A Simple Adaptive Algorithm for Principle Component and Independent Component Analysis*

Hyun-Chool SHIN†*, Nonmember, Hyoung-Nam KIM††, Member, and Woo-Jin SONG†††, Nonmember

SUMMARY  In this letter we propose a simple adaptive algorithm which solves the unit-norm constrained optimization problem. Instead of conventional parameter norm based normalization, the proposed algorithm incorporates single parameter normalization which is computationally much simpler. The simulation results illustrate that the proposed algorithm performs as good as conventional ones while being computationally simpler.

key words: constrained optimization, unit-norm, principle component analysis, independent component analysis

1. Introduction

Constrained optimization is the minimization of an objective function subjected to constraints on the possible values of the independent parameters. A typical problem arising in signal processing for the parameter vector \( \mathbf{w} = [w_1, w_2, \ldots, w_K]^T \) is to minimize a quadratic cost function

\[
J(\mathbf{w}) = E[(d(i) - \mathbf{w}^T \mathbf{x}_i)^2]
\]

subject to

\[
\|\mathbf{w}\|^2 = \sum_{k=1}^{K} |w_k|^2 = 1,
\]

where \( d(i) \) is the desired data and \( \mathbf{x}_i \) denotes the \((K\times1)\) input vector. This is one of the basic and fundamental problems in communications, controls and adaptive signal processing [1]–[5].

The well-known stochastic gradient algorithm for minimizing (1) has the following form

\[
\begin{align*}
\tilde{\mathbf{w}}_i &= \mathbf{w}_{i-1} + \mu e(i) \mathbf{x}_i, \quad \text{and} \\
\mathbf{w}_i &= \frac{\tilde{\mathbf{w}}_i}{\|\tilde{\mathbf{w}}_i\|}.
\end{align*}
\] (2)

where \( e(i) = d(i) - \mathbf{w}_{i-1}^T \mathbf{x}_i \), \( \mathbf{w}_i \) is an estimate for the optimal estimate \( \mathbf{w}^* \) at iteration \( i \), and \( \mu \) is the step-size [5].

The adaptive algorithm in (2) is realized in two steps. The first step is the same as the conventional least mean square (LMS) algorithm which minimizes \( J(\mathbf{w}) \) itself without any constraint. Then, in the second step, the constraint is separately kept by normalizing \( \tilde{\mathbf{w}}_i \) by \( \|\mathbf{w}_i\| \) so that the algorithm satisfies \( \|\mathbf{w}_i\| = 1 \) at every iteration.

2. Single Parameter Normalization

Here we propose a simpler adaptive algorithm which solves the unit-norm constrained optimization problem. Instead of conventional parameter norm based normalization in (2), the proposed method incorporates single parameter normalization (SPN):

\[
\begin{align*}
\tilde{\mathbf{w}}_i &= \mathbf{w}_{i-1}^{spn} + \mu e(i) \mathbf{x}_i, \quad \text{and} \\
\mathbf{w}_i^{spn} &= \frac{\tilde{\mathbf{w}}_i}{\|\tilde{\mathbf{w}}_i\|},
\end{align*}
\] (3)

where \( \tilde{\mathbf{w}}_k(i) \) is an arbitrary \( k \)-th single parameter in \( \tilde{\mathbf{w}}_i \) \((\tilde{\mathbf{w}}_k(i) \neq 0)\). To avoid notational confusion between (2) and (3), we denote an estimate of \( \mathbf{w}^* \) as \( \mathbf{w}_i^{spn} \).

To give a better insight, we compare the two schemes of (2) and (3) from a geometric perspective in Fig. 1. Without loss of generality, assume that the current estimate at the \((i-1)\)-th iteration is \( \mathbf{w}_{i-1} = \mathbf{w}_{i-1}^{spn} \). After the update done by the upper equation in (2) and (3), the two algorithms result in the same intermediate estimate \( \tilde{\mathbf{w}}_i \) since the two upper equations for update are exactly the same. Then the normalization step in (2) moves \( \tilde{\mathbf{w}}_i \) onto unit circle or hypersphere, \( \|\mathbf{w}\| = 1 \).

On the other hand, the proposed normalization step places

**Fig. 1** Geometric illustration of two normalization methods.
\(\hat{w}_i\) on the hyperplane of \(|w_k| = 1\). The single parameter normalization step in (3) does not change the direction of \(\hat{w}_i\). This means that the two estimates are the same within a scale.

Table 1 describes the computational complexity required for performing the normalization steps. To calculate the CPU time needed for two normalization procedures, the simulation was performed using the Matlab version 7.0 on Intel(R) Core(TM)2 1.86 GHz processor with 1G RAM. For \(K=16\), the CPU times for 10000 normalizations in Eq. (2) and Eq. (3) are 0.094 sec. and 0.047 sec., respectively.

### 3. Applications

To demonstrate the proposed method indeed solves the constrained optimization problem, we apply the method to two well-known signal processing problems: principle component analysis (PCA) and independent component analysis (ICA).

#### 3.1 Principle Component Analysis (PCA)

PCA is a well-established technique for dimension reduction. Its applications include data compression, image processing, data visualization, pattern recognition, and time series prediction [2]–[4]. Let \(x \in \mathbb{R}^K\) denote an \(K\)-dimensional zero mean random vector. Consider a single neural network whose output \(y\) is given by

\[
y = w^T x.
\]

The PCA aims at finding a vector \(w\) such that the variance of \(y\) is maximized, i.e., the cost function in (1) and \(e(i)\) in (2) become

\[
J(w) = E[|w^T x_i|^2]
\]

subject to \(||w|| = 1\) and \(e(i) = w^T x_i = y(i)\), respectively and thus the optimum is

\[
w^* = \arg \max J(||y(i)||^2).
\]

It is well-known that the solution \(w^*\) corresponds to the normalized eigenvector associated with the largest eigenvalue of the covariance matrix \(R_{xx} = E[xx^T]\). A conventional adaptive algorithm finding \(\hat{w}\) is

\[
\hat{w}_i = w_{i-1} + \mu y_i x_i, \quad \text{and} \quad \hat{w}_i = \frac{\hat{w}_i}{||\hat{w}_i||}
\]

(7)

Here we propose a simpler PCA algorithm based on single parameter normalization:

\[
\hat{w}_i = w_{i-1} + \mu y_i x_i, \quad \text{and} \quad w_i = \frac{\hat{w}_i}{||\hat{w}_i||}
\]

(8)

In the simulation the input vector \(x\) is chosen to have the covariance matrix of

\[
R_{xx} = \begin{pmatrix}
0.10 & 0.70 & 0.49 & 0.34 \\
0.70 & 0.10 & 0.70 & 0.49 \\
0.49 & 0.70 & 1.00 & 0.70 \\
0.34 & 0.49 & 0.70 & 1.00
\end{pmatrix}
\]

whose the normalized eigenvector associated with the largest eigenvalue is

\[
w^* = [0.4615 \ 0.5357 \ 0.5357 \ 0.4615]^T.
\]

We set \(k = 2\) and \(\mu = 0.001\). For comparison we plot the norm squared parameter error, \(||w_i - w^*||^2\). Note that the normalization by norm for plotting is just for fair comparison, but during the adaptation, such normalization is never used in the proposed scheme in Eq. (7). Each plot is averaged over 100 independent trials. As shown in Fig. 2, the proposed adaptive algorithm converges as good as the conventional one while being computationally simpler.

#### 3.2 Independent Component Analysis (ICA)

ICA is a recently developed method whose goal is to find a linear representation of non-Gaussian data so that the components are statistically independent. The main applications of ICA are in blind source separation, feature extraction, and blind deconvolution [6]–[8].

Let us denote the observed vector and the independent component vector as \(x \in \mathbb{R}^K\) and \(s \in \mathbb{R}^N\), respectively. Then the linear relationship is given by

\[
x = As,
\]

(9)

where \(A\) is an unknown \(K \times N\) matrix, called the mixing matrix. The basic problem of ICA is to estimate the realizations...
of the original independent vector \( s \) using only observations of the vector \( x \). Here we derive a new ICA algorithm based on single parameter normalization and compare the behaviors of the algorithm with conventional one. The ICA aims at finding a vector \( w \) such that the variance of \( y^2 \) is maximized, i.e., the cost function in (1) and \( e(i) \) in (2) become

\[
J(w) = E[(w^T x_i)^4]
\]

subject to \( ||w|| = 1 \) and \( e(i) = w^T x_i = y(i) \), respectively and thus the optimum is

\[
w^o = \arg \max E||y(i)||^4.
\]

A conventional adaptive algorithm is given by

\[
y(i) = w_{i-1}^T x_i,
\]

\[
\hat{w}_i = w_{i-1} - \mu |y(i)|^2 \cdot y(i) \cdot x_i, \quad \text{and}
\]

\[
w_i = \frac{\hat{w}_i}{||\hat{w}_i||}.
\]

The proposed adaptive algorithm for ICA is

\[
y(i) = w_{i-1}^T x_i,
\]

\[
\hat{w}_i = w_{i-1} - \mu |y(i)|^2 \cdot y(i) \cdot x_i, \quad \text{and}
\]

\[
w_i = \frac{\hat{w}_i}{||\hat{w}_i||}.
\]

The simulation model is similar with that in [5]. We generate \( s_i = [s_1(i) \ s_2(i) \ s_3(i)]^T \) where \( s_1(i) \) is an i.i.d. binary distributed signal and \( s_2(i) \) and \( s_3(i) \) are i.i.d. Laplacian distributed signals with p.d.f. \( P(s) = e^{-|s|} / \sqrt{2} \). The mixing matrix \( A \) is set to be the covariance matrix of \( x \)

\[
R_{xx} = \begin{pmatrix} 0.9 & 0.4 & 0.7 \\ 0.4 & 0.3 & 0.5 \\ 0.7 & 0.5 & 1.0 \end{pmatrix}.
\]

We want to estimate the binary source \( s_1(i) \) from the mixed component \( x_i \). We set \( k = 3 \) and \( \mu = 0.001 \). For the comparison we plot the norm squared error, \( ||s_1(i) - y(i)||^2 \) in Fig. 3. Each plot is averaged over 100 independent trials. As expected, the proposed adaptive algorithm shows a good performance with lower computational complexity.

4. Conclusions

We have proposed a simple adaptive algorithm for the unit-norm constrained optimization. For PCA and ICA applications, the proposed single parameter normalization works as good as the conventional norm-based normalization while being computationally simpler. Although only two applications have been illustrated here, we believe the proposed method may trigger other applications based on the unit-norm constrained optimization.

The performance of the proposed scheme may depend on the choice of normalization parameter \( k \). More importantly for stability, we need to avoid normalization by zero or very small value. So at some intervals we recommend to change normalization parameter, \( k \) whose absolute value is maximum. The issues on optimum \( k \) and stable normalization remain for further issue.

References