

Optimization using Fourier Expansion over a Geodesic for Non-Negative ICA

Mark D. Plumbley

Department of Electronic Engineering, Queen Mary, University of London, Mile End Road, London E1 4NS, United Kingdom. Email: mark.plumbley@elec.qmul.ac.uk

Abstract. We propose a new algorithm for the non-negative ICA problem, based on the rotational nature of optimization over a set of square orthogonal (orthonormal) matrices \mathbf{W} , i.e. where $\mathbf{W}^T \mathbf{W} = \mathbf{W} \mathbf{W}^T = \mathbf{I}_n$. Using a truncated Fourier expansion of $J(t)$, we obtain a Newton-like update step along the steepest-descent geodesic, which automatically approximates to a usual (Taylor expansion) Newton update step near to a minimum. Experiments confirm that this algorithm is effective, and it compares favourably with existing non-negative ICA algorithms. We suggest that this approach could be modified for other algorithms, such as the normal ICA task.

1 Introduction

The task of non-negative independent component analysis (*non-negative ICA*) is to estimate the source vectors $\mathbf{s} = (s_1, \dots, s_n)$ and mixing matrix \mathbf{A} in the linear generative model $\mathbf{x} = \mathbf{A}\mathbf{s}$ given a set of observation vectors $\mathbf{x} = (x_1, \dots, x_n)$, where the sources are *non-negative*, i.e. $\Pr(s_i < 0) = 0$, and *independent*, i.e. $p(s_i s_j) = p(s_i)p(s_j)$ if $i \neq j$. We can also write this in matrix form as $\mathbf{X} = \mathbf{A}\mathbf{S}$ where each column of \mathbf{X} and \mathbf{S} represent a sample of \mathbf{x} and \mathbf{s} respectively.

There are two particular reasons why the non-negative ICA problem is interesting. Firstly, many real-world problems such as the analysis of images, text or musical signals, contain mixtures of sources which are non-negative [1]. Secondly, the non-negativity constraint introduces new approaches which are not available to the more general ICA problem [2–4]. Specifically, in previous work, we showed that for sources for which $\Pr(s < \delta) > 0$ for any $\delta > 0$, which we term *well-grounded*, the sources will be identified by finding a rotation of pre-whitened observations which is non-negative [5]. We also introduced a number of algorithms to perform this rotation [6, 7]. Some of these algorithms use the concept of a *geodesic search*, analogous to a line search, but on the manifold of orthogonal rotation matrices (see e.g. [8–12]). We previously used the tangent gradient at a point to determine the geodesic direction, and then perform a line search along that geodesic [6].

In this paper we explore the use of second order information to assist this line search, deriving a convenient form for the second derivative of mean squared non-negative reconstruction error on the geodesic.

Then, since we are on a rotation-like geodesic, we propose the use of a first order Fourier expansion around the optimum point to find the zero derivative point along the line, rather than using the usual second order Taylor expansion, leading to the normal Newton method. This will allow us to take large steps towards the bottom of the solution, even if we are near to a ‘peak’ where the Newton method would converge to the maximum instead of a minimum.

2 Non-negative ICA System

The non-negative ICA system we consider is similar to that in [6]. Given a sequence of observed n -dimensional data vectors \mathbf{x} , we first carry out a pre-whitening step [13], although being careful not to zero-mean the data in the process, since this would lose any information about the non-negativity of the sources [5]. Let $\Sigma_{\mathbf{x}} \equiv E((\mathbf{x} - \mu_{\mathbf{x}})(\mathbf{x} - \mu_{\mathbf{x}})^T)$, where $\mu_{\mathbf{x}} = E(\mathbf{x})$ is the mean of \mathbf{x} . We form the eigenvector-eigenvalue decomposition $\Sigma_{\mathbf{x}} = \mathbf{E}\mathbf{D}\mathbf{E}^T$ where $\mathbf{D} = \text{diag}(d_1, \dots, d_n)$ is a diagonal matrix containing the eigenvalues of $\Sigma_{\mathbf{x}}$, and $\mathbf{E} = (\mathbf{e}_1, \dots, \mathbf{e}_n)$ is a square orthonormal matrix whose columns are the corresponding eigenvectors. Then the pre-whitened data is given by the sequence of vectors $\mathbf{z} = \mathbf{V}\mathbf{x}$ where $\mathbf{V} = \mathbf{M}\mathbf{D}^{-1/2}\mathbf{E}^T$ for some square orthonormal matrix \mathbf{M} . For example, we can choose $\mathbf{M} = \mathbf{I}_n$ so we have simply $\mathbf{V} = \mathbf{D}^{-1/2}\mathbf{E}^T$. It is easy to verify that $\Sigma_{\mathbf{z}} \equiv E((\mathbf{x} - \bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})^T) = \mathbf{I}_n$

Given an $n \times n$ orthonormal weight matrix \mathbf{W} , i.e. $\mathbf{W}^T\mathbf{W} = \mathbf{W}\mathbf{W}^T = \mathbf{I}_n$, we calculate $\mathbf{y} = \mathbf{W}\mathbf{x}$, together with positive rectified version $\mathbf{y}_+ = (y_1^+, \dots, y_n^+)$ where $y_i^+ = g_+(y_i) \equiv \max(y_i, 0)$. We often regard \mathbf{y}_+ as the ‘output’ of the system, together with a complementary error vector $\mathbf{y}_- = \mathbf{y} - \mathbf{y}_+$.

Typically a sequence of p samples of n -dimensional input vectors \mathbf{x} is represented as the columns of a $n \times p$ matrix \mathbf{X} , and $\Sigma_{\mathbf{x}}$ is estimated from these data samples. We then have corresponding matrices $\mathbf{Z} = \mathbf{V}\mathbf{X}$, $\mathbf{Y} = \mathbf{W}\mathbf{Z}$, $\mathbf{Y}_+ = g_+(\mathbf{Y})$ and $\mathbf{Y}_- = \mathbf{Y} - \mathbf{Y}_+$ where $g_+(\cdot)$ is applied element-wise to the matrix \mathbf{Y} .

Now, at each update step we shall multiplicatively update \mathbf{W} by some square orthonormal ‘rotation’ matrix $\mathbf{R} \in SO(n)$, i.e. $\mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}_n$ and $\det \mathbf{R} = +1$. We can easily see that the update $\mathbf{W}_{new} \leftarrow \mathbf{R}\mathbf{W}_{old}$ will retain the orthonormality of \mathbf{W} , since e.g. $\mathbf{R}\mathbf{W}(\mathbf{R}\mathbf{W})^T = \mathbf{R}\mathbf{W}\mathbf{W}^T\mathbf{R}^T = \mathbf{I}_n$ [10]. We often find it convenient to express \mathbf{Y}_{new} in terms of \mathbf{W} and \mathbf{R} directly, i.e. $\mathbf{Y}_{new} = \mathbf{R}\mathbf{W}_{old}\mathbf{Z}$, or simply $\mathbf{Y} = \mathbf{R}\mathbf{W}\mathbf{Z}$. Thus both \mathbf{R} and \mathbf{W} are always elements of the set of $n \times n$ orthogonal (orthonormal) rotation matrices.

Now we can write an orthonormal rotation matrix $\mathbf{R} \in SO(n)$ as the exponential of a skew-symmetric matrix, i.e. $\mathbf{R} = e^{\mathbf{B}}$ where $\mathbf{B}^T = -\mathbf{B}$ is skew-symmetric, $\mathbf{B} \in so(n)$ [14]. We use the non-zero elements $\{\phi_{ij} \mid i < j\}$ of $\Phi = \text{UT}_+(\mathbf{B})$ to be the coordinates of an $(n(n-1)/2)$ -dimensional parameter space, where the strict upper triangle operator $\text{UT}_+(\cdot)$ sets elements on or below the diagonal to zero. For example, in the special case of $n = 2$, we have

$$\mathbf{B} = \begin{pmatrix} 0 & \phi \\ -\phi & 0 \end{pmatrix} \quad \text{giving} \quad \mathbf{R} = \begin{pmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{pmatrix}.$$

3 Optimization by Rotation in Steepest-Descent Geodesic

For our non-negative ICA algorithm, it is sufficient to minimize the distortion measure

$$J = \frac{1}{2} \|\mathbf{Y}_-\|_F^2 = \frac{1}{2} \sqrt{\sum_{ij} y_{ij}^-} \quad (1)$$

where $\|\cdot\|_F$ is the Frobenius norm, which will be zero if and only if the sequence of output vectors \mathbf{y}_+ ($= \mathbf{y}$ if $J = 0$) is some positive scaling and/or permutation of the non-negative sources \mathbf{s} [5]. Calculating the derivative of J with respect to Φ , we find [6]

$$\nabla_{\Phi} J = \text{UT}_+(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T) \quad (2)$$

where $[\nabla_{\Phi} J]_{ij} \equiv dJ/d\phi_{ij}$. Let us define an inner product in Φ -space of $\langle \Phi, \Theta \rangle = \sum_{ij} \phi_{ij} \theta_{ij}$ and let the corresponding norm $|\Phi| = \sqrt{\langle \Phi, \Phi \rangle} = \|\Phi\|_F$ be the distance measure. Then the matrix gradient $-\nabla_{\Phi} J$ in (2) is the *steepest descent* gradient for J in Φ -space,

$$\theta = |\nabla_{\Phi} J| = \frac{1}{2} \|\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T\|_F \quad (3)$$

is the norm of this gradient, and the matrix $\mathbf{H}_{\Phi} = -\nabla_{\Phi} J/\theta = -\text{UT}_+(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T)/\theta$ is the unit-norm steepest descent direction. For the zero gradient case $\theta = 0$ the matrix \mathbf{H}_{Φ} is undefined.

Starting from $\Phi(0) = \mathbf{0}$ and hence $\mathbf{R}(0) = \mathbf{I}_n$, using $\Phi(t) = t\mathbf{H}_{\Phi}$ defines a geodesic $\mathbf{R}(t) = e^{\mathbf{B}}(t)$ parametrized by t in \mathbf{R} -space, where $\mathbf{B}(t) = \Phi(t) - \Phi(t)^T$ [10]. If we define

$$\mathbf{H} = \mathbf{H}_{\Phi} - \mathbf{H}_{\Phi}^T = -(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T)/\theta \quad (4)$$

as the equivalent steepest descent direction in \mathbf{B} -space, we can therefore also write $\mathbf{B}(t) = t\mathbf{H}$. Thus we can reduce J by performing a ‘‘line search’’ to minimize $J(t)$ along this steepest-descent geodesic and then repeat in a new direction. There are a number of algorithms which can be used, from a small update step leading to gradient flow [10, 12], or larger steps based on approximating the $J(t)$ by a quadratic function [6]. However, we shall next consider a modification of this approach, based on the rotational nature of $\mathbf{R}(t)$.

3.1 Rotational Geometry of \mathbf{R} for $n \leq 3$

Let us now consider the case of rotations \mathbf{R} for $n = 2$ and $n = 3$ (for $n = 1$ we have trivially $\mathbf{R} = \mathbf{I}_1 = 1$ and a single point search space $\{\mathbf{B}\} = \{\Phi\} = \{0\}$). In the general case, we use the matrix exponential $\mathbf{R}(t) = e^{\mathbf{B}}(t) = \exp(t\mathbf{H})$ but for $n \leq 3$ this matrix exponential takes a convenient and easy to calculate form. For $n = 2$ we get a simple Givens rotation [15]

$$\mathbf{H} = \pm \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \quad \mathbf{B}(t) = \pm \begin{pmatrix} 0 & t \\ -t & 0 \end{pmatrix} \quad \mathbf{R} = \begin{pmatrix} \cos t & \pm \sin t \\ \mp \sin t & \cos t \end{pmatrix} \quad (5)$$

while for $n = 3$ we can use the normalized Rodrigues formula [16, 14]

$$\mathbf{R}(t) = \mathbf{I}_n + \sin t \mathbf{H} + (1 - \cos(t)) \mathbf{H}^2 \quad (6)$$

where \mathbf{H} is the normalized skew-symmetric matrix as constructed above. We can easily see this also applies in the $n = 2$ case in (5) since $\mathbf{H}^2 = -\mathbf{I}_2$ for $n = 2$. For both of these cases, we can see that $\mathbf{R}(t) = \mathbf{R}(t + 2k\pi)$ and hence we are looking for a minimum of a function $J(t)$ which repeats every 2π .

Of course, it is possible that $J(t)$ repeats at some smaller interval $2\pi/l$ for integer l . In fact, for the usual (not non-negative) ICA problem, the solutions \mathbf{Y} and $-\mathbf{Y}$ are considered to be equivalent, so $\mathbf{R}(t)$ and $\mathbf{R}(t + \pi)$ would also be equivalent, yielding a $J(t)$ which repeats every π (or even $\pi/2$ for $n = 2$, since a quarter-turn will also align with a solution). However, this is not the case for non-negative ICA, where the solution $-\mathbf{Y}$ is not equivalent to \mathbf{Y} .

3.2 Fourier Expansion of $J(t)$

If we are close to the minimum of $J(t)$ along the line, we could use a Taylor expansion about the minimum t^* to write

$$J(t) \approx a_0 + a_1(t - t^*) + a_2(t - t^*)^2$$

for which we have $J'(t) \approx a_1 + 2a_2(t - t^*)$ and $J''(t) \approx 2a_2$. Since $J'(t) = 0$ at the minimum point $t = t^*$, we must have $a_1 = 0$, leading to an estimate of distance from t^* of $(t - t^*) \approx J'(t)/J''(t)$. We therefore estimate that the minimum is at $\hat{t} = t - J'(t)/J''(t)$, i.e. a Newton update step. However, the Newton update method can suffer from problems away from the minimum. In particular, if we are close to the maximum rather than the minimum, the Newton method will converge to the maximum (where $J'(t)$ is also zero) instead of the minimum.

In the present system, we can use an alternative approach. Since we know that $J(t)$ repeats every $t = 2k\pi$, we can use a Fourier expansion instead. In its simplest form, we get

$$J(t) \approx a_0 - a_1 \cos(t - t^*) \quad (7)$$

where we use the minus cosine so that the minimum of $J(t)$ is at $t = t^*$. Proceeding as for the Newton method from the Taylor expansion, we get $J'(t) \approx a_1 \sin(t - t^*)$ and $J''(t) \approx a_1 \cos(t - t^*)$, leading to an estimate for the minimum of $\hat{t} = t - \arctan(J'(t), J''(t))$ where $\arctan(\cdot, \cdot)$ is a four-quadrant arc tan function defined such that $\psi = \arctan(\sin \psi, \cos \psi)$ for all $-\pi < \psi \leq \pi$ (see e.g. the Matlab function `atan2`). We notice that $\arctan(J'(t), J''(t)) \approx J'(t)/J''(t)$ for small $t - t^*$, leading to the Newton method as $t \rightarrow t^*$.

For $n > 3$ the situation is more complex [10, 14], with multiple orthogonal rotations so that $\mathbf{R}(t)$ does not repeat every $t = 2\pi$. Nevertheless, we have found experimentally that one rotation direction often dominates a long way from the solution, when a large change to t is required. For small t multiple rotations do emerge, but in this range we approximate the Newton method, so the non-repeating distant behaviour does not seem to be a concern.

3.3 Calculating the Line Derivatives

Once the geodesic (line) is defined by θ and \mathbf{H} as in (3) and (4), to move to the estimated minimum along this geodesic using either the Fourier expansion or the Newton (Taylor expansion) method, we need to calculate the line derivatives $J'(t)$ and $J''(t)$. First differentiating (1) with respect to y_{ij} yields $dJ/dy_{ij} = y_{ij}^- (dy_{ij}^-/dy_{ij})$. We notice that the slope dy_{ij}^-/dy_{ij} is discontinuous at $y_{ij} = 0$, changing between 1 and 0. However, the product $y_{ij}^- (dy_{ij}^-/dy_{ij})$ is well defined, since the discontinuity in dy_{ij}^-/dy_{ij} is ‘hidden’ by the zero in y_{ij}^- . Letting $\mathbf{K}_- = [k_{ij}^-]$ be an indicator matrix for \mathbf{Y}_- , such that $k_{ij}^- = 1$ if $y_{ij} < 0$, and zero otherwise, so $y_{ij}^- = k_{ij}^- y_{ij}$, we get $dJ/dy_{ij} = k_{ij}^- y_{ij} = y_{ij}^-$ so $J'(t) = \langle \mathbf{Y}_-, \mathbf{Y}'(t) \rangle$. Differentiating $\mathbf{Y}(t) = \mathbf{R}\mathbf{W}\mathbf{X} = e^{t\mathbf{H}}\mathbf{W}\mathbf{X}$ with respect to t we get $\mathbf{Y}'(t) = \mathbf{H}e^{t\mathbf{H}}\mathbf{W}\mathbf{X} = \mathbf{H}\mathbf{Y}$ so $J'(t) = \langle \mathbf{Y}_-, \mathbf{H}\mathbf{Y} \rangle = \text{trace}(\mathbf{Y}_-^T \mathbf{H}\mathbf{Y})$ which, substituting for \mathbf{H} from (4), eventually gives

$$J'(t) = -2\theta = -\|\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y}\mathbf{Y}_-^T\|_F. \quad (8)$$

For the second derivative, we would like to differentiate $J'(t) = \langle \mathbf{Y}_-, \mathbf{H}\mathbf{Y} \rangle$. This is no longer strictly differentiable, since the slope dy_{ij}^-/dy_{ij} jumps from 1 for $y_{ij} < 0$ to 0 for $y_{ij} > 0$ as we remarked above. However, for the purposes of optimization of J we will define $dy_{ij}^-/dy_{ij} = 0$ for $y_{ij} = 0$, giving $dy_{ij}^-/dy_{ij} = k_{ij}^-$. Writing $\mathbf{Y}_- = \mathbf{K}_- \circ \mathbf{Y}$ where \circ represents element-wise multiplication, and noting that $\langle \mathbf{K}_- \circ \mathbf{Y}, \mathbf{H}\mathbf{Y} \rangle = \langle \mathbf{K}_- \circ \mathbf{Y}, \mathbf{K}_- \circ (\mathbf{H}\mathbf{Y}) \rangle$, we get

$$\begin{aligned} J''(t) &= \langle \mathbf{K}_- \circ \mathbf{Y}'(t), \mathbf{K}_- \circ (\mathbf{H}\mathbf{Y}) \rangle + \langle \mathbf{Y}_-, \mathbf{H}\mathbf{Y}'(t) \rangle \\ &= \|\mathbf{K}_- \circ (\mathbf{H}\mathbf{Y})\|_F^2 + \langle \mathbf{Y}_-, \mathbf{H}^2 \mathbf{Y} \rangle. \end{aligned} \quad (9)$$

4 Proposed Algorithm

Given an input data matrix \mathbf{X} , whitened to give $\mathbf{Z} = \mathbf{V}\mathbf{X}$ as described above, the algorithm is as follows:

1. Initialize $\mathbf{W} = \mathbf{I}_n$
2. Calculate $\mathbf{Y} = \mathbf{W}\mathbf{Z}$, $\mathbf{Y}_- = g_-(\mathbf{Y})$ and θ as in (3).
3. If $\theta = 0$, finish.
4. Calculate \mathbf{H} as in (4), $J'(t)$ as in (8) and $J''(t)$ as in (9), and set $t_1 = -\arctan(J'(t), J''(t))$.
5. Calculate $\mathbf{B} = t_1 \mathbf{H}$ and $\mathbf{R} = e^{\mathbf{B}}$ (using e.g. the Rodrigues formula for $n \leq 3$).
6. Update $\mathbf{W} \leftarrow \mathbf{R}\mathbf{W}$.
7. Repeat from step 2 until $\theta = 0$.

To use the Newton method instead of a Fourier expansion update, set $t_1 = -J'(t)/J''(t)$ in step 4. While this algorithm has been derived specifically for the non-negative ICA problem, it should be possible to modify it to other tasks for optimization over orthonormal matrices, modifying as necessary if $J(t)$ repeats at $t = \pi$, as for normal ICA, instead of $t = 2\pi$ as for non-negative ICA.

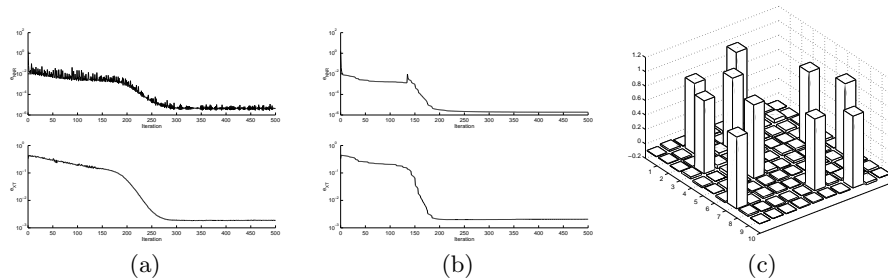


Fig. 1. Results on artificial data for $n = 10$, showing (a) learning curve for the geodesic ‘single step’ algorithm in [6], (b) learning curve for the Fourier step algorithm described in the current paper, and (c) the values of \mathbf{WVA} for (b) after 250 iterations, showing this approximates a positive permutation matrix.

5 Experiments

Experiments were carried out in Matlab to confirm the operation of the algorithm. For artificial data, we generated $p = 1000$ unit variance random source data vectors \mathbf{s}_p , mixed using a random matrix \mathbf{A} as for the non-negative PCA experiments described in [7]. To measure the separation performance of the algorithm, we use two performance measures: a nonnegative reconstruction error

$$e_{NNR} = \frac{1}{np} \|\mathbf{Z} - \mathbf{W}^T \mathbf{Y}_+\|_{\mathbf{F}}^2 \quad (10)$$

which, for orthonormal \mathbf{W} , is a scaled version of J ; and a cross-talk error

$$e_{XT} = \frac{1}{n^2} \|\text{abs}(\mathbf{WVA})^T \text{abs}(\mathbf{WVA}) - \mathbf{I}_n\|_{\mathbf{F}}^2 \quad (11)$$

where $\text{abs}(\mathbf{M})$ is the matrix of absolute values of the elements of \mathbf{M} , which is zero only if $\mathbf{y} = \mathbf{WVA}\mathbf{s}$ is a positive permutation of the sources, i.e. only if the sources have been successfully separated.

For $n = 10$, after 250 iterations (18.8s of CPU time on an 850MHz Pentium 3), we had $e_{NNR} = 2.20 \times 10^{-6}$ and $e_{XT} = 2.00 \times 10^{-3}$, bettering the non-negative PCA algorithm [7] which took 10^5 iterations to obtain $e_{NNR} = 5.02 \times 10^{-4}$ and $e_{XT} = 0.0553$ (called e_{MSE} and e_{Perm} respectively in [7]). We also compared this with the geodesic ‘single step’ algorithm introduced in [6] which is almost as fast as the current algorithm, but exhibits a very noisy learning curve (Fig. 1(a)), indicating the assumption in that algorithm of a quadratic bowl with $J = 0$ at the bottom is not completely justified. For the current algorithm, the Fourier expansion leads to a particularly fast initial one or two iterations, with an unexpected increase in reconstruction error e_{NNR} around iteration 140 coinciding with a corresponding step decrease in crosstalk error e_{XT} . For details see Fig. 1. Similar results were observed in an image separation task (not shown).

We also applied this algorithm to a music analysis problem. Here a small segment of a Liszt ‘Etude’ has been played on a MIDI synthesized piano, pre-processed into power spectrogram with $p = 467$ frames and reduced to 10 dimensions using PCA (for details of this task, see [6]). The geodesic ‘single step’ algorithm becomes unstable very quickly on this task, since $J(t)$ does not reach zero at the best solution. We therefore compared this to the geodesic flow algorithm [10, 6], using $\mathbf{B} = -\mu(\mathbf{Y}_- \mathbf{Y}^T - \mathbf{Y} \mathbf{Y}_-^T)$ in the update algorithm with the update factor $\mu = 0.001$ chosen experimentally to yield fastest convergence without instability (Fig. 2). The Fourier update algorithm is similar in speed

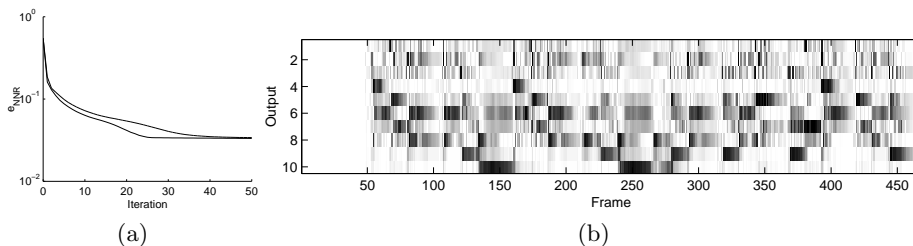


Fig. 2. Results on music analysis task, showing (a) learning curves for geodesic flow (upper curve) and Fourier update algorithm (lower curve), and (b) output for the Fourier update algorithm showing identified notes. See [6] for task details.

to the fastest geodesic flow algorithm (actually slightly faster in this case), but does not require any update parameter μ to be selected.

6 Conclusions

The non-negative ICA problem can be tackled using algorithms which minimize a cost function J over the space of orthogonal (orthonormal) rotation matrices \mathbf{W} , i.e. where $\mathbf{W}\mathbf{W}^T = \mathbf{W}^T\mathbf{W} = \mathbf{I}_n$. For the special case of $n \leq 3$, a geodesic $\mathbf{R}(t) = e^{t\mathbf{H}}$ where \mathbf{H} is skew-symmetric yields a single plane of rotation, giving a cost function $J(t)$ that repeats every $t = 2k\pi$.

We proposed an algorithm that takes advantage of the cyclical nature of $J(t)$, using a truncated Fourier expansion of $J(t)$ to yield a Newton-like update step along the steepest-descent geodesic. This automatically approximates to a usual Newton update step near to a minimum. Experiments confirm that this algorithm is effective, even for $n > 3$, and it compares favourably with existing non-negative ICA algorithms.

We suggest that this approach could be modified for other tasks requiring optimization over orthogonal matrices, such as the standard prewhitened ICA problem, but using a rotation period of π (or $\pi/2$) instead of 2π .

7 Acknowledgements

This work was partially supported by EPSRC grant GR/R54620, and by the EU-FP6-IST-507142 project SIMAC (Semantic Interaction with Music Audio Contents: www.semanticaudio.org). The music sequence is used by permission of the Classical Piano Midi Page www.piano-midi.de, copyright Bernd Krueger.

References

1. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In Leen, T.K., Dietterich, T.G., Tresp, V., eds.: *Advances in Neural Information Processing Systems 13*, MIT Press (2001) 556–562 Proceedings of NIPS*2000.
2. Paatero, P., Tapper, U.: Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics* **5** (1994) 111–126
3. Harpur, G.F.: *Low Entropy Coding with Unsupervised Neural Networks*. PhD thesis, Department of Engineering, University of Cambridge (1997)
4. Cichocki, A., Georgiev, P.: Blind source separation algorithms with matrix constraints. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* **E86–A** (2003) 522–531
5. Plumbley, M.D.: Conditions for nonnegative independent component analysis. *IEEE Signal Processing Letters* **9** (2002) 177–180
6. Plumbley, M.D.: Algorithms for nonnegative independent component analysis. *IEEE Transactions on Neural Networks* **14** (2003) 534–543
7. Plumbley, M.D., Oja, E.: A “nonnegative PCA” algorithm for independent component analysis. *IEEE Transactions on Neural Networks* **15** (2004) 66–76
8. Brockett, R.W.: Dynamical systems that sort lists, diagonalize matrices, and solve linear programming problems. *Linear Algebra and its Applications* **146** (1991) 79–91
9. Edelman, A., Arias, T.A., Smith, S.T.: The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.* **20** (1998) 303–353
10. Nishimori, Y.: Learning algorithm for ICA by geodesic flows on orthogonal group. In: *Proceedings of the International Joint Conference on Neural Networks (IJCNN’99)*. Volume 2., Washington, DC (1999) 933–938
11. Douglas, S.C.: Self-stabilized gradient algorithms for blind source separation with orthogonality constraints. *IEEE Transactions on Neural Networks* **11** (2000) 1490–1497
12. Fiori, S.: A theory for learning by weight flow on Stiefel-Grassman manifold. *Neural Computation* **13** (2001) 1625–1647
13. Hyvärinen, A., Karhunen, J., Oja, E.: *Independent Component Analysis*. John Wiley & Sons (2001)
14. Gallier, J., Xu, D.: Computing exponentials of skew-symmetric matrices and logarithms of orthogonal matrices. *International Journal of Robotics and Automation* **18** (2003) 10–20
15. Comon, P.: Independent component analysis - a new concept? *Signal Processing* **36** (1994) 287–314
16. Fiori, S., Rossi, R.: Stiefel-manifold learning by improved rigid-body theory applied to ICA. *International Journal of Neural Systems* **13** (2003) 273–290