergence. It can be shown that one sufficient condition to guarantee the norm convergence of the proposed algorithm is

$$f(t) \leq \frac{z^2(t)}{\boldsymbol{W}^T(t)\boldsymbol{W}(t)} g(t) \tag{17}$$

Moreover, if we select

$$f(t) = \frac{z^2(t)}{\boldsymbol{W}^T(t)\boldsymbol{W}(t)} g(t) \tag{18}$$

then the norm of $\boldsymbol{W}(t)$ will be invariant during the learning phase and equal to the norm of the initial weight vector $\boldsymbol{W}(0)$, that is,

$$\| \boldsymbol{W}(t) \| = \| \boldsymbol{W}(0) \|, \quad t > 0, \tag{19}$$

$$\lim_{t\to\infty} \boldsymbol{W}(t) = \pm \| \boldsymbol{W}(0) \| \boldsymbol{S}_N \tag{20}$$

This invariant property means that only the direction convergence is involved in the learning phase and suggests that another sufficient condition to guarantee the norm convergence is

$$f(t) = \frac{z^2(t)}{\boldsymbol{W}^T(0)\boldsymbol{W}(0)} g(t) \tag{21}$$

In addition, this property is also very useful in designing the hardware implementation of the proposed algorithm.

Based on the above, we can select appropriate functions $g(t)$ and $f(t)$ so as to guarantee the direction convergence and the norm convergence without any shortcomings mentioned above. A very simple example to select $g(t)$ and $f(t)$ is

$$g(t) = \boldsymbol{W}^T(t)\boldsymbol{W}(t) \tag{22}$$

$$f(t) = z^2(t) \tag{23}$$

According to (21), we can also select

$$g(t) = \boldsymbol{W}^T(0)\boldsymbol{W}(0) \tag{24}$$

$$f(t) = z^2(t) \tag{25}$$

3. FURTHER DISCUSSIONS

It is easy to show that the other available MCA algorithms are all special cases of the generalized algorithm (2), that is,

1. for the constrained anti-Hebbian learning algorithm presented in [2,3], $g(t) = 1$, $f(t) = z^2(t)$.
2. for the normalized version of the constrained anti-Hebbian learning algorithm proposed in [3], $g(t) = 1$, $f(t) = \frac{z^2(t)}{\boldsymbol{W}^T(t)\boldsymbol{W}(t)}$;
3. for another constrained anti-Hebbian learning algorithm reported in [2,6], $g(t) = 1$, $f(t) = z^2(t) + 1 - \boldsymbol{W}^T(t)\boldsymbol{W}(t)$;
4. for the algorithm proposed in [4], $g(t) = 1$, $f(t) = z(t)x_N(t)$;
5. for the algorithm suggested in [5], $g(t) = 1$, $f(t) = 2K(1 - \boldsymbol{W}^T(t)\boldsymbol{W}(t))$, where K is a positive parameter which is determined by the trace of the autocorrelation matrix.

For wider applications, the proposed algorithm (2) can be extended to the complex-valued case. In the complex-valued case, (2) becomes

$$\boldsymbol{W}_c(t+1) = \boldsymbol{W}_c(t) - \gamma(t)(g(t)\boldsymbol{X}_c(t)z_c(t) - f(t)\boldsymbol{W}_c(t)) \tag{26}$$

where

$$z_c(t) = \boldsymbol{X}_c^T(t)\boldsymbol{W}_c(t) \tag{27}$$

$$z_c(t) = \begin{pmatrix} z_r(t) \\ z_i(t) \end{pmatrix}, \quad \boldsymbol{X}_c(t) = \begin{pmatrix} \boldsymbol{X}_r(t) & -\boldsymbol{X}_i(t) \\ \boldsymbol{X}_i(t) & \boldsymbol{X}_r(t) \end{pmatrix}$$

$$\boldsymbol{W}_c(t) = \begin{pmatrix} \boldsymbol{W}_r(t) \\ \boldsymbol{W}_i(t) \end{pmatrix}$$

and $\boldsymbol{X}(t) = \boldsymbol{X}_r(t) + j\boldsymbol{X}_i(t)$, $\boldsymbol{W}(t) = \boldsymbol{W}_r(t) + j\boldsymbol{W}_i(t)$, $z(t) = z_r(t) + jz_i(t)$, and $z(t) = \boldsymbol{W}^H(t)\boldsymbol{X}(t)$.

It is worthy mentioning that if we simply reverse the sign of the second and the third term of the algorithm (2), that is,

$$\boldsymbol{W}(t+1) = \boldsymbol{W}(t) + \gamma(t)(z(t)g(t)\boldsymbol{X}(t) - f(t)\boldsymbol{W}(t)) \tag{28}$$

then the weight vector provided by the algorithm (28) will converge to the principal component $\boldsymbol{S}_1$ on the condition that $\boldsymbol{W}^T(0)\boldsymbol{S}_1 \neq 0$, $g(t) > 0$ and

$$f(t) \geq \frac{z^2(t)}{\boldsymbol{W}^T(t)\boldsymbol{W}(t)} g(t).$$

4. SIMULATIONS

We have simulated the proposed MCA algorithm. Two examples are given in this paper. We generated a random vector $\boldsymbol{X}(t)$ according to one of the most common models in signal processing, that is, $x_i(t) = \sum_{j=1}^{L} a_j(t)\, cos(2(i-1)\pi f_j + b_j(t)) + c_i(t)$ where $a_j(t)$ is a magnitude parameter sequence, f_j is a frequency parameter, $b_j(t)$ is the initial phase which is an uniform sequence in $[0, 2\pi]$,

$c_i(t)$ is a zero-mean random sequence. We selected nonlinear functions as (24) and (25) show. In the first example, the smallest eigenvalue is single. In the second example, the smallest eigenvalue is multiple. For simplicity, in the simulations we let $\gamma(t)$ be unchanged during the learning phase. A more sophisticated and better selection of $\gamma(t)$ could be made according to the Robbins-Monro procedures [3]. $\boldsymbol{R}$ is the autocorrelation matrix with $N \times N$ elements, $\boldsymbol{W}(0)$ is the initial vector. $\boldsymbol{W}(f)$ is the weight vector provided by the algorithm (2) in the convergent state. λ_a is the exact smallest eigenvalue and λ_f is obtained by computing $\lambda_f = \frac{\boldsymbol{W}^T(f)\boldsymbol{R}\boldsymbol{W}^T(f)}{\boldsymbol{W}^T(f)\boldsymbol{W}^T(f)}$. Obviously, $\lambda_f \approx \lambda_a$. For further illustration, Figures 1 and 2 describe the learning process of the weight vector $\boldsymbol{W}(t)$.

Example 1:

$$\boldsymbol{R} = \begin{pmatrix} 13.5000 & 11.2490 & 7.8627 & 3.3117 \\ 11.2490 & 13.5000 & 11.2490 & 7.8627 \\ 7.8627 & 11.2490 & 13.5000 & 11.2490 \\ 3.3117 & 7.8627 & 11.2490 & 13.5000 \end{pmatrix}$$

$$\boldsymbol{W}(f) = \begin{pmatrix} 0.5098 & -0.8000 & 0.3088 & 0.1008 \end{pmatrix}^T$$

$$\boldsymbol{W}(0) = \begin{pmatrix} 0.5000 & -0.5000 & 0.5000 & 0.5000 \end{pmatrix}^T$$

$$\lambda_a = 1.0026, \quad \lambda_f = 1.0470$$

Example 2:

$$\boldsymbol{R} = \begin{pmatrix} 5.5000 & 3.6406 & 1.3906 & -1.3906 \\ 3.6406 & 5.5000 & 3.6406 & 1.3906 \\ 1.3906 & 3.6406 & 5.5000 & 3.6406 \\ -1.3906 & 1.3906 & 3.6406 & 5.5000 \end{pmatrix}$$

$$\boldsymbol{W}(f) = \begin{pmatrix} 0.6557 & -0.6052 & -0.0826 & 0.4469 \end{pmatrix}^T$$

$$\boldsymbol{W}(0) = \begin{pmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0000 \end{pmatrix}^T$$

$$\lambda_a = 1.0000, \quad \lambda_f = 1.0004$$

The simulation results have demonstrated the accuracy of the above analyses and the effectiveness of the proposed algorithm.

5. ACKNOWLEDGEMENT

This work is supported by the German Research Foundation (DFG).

REFERENCES

[1] Haykin, S. "Neural Networks Expand SP's Horizons," IEEE Signal Processing Magazine, Vol.13, No.2, 1996, pp.24-49.

[2] Oja, E., "Principal Components, Minor Components and Linear Neural Networks," Neural Networks, Vol.5, 1992, pp.927-935.

[3] Xu, L., Oja, E. and Suen, C.Y., "Modified Hebbian Learning for Curve and Surface Fitting," Neural Networks, Vol.5, 1992, pp.441-457.

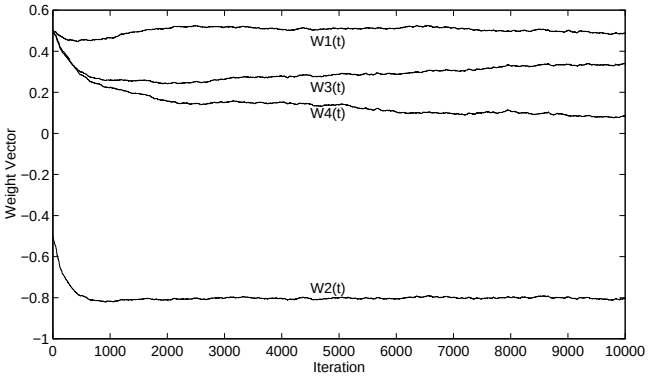

FIGURE 1. The learning process of the weight vector $\boldsymbol{W}(t)$ in Example 1.

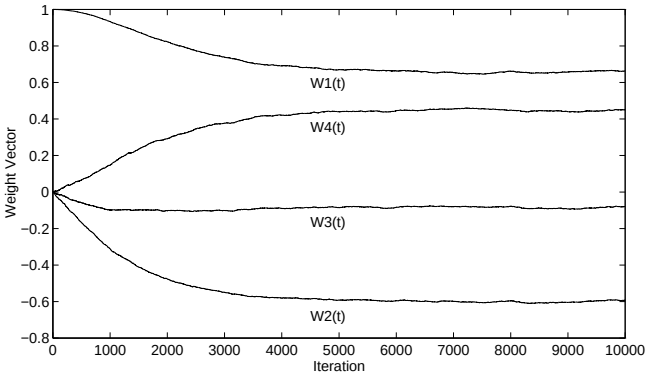

FIGURE 2. The learning process of the weight vector $\boldsymbol{W}(t)$ in Example 2.

[4] Gao, K., Ahmad, M.O. and Swamy, M.N.S., "A Constrained Anti-Hebbian Learning for Total Least-Squares Estimation with Applications to Adaptive FIR and IIR Filtering," IEEE Transactions on Circuits and Systems: II, Vol.41, 1994, pp. 718-729.

[5] Mathew, G. and Reddy, V.U., "Development and Analysis of a Neural Network Approach to Pisarenko's Harmonic Retrieval Method," IEEE Transactions on Signal Processing, Vol. 42, 1994, pp.663-667.

[6] Oja, E. and Wang. L., "Robust Fitting by Nonlinear Neural Units," Neural Networks, Vol.9, No.3, 1996, pp.435-444.