BLIND SOURCE SEPARATION USING LEAST-SQUARES TYPE ADAPTIVE ALGORITHMS

Juha Karhunen and Petteri Pajunen

Helsinki University of Technology
Laboratory of Computer and Information Science
Rakentajanaukio 2 C, FIN-02150 Espoo, FINLAND
e-mail: Juha.Karhunen@hut.fi, Petteri.Pajunen@hut.fi

ABSTRACT

In this paper adaptive least-squares type algorithms are introduced for blind source separation. They are based on minimizing a criterion used in context with nonlinear PCA (Principal Component Analysis) networks. The new algorithms converge clearly faster and provide more accurate results than typical current adaptive blind separation algorithms based on instantaneous gradients. They are also applicable to the difficult case of nonstationary mixtures. The proposed algorithms have a close relationship to a nonlinear extension of Oja's PCA learning rule. A batch algorithm based on the same criterion is also presented.

1. INTRODUCTION

Blind source separation (BSS) has lately become a popular research area both in statistical signal processing and unsupervised neural learning. In BSS, the goal is to separate mutually statistically independent but otherwise unknown source signals from their linear mixtures without knowing the mixing coefficients. BSS techniques are needed in many applications of signal processing [1].

Conventional linear discrete-time signal processing systems are not adequate in the BSS problem, because some higher-order statistics must be employed for achieving separation. In neural (or adaptive) BSS methods [2, 3, 4, 5], this is typically done implicitly by using suitable nonlinearities in the learning algorithms. Different neural approaches to BSS and to the closely related Independent Component Analysis (ICA) are reviewed in the recent tutorial paper [5]. Several authors have independently shown that fairly simple neural rules are able to learn a satisfactory separating solution in many instances. In particular, we have recently shown that certain nonlinear PCA type neural algorithms can sometimes successfully separate even 10 sources on certain conditions [6].

However, the existing adaptive and neural separation algorithms are usually based on a crude instantaneous estimate of the stochastic gradient. Such algorithms are fairly simple, but require a careful choice of the learning parameter(s) for acceptable performance. If the learning parameter is too small, convergence can be intolerably slow. On the other hand the algorithm may become unstable if the learning parameter is chosen too large. It is difficult to apply these algorithms to large-scale problems because of the poor accuracy of the instantaneous gradient estimate. Furthermore, convergence speed can greatly depend on initial values.

In [7] Yang introduced computationally efficient approximate recursive least-squares algorithms for tracking signal subspaces or principal (PCA) eigenvectors of the data covariance matrix. However, these algorithms cannot be used for the BSS problem because they utilize second-order statistics only. In this paper we combine our earlier results with Yang's algorithms. By incorporating nonlinearities, we get new adaptive BSS algorithms which converge much faster than existing neural blind separation algorithms.

2. BLIND SOURCE SEPARATION

The blind source separation problem has the following basic form. Assume that there exist m zero mean source signals $s_1(t), \dots, s_m(t)$ that are scalar-valued and mutually statistically independent (or as independent as possible) at each time instant t. The original sources $s_i(t)$ are unknown, and all that we have are n possibly noisy but different linear mixtures $x_1(t), \dots, x_n(t)$ of the sources. The mixing coefficients are some unknown constants. In blind source separation, the task is to find the waveforms $\{s_i(t)\}$ of the sources, knowing only the mixtures $x_j(t)$.

Denote by $\mathbf{x}(t) = [x_1(t), \dots, x_n(t)]^T$ the n-dimensional t:th data vector made up of the mixtures at discrete time (or point) t. The BSS signal model then takes the form

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t). \tag{1}$$

Here $\mathbf{s}(t) = [s_1(t), \dots, s_m(t)]^T$ is the source vector, and $\mathbf{A}$ is a constant full-rank $n \times m$ mixing matrix whose elements are the unknown coefficients of the mixtures. The additive noise term $\mathbf{n}(t)$ is often omitted from (1), because it is usually impossible to separate noise from the source signals.

The number of available different mixtures n must be at least as large as the number of sources m. Usually m is assumed known, and often $m = n$. Furthermore, each source signal $s_i(t)$ is assumed to be a stationary zero-mean stochastic process. Only one of the sources is allowed to have a Gaussian distribution.

In neural and adaptive BSS, an $m \times n$ separating matrix $\mathbf{B}(t)$ is updated so that the m-vector

$$\mathbf{y}(t) = \mathbf{B}(t)\mathbf{x}(t) \tag{2}$$

becomes an estimate $\mathbf{y}(t) = \hat{\mathbf{s}}(t)$ of the original independent source signals. In neural realizations, $\mathbf{y}(t)$ is the output vector of the network, and the matrix $\mathbf{B}(t)$ is the total weight matrix between the input and output layers. The estimate $\hat{s}_i(t)$ of the i:th source signal may appear in any component $y_j(t)$ of $\mathbf{y}(t)$. The amplitudes of the estimates $y_j(t)$ are typically scaled so that they have unit variance.

In many BSS algorithms, the data vectors $\mathbf{x}(t)$ are preprocessed by whitening them. This is done by applying a linear transform which makes the covariance matrix of the whitened vectors $m \times m$ unit matrix $\mathbf{I}_m$. Standard PCA is often used for this purpose [5]. In tracking applications it is necessary to use adaptive whitening. One such update rule is

$$\Delta \mathbf{V}(t) = \mu[\mathbf{I} - \mathbf{v}(t)\mathbf{v}^T(t)]\mathbf{V}(t) \tag{3}$$

where $\mathbf{V}(t)$ is the whitening matrix and $\mathbf{v}(t) = \mathbf{V}(t)\mathbf{x}(t)$ is the whitened vector. After prewhitening the separation task becomes somewhat easier, because the components of the whitened vectors $\mathbf{v}(t)$ are uncorrelated. Also the subsequent separating matrix, denoted here for clarity by $\mathbf{W}^T(t)$, can be taken orthogonal: $\mathbf{W}^T(t)\mathbf{W}(t) = \mathbf{I}_m$.

In the standard stationary case, the whitening and separating matrices converge to some constant values during learning. However, the same model can be used in nonstationary situations by keeping these matrices time-varying.

As a separating criterion, we use the sum of kurtoses $\mathrm{E}\{y_i(t)^4\} - 3[\mathrm{E}\{y_i(t)^2\}]^2$ of the outputs of the network, because this provides simple but yet sufficiently efficient neural algorithms. The theory is presented in more detail in [5]. This criterion is minimized for sub-Gaussian sources (for which the kurtosis is negative) and maximized for super-Gaussian sources (having a positive kurtosis).

3. NONLINEAR PCA CRITERION

We have previously studied various robust and nonlinear extensions of PCA neural networks and learning algorithms in several papers; see [8]. In particular, we have considered the cost function

$$J_1(\mathbf{W}) = \mathrm{E}\{\|\mathbf{x} - \mathbf{W}\mathbf{g}(\mathbf{W}^T\mathbf{x})\|^2\}. \tag{4}$$

Here $\mathbf{g}(\mathbf{y})$ denotes the vector which is obtained by applying an odd nonlinear function $g(t)$ to each component of the vector $\mathbf{y}$. A typical choice is $g(t) = \tanh(t)$. The criterion (4) can be approximately minimized with respect to the weight matrix $\mathbf{W}$ using the nonlinear PCA subspace rule derived in [9]:

$$\Delta \mathbf{W}(t) = \mu(t)[\mathbf{x}(t) - \mathbf{W}(t)\mathbf{g}(\mathbf{y}(t))]\mathbf{g}(\mathbf{y}^T(t)) \tag{5}$$

Here $\Delta\mathbf{W}(t) = \mathbf{W}(t{+}1) - \mathbf{W}(t)$, and $\mu(t)$ is a positive learning parameter, usually a small constant. The algorithm (5) utilizes instantaneous stochastic gradient.

Later on, we have shown [6] that if the mixture vectors $\mathbf{x}(t)$ are prewhitened, $\mathbf{W}(t)$ becomes an orthogonal $m \times m$ separating matrix. For sub-Gaussian sources, the sigmoidal nonlinearity $g(t) = \tanh(t)$ provides separation [6]. The nonlinear PCA rule (5) can be applied also for super-Gaussian sources using Fahlman type nonlinearities [10].

Theoretical justifications for the use of the criterion (4) and the algorithm (5) in separation are presented in [6] and [10]. In particular, the local minima of $J_1(\mathbf{W})$ correspond to separating orthogonal matrices $\mathbf{W}$ for prewhitened data vectors.

4. NONLINEAR LEAST-SQUARES ALGORITHMS

In the linear special case $g(t) = t$ the criterion (4) reduces to

$$J_2(\mathbf{W}) = \mathrm{E}\{\|\mathbf{x} - \mathbf{W}\mathbf{W}^T\mathbf{x}\|^2\}. \tag{6}$$

It is well known [9] that the cost function (6) is minimized by any orthogonal matrix $\mathbf{W}$ whose columns lie in the PCA (signal) subspace defined by the principal eigenvectors of the data covariance matrix $\mathrm{E}\{\mathbf{x}\mathbf{x}^T\}$. Yang has recently derived in [7] a recursive least-squares algorithm called PAST for minimization of the linear PCA criterion (6). The PAST algorithm can be used for adaptive estimation and tracking of the PCA/signal subspace.

In this paper we extend Yang's PAST algorithm so that it can be used for minimizing the more general nonlinear PCA criterion (4). One iteration of our modified PAST algorithm is

$$\begin{aligned}
\mathbf{z}(t) &= \mathbf{g}(\mathbf{W}^T(t-1)\mathbf{x}(t)) \\
\mathbf{h}(t) &= \mathbf{P}(t-1)\mathbf{z}(t) \\
\mathbf{m}(t) &= \mathbf{h}(t)/(\beta + \mathbf{z}^T(t)\mathbf{h}(t)) \\
\mathbf{P}(t) &= \frac{1}{\beta}\,\mathrm{U}\left[\mathbf{P}(t-1) - \mathbf{m}(t)\mathbf{h}^T(t)\right] \\
\mathbf{e}(t) &= \mathbf{x}(t) - \mathbf{W}(t-1)\mathbf{z}(t) \\
\mathbf{W}(t) &= \mathbf{W}(t-1) + \mathbf{e}(t)\mathbf{m}^T(t)
\end{aligned} \tag{7}$$

The constant $0 < \beta \leq 1$ is a forgetting term which should be close to unity. U denotes that only the upper triangular part of the argument is computed. Its transpose is then copied to the lower triangular part so that the matrix becomes symmetric. The initial values $\mathbf{W}(0)$ and $\mathbf{P}(0)$ can be chosen for example to $m \times m$ unit matrices. In applying the algorithm (7) to the BSS problem, the data vectors $\mathbf{x}(t)$ must be prewhitened according to the previous theory.

The algorithm (7) can be derived as follows [7, 11]. The unknown expectation in (4) is first replaced by the respective sample mean, leading to a similar least-squares criterion. Then the unknown vector $\mathbf{g}(\mathbf{W}^T(t)\mathbf{x}(t))$ is approximated by $\mathbf{z}(t) = \mathbf{g}(\mathbf{W}^T(t-1)\mathbf{x}(t))$. This vector can be computed because the previous weight matrix $\mathbf{W}^T(t-1)$ is known in the adaptive algorithm. The approximation error is usually small because the update $\Delta\mathbf{W}(t)$ becomes small compared with $\mathbf{W}(t)$ after initial convergence. These steps lead to the modified least-squares type criterion

$$J_3(\mathbf{W}(t)) = \sum_{i=1}^{t} \beta^{t-i} \parallel \mathbf{x}(i) - \mathbf{W}(t)\mathbf{z}(i) \parallel^2 . \tag{8}$$

We have included a forgetting factor β into the criterion (8). If $\beta = 1$, no forgetting of old data takes place. Choosing $\beta < 1$ is useful especially in tracking nonstationary changes in the data. The cost function (8) has now the standard form used in recursive least-squares methods [11]. Several algorithms exist for finding the optimal weight matrix $\mathbf{W}(t)$ iteratively. We have here used the efficient algorithm (7).

The algorithm (7) can be regarded either as a neural network learning algorithm or adaptive signal processing algorithm. It doesn't require any matrix inversions, the most complicated operation being a division by a scalar. The algorithm can be modified easily so that the weight vectors are computed sequentially using a deflation technique, resulting in the following algorithm:

$$
\begin{aligned}
&\mathbf{x}_1(t) = \mathbf{x}(t); \\
&\text{For each } i = 1, \dots, m \text{ compute} \\
&\quad z_i(t) = g(\mathbf{w}_i^T(t-1)\mathbf{x}_i(t)), \\
&\quad d_i(t) = \beta d_i(t-1) + [z_i(t)]^2, \\
&\quad \mathbf{e}_i(t) = \mathbf{x}_i(t) - \mathbf{w}_i(t-1) z_i(t), \\
&\quad \mathbf{w}_i(t) = \mathbf{w}_i(t-1) + \mathbf{e}_i(t)[z_i(t)/d_i(t)], \\
&\quad \mathbf{x}_{i+1}(t) = \mathbf{x}_i(t) - \mathbf{w}_i(t) z_i(t).
\end{aligned}
\tag{9}
$$

In the case of only one weight vector ($m = 1$), both (7) and (9) reduce to the form

$$
\mathbf{w}(t) = \mathbf{w}(t-1) + \frac{1}{d(t)}[\mathbf{x}(t) - \mathbf{w}(t-1)z(t)]z(t) \tag{10}
$$

where $z(t) = g(\mathbf{w}^T(t-1)\mathbf{x}(t)) = g(y(t))$, and $d(t) = \beta d(t-1) + [z(t)]^2$. A comparison with the nonlinear PCA learning rule (5) shows that these two algorithms are the same except for the scalar learning parameter. In (10), the learning parameter $1/d(t)$ is determined automatically from the properties of the data so that it is roughly optimal due to the minimization of the least-squares criterion (8). On the other hand, in (5) the learning parameter $\mu(t)$ is usually a constant which is chosen by somewhat ad hoc manner or by tuning it to the average properties of the data. It is just this nearly optimal choice of the learning parameter that yields the recursive least-squares algorithms their superior convergence properties compared to standard stochastic gradient type learning algorithms such as (5).

5. BATCH ALGORITHM

The form of the cost function (4) suggests a straightforward batch algorithm for optimizing the matrix $\mathbf{W}$. If we assume for the moment that the matrix $\mathbf{W}$ in the term $\mathbf{g}(\mathbf{W}^T\mathbf{x})$ is constant, we can consider (4) as the least-square error for the linear model

$$
\mathbf{X} = \mathbf{W}\mathbf{g}(\mathbf{W}^T\mathbf{X}) + \mathbf{e} = \mathbf{W}\mathbf{G} + \mathbf{e}, \tag{11}
$$

where

$$
\begin{aligned}
\mathbf{G} &= [\mathbf{g}(\mathbf{W}^T\mathbf{x}(1)), \dots, \mathbf{g}(\mathbf{W}^T\mathbf{x}(N))], \\
\mathbf{X} &= [\mathbf{x}(1), \dots, \mathbf{x}(N)].
\end{aligned}
$$

The best approximate solution in the least-squares sense is [12]

$$
\mathbf{W} = \mathbf{X}\mathbf{G}^+ = \mathbf{X}\mathbf{G}^T(\mathbf{G}\mathbf{G}^T)^{-1}, \tag{12}
$$

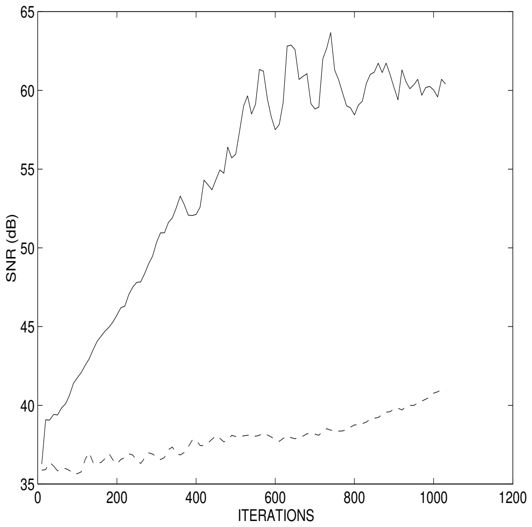

Figure 1: Average signal-to-noise ratios in dB. Solid line: nonlinear least-squares algorithm. Dashed line: nonlinear PCA subspace rule.

where $\mathbf{G}^+$ is the pseudoinverse of $\mathbf{G}$. By setting an initial value for $\mathbf{W}$, for example the unit matrix, and iterating the formula (12) we get an iterated least-squares batch algorithm for minimizing (4). A sequential version of this batch algorithm can be derived in a similar manner [12].

If the number N of the data vectors $\mathbf{x}$ is large, the computational load of the batch algorithm (12) becomes clearly higher than that of the recursive algorithm (7). In this case, one can use smaller subsequent segments of the data matrix $\mathbf{X}$ in each iteration of the batch algorithm. For more details and experimental results, see [12].

6. EXPERIMENTS

In the first experiment the convergence speed of the proposed nonlinear least-squares type algorithm (7) was compared to the nonlinear PCA algorithm (5). Four mixtures of four sub-Gaussian source signals (a sinusoid, uniformly distributed white noise, a ramp signal, and a binary signal) were constructed. The elements of the mixing matrix were Gaussian random numbers. These mixtures were first whitened and then the separating matrix was estimated using the algorithms (7) and (5). In both algorithms, the nonlinear learning function was the sigmoid $g(t) = \tanh(t)$. The learning rate was $\mu = 0.01$ for the nonlinear PCA algorithm (5), and the forgetting factor was $\beta = 0.99$ for the nonlinear least-squares algorithm (7). The resulting average signal-to-noise ratios computed after every tenth iteration are shown in Fig. 1. In computing the SNRs, noise denotes the error between the true source signal and its best estimate. Thus Fig. 1 desribes the quality of the separation results, and shows that the least-squares type algorithm (7) converges dramatically faster than the nonlinear PCA algorithm (5).

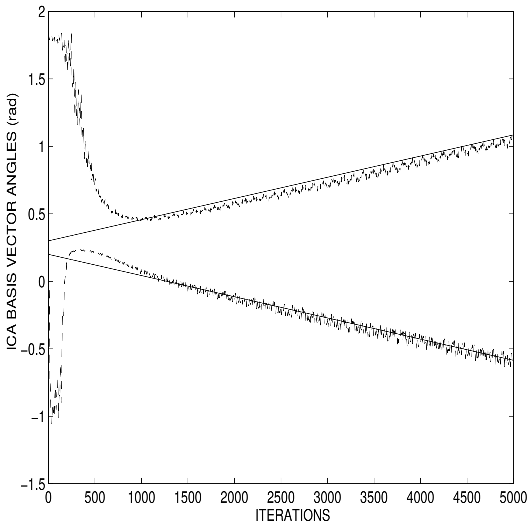

Figure 2: The estimation of the angles θ_1 and θ_2. The two solid diverging lines are the correct values of θ_i. The oscillating curves are the estimated angles.

The results were qualitatively similar in all the computer simulations that we have made thus far. After a sufficiently large number of iterations, the nonlinear PCA algorithm and other similar stochastic gradient type separation algorithms usually converge close to a separating solution. However, the number of iterations needed may vary greatly depending on the initial values and learning parameters. These algorithms have sometimes difficulties in converging at all especially if the number of sources m is not small. The nonlinear least-squares algorithm (7) converges much faster.

A simple tracking experiment was made by using a time-dependent mixing matrix

$$\mathbf{A}(t) = \begin{pmatrix} \cos(\theta_1(t)) & \cos(\theta_2(t)) \\ \sin(\theta_1(t)) & \sin(\theta_2(t)) \end{pmatrix}$$

The columns of $\mathbf{A}(t)$ are unit vectors with angle $\theta_i(t)$. The independent source signals were a sinusoid and a ramp signal, and the total number of samples was 5000. The angles were initialized to $\theta_1(1) = 0.3$ and $\theta_2(1) = 0.2$. During the simulation, they linearly changed to the final values $\theta_1(5000) = 0.3 + \pi/4$ and $\theta_2(5000) = 0.2 - \pi/4$ as shown by the solid lines in Fig. 2. The algorithm (3) was used for whitening. From the separating matrix $\mathbf{B}(t)$ estimated using the nonlinear least-squares algorithm (7), we computed the respective estimated angles. These are depicted by the more randomly fluctuating curves in Fig. 2. Fig. 2 shows that the algorithm (7) is able to track the changes in the mixing matrix with good accuracy after the initial transient phase.

7. CONCLUSIONS

In this paper, we have introduced fast adaptive RLS type algorithms for blind source separation. The proposed algorithms provide good performance at a moderate computational cost. They can be used also in nonstationary situations.

8. REFERENCES

[1] J. Karhunen, A. Hyvärinen, R. Vigario, J. Hurri, and E. Oja, "Applications of neural blind separation to signal and image processing." To appear in *Proc. 1997 Int. Conf. on Acoustics, Speech, and Signal Proc. (ICASSP-97)*, Munich, Germany, April 1997.

[2] J.-F. Cardoso and B. Laheld, "Equivariant adaptive source separation." to appear in *IEEE Trans. on Signal Processing*, vol. 44, no. 12, Dec. 1996.

[3] A. Bell and T. Sejnowski, "An information-maximisation approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, 1995.

[4] C. Jutten and J. Herault, "Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, 1991.

[5] J. Karhunen, "Neural approaches to independent component analysis and source separation," in *Proc. of the 4th European Symposium on Artificial Neural Networks (ESANN'96)*, (Bruges, Belgium), pp. 249–266, April 1996.

[6] E. Oja and J. Karhunen, "Signal separation by nonlinear Hebbian learning," in *Computational Intelligence – A Dynamic System Perspective* (M. Palaniswami *et al.*, eds.), pp. 83–97, IEEE Press, 1995.

[7] B. Yang, "Projection approximation subspace tracking," *IEEE Transactions on Signal Processing*, vol. 43, pp. 95–107, January 1995.

[8] E. Oja, J. Karhunen, and A. Hyvärinen, "From neural PCA to neural ICA," in *NIPS Post-Conference Workshop on Blind Signal Processing*, (Snowmass, Colorado, USA), December 1996.

[9] J. Karhunen and J. Joutsensalo, "Representation and separation of signals using nonlinear PCA type learning," *Neural Networks*, vol. 7, no. 1, pp. 113–127, 1994.

[10] M. Girolami and C. Fyfe, "Stochastic ICA contrast maximisation using Oja's nonlinear PCA algorithm." Submitted to *Int. J. of Neural Systems*, 1996.

[11] J. Mendel, *Lessons in Estimation Theory for Signal Processing, Communications, and Control.* Prentice-Hall, 1995.

[12] P. Pajunen and J. Karhunen, "Iterated least-squares batch algorithms for blind source separation." Submitted to *IEEE Signal Processing Letters*, 1997.