Some Gradient Based Joint Diagonalization Methods for ICA

Bijan Afsari and P.S. Krishnaprasad

Institute for Systems Research , University of Maryland
College Park, Maryland 20742, USA
{bijan, krishna}@isr.umd.edu

Abstract. We present a set of gradient based orthogonal and non-orthogonal matrix joint diagonalization algorithms. Our approach is to use the geometry of matrix Lie groups to develop continuous-time flows for joint diagonalization and derive their discretized versions. We employ the developed methods to construct a class of Independent Component Analysis (ICA) algorithms based on non-orthogonal joint diagonalization. These algorithms pre-whiten or sphere the data but do not restrict the subsequent search for the (reduced) un-mixing matrix to orthogonal matrices, hence they make effective use of both second and higher order statistics.

1 Introduction

Simultaneous or Joint Diagonalization (JD) of a set of estimated statistics matrices is a part of many algorithms, especially in the field of ICA and Blind Source Separation (BSS). The early methods developed for JD were those that restrict the joint diagonalizer to belong to the compact Lie group of orthogonal matrices O(n)[5]. Accordingly the JD problem is defined as minimization of a function of the form:

$$J_1(\Theta) = \sum_{i=1}^{n} \left\| \Theta C_i \Theta^T - \mathrm{diag}(\Theta C_i \Theta^T) \right\|_F^2 \tag{1}$$

where $\{C_i\}_{i=1}^N$ is the set of symmetric matrices to be diagonalized, $\Theta \in O(n)$ is the joint diagonalizer sought, diag(A) is the diagonal part of A and $\|A\|_F$ denotes the Frobenius norm of matrix A. We remind the reader that due to compactness of O(n) we know in advance that $J_1(\Theta)$ has a minimum on O(n). Different methods for minimization of this cost function in the context of Jacobi methods [5],[3] and optimization on manifolds [10],[12] have been proposed. Here we shall give a gradient flow expression for this problem, which to our knowledge is referred to in some papers without explicit representation [9].

Non-orthogonal JD is very appealing in the context of noisy ICA. Consider the standard ICA model:

$$\mathbf{x}_{n\times 1} = A_{n\times n}\mathbf{s}_{n\times 1} + \mathbf{n}_{n\times 1} = \mathbf{z} + \mathbf{n} \tag{2}$$

with $\mathbf{n}$ a Gaussian noise vector(all random variables are assumed to be of mean zero). We know that if $\{C_i\}_{i=1}^N$ is a collection of matrix slices of cumulant tensor of $\mathbf{x}$ of order higher than two and B is an un-mixing matrix belonging to the Lie group of non-singular matrices GL(n) then BC_iB^T's are diagonal. The problem of non-orthogonal JD has been addressed by few authors among them: [13] [11] [14]. Defining a suitable cost function for non-orthogonal JD seems to be difficult due to non-compactness of GL(n). In Section (3) we consider extension of J_1 to GL(n) or SL(n) (the group of non-singular matrices with unity determinant) using the scale ambiguity inherent in the ICA problem and we shall derive gradient based continuous flows and their discrete versions for non-orthogonal JD. Although these algorithms are general, they perform much better if the matrix sought is close to orthogonal or a perturbation of the initial condition (the identity matrix in most cases). This is a manifestation of the fact the JD problem is easier to solve on a compact set or locally. Based on this observation, in Section (4) we develop an ICA algorithm based on non-orthogonal JD. These algorithms have the property that although they first sphere the data they do not confine the JD search afterwards to O(n), i.e. they perform non-orthogonal JD after the data is whitened. In Section (5) we present some simulations comparing the performance of the developed ICA algorithms and the celebrated JADE algorithm [5] in noise.

Notation: In the sequel tr(A) is the trace of the matrix A, $\dot{x}$ denotes the time derivative of the variable x, T_pM represents the tangent space to the manifold M at point p and $I_{n\times n}$ is the $n \times n$ identity matrix. All random variables are in boldface small letters and are assumed to be zero mean.

2 Gradient Based Orthogonal JD

Considering O(n) as a Riemannian Lie group with the Riemannian metric defined as $\langle \xi, \eta \rangle_\Theta = \mathrm{tr}((\xi\Theta^T)^T\eta\Theta^T) = \mathrm{tr}(\xi^T\eta)$ for $\xi, \eta \in T_\Theta O(n)$ and, following [8], it is easy to find the gradient flow for minimization of $J_1(\Theta)$ as:

$$\dot{\Theta} = -\Delta\Theta = \sum_{i=1}^{N} \left[\mathrm{diag}(\Theta C_i \Theta^T), \Theta C_i \Theta^T\right] \Theta, \qquad \Theta(0) = I_{n\times n} \tag{3}$$

where $[X, Y] = XY - YX$ is the Lie bracket. In [6], a result is given which is essentially the same as (3) but with a different point of view and representation. In discretization of a flow on O(n) it is difficult to ensure that the updates keep the answer always orthogonal. Different methods have been proposed to address this [4], [10], [12]. We mention that in the context of ICA an Euler discretization with small enough fixed step-size, which is equivalent to steepest descent algorithm, is promising.

3 Non-Orthogonal JD Based on the Gradient of J_1

Consider a set of symmetric matrices $\{C_i\}_{i=1}^N$ that are assumed to have an exact joint diagonalizer in GL(n). Then the cost function $J_1(B)$ with $B \in$GL(n) has

a minimum of zero. It may seem appropriate to define this as a cost function for JD in the non-orthogonal case. However we can see that this cost function can be reduced by reducing the norm of B. In other words this cost function is not scale-invariant, i.e. $J_1(\Lambda B) \neq J_1(B)$ for non-singular diagonal Λ. By scale-invariance for a JD cost function in terms of un-mixing matrix B, we mean that it does not change under left multiplication of the argument by diagonal matrices in the same manner that mutual information is scale-invariant. In the following we provide some remedies to deal with scale variability of $J_1(B)$.
We consider $\mathrm{GL}(n)$ as a Riemannian manifold with the Riemannian metric (also known as Natural Riemannian metric [2]):

$$\langle \xi, \eta \rangle_B = \mathrm{tr}((\xi B^{-1})^T \eta B^{-1}) = \mathrm{tr}(B^{-T} \xi^T \eta B^{-1}) = \mathrm{tr}(\eta (B^T B)^{-1} \xi^T) \quad (4)$$

for $\xi, \eta \in T_B GL(n)$. Again it is easy to see that the gradient flow for minimization of $J_1(B)$ is:

$$\dot{B} = -\Delta B \quad (5)$$

with

$$\Delta = \sum_{i=1}^{N} \big(BC_iB^T - \mathrm{diag}\big(BC_iB^T\big)\big)BC_iB^T \quad (6)$$

and $B(0) \in \mathrm{GL}(n)$. An interesting observation is that the equilibria of this flow found by letting $\Delta = 0$ satisfy $BC_iB^T = \mathrm{diag}(BC_iB^T)$ for all $1 \leq i \leq N$. Therefore unless C_i's have an exact joint diagonalizer flow in (5) has no equilibria, which confirms our argument that $J_1(B)$ is not a suitable criterion for non-orthogonal JD. We recall that in [11] a scale invariant cost function is introduced that is applicable only for positive definite C_i's.

One way to ameliorate the problem with non-compactness of $\mathrm{GL}(n)$ and scale variability of $J_1(B)$ is to consider minimization of $J_1(B)$ over $\mathrm{SL}(n)$. Obviously $\mathrm{SL}(n)$ is not a compact group and $\det(B) = 1$ does not put any upper bound on $\|B\|$, but it requires $\|B\|_2 \geq 1$ and this prevents converging to the trivial infimum of $J_1(B)$ at $B = 0$. By restricting B to be in $\mathrm{SL}(n)$, we identify all matrices of the form αB for $\alpha \in \mathbb{R} - \{0\}$ with B. It is easy to show that the orthogonal projection of *any* matrix $A_{n\times n}$ on the space of matrices with zero trace is given by: $A^0 = A - \frac{tr(A)}{n} I_{n\times n}$. Accordingly the projection of the gradient flow found in (5) to $\mathrm{SL}(n)$ is the gradient flow:

$$\dot{B} = -\Delta^0 B, \quad B(0) = I \quad (7)$$

with Δ as in (6).

A more general way to deal with non-compactness of $\mathrm{GL}(n)$ and scale-variability of $J_1(B)$ is to project its gradient on to a subspace such that the projection does not reduce the cost function due to row scaling. This approach maybe considered as equivalent to identifying B and ΛB for all non-singular diagonal Λ. This method leads to a nonholonomic flow [1]. The projected flow is derived from (5) by projecting Δ in (6) to the space of zero diagonal matrices. Letting $\Delta^\perp = \Delta - \mathrm{diag}(\Delta)$, the projected flow is given by:

$$\dot{B} = -\Delta^\perp B, \quad B(0) = I_{n\times n} \quad (8)$$

where Δ is the same as in (6). From the definition of gradient with respect to the Riemannian metric (4) we have, along trajectories of (8):

$$\dot{J}_1 = \mathrm{tr}((\nabla J_1 B^{-1})^T \dot{B} B^{-1}) = -\mathrm{tr}(\Delta^T \Delta^{\perp}) = -\mathrm{tr}(\Delta^{\perp T} \Delta^{\perp}) = -\sum_{i \neq j} \Delta_{ij}^2 \leq 0 \tag{9}$$

So, as long as Δ is not diagonal, (8) is a descent flow. Note that the equilibria of the flow (8) is exactly the set $\{B \in GL(n) | \Delta \text{ is diagonal}\}$. On the other hand if $B(0) \in \mathrm{SL}(n)$ then (8) restricts to a flow on $\mathrm{SL}(n)$ and $\|B(t)\|_2 \geq 1$.

By picking small enough step-size we expect to have discretizations of (7) and (8) that decrease the cost function at each step and keep the trajectory on $\mathrm{SL}(n)$ as much as possible. These two flows have the general form:

$$\dot{B} = -XB \tag{10}$$

where X is defined accordingly. The Euler discretization will be:

$$B_{k+1} = (I - \mu_k X_k) B_k, \quad B_0 = I \quad k \geq 0 \tag{11}$$

In practice we can choose a fixed small step-size and change it if we observe instability. A pseudo code for this algorithm is:

Algorithm 1:

```
1. set $\mu$ and $\epsilon$.
2. set $B_0 = I_{n \times n}$ or “to a good initial guess”.
3. while $\|X_k\|_F > \epsilon$ do
   $B_{k+1} = \big(I - \mu X_k\big) B_k$
   if $\|B_{k+1}\|_F$ is “big” then “reduce” $\mu$ and goto 2.
4. end
```

It is possible to modify the flow (8) such that its discretization yields $\det(B_k) = 1$, by construction. Let X^L (X^U) denote a lower (upper) triangular matrix that has the same lower (upper) part as X. Consider the lower-triangular version of (8) $\dot{B} = -\Delta^{\perp L} B$. Note that by the Euler discretization $B_{k+1} = (I - \mu \Delta_k^{\perp L}) B_k$ and if $B_0 = I$ then $\det(B_k) = 1$, by construction. The same is true if we consider the upper triangular version of (8). Therefore based on the LU factorization of the un-mixing matrix we can have an iterative algorithm that alternatively looks for upper and lower triangular factors of the un-mixing matrix and keeps the determinant unity by construction. A pseudo code for this method is:

Algorithm 2:

Consider the set $\{C_i\}_{i=1}^N$ of symmetric matrices. Let (a): $\dot{U} = -\Delta^{\perp U} U$ and (b): $\dot{L} = -\Delta^{\perp L} L$ with $B = U(0) = L(0) = I$ be the corresponding upper and lower triangularized versions of (8).

```
1. Use Algorithm 1 to find $U$ the solution to (a).
2. set $C_i \leftarrow U C_i U^T$.
3. Use Algorithm 1 to find $L$ the solution to (b)
4. set $C_i \leftarrow L C_i L^T$.
5. set $B \leftarrow L\,UB$
6. if $\|LU - I\|_F$ is “small” end, else goto 1
```

4 A Family of ICA Algorithms Based on JD

Here we introduce a general scheme for an ICA algorithm. Consider the data model (2). If we lack information about noise, we use the correlation matrix of $\mathbf{x}$ instead of that of $\mathbf{z}$ to find a whitening or sphering matrix W. In this case the sphered signal:

$$\mathbf{y} = W\mathbf{x} = WA\mathbf{s} + W\mathbf{n} = A_1\mathbf{s} + \mathbf{n}_1 \tag{12}$$

is such that the reduced mixing matrix A_1 can not assumed to be orthogonal as in the noiseless case, however it can assumed to be close to orthogonal where the orthogonality error depends on the signal and noise power and condition number of the matrix A [7]. Note that, by Gaussianity of noise all the higher order cumulant matrix slices of $\mathbf{y}$ are diagonalizable by A_1. Applicability of the JADE algorithm which jointly diagonalizes a set of fourth order cumulant slices of $\mathbf{y}$ by an orthogonal matrix will be limited in this case because it reduces the degrees of freedom in the optimization problem involved or in other words leaves the bias introduced in the whitening phase un-compensated. An algorithm such as JADE or mere "sphereing" brings the data (globally) close to independence but we can proceed further by (locally) finding a non-orthogonal un-mixing matrix and reduce mutual information further. This local un-mixing matrix can be incorporated into the whole answer by multiplication due to the multiplicative group structure of the ICA problem. We shall use this idea in developing a new ICA method based on non-orthogonal JD. We emphasize that after whitening although we look for a non-orthogonal joint diagonalizer the fact that it is close to orthogonal makes the search much easier in practice.
Consider the data model (2). The general scheme for ICA based on non-orthogonal JD of fourth (or higher) order cumulant slices is comprised of the following steps:

1. Whiten $\mathbf{x}$, let W be a whitening matrix, compute $\mathbf{y} = W\mathbf{x}$ and set $B = W$.
2. Estimate $C = \{C_i\}_{i=1}^N$ a subset of the fourth order cumulant matrix slices of $\mathbf{y}$.
3. Jointly diagonalize $C = \{C_i\}_{i=1}^N$ by an orthogonal matrix Θ and set $C_i \leftarrow \Theta C_i \Theta^T$.
4. Jointly diagonalize $C = \{C_i\}_{i=1}^N$ by a non-orthogonal matrix B_{JDN} (using any algorithm such as Algorithms 1 or 2), set $C_i \leftarrow B_{JDN} C_i B_{JDN}^T$ and set $B \leftarrow B_{JDN}\Theta B$.
5. If necessary goto step (3)
6. Compute the recovered signal $\hat{\mathbf{x}} = B\mathbf{x}$

Steps (1-3) comprise the JADE algorithm. In experiments that the model (2) truly holds, inclusion of step (3) proves to be redundant, but in cases where the model does not hold, compactness of $\mathrm{O}(n)$ can be helpful as well as repeating steps (3) and (4). The justification for adopting this scheme is four-fold:

1. Usually by whitening the data the mutual information is reduced so the whitened data is closer to independence.
2. In most cases whitening the data reduces the dynamic range of $\|C_i\|$'s and enables better convergence for numerical methods thereafter.

3. Although estimation of the correlation matrix of $\mathbf{z}$ in (2) from observation data $\mathbf{x}$ is biased it has less variance than the estimated higher order cumulant slices (this is pronounced especially in small sample sizes). Therefore it is meaningful to use as much information as possible from this correlation matrix provided we can avoid the harm of the "bias" it introduces.
4. As we mentioned before, solving the ICA or JD problem for $\mathbf{y}$ is more local than the one for $\mathbf{x}$. Also the fact that A_1 in (12) is close to orthogonal makes the non-orthogonal JD of the cumulant slices of $\mathbf{y}$ instead of those of $\mathbf{x}$ much more efficient and easier.

In the sequel we consider ICA algorithms that are comprised of steps 1,2,4. An algorithm with its JD part based on the discrete version of flow (7) is referred to as SL(n)-JD, an algorithm with its JD part based on the discrete version of flow (8) is referred to as NH-JD and an algorithm based on the LU factorization (Algorithm 2) is referred to as LU-JD.

5 Simulations

In this section we compare the performance of the developed set of algorithms with the standard JADE in the presence of noise. We consider

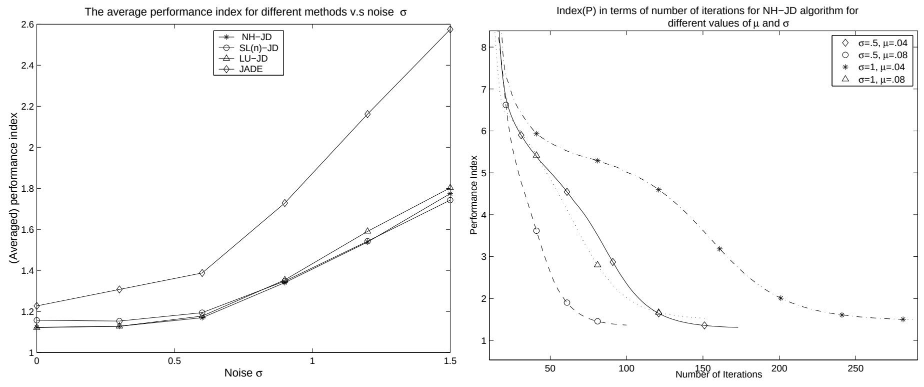
$$\mathbf{x} = A\mathbf{s}_{n\times 1} + \sigma\mathbf{n} \tag{13}$$

where $\mathbf{n}$ is zero mean Gaussian noise with identity correlation matrix then σ^2 indicates the power of noise. We consider $n = 5$ sources. Two of them are uniformly distributed in $[-\frac{1}{2}, \frac{1}{2}]$ and another two are two-side exponentially distributed with parameter $\lambda = 1$ and mean zero and the fifth one is one-side exponential with parameter $\lambda = 1$. The matrix A is randomly generated and to fit in the page the entries are truncated to integers:

$$A = \begin{bmatrix} -4 & 11 & -1 & 1 & 2 \\ -16 & 11 & 7 & 10 & -13 \\ 1 & 0 & -5 & 0 & 7 \\ 2 & 3 & 21 & 0 & 16 \\ -11 & 1 & -1 & -8 & -6 \end{bmatrix}$$

We generate $T = 3500$ samples of data and mix the data through A. Next we run four ICA algorithms. Three algorithms SL(n)-JD, NH-JD and LU-JD in addition to the standard JADE are applied to the data. $N = n^2 = 25$ fourth order cumulant matrix slices are used. For SL(n)-JD and NH-JD $\mu = .01$ and $\epsilon = .01$ are used (see Algorithm 1). For LU-JD $\mu = .05$, $\epsilon = .01$ are used (see Algorithm 2) and the LU iteration is performed five times. These values are not optimal, they were chosen based on few tries. Implementations are in $MATLAB^{®}$ code and the $MATLAB^{®}$ code for JADE was downloaded from: http://tsi.enst.fr/c̃ardoso/icacentral/Algos. The performance measure used is the distance of the product of the estimated un-mixing and the mixing matrix, i.e. $P = BA$, from essential diagonality:

$$\text{Index}(P) = \sum_{i=1}^{n}(\sum_{j=1}^{n}\frac{|p_{ij}|}{\max_k |p_{ik}|} - 1) + \sum_{j=1}^{n}(\sum_{i=1}^{n}\frac{|p_{ij}|}{\max_k |p_{kj}|} - 1) \tag{14}$$



(a) (b)

Fig. 1. (a) Average in-noise-performance index (every point is averaged over 100 trials) of different JD based ICA algorithms. The average Index(P) is plotted versus σ. (b) Index(P) in terms of iteration number for NH-JD algorithm for noise $\sigma = 0.5, 1$ and step-size $\mu = 0.04, 0.08$

For each value of σ the experiment is run $k = 100$ times and the performance measure is averaged over the trials. Figure (1.a) shows the results. We can see that the introduced algorithms all have almost the same performance and outperform the standard JADE especially in high level Gaussian noise. In Figure (1.b) we consider the behavior of Index(P) in terms of number of iterations in the JD part of the NH-JD algorithm for a single realization of data generated as above. Two different values of $\sigma = .5, 1$ and step-size $\mu = .04, .08$ are examined. All the initial conditions in the JD part (i.e. Algorithm 1) are $B_0 = I_{5\times 5}$. As the figure illustrates more iterations are required as noise power increases. By increasing step-size one may combat this, however dramatic increase of μ may result in instability of the algorithm. The run-time for these algorithms (in $MATLAB^{\circledR}$ code) is higher than JADE's, although we expect faster performance in low-level codes or DSPs. Part of this slower convergence can be attributed to the nature of gradient based methods which have linear convergence. One idea for speed improvement can be to use the result from JADE as the initial condition for the non-orthogonal JD methods introduced here.

6 Conclusion

We introduced gradient based flows for orthogonal and non-orthogonal JD of a set symmetric matrices and developed a family of ICA algorithms upon non-orthogonal JD. The non-orthogonal flows are derived based on defining suitable metrics and the geometry of the groups GL(n) and SL(n). The main drawback of gradient based JD methods is their slow convergence but their implementation requires only addition and multiplication. The developed ICA algorithms have the property that after whitening the data they do not confine the search space to orthogonal matrices. This way we can take advantage of both second order statistics (which has less variance) and higher order statistics which are blind to Gaussian noise. Numerical simulations show better performance for the proposed algorithms than for the standard JADE algorithm in Gaussian noise.

7 Acknowledgments

This research was supported in part by Army Research Office under ODDR&E MURI01 Program Grant No. DAAD19-01-1-0465 to the Center for Communicating Networked Control Systems (through Boston University).

References

1. S.Amari, T.-P Chen, A. Chichoki, Non-holonomic constraints in learning algorithms for blind source separation , preprint, 1997.
2. S. Amari: Natural Gradient Adaptation. In S. Haykin (ed): Unsupervised Adaptive Filtering, Vlume I, Blind Source Separation, Wiley Interscience, 2000.
3. A. Bunse-Gerstner, R. Byers and V. Mehrmann: Numerical Methods For Simultaneous Diagonalization, SIAM Journal on Matrix Analysis and Applications, vol. 4, pp. 927-949, 1993.
4. M.P. Calvo, A. Iserles and A. Zanna: Runge-Kutta methods for orthogonal and isospectral flows, Appl. Num. Maths 22 (1996)
5. J.F. Cardoso and A. Soulumiac: Blind Beamforming For Non-Gauusian Signals, IEE-Proceedings, Vol.140, No 6, Dec 1993
6. J.F. Cardoso: Perturbation of joint diagonalizers, Technical report, Telecom Paris, 1994.
7. J.F. Cardoso: On the performance of orthogonal source separation algorithms, EUSIPCO-94, Edinburgh.
8. U.Helmke and J.B. Moore: Optimization and Dynamical Systems, Springer-Verlag, 1994
9. G. Hori, J.H. Manton: Critical Point Analysis of Joint Diagonalization Criteria, 4th International Symposium on Independent Component Analysis and Blind Signal Separation (ICA2003), April 2003, Nara, Japan
10. J.H Manton: Optimization Algorithms Exploiting Unitary Constraints, IEEE Transactions on Signal Processing, Vol. 50, No. 3, March 2002
11. D.T. Pham: Joint Approximate Diagonalization of Positive Definite Hermitian Matrices, SIAM Journal of Matrix Analysis and Applications, Vol. 22, No. 4, pp. 136-1152.
12. I. Yamada, T. Ezaki: An Orthogonal Matrix Optimization by Dual Cayley Parametrization Technique, 4th International Symposium on Independent Component Analysis and Blind Signal Separation (ICA2003), April 2003, Nara, Japan
13. A.Yeredor: Non-Orthogonal Joint Diagonalization in the Least-Squares Sense With Application in Blind Source Separation, IEEE Transactions on Signal Processing, Vol 50, No.7.July 2002.
14. A.Ziehe, P. Laskov, K. Muller, G. Nolte: Linear Least-squares Algorithm for Joint Diagonalization, 4th International Symposium on Independent Component Analysis and Blind Signal Separation (ICA2003), Nara, Japan, April 2003.