A Simple and Accurate ICA Algorithm for Separating Mixtures of Up to Four Independent Components

TANG Ying[1] LI Jian-Ping[1] WU Huai[1]

Abstract This paper introduces an algorithm for independent component analysis (ICA) using explicit closed forms of two-, three- and four-dimensional antisymmetric matrix exponentials, based on which both the search direction and matrix exponentials can be directly computed in each iteration without any approximation. In addition, two errors have been corrected for the representation of four-dimensional antisymmetric matrix exponentials that were established in other works. Simulations show that the algorithm converges fast and can achieve better performance than the well-known Extended InfoMax and FastICA algorithms for mixtures of up to four independent components.

Key words Independent component analysis (ICA), matrix exponential, closed form, antisymmetric matrix

DOI 10.3724/SP.J.1004.2011.00794

The simplest problem setting of independent component analysis (ICA) involves recovering the independent source signals $\boldsymbol{s}(t) = [s_1(t), \cdots, s_n(t)]^{\mathrm{T}}$ from the observed data vector $\boldsymbol{x}(t) = [x_1(t), \cdots, x_n(t)]^{\mathrm{T}} = A\boldsymbol{s}(t)$, where A is a mixing matrix with full rank[1−2]. The solution of the ICA problem is to find a matrix B such that the estimate $\boldsymbol{y}(t) = B\boldsymbol{x}(t) = BA\boldsymbol{s}(t)$ is equal to $\boldsymbol{s}(t)$ within scale changes and permutations.

It is typical to tackle the ICA problem in two stages. First, the observation vector $\boldsymbol{x}(t)$ is whitened to get $\boldsymbol{z}(t) = W\boldsymbol{x}(t)$ by using a principal component analysis algorithm[3], where W is the whitening matrix. Then, based on the InfoMax principle[4] the ICA problem is reduced to find an orthogonal matrix U to maximize the following cost function:

$$\begin{aligned} &\phi = E\left\{\sum_{i=1}^{n} \log g_i'(y_i)\right\} \\ \text{s.t.} \quad &\boldsymbol{y} = U\boldsymbol{z} = [y_1, \cdots, y_n]^{\mathrm{T}} \end{aligned} \tag{1}$$

where $E(\cdot)$ is the expectation operator and $g_i(\cdot)$ is a squashing function. The selection of $g_i(\cdot)$ in (1) plays an important role in the ICA problem. Ideally, $g_i(\cdot)$ may be selected as the cumulative distribution function (CDF) of s_i. Practical choices of the squashing functions are discussed in Section 2.

Let the set of n-by-n orthogonal matrices be $O(n)$, and the subset of $O(n)$ be $SO(n)$ in which the determinant of the matrix is 1. Clearly, it is sufficient to search for U in $SO(n)$ because we can only find U with ambiguities in permutations. Thus the problem in (1) can be rewritten as

$$\begin{aligned} &\max\ \phi = E\left\{\sum_{i=1}^{n} \log g_i'(y_i)\right\} \\ \text{s.t.} \quad &\boldsymbol{y} = U\boldsymbol{z} = [y_1, \cdots, y_n]^{\mathrm{T}}, \quad U \in SO(n) \end{aligned} \tag{2}$$

Denote by $so(n)$ the set of n-by-n antisymmetric matrices. Then, the following lemma is proved in the Appendix.

Lemma 1. If and only if there is a matrix $C \in so(n)$ satisfying $\mathrm{e}^C = U$, then there exists $U \in SO(n)$.

Based on the above lemma, (2) can be reformulated as

$$\begin{aligned} &\max\ \phi = E\left\{\sum_{i=1}^{n} \log g_i'(y_i)\right\} \\ \text{s.t.} \quad &\boldsymbol{y} = \mathrm{e}^C\boldsymbol{z} = [y_1, \cdots, y_n]^{\mathrm{T}}, \quad C \in so(n) \end{aligned} \tag{3}$$

The above formulation has the advantage that the antisymmetric constraints can be easily implemented although matrix exponentials are required. However, most methods for matrix exponentials are approximate ones whose closed-form expressions are rarely available[5].

This paper addresses the separation of low-order mixtures using explicit closed forms of exponentials of antisymmetric matrices up to order 4. Unlike those geodesic-based ICA methods involving matrix exponentials[6−10], the method of this paper is based on simple matrix analysis and does not require complex mathematical concepts.

1 Closed forms of low-order antisymmetric matrix exponentials

For any square matrix P of order n, the matrix exponential e^P is defined as $\mathrm{e}^P = I_n + \sum_{k=1}^{\infty} \frac{P^k}{k!}$, where I_n is the identity matrix of order n. To the best of our knowledge, in general there are no explicit closed forms for $\mathrm{e}^C, C \in so(n)$, except for $n = 2$, 3, 4. In what follows, $c_1, \cdots, c_6$, are six real numbers used to construct the antisymmetric matrix C of orders 2, 3 and 4.

In the case of $n = 2$, we have the following simple formula:

$$C = \begin{bmatrix} 0 & c_1 \\ -c_1 & 0 \end{bmatrix} \quad \Rightarrow \quad \mathrm{e}^C = \begin{bmatrix} \cos c_1 & \sin c_1 \\ -\sin c_1 & \cos c_1 \end{bmatrix} \tag{4}$$

In the case of $n = 3$, we have the following formula due to Rodrigues[11−12]:

$$C = \begin{bmatrix} 0 & c_3 & c_2 \\ -c_3 & 0 & c_1 \\ -c_2 & -c_1 & 0 \end{bmatrix} \Rightarrow \mathrm{e}^C = I_n + \frac{\sin\beta}{\beta} C + \frac{1-\cos\beta}{\beta^2} C^2 \tag{5}$$

where $\beta = \sqrt{c_1^2 + c_2^2 + c_3^2}$.

The following method for evaluating $\mathrm{e}^C, C \in so\,(4)$, was obtained in [13]. Let

$$C = \begin{bmatrix} 0 & c_6 & c_5 & c_3 \\ -c_6 & 0 & c_4 & c_2 \\ -c_5 & -c_4 & 0 & c_1 \\ -c_3 & -c_2 & -c_1 & 0 \end{bmatrix} \tag{6}$$

Manuscript received October 16, 2009; accepted February 22, 2011
Supported by National Natural Science Foundation of China (61003121), Multimodel Biometric Encryption and Its Implementation (January 2011∼December 2013)
1. School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, P.R. China

and let

$$s = (c_1c_6 + c_3c_4 - c_2c_5)^2, \qquad t = \sum_{i=1}^{6} c_i^2 \tag{7}$$

Clearly, there exists $t^2 - 4s \geq 0$. Define

$$\begin{aligned} \gamma &= \sqrt{\frac{t - \sqrt{t^2 - 4s}}{2}}, \qquad \mu = \sqrt{\frac{t + \sqrt{t^2 - 4s}}{2}} \\ \alpha_1 &= \frac{\gamma \sin\mu - \mu \sin\gamma}{\gamma\mu(\gamma^2 - \mu^2)}, \qquad \alpha_2 = \frac{\cos\mu - \cos\gamma}{\gamma^2 - \mu^2} \\ \alpha_3 &= \frac{\sin\gamma}{\gamma} + \alpha_1\gamma^2, \qquad \alpha_4 = \cos\gamma + \alpha_2\gamma^2 \end{aligned} \tag{8}$$

If $t^2 - 4s = 0$, i.e.,

$$c_1 = c_6, \quad c_2 = -c_5, \quad c_3 = c_4$$

or

$$c_1 = -c_6, \quad c_2 = c_5, \quad c_3 = -c_4 \tag{9}$$

we have

$$\mathrm{e}^C = \frac{\sin\nu}{\nu} C + \cos\nu I_4, \quad \nu = \sqrt{\frac{t}{2}} \tag{10}$$

If $t^2 - 4s > 0$, we have

$$\mathrm{e}^C = \alpha_1 C^3 + \alpha_2 C^2 + \alpha_3 C + \alpha_4 I_4 \tag{11}$$

The above results are the same as those in [13] except for two typographical errors in the expressions of α_1 and α_3.

2 Computation procedure for low-dimensional ICA

We use stochastic gradient adaptation to solve (3) with the expectation operator replaced by an estimate obtained from a single-sample realization. In what follows, the derivations assume single-sample estimates. The extensions to the case of block-based adaptation are straightforward, and the experimental results presented in the next section are based on updating the parameters every 100 samples.

This is an unconstrained optimization problem involving $\frac{n(n-1)}{2}$ parameters of the antisymmetric matrix C. For simplicity, let $\boldsymbol{c}(k) = [c_1(k), \cdots, c_m(k)]^{\mathrm{T}}, m = \frac{n(n-1)}{2}$, denote the vector of the estimated parameters after the kth iteration. We consider the following iterative algorithm to update $\boldsymbol{c}(k)$:

$$\boldsymbol{c}(k+1) = \boldsymbol{c}(k) + \mu(k) \cdot \boldsymbol{\nabla\phi}(k), \quad k = 0, 1, 2, \cdots \tag{12}$$

where $\mu(k)$ is the step size at the kth iteration and $\boldsymbol{\nabla\phi}(k)$ is the gradient of ϕ with respect to $\boldsymbol{c}(k)$:

$$\boldsymbol{\nabla\phi}(k) = \left[\frac{\partial\phi(k)}{\partial c_1(k)}, \cdots, \frac{\partial\phi(k)}{\partial c_m(k)}\right]^{\mathrm{T}} \tag{13}$$

As defined in (3), $\phi(k) = \Sigma_{i=1}^{n} \log g_i'(y_i(k))$, where $y_i(k)$ is the ith element of $\boldsymbol{y}(k) = \mathrm{e}^{C(k)}\boldsymbol{z}$ and $C(k)$ is the antisymmetric matrix generated by vector $\boldsymbol{c}(k)$.

Getting precise representations of $\boldsymbol{\nabla\phi}$ for $n = 2, 3, 4$, is the main task of this section. The calculation depends on the choice of $g_i(\cdot)$ in (3). It is known that the local convergence can be ensured if $g_i(\cdot)$ is selected to be the CDF of sources[14]. Several practical choices are discussed in [4, 14 ∼ 16]. The following form for $\frac{g''(y_i(k))}{g'(y_i(k))}$ was proposed in [15] for sources with both sub- and super-Gaussian distributions[1]:

$$\frac{g''(y_i(k))}{g'(y_i(k))} = -y_i(k) - \mathrm{sign}(\kappa_i(k)) \tanh y_i(k) \tag{14}$$

where $\kappa_i(k)$ takes the following form[16]:

$$\kappa_i(k) = E\{\mathrm{sech}^2 y_i(k) \cdot y_i^2(k) - y_i(k) \cdot \tanh y_i(k)\} \tag{15}$$

Based on (3), we have

$$\boldsymbol{\nabla\phi}(k) = [\boldsymbol{\nabla y_1}(k), \cdots, \boldsymbol{\nabla y_n}(k)] \left[\frac{g''(y_1(k))}{g'(y_1(k))}, \cdots, \frac{g''(y_n(k))}{g'(y_n(k))}\right]^{\mathrm{T}} \tag{16}$$

where

$$\boldsymbol{\nabla y_i}(k) = \left[\frac{\partial y_i(k)}{\partial c_1(k)}, \cdots, \frac{\partial y_i(k)}{\partial c_m(k)}\right]^{\mathrm{T}} \tag{17}$$

for $i = 1, \cdots, n, \quad m = \frac{n(n-1)}{2}$.

In what follows, we will develop explicit expressions for $\boldsymbol{\nabla y_i}(k), i = 1, \cdots, n$, for $n = 2, 3, 4$. Throughout the following derivations, vector $\boldsymbol{z} = [z_1, \cdots, z_n]^{\mathrm{T}}$ denotes whitened signals.

2.1 Case of $n = 2$

This is a trivial case. From (4), we have

$$\begin{aligned} \nabla y_1(k) &= -\sin c_1(k) \cdot z_1 + \cos c_1(k) \cdot z_2 \\ \nabla y_2(k) &= -\cos c_1(k) \cdot z_1 - \sin c_1(k) \cdot z_2 \end{aligned} \tag{18}$$

2.2 Case of $n = 3$

For simplicity, let $\boldsymbol{d}(k) = C(k)\boldsymbol{z} = [d_1(k), d_2(k), d_3(k)]^{\mathrm{T}}$ and $\boldsymbol{f}(k) = C^2(k)\boldsymbol{z} = [f_1(k), f_2(k), f_3(k)]^{\mathrm{T}}$, where $C(k)$ is generated by $\boldsymbol{c}(k)$ using (5). Let $\boldsymbol{\nabla d_i}(k) = [\frac{\partial d_i(k)}{\partial c_1(k)}, \cdots, \frac{\partial d_i(k)}{\partial c_3(k)}]^{\mathrm{T}}$ and $\boldsymbol{\nabla f_i}(k) = [\frac{\partial f_i(k)}{\partial c_1(k)}, \cdots, \frac{\partial f_i(k)}{\partial c_3(k)}]^{\mathrm{T}}$ for $i = 1, 2, 3$.

By direct computation, it is easy to get

$$\begin{aligned} \boldsymbol{\nabla d_1}(k) &= [\ 0 \quad z_3 \quad z_2\ ]^{\mathrm{T}} \\ \boldsymbol{\nabla d_2}(k) &= [\ z_3 \quad 0 \quad -z_1\ ]^{\mathrm{T}} \\ \boldsymbol{\nabla d_3}(k) &= [\ -z_2 \quad -z_1 \quad 0\ ]^{\mathrm{T}} \end{aligned} \tag{19}$$

and

$$\begin{aligned} \boldsymbol{\nabla f_1}(k) &= \begin{bmatrix} 0 & -c_2(k) & c_3(k) \\ -2c_2(k) & -c_1(k) & 0 \\ -2c_3(k) & 0 & c_1(k) \end{bmatrix} \boldsymbol{z} \\ \boldsymbol{\nabla f_2}(k) &= -\begin{bmatrix} c_2(k) & 2c_1(k) & 0 \\ c_1(k) & 0 & c_3(k) \\ 0 & 2c_3(k) & c_2(k) \end{bmatrix} \boldsymbol{z} \\ \boldsymbol{\nabla f_3}(k) &= \begin{bmatrix} c_3(k) & 0 & -2c_1(k) \\ 0 & -c_3(k) & -2c_2(k) \\ c_1(k) & -c_2(k) & 0 \end{bmatrix} \boldsymbol{z} \end{aligned} \tag{20}$$

[1] $g'(y_i(k))$ and $g''(y_i(k))$ are the first- and second-order derivatives of g with respect to $y_i(k)$

We can now use the above results, the closed-form expression for e^C in (5) and $\boldsymbol{y}(k)=\mathrm{e}^{C(k)}\boldsymbol{z}$ to get

$$\frac{\partial y_i(k)}{\partial c_j(k)}=\frac{\sin\beta(k)}{\beta(k)}\frac{\partial d_i(k)}{\partial c_j(k)}+c_j(k)\frac{\beta(k)\cos\beta(k)-\sin\beta(k)}{\beta^3(k)}d_i(k)+\frac{1-\cos\beta(k)}{\beta^2(k)}\frac{\partial f_i(k)}{\partial c_j(k)}+c_j(k)\frac{\beta(k)\sin\beta(k)-2(1-\cos\beta(k))}{\beta^4(k)}f_i(k) \tag{21}$$

where $i,j=1,2,3$ and $\beta(k)=\sqrt{c_1^2(k)+c_2^2(k)+c_3^2(k)}$. Equation (21) can be rewritten using the following vector form:

$$\nabla\boldsymbol{y_i}(k)=\left[\begin{array}{c}c_1(k)\\c_2(k)\\c_3(k)\end{array}\right]\left[\frac{\beta(k)\sin\beta(k)-2(1-\cos\beta(k))}{\beta^4(k)}f_i(k)+\frac{\beta(k)\cos\beta(k)-\sin\beta(k)}{\beta^3(k)}d_i(k)\right]+\frac{1-\cos\beta(k)}{\beta^2(k)}\nabla\boldsymbol{f_i}(k)+\frac{\sin\beta(k)}{\beta(k)}\nabla\boldsymbol{d_i}(k)$$

2.3 Case of $n=4$

Let $\boldsymbol{p}(k)=C(k)\boldsymbol{z}=[p_1(k),\cdots,p_4(k)]^{\mathrm{T}}$, $\boldsymbol{q}(k)=C^2(k)\boldsymbol{z}=[q_1(k),\cdots,q_4(k)]^{\mathrm{T}}$ and $\boldsymbol{h}(k)=C^3(k)\boldsymbol{z}=[h_1(k),\cdots,h_4(k)]^{\mathrm{T}}$, where $C(k)$ is created by $\boldsymbol{c}(k)$ using (6). Let $\nabla\boldsymbol{p_i}(k),\nabla\boldsymbol{q_i}(k)$, $\nabla\boldsymbol{h_i}(k)$, and $\nabla\boldsymbol{\alpha_i}(k)$ be the gradient vectors of $p_i(k),q_i(k)$, $h_i(k)$ and $\alpha_i(k)$, respectively, with respect to vector $\boldsymbol{c}(k)$. Here, $\alpha_i(k)$ is the value of α_i defined in (8) at the kth iteration.

If e^C takes the form as (11), we have

$$\begin{aligned}\nabla\boldsymbol{y_i}(k)=&h_i(k)\nabla\boldsymbol{\alpha_1}(k)+\alpha_1(k)\nabla\boldsymbol{h_i}(k)+q_i(k)\nabla\boldsymbol{\alpha_2}(k)+\\&\alpha_2(k)\nabla\boldsymbol{q_i}(k)+p_i(k)\nabla\boldsymbol{\alpha_3}(k)+\\&\alpha_3(k)\nabla\boldsymbol{p_i}(k)+z_i\nabla\boldsymbol{\alpha_4}(k)\end{aligned} \tag{22}$$

for $i=1,\cdots,4$. All the components of (23) are explicitly evaluated in the appendix.

If the condition (9) holds, e^C takes the form as (10) and it is straightforward to show that

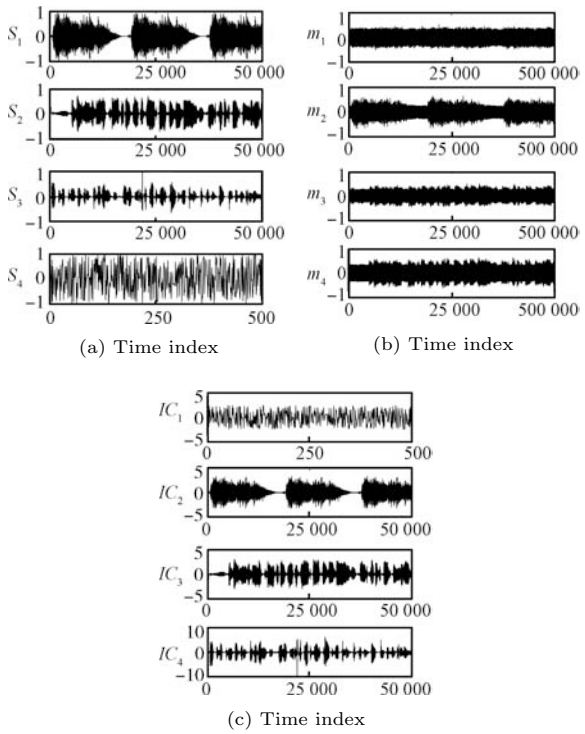
$$\nabla\boldsymbol{y_i}(k)=\frac{\nu(k)\cos\nu(k)-\sin\nu(k)}{\nu^2(k)}p_i(k)\nabla\boldsymbol{\nu}(k)+\frac{\sin\nu(k)}{\nu(k)}\nabla\boldsymbol{p_i}(k)-z_i\sin\nu(k)\nabla\boldsymbol{\nu}(k) \tag{23}$$

for $i=1,\cdots,4$. In the above equation, the gradient vector $\nabla\boldsymbol{p_i}(k)$ is computed in the appendix, $\nu(k)$ is the value of ν defined in (10) at the kth iteration and $\nabla\boldsymbol{\nu}(k)=[\frac{\partial\nu(k)}{\partial c_1(k)},\cdots,\frac{\partial\nu(k)}{\partial c_6(k)}]^{\mathrm{T}}=\frac{\sqrt{2}}{2\sqrt{t(k)}}[c_1(k),\cdots,c_6(k)]^{\mathrm{T}}$ where $t(k)$ is the value of t defined in (7) at the kth iteration.

The complete algorithm for four-dimensional whitened mixtures is summarized in Table 1.

3 Simulation results

In this section, we present the experimental results for the case of $n=4$. In this simulation, the first three source signals were super-Gaussian speech signals[2] and the fourth was a uniformly distributed sub-Gaussian noise sequence in the interval $[-1,1]$. The signals as well as their mixtures created by a random mixing matrix whose elements were uniformly distributed in the range $[-1,1]$ were 50 000 samples long. The algorithm of this paper was implemented in a block-adaptive manner, with each block containing 100 samples. All the elements of the initial vector $\boldsymbol{c}(0)$ were set to 0.01. The step size $\mu=0.02$ was used in the first 100 iterations. Subsequent to that, μ was set to 0.002. The iterations were terminated when $|\frac{\nabla\boldsymbol{\phi}(k)}{||\nabla\boldsymbol{\phi}(k)||}|<0.001$. If necessary, the algorithm would iterate on the mixture a second time after the 50 000 samples were exhausted. Five hundred independent experiments were conducted, in which the mixing matrix and the sub-Gaussian noise sequence changed with each run. Fig. 1 shows the results from a ty-



(a) Time index (b) Time index

(c) Time index

Fig. 1 (a) Source waveforms; (b) typical mixtures; and (c) separated signals using the proposed algorithm for a typical run

Table 1 Algorithm for the case of $n=4$

Input: The whitened signals $\boldsymbol{z}$, step size μ, block size l and stopping condition ε.
Initialization: Set $\boldsymbol{c}(0)$ to a nonzero vector. Let $k=0$.
*During the kth iteration:
Create $C(k)$ from $\boldsymbol{c}(k)$ using (6). Compute $\mathrm{e}^{C(k)}$ using (10) or (11);
$\boldsymbol{y}(kl+v)=\mathrm{e}^{C(k)}\boldsymbol{z}(kl+v),\quad v=0,\cdots,l-1;$
$\kappa_i=\frac{1}{l}\sum_{v=0}^{l-1}\mathrm{sech}^2 y_i(kl+v)\times y_i^2(kl+v)-y_i(kl+v)\times\tanh y_i(kl+v);$
$\frac{g''(y_i(kl+v))}{g'(y_i(kl+v))}=-y_i(kl+v)-\mathrm{sign}(\kappa_i(k))\tanh y_i(kl+v);$
Compute $\nabla\boldsymbol{y_i}(kl+v)$ using (23) or (24) for $i=1,\cdots,4;$
$\nabla\boldsymbol{\phi}(k)=\frac{1}{l}\sum_{v=0}^{l-1}\{[\nabla\boldsymbol{y_1}(kl+v),\cdots,\nabla\boldsymbol{y_4}(kl+v)]\times[\frac{g''(y_1(kl+v))}{g'(y_1(kl+v))},\cdots,\frac{g''(y_4(kl+v))}{g'(y_4(kl+v))}]^{\mathrm{T}}\};$
$\boldsymbol{c}(k+1)=\boldsymbol{c}(k)+\mu\cdot\nabla\boldsymbol{\phi}(k);$
If $\lvert\frac{\nabla\boldsymbol{\phi}(k)}{\lvert\lvert\nabla\boldsymbol{\phi}(k)\rvert\rvert}\rvert<\varepsilon$, end iteration, otherwise let $k=k+1$ and go to *;

[2] From http://www.cis.hut.fi/projects/ica/cocktail/cocktail_en.cgi

Table 2 The mean normalized kurtosis and steady-state PI of three algorithms over the convergent cases of 500 runs

	Source signals	Our algorithm	Extended InfoMax	FastICA
Sound 1	1.2684	1.2684	1.2685	1.2685
Sound 2	1.1934	1.1936	1.1953	1.1953
Sound 3	15.873	15.871	15.887	15.906
Noise	−1.2059	−1.2063	−1.2075	−1.2068
Steady-state PI		0.1118	0.1273	0.11468
Convergent cases		94 %	81 %	76 %

pical run based on the proposed algorithm. Only the first 500 samples of the noise sequence were plotted in the figure to clearly show its waveform. From Fig. 1, we can see that the propsoed algorithm achieved good separation in this example. Fig. 2 displays two learning curves based on our algorithm and the Extended InfoMax algorithm[16] obtained by averaging the following performance measure[1]:

$$PI(k) = \sum_{i=1}^{n}\left(\sum_{j=1}^{n}\frac{|t_{ij}(k)|}{\max_h|t_{ih}(k)|}-1\right)+ \sum_{j=1}^{n}\left(\sum_{i=1}^{n}\frac{|t_{ij}(k)|}{\max_h|t_{hj}(k)|}-1\right) \tag{24}$$

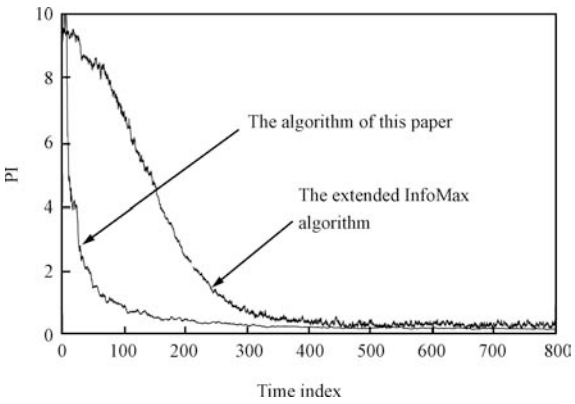


Fig. 2 Average learning curves of the proposed algorithm and the Extended InfoMax alorithm over the converged cases in 500 runs

over the converged cases of 500 runs. In (25), $t_{ij}(k)$ is the (i,j)-element of matrix $T(k) = \mathrm{e}^{C(k)}WA$. Clearly, the smaller such measure is, the better the performance of an algorithm is. From Fig. 2, we can see that the proposed algorithm converges significantly faster than the Extended InfoMax algorithm while performing at least as well as or better than the Extended InfoMax algorithm. Because the FastICA algorithm[17] extracts the source signals sequentially, it is impossible to plot a similar curve for that algorithm.

Comparisons of the three algorithms after convergence is provided in Table 2 in terms of the mean normalized kurtosis (the kurtosis of a signal s_i normalized to have unit variance, also known as the normalized kurtosis, is given by $E(s_i^4)-3$), and the steady-state PI. We make a few observations here. Possibly because of the multimodal nature of the performance surface, none of the algorithms converged in all runs (the algorithm was deemed to have converged if $PI(k)$ reached 0.2). As tabulated in Table 2, of the three algorithms, our method achieved satisfactory separation 94 % of the 500 runs while the other two algorithms converged only 81 % and 76 % of the 500 runs, indicating that the proposed approach may have superior convergence properties than the Extended InfoMax and FastICA algorithms. This may be because our method updates the parameters strictly in the smaller space $SO(n)$, instead of the space of general matrices, while the other two do not have such advantage. The results in Table 2 also indicate that the proposed algorithm exhibits comparable or slightly better separation capability than the other two algorithms.

4 Conclusions

This paper proposed an ICA algorithm for separating low-dimensional mixtures using explicit closed forms of two-, three- and four-dimensional antisymmetric matrix exponentials along with two corrections for the representation of four-dimensional antisymmetric matrix exponentials introduced in other papers. The resulting algorithm is computationally straightforward. Experimental results have indicated that the proposed algorithm exhibits faster convergence and better separation than the two competing algorithms from the literature. Consequently, we believe that this new approach is a viable alternative to the ICA algorithms available in the literature.

Appendix

1) Proof of Lemma 1:

If $C \in so(n)$, we have $\mathrm{e}^C\mathrm{e}^{C^{\mathrm{T}}} = \mathrm{e}^{C+C^{\mathrm{T}}} = I_n$ and $\det(\mathrm{e}^C) = \mathrm{e}^{\mathrm{trace}(C)} = 1$. Next, we show that the converse is also true.

Since U is an orthogonal matrix, there exists an orthogonal matrix H such that[18]

$$U = H\cdot\mathrm{diag}\{\underbrace{1,\cdots,1}_{r},\underbrace{-1,\cdots,-1}_{s},\underbrace{B_1,\cdots,B_t}_{t}\}\cdot H^{\mathrm{T}} \tag{25}$$

where

$$B_i = \begin{bmatrix}\cos\theta_i & \sin\theta_i\\ -\sin\theta_i & \cos\theta_i\end{bmatrix},\quad i=1,\cdots,t$$

Since $\det(U) = 1$, s in (26) must be even, i.e., $s = 2q$, $q = 0, 1, \cdots$. Let

$$F_\alpha = \begin{bmatrix}0 & \alpha\\ -\alpha & 0\end{bmatrix},\quad \alpha\in\mathbf{R}$$

It is straightforward to verify

$$\mathrm{e}^{F_\pi} = \begin{bmatrix}-1 & 0\\ 0 & -1\end{bmatrix},\ \mathrm{e}^{F_{\theta_i}} = \begin{bmatrix}\cos\theta_i & \sin\theta_i\\ -\sin\theta_i & \cos\theta_i\end{bmatrix}$$

for $i = 1,\cdots,t$. Then, we construct C as

$$C = H\cdot\mathrm{diag}\{\underbrace{0,\cdots,0}_{r},\underbrace{F_\pi,\cdots,F_\pi}_{q},\underbrace{F_{\theta_1},\cdots,F_{\theta_t}}_{t}\}\cdot H^{\mathrm{T}}$$

Using the definition of e^C and the orthogonality of H, we can write

$$\mathrm{e}^C = H \cdot \mathrm{diag}\{\underbrace{\mathrm{e}^0, \cdots, \mathrm{e}^0}_{r}, \underbrace{\mathrm{e}^{F_\pi}, \cdots, \mathrm{e}^{F_\pi}}_{q}, \underbrace{\mathrm{e}^{F_{\theta_1}}, \cdots, \mathrm{e}^{F_{\theta_t}}}_{t}\} \cdot H^{\mathrm{T}} = U$$

thus proving the lemma. □

2) Computation of the components in (23)

In what follows, we provide details of the computation of several variables in (23). The iteration number k is omitted in the following derivations for simplicity. Denote $\gamma_i = \frac{\partial \gamma}{\partial c_i}$, $\mu_i = \frac{\partial \mu}{\partial c_i}$, $\lambda_{ij} = c_i c_j$. Based on (7) and (8), we have

$$\begin{aligned} \gamma_i &= \frac{c_i\sqrt{t^2-4s} - tc_i + 2\eta(\lambda_{16}+\lambda_{34}-\lambda_{25})c_{7-i}}{2\gamma\sqrt{t^2-4s}} \\ \mu_i &= \frac{c_i\sqrt{t^2-4s} + tc_i - 2\eta(\lambda_{16}+\lambda_{34}-\lambda_{25})c_{7-i}}{2\mu\sqrt{t^2-4s}} \end{aligned} \tag{26}$$

where η satisfies

$$\eta = \begin{cases} 1, & \text{if } i = 1,3,4,6 \\ -1, & \text{if } i = 2,5 \end{cases} \tag{27}$$

Let $t_1, \cdots, t_6$, defined as follows:

$$\begin{aligned} t_1 &= \gamma\mu(\gamma^2-\mu^2), \\ t_2 &= (\gamma_i \sin\mu + \gamma\mu_i\cos\mu - \mu_i\sin\gamma - \mu\gamma_i\cos\gamma)t_1, \\ t_3 &= (\gamma\sin\mu - \mu\sin\gamma)(3\gamma^2\mu\gamma_i + \gamma^3\mu_i - \gamma_i\mu^3 - 3\gamma\mu^2\mu_i), \\ t_4 &= \gamma^2 - \mu^2, \\ t_5 &= (\gamma_i\sin\gamma - \mu_i\sin\mu)t_4, \\ t_6 &= 2(\cos\mu - \cos\gamma)(\gamma\gamma_i - \mu\mu_i). \end{aligned}$$

From (8), it is straightforward to verify that

$$\begin{aligned} \frac{\partial\alpha_1}{\partial c_i} &= \frac{t_2 - t_3}{t_1^2} \\ \frac{\partial\alpha_2}{\partial c_i} &= \frac{t_5 - t_6}{t_4^2} \\ \frac{\partial\alpha_3}{\partial c_i} &= \frac{\gamma\gamma_i\cos\gamma - \gamma_i\sin\gamma}{\gamma^2} + \frac{\partial\alpha_1}{\partial c_i}\gamma^2 + 2\alpha_1\gamma\gamma_i \\ \frac{\partial\alpha_4}{\partial c_i} &= -\gamma_i\sin\gamma + \frac{\partial\alpha_2}{\partial c_i}\gamma^2 + 2\alpha_2\gamma\gamma_i \end{aligned} \tag{28}$$

Direct evaluation using the definition of $\boldsymbol{\nabla h_i}, i = 1,2,3,4$, gives

$$\boldsymbol{\nabla h_1} = \begin{bmatrix} 0 & \lambda_{34}-\lambda_{25} & -2\lambda_{15} & \lambda_{46}-2\lambda_{13} \\ 0 & -2\lambda_{26} & -\lambda_{34}-\lambda_{16} & -\lambda_{45}-2\lambda_{23} \\ 0 & \lambda_{14}-2\lambda_{36} & -2\lambda_{35}-\lambda_{24} & \lambda_{44}-2\lambda_{33}-t \\ 0 & -2\lambda_{46} & -2\lambda_{45} & \lambda_{16}-\lambda_{25} \\ 0 & -2\lambda_{56}-\lambda_{12} & \lambda_{22}-2\lambda_{55}-t & -2\lambda_{35}-\lambda_{24} \\ 0 & \lambda_{11}-2\lambda_{66}-t & -2\lambda_{56}-\lambda_{12} & \lambda_{14}-2\lambda_{36} \end{bmatrix} \boldsymbol{z}$$

$$\boldsymbol{\nabla h_2} = \begin{bmatrix} \lambda_{25}-\lambda_{34} & 0 & \lambda_{36}-2\lambda_{14} & -\lambda_{56}-2\lambda_{12} \\ 2\lambda_{26}+\lambda_{15} & 0 & -\lambda_{35}-2\lambda_{24} & \lambda_{55}-2\lambda_{22}-t \\ 2\lambda_{36} & 0 & \lambda_{16}-\lambda_{25} & -\lambda_{45}-2\lambda_{23} \\ 2\lambda_{46}-\lambda_{13} & 0 & \lambda_{33}-2\lambda_{44}-t & -\lambda_{35}-2\lambda_{24} \\ 2\lambda_{56} & 0 & -2\lambda_{45} & -\lambda_{34}-\lambda_{16} \\ t-\lambda_{11}+2\lambda_{66} & 0 & \lambda_{13}-2\lambda_{46} & -2\lambda_{26}-\lambda_{15} \end{bmatrix} \boldsymbol{z}$$

$$\boldsymbol{\nabla h_3} = \begin{bmatrix} \lambda_{26}+2\lambda_{15} & 2\lambda_{14}-\lambda_{36} & 0 & \lambda_{66}-2\lambda_{11}-t \\ \lambda_{16}+\lambda_{34} & \lambda_{35}+2\lambda_{24} & 0 & -\lambda_{56}-2\lambda_{12} \\ 2\lambda_{35}+\lambda_{24} & \lambda_{25}-\lambda_{16} & 0 & \lambda_{46}-2\lambda_{13} \\ 2\lambda_{45}+\lambda_{23} & t-\lambda_{33}+2\lambda_{44} & 0 & \lambda_{36}-2\lambda_{14} \\ t-\lambda_{22}+2\lambda_{55} & 2\lambda_{45}+\lambda_{23} & 0 & -\lambda_{26}-2\lambda_{15} \\ 2\lambda_{56} & 2\lambda_{46}-\lambda_{13} & 0 & \lambda_{34}-\lambda_{25} \end{bmatrix} \boldsymbol{z}$$

$$\boldsymbol{\nabla h_4} = \begin{bmatrix} 2\lambda_{13}-\lambda_{46} & \lambda_{56}+2\lambda_{12} & t-\lambda_{66}+2\lambda_{11} & 0 \\ \lambda_{45}+2\lambda_{23} & t-\lambda_{55}+2\lambda_{22} & \lambda_{56}+2\lambda_{12} & 0 \\ t-\lambda_{44}+2\lambda_{33} & \lambda_{45}+2\lambda_{23} & 2\lambda_{13}-\lambda_{46} & 0 \\ \lambda_{25}-\lambda_{16} & \lambda_{35}+2\lambda_{24} & 2\lambda_{14}-\lambda_{36} & 0 \\ 2\lambda_{35}+\lambda_{24} & \lambda_{16}+\lambda_{34} & \lambda_{26}+2\lambda_{15} & 0 \\ 2\lambda_{36}-\lambda_{14} & 2\lambda_{26}+\lambda_{15} & \lambda_{25}-\lambda_{34} & 0 \end{bmatrix} \boldsymbol{z} \tag{29}$$

where t is defined in (7). Similarly, there exists

$$\boldsymbol{\nabla q_1} = \begin{bmatrix} 0 & 0 & -c_3 & c_5 \\ 0 & -c_3 & 0 & c_6 \\ -2c_3 & -c_2 & -c_1 & 0 \\ 0 & -c_5 & c_6 & 0 \\ -2c_5 & -c_4 & 0 & c_1 \\ -2c_6 & 0 & c_4 & c_2 \end{bmatrix} \boldsymbol{z}, \boldsymbol{\nabla q_2} = \begin{bmatrix} 0 & 0 & -c_2 & c_4 \\ -c_3 & -2c_2 & -c_1 & 0 \\ -c_2 & 0 & 0 & -c_6 \\ -c_5 & -2c_4 & 0 & c_1 \\ -c_4 & 0 & -c_6 & 0 \\ 0 & -2c_6 & -c_5 & -c_3 \end{bmatrix} \boldsymbol{z}$$

$$\boldsymbol{\nabla q_3} = \begin{bmatrix} -c_3 & -c_2 & -2c_1 & 0 \\ 0 & -c_1 & 0 & -c_4 \\ -c_1 & 0 & 0 & -c_5 \\ c_6 & 0 & -2c_4 & -c_2 \\ 0 & -c_6 & -2c_5 & -c_3 \\ c_4 & -c_5 & 0 & 0 \end{bmatrix} \boldsymbol{z}, \ \boldsymbol{\nabla q_4} = \begin{bmatrix} c_5 & c_4 & 0 & -2c_1 \\ c_6 & 0 & -c_4 & -2c_2 \\ 0 & -c_6 & -c_5 & -2c_3 \\ 0 & c_1 & -c_2 & 0 \\ c_1 & 0 & -c_3 & 0 \\ c_2 & -c_3 & 0 & 0 \end{bmatrix} \boldsymbol{z} \tag{30}$$

$$\begin{aligned} \boldsymbol{\nabla p_1} &= [\ 0 \quad 0 \quad z_4 \quad 0 \quad z_3 \quad z_2\ ]^{\mathrm{T}} \\ \boldsymbol{\nabla p_2} &= [\ 0 \quad z_4 \quad 0 \quad z_3 \quad 0 \quad -z_1\ ]^{\mathrm{T}} \\ \boldsymbol{\nabla p_3} &= [\ z_4 \quad 0 \quad 0 \quad -z_2 \quad -z_1 \quad 0\ ]^{\mathrm{T}} \\ \boldsymbol{\nabla p_4} &= [\ -z_3 \quad -z_2 \quad -z_1 \quad 0 \quad 0 \quad 0\ ]^{\mathrm{T}} \end{aligned} \tag{31}$$

References

1 Choi S, Cichocki S, Park H M, Lee S Y. Blind source separation and independent component analysis: a review. *Neural Information Processing-Letters and Reviews*, 2005, **6**(1): 1−57

2 Xiao Ming, Xie Sheng-Li, Fu Yu-Li. Underdetermined blind source separation algorithm based on normal vector of hyperplane. *Acta Automatica Sinica*, 2008, **34**(2): 142−149 (in Chinese)

3 Tang Ying, Li Jian-Ping. A new algorithm of ICA: using the parameterized orthogonal matrixes of any dimensions. *Acta Automatica Sinica*, 2008, **34**(1): 31−39 (in Chinese)

4 Bell A J, Sejnowski T J. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 1995, **7**(6): 1129−1159

5 Ashi H A, Cummings L J, Matthews P C. Comparison of methods for evaluating functions of a matrix exponential. *Applied Numerical Mathematics*, 2009, **59**(3−4): 468−486

6 Abrudan T E, Eriksson J, Koivunen V. Steepest descent algorithms for optimization under unitary matrix constraint. *IEEE Transactions on Signal Processing*, 2008, **56**(3): 1134−1147

7 Fiori S. Quasi-geodesic neural learning algorithms over the orthogonal group: a tutorial. *Journal of Machine Learning Research*, 2005, **6**: 743−781

8 Fiori S, Tanaka T. An algorithm to compute averages on matrix lie groups. *IEEE Transactions on Signal Processing*, 2009, **57**(12): 4734−4743

9 Nishimori Y, Akaho S. Learning algorithms utilizing quasi-geodesic flows on the stiefel manifold. *Neurocomputing*, 2005, **67**: 106−135

10 Plumbley M D. Algorithms for nonnegative independent component analysis. *IEEE Transactions on Neural Networks*, 2003, **14**(3): 534−543

11 Gallier J, Xu D. Computing exponentials of skew-symmetric matrices and logarithms of orthogonal matrices. *International Journal of Robotics and Automation*, 2002, **17**(4): 10−20

12 Moakher M. Means and averaging in the group of rotations. *SIAM Journal on Matrix Analysis and Applications*, 2002, **24**(1): 1−16

13 Politi T. A formula for the exponential of a real skew-symmetric matrix of order 4. *Bit Numerical Mathematics*, 2001, **41**(4): 842−845

14 Pham D T, Garrat P. Blind separation of mixture of independent sources through a quasi-maximum likelihood approach. *IEEE Transactions on Signal Processing*, 1997, **45**(7): 1712−1725

15 Xu L. One-bit-matching theorem for ICA, convex-concave programming on polyhedral set, and distribution approximation for combinatorics. *Neural Computation*, 2007, **19**(2): 546−569

16 Lee T W, Girolami M, Sejnowski T J. Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources. *Neural Computation*, 1999, **11**(2): 417−441

17 Shen H, Kleinsteuber M, Huper K. Local convergence analysis of fastICA and related algorithms. *IEEE Transactions on Neural Networks*, 2008, **19**(6): 1022−1032

18 Gloub G H, Loan C F V. *Matrix Computation (Third Edition)*. Baltimore: The John Hopkins University Press, 1996. 341−342

TANG Ying Ph. D. candidate at the School of Computer Science and Engineering, University of Electronic Science and Technology of China. He received his bachelor degree from the School of Automation, East China University of Science and Technology in 2002 and his master degree form the School of Automation, University of Electronic Science and Technology of China in 2006. His research interest covers geometric methods and nonlinear dynamics in machine learning, signal processing, and neural network. Corresponding author of this paper.
E-mail: mathtygo@yahoo.com

LI Jian-Ping Ph. D., professor at the International Centre for Wavelet Analysis and Its Applications, the School of Computer Science and Engineering, University of Electronic Science and Technology of China. His research interest covers wavelets analysis and information security.
E-mail: jpli2222@uestc.edu.cn

WU Huai Ph. D. candidate at the School of Computer Science and Engineering, University of Electronic Science and Technology of China. She received her bachelor and master degrees form the University of Electronic Science and Technology of China in 2004 and 2007, respectively. Her research interest covers cryptography and wireless network.
E-mail: daisylilac@126.com