Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Solving Joint Diagonalization Problems via a Riemannian Conjugate Gradient Method in Stiefel Manifold

Hugo Lara Urdaneta[1]
Campus de Blumenau, UFSC, Blumenau, SC
Harry Fernando Oviedo Leon [2]
Mathematics Research Center, CIMAT A.C. Guanajuato, Mexico

Abstract. We present a Riemannian Conjugate Gradient (RCG) algorithm to solve the Joint Diagonalization Problem in Stiefel manifolds. We use the special structure of this manifold to build a procedure which avoids extra-calculations associated to vector transportation. Numerical experiments which compare our algorithm with other known procedures are also provided.

Palavras-chave. Optimization on Manifolds, Stiefel Manifold, Conjugate Gradient Method, Vector Transport.

1 Introduction

The joint diagonalization problem (JDP) for N symmetric $n\times n$ matrices $A_1, A_2, \ldots, A_N$ consists of finding an orthogonal matrix of size $n \times p$, which maximizes the sum of squares of the diagonal entries at $X^T A_l X$, $l = 1, \ldots, N$ (see [12]). Getting a solution for this problem is of great value in applications such as Independent Component Analysis (ICA), and Blind Source Separation Problem, among others [3]. The problem JDP can be cast as the following optimization problem:

$$\begin{array}{cc} \text{minimize} & \mathcal{F}(X) = -\sum_{l=1}^{N} ||diag(X^T A_l X)||^2 \\ \text{subject to} & X^T X = I, \end{array} \tag{1}$$

Algorithms that addresses this quadratically constrained nonlinear optimization problem are diverse, most of them using the associated first order optimality conditions and based on the gradient of the Lagrangean function. This kind of method build a sequence $\{X^k\}$ of iterates that converges to optimal solutions, which are feasible to the orthogonality constraints. Nevertheless, this sequence is feasible (orthogonal) only at optimality. In some cases, it is important to build a sequence $\{X^k\}$ of feasible matrices converging to optimality, in virtue that the structure is kept in the search ([2]). The idea of manifold optimization methods is to use the tools about constructing optimization algorithms in $\mathbb{R}^n$, but following the manifold, ensuring feasibility in each iteration.

[1]hugo.lara.urdaneta@ufsc.br
[2]harry.oviedo@cimat.mx

The next section is devoted to some notation and background on Riemannian manifolds, including RCG methods, and tools on Stiefel manifolds. The third section exhibit our proposed algorithm, and comment it convergence properties. Section 4 deploy numerical experiments, and finally we provide conclusion remarks.

2 Notation and Background

Let A be an $n \times n$ matrix with real entries. We say that A is skew symmetric if $A^\top = -A$. The trace $Tr[A]$ of A is defined as the sum of the diagonal entries. The Euclidean inner product of two matrices $A, B \in \mathbb{R}^{m\times n}$ is defined as $\langle A, B\rangle_e := \sum_{i,j} a_{ij} b_{ij} = Tr[A^\top B]$, where a_{ij} and b_{ij} denote the (i,j) component of the respective matrix. The canonical inner product associated to $X \in \mathbb{R}^{m\times n}$ is defined as $\langle A, B\rangle_c := Tr[A^\top(I - \frac{1}{2}XX^\top)B]$. The Frobenius norm of A is defined as the metric induced by the Euclidean inner product, that is, $||A||_F = \sqrt{\langle A, A\rangle_e}$. Let $\mathcal{F} : \mathbb{R}^{n\times p} \to \mathbb{R}$ be a differentiable function, and denote by $G := \mathcal{DF}(X) := \left(\frac{\partial \mathcal{F}(X)}{\partial x_{ij}}\right)$ the matrix of partial derivatives of $\mathcal{F}$ with respect to X (the Euclidean gradient of $\mathcal{F}$). The directional derivative of $\mathcal{F}$ along a given matrix $Z \in \mathbb{R}^{n\times p}$ at a given point X is defined by $\mathcal{DF}(X)[Z] := \lim_{t\to 0} \frac{\mathcal{F}(X+tZ)-\mathcal{F}(X)}{t} = \langle G, Z\rangle_e$. Let $\mathcal{E}$ be an Euclidean space. A Riemannian manifold $\mathcal{M}$ is a manifold whose tangent spaces $T_x\mathcal{M}$ at $x \in \mathcal{M}$ are endowed with a smooth local inner product (the Riemannian metric) $g(\eta_x, \xi_x) = \langle \eta_x, \xi_x\rangle_x$, where $\eta_x, \xi_x \in T_x\mathcal{M}$. Let $f : \mathcal{M} \to \mathbb{R}$ be a differentiable scalar field on a Riemannian manifold $\mathcal{M}$. The Riemannian gradient $\mathit{gradf}(x)$ of f at x is defined as the unique element of $T_x\mathcal{M}$ that satisfies $\langle \mathit{gradf}(x), \xi\rangle_x = \mathcal{D}f(X)[\xi], \quad \forall \xi \in T_x\mathcal{M}$. Now, let $f : \mathcal{E} \to \mathbb{R}$ be a differentiable objective function that we want to minimize on a Riemannian submanifold, and let $\nabla f(x)$ be the Euclidean gradient of f at $x \in \mathcal{E}$. We can build a Riemannian gradient for f at $x \in \mathcal{M}$ by projecting the Euclidean gradient $\nabla f(x)$ onto $T_x\mathcal{M}$, that is $\mathit{gradf}(x) = P_{T_x\mathcal{M}}[\nabla f(x)]$.

Two other concepts regarded to manifolds are of our interest: A *retraction* R is a smooth map from the tangent bundle $T\mathcal{M} := \bigcup_{x\in\mathcal{M}} T_x\mathcal{M}$ onto $\mathcal{M}$ which allows us to go back onto the manifold. A *vector transport* $\mathcal{T}_{\eta_x}(\xi_x)$ facilitates the movement from a tangent manifold to another. Both of the objects belong to tangent subspaces of the manifold, which facilitates the operations related to algorithmic procedures. In [2] are provided formal definitions, and studied their properties. Since $T_x\mathcal{M}$ is an affine set on $\mathcal{E}$, then we can define $\mathcal{T}_{\eta_x}(\xi_x) = P_{T_{R_x(\eta_x)}}(\xi_x)$.

2.1 Riemannian conjugate gradient methods

Now we provide a brief review of the Riemannian conjugate gradient methods (RCG). A complete description of these methods appear in [2,4]. Manifold-constrained optimization refers to a class of problems of the form, $\min f(x) \quad s.t. \quad x \in \mathcal{M}$, where $f : \mathcal{M} \to \mathbb{R}$ is a given smooth real-valued function, and $\mathcal{M}$ is a Riemannian manifold, that is a manifold equipped with an inner product on the associated tangent spaces. A general iteration of a RCG update the iterate by the following scheme, starting in $x_0 \in \mathcal{M}$ with $\eta_0 = -\mathit{gradf}(x_0)$, $x_{k+1} = R_{x_k}(\tau_k \eta_k)$, where $R_{x_k}() : T_{x_k}\mathcal{M} \to \mathcal{M}$ is a retraction, $\tau_k > 0$ is the step-size and $\eta_k \in T_{x_k}\mathcal{M}$ is given by the recursive formula, $\eta_{k+1} = -gradf(x_{k+1}) + \beta_{k+1}\mathcal{T}_{\tau_k\eta_k}(\eta_k)$, a linear combination of the Riemannian Gradient and a transported (to the current iterate tangent space) version of the previous search direction. There are several expressions to update the parameter β_{k+1} of the equation above. Some of the most popular are, the β of Fletcher-Reeves $\beta_{k+1}^{FR} = \frac{\langle gradf(x_{k+1}), gradf(x_{k+1})\rangle_{x_{k+1}}}{\langle gradf(x_k), gradf(x_k)\rangle_{x_k}}$, and the β of Polak-Ribière, $\beta_{k+1}^{PR} = \frac{\langle gradf(x_{k+1}), gradf(x_{k+1}) - \mathcal{T}_{\tau_k\eta_k}(gradf(x_k))\rangle_{x_{k+1}}}{\langle gradf(x_k), gradf(x_k)\rangle_{x_k}}$. It has been proposed RCG algorithms to solve particular cases of (1) (see [1, 2, 4, 14]). Since vector transportation between different tangent spaces are needed in the formulas, then in each iteration an amount of computational effort should be dedicated to this task. If $\mathcal{M}$ is an embedded submanifold of a Euclidean space $\mathcal{E}$, with an associated retraction $R_x(\cdot)$, then $T_y\mathcal{M} \subset \mathcal{E}$ for all $y \in \mathcal{M}$, and so, a vector transport can be defined by $\mathcal{T}_{\eta_x}(\xi_x) = P_{R_x(\eta_x)}(\xi_x)$,

The use of vector transports makes the RCG algorithms to perform more matrix multiplications per iteration than the *Riemannian steepest descent.* In [14], Zhu introduce two novel vector transports associated with the Cayley transform retraction for Stiefel-constrained optimization. However its two vector transports require to invert a matrix, which can be computationally expensive.

2.2 Tools on Stiefel manifold

In this subsection, we briefly review some tools about the Stiefel manifold $St(n,p)$, which is the feasible set of the problem (1), as discussed in [2, 4]. It is well known that the Stiefel manifold $St(n,p)$ is an embedded submanifold of $\mathbb{R}^{n\times p}$. The tangent space of $St(n,p)$ at $X \in St(n,p)$ is $T_XSt(n,p) = \{Z \in \mathbb{R}^{n\times p} : Z^\top X + X^\top Z = 0\}$. It is easy to verify that the tangent space coincide with $\Omega_X = \{Z \in \mathbb{R}^{n\times p} : Z = WX, \text{ for some skew-symmetric matrix } W \in \mathbb{R}^{p\times p}\}$. The canonical projection onto $T_XSt(n,p)$ is

$$Pc_{T_XSt(n,p)}[\xi] = \xi - X\xi^TX. \tag{2}$$

Now we consider the function $\mathcal{F}$ defined in problem (1). To calculate the Riemannian gradient of $\mathcal{F}$ associated to the Stiefel manifold, we need to project the (Euclidean) gradient onto the tangent space of $St(n,p)$ at the current point. We first calculate the Euclidean gradient $\nabla\mathcal{F}$. Observe first that for each $l = 1,\ldots,N$, $||diag(X^TA_lX)||_F^2 = tr(diag(X^TA_lX)^2) = tr(X^TA_lXdiagX^TA_lX)$. The matrix of partial derivatives for this expression is given by $4A_lXdiag(X^TA_lX)$. The last expression leads us to our Euclidean gradient: $\nabla\mathcal{F}(X) = -4\sum_{l=1}^N A_lXdiag(X^TA_lX)$. The Riemannian gradient of $\mathcal{F}$ in $St(n,p)$ is deduced by using the projection in (2): $grad\mathcal{F}(X) = \nabla\mathcal{F}(X) - X\nabla\mathcal{F}(X)^TX$. This Riemannian gradient is actively used in our proposed algorithm.

3 A Riemannian Conjugate Gradient Algorithm

In this section, we describe the algorithm formulated in [7], specially adapted to our problem. The merit of such algorithm is focused on the savings due to our vector transport. Our updating formula, differently of the general RCG, does not need to use parallel transport explicitly. This novelty is what differentiates our proposal from the standard RCG methods discussed in section 3. Let X_k be the current iterate, and define the new iterate $Y_k(\overline{\tau})$ as a point on the curve $Y_k(\tau) = Qf(X_k + \tau Z_k)$, where $\tau > 0$ is the step-size, $Z_k = W_k X_k$, and $Qf(\cdot)$ denote the mapping that sends a matrix to the Q factor of its QR decomposition such that the R factor is an upper triangular $p \times p$ matrix with strictly positive diagonal elements. It is known that the $Qf(\cdot)$ is a retraction on Stiefel manifolds (see [2]). The updating scheme for Z_k involves calculating W_k recursively, starting in $W_0 = -A_0$, by $W_k = -A_k + \beta_k W_{k-1}$, where $A_k := A(X_k) = G_k X_k^\top - X_k G_k^\top$, is the gradient operator as defined in section above. The parameter is updated by

$$\beta_k = \begin{cases} \beta_k^{FR-PR} & \text{if } \mathcal{DF}(X_k)[Z_k] < 0 \\ 0 & \text{in other case,} \end{cases} \tag{3}$$

where

$$\beta_k^{FR-PR} = \begin{cases} -\beta_k^{FR} & \text{if } \beta_k^{PR} < -\beta_k^{FR} \\ \beta_k^{PR} & \text{if } |\beta_k^{PR}| < \beta_k^{FR} \\ \beta_k^{FR} & \text{if } \beta_k^{PR} > \beta_k^{FR}, \end{cases}$$

and $\beta_k^{FR}, \beta_k^{PR}$ are the well-known formulas of Fletcher-Reeves and Polak-Ribière [5], respectively. The choice for this updating formula yields a descent direction algorithm, as proven in [7].

Now we present a new transport for Stiefel manifolds. Consider $X \in St(n,p)$, and $\xi_X, \eta_X, \in T_X St(n,p)$. Given a retraction $R_X(\cdot)$ on Stiefel manifolds, we define our vector transport as,

$$\mathcal{T}_{\eta_X}(\xi_X) := W R_X(\eta_X). \tag{4}$$

In [7] we demonstrate that in fact $\mathcal{T}_{\eta_X}(\xi_X)$ is a vector transport on $St(n,p)$. Observe that the calculation of our vector transport (4), is much simpler than those existing in the literature. This only requires compute a matrix multiplication, hence this vector transport can generate more efficient Riemannian conjugate gradient algorithms. Using this particular transport vector, it can be shown that the direction search given by Z_k is indeed a RCG direction. This connects our proposal with the standard Riemannian conjugate methods presented in section 2.1. In fact, it is easy to show that $Z_{k+1} = -\nabla_c \mathcal{F}(X_{k+1}) + \beta_{k+1} W_k X_{k+1}$, and by using our vector transport $Z_{k+1} = -\nabla_c \mathcal{F}(X_{k+1}) + \beta_{k+1} \mathcal{T}_{\tau_k Z_k}(Z_k)$, therefore, the direction search that we propose, corresponds to a Riemannian conjugate gradient direction, for which, the calculation of the vector transport is implicit.

Typically, monotone line search algorithms construct a sequence $\{X_k\}$ such that the sequence $\{\mathcal{F}(X_k)\}$ of objective values is monotone decreasing. Generally these methods calculate the step size τ oriented by minimum values for $\{\mathcal{F}(Y_k(\tau))\}$. Approximated minimal solutions are sufficient to ensure the convergence properties. In particular we use the so called Armijo Rule [5]. Now we propose our algorithm.

Algorithm 1 Quasi Conjugated Gradient Method with Armijo's Inexact Line Search

```
Require: $X_0 \in \mathsf{St}(n,p)$, $\rho, \epsilon, \delta \in (0,1)$, $G_0 = \mathcal{DF}(X_0)$ $A_0 = G_0 X_0^\top - X_0 G_0^\top$, $W_0 = -A_0$,
    $Z_0 = W_0 X_0$, $k = 0$.
Ensure: $X^*$ an $\epsilon$-stationary point.
 1: while $||A_k||_F > \epsilon$ do
 2:     Step size selection: take an initial step-size $\tau = \mu_0$ where $\mu_0 > 0$.
 3:     while $\mathcal{F}(Y_k(\tau)) > \mathcal{F}(X_k) + \rho\tau\mathcal{DF}(X_k)[Z_k]$ do
 4:         $\tau = \delta\tau$,
 5:     end while
 6:     $X_{k+1} = Y_k(\tau) := Qf(X_k + \tau Z_k)$,
 7:     Calculate, $G_{k+1} = \mathcal{DF}(X_{k+1})$ and $A_{k+1} = G_{k+1}X_{k+1}^\top - X_{k+1}G_{k+1}^\top$,
 8:     Update $W_{k+1} = -A_{k+1} + \beta_{k+1}W_k$ with $\beta_{k+1}$ as in (5),
 9:     $Z_{k+1} = W_{k+1}X_{k+1}$,
10:     $k = k + 1$,
11: end while
12: $X^* = X_k$.
```

The bottleneck of the previous algorithm is found in step 3, where several QR factorization must be calculated. In order to solve efficiently the QR factorization of the matrix $M = X_k + \tau Z_k$ we use Cholesky factorization. The Riemannian conjugate gradient methods require that the step size τ satisfy the strong Wolfe conditions [5], (whose Riemannian version appear in [8, 14]) in each iteration, and furthermore, the vector transport needs to satisfy the non-expansive Ring-Wirth condition [8], to guarantee global convergence. By means of a direct calculation, it can be verified that the vector transport given in (4) satisfies this condition for the case of $n = p$, nevertheless, when $p < n$, this condition is not necessarily fulfilled. There are ways to rescale any vector transport to obtain another one that satisfies the Ring-Wirth condition [10]. However, if we set the algorithm to periodically restart the direction at $\beta_k = 0$, then we can guarantee that all these Z_k are descent directions, and then the procedure becomes a line search method on matrix manifold. Observe that in our algorithm, we use this restart and in this way, we have that $\{Z_k\}$ gradient-related sequence that is contained in the tangent bundle of the Stiefel manifold. In addition, the mapping $R_{X_k}(\eta_k) := Qf(X_k + \eta_k)$ is a retraction (see [2]). Therefore, the results of global convergence that appear in [2] regarding line-search methods on manifolds using retractions apply directly to Algorithm 1, which allows us to conclude that $\lim_{k\to\infty} ||\nabla_e \mathcal{F}(X_k)||_F = 0$.

4 Numerical Experiments

In this section, we perform some numerical experiments to investigate the efficiency of the proposed method. All algorithms were implemented using MATLAB 7.10 on a intel(R) CORE(TM) i7-4770, CPU 3.40 GHz with 500 Gb of HD and 16 Gb of Ram. We present a comparative study of our algorithm versus other state-of-the-art algorithms for several instances of the problems *Joint diagonalization problem*. In the following tables, we show

Table 1: Numerical results for Experiment 1 with $N = 10$

	Ex.5 $n = 30$, $p = 10$						Ex.6 $n = 50$, $p = 30$					
Methods	**Nitr**	**Time**	**NrmG**	**Fval**	**Feasi**	**Error**	**Nitr**	**Time**	**NrmG**	**Fval**	**Feasi**	**Error**
QR_CG	151.3	0.038	9.14e-6	-69.3	7.27e-16	0.0203	248.7	0.179	9.25e-06	-1.56e+2	1.44e-15	0.0353
OptStiefel	97.6	0.025	8.84e-6	-69.3	7.96e-16	0.0203	137.6	0.103	9.09e-06	-1.56e+2	3.31e-14	0.0353
Grad_retrac	97	0.031	8.81e-6	-69.3	1.84e-14	0.0202	139.2	0.126	9.02e-06	-1.56e+2	1.83e-14	0.0353
	Ex.7 $n = 50$, $p = 50$						Ex.8 $n = 100$, $p = 40$					
QR_CG	258.5	0.25	9.29e-6	-1.67e+e2	2.23e-15	0.2475	395.2	0.497	9.38e-6	-2.6e+e2	1.76e-15	0.1046
OptStiefel	145.6	0.141	9.09e-6	-1.67e+e2	4.95e-14	0.2475	206.3	0.28	9.26e-6	-2.6e+e2	2.48e-15	0.1046
Grad_retrac	1.46e+2	0.187	9.05e-6	-1.67e+e2	2.12e-14	0.2475	1.99e+2	0.317	9.32e-6	-2.6e+e2	2.27e-14	0.1046

Table 2: Numerical results for Experiment 2 with $N = 10$

	Ex.1 $n = 30$, $p = 10$					Ex.2 $n = 50$, $p = 30$				
Methods	**Nitr**	**Time**	**NrmG**	**Fval**	**Feasi**	**Nitr**	**Time**	**NrmG**	**Fval**	**Feasi**
QR_CG	208.7	0.052	3.83e-5	-2.10e+5	6.58e-16	518.2	0.328	2.90e-4	-1.40e+6	1.35e-15
OptStiefel	161.1	0.036	1.22e-5	-2.11e+5	8.02e-16	430.3	0.268	7.94e-6	-1.40e+6	4.90e-14
Grad_retrac	190.9	0.076	1.50e-5	-2.11e+5	2.13e-14	694.2	0.97	2.68e-4	-1.40e+6	9.23e-15
	Ex.3 $n = 50$, $p = 50$					Ex.4 $n = 100$, $p = 40$				
QR_CG	765.9	0.61	5.00e-4	-1.74e+6	2.05e-15	842.6	0.956	1.06e-2	-8.49e+6	1.57e-15
OptStiefel	597.5	0.456	8.14e-6	-1.74e+6	6.11e-14	852.2	0.979	2.06e-1	-8.49e+6	2.44e-15
Grad_retrac	2443.5	5.75	5.00e-4	-1.74e+6	9.46e-15	1539.3	4.663	3.30e-3	-8.49e+6	6.74e-15

the average of each values to be compared in a total of 100 executions. In addition, the maximum number of iterations is set at 8000, we use a tolerance for the gradient norm of $\epsilon = $ 1e-5 and we take the following values, $xtol =$ 1e-14 and $ftol =$ 1e-14 as tolerances for the other stopping criteria. First, we perform an experiment varying p and n (this experiment was taken from [9]). In this case, we build the matrices A_1, A_2,..., A_N as follows. We randomly generate $N = 10$ $n \times n$ diagonal matrices Λ_1, Λ_2,..., Λ_N, and also choose randomly the $n \times n$ orthogonal matrix P, where the diagonal entries $\lambda_1^{(l)}$, $\lambda_2^{(l)}$,..., $\lambda_n^{(l)}$ of each Λ_l are positive and in descending order. We then put A_1, A_2,..., A_N as $A_l = P\Lambda_l P^\top$ for all $l = 1, 2, ..., N$. Observe that $X^* = PI_{n,p}$ is an optimal solution to the problem (1). As a starting point, we compute an approximate solution $X_0 = Qf(X^* + X^{rand})$, where X^{rand} is a randomly chosen $n \times p$ matrix such that $\max_{i,j}\{|x_{ij}|\} \leq 0.01$, where x_{ij} denotes the entry (i, j) of matrix X^{rand}. The numerical results, associated with this experiment for several values of n and p, are contained in Table 1. From these table we can observe that, all the methods compared show a similar performance in terms of CPU time, but our method performs more iterations. However, the three algorithms get good solutions.

Next, we carry out another experiment, in order to test the methods starting from a initial point randomly generated and not necessarily close to a solution. We set $N = 10$ and build 100 JDPs generating each of the N matrices $A_1, A_2, \ldots, A_N$ in the following way, first we generate randomly a matrix $\bar{A} \in \mathbb{R}^{n\times n}$ with all its entries following a Gaussian distribution, then we build A_l for $A_l = \bar{A}^\top \bar{A}$ and thus ensure that every A_l is symmetric. On the other hand, we generate at the initial point X_0 randomly on the Stiefel manifold. Table 2 shows the results obtained by the methods for different problem sizes. Based on the results summarized in Table 2, we observe that **OptStiefel** and our **QR_CG** are much more efficient methods that **Grad_retrac** in terms of CPU time. furthermore, our method presents competitive results compared to those obtained by the other two methods.

References

[1] T. Abrudan, J. Eriksson, and V. Koivunen. Conjugate gradient algorithm for optimization under unitary matrix constraint, *Signal Processing*, Elsevier, 89:1704–1714, 2009.

[2] P.A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, Princeton, NJ, 2008.

[3] B. Afsari, and P. Krishnaprasad. Some gradient based joint diagonalization methods for ICA, *ICA*, Springer, 3195:437–444, 2004.

[4] A. Edelman, T. Arias, and S. T. Smith, The geometry of algorithms with orthogonality constraints, *SIAM journal on Matrix Analysis and Applications*, SIAM, 20:303–353, 1998.

[5] J. Nocedal, and S. J. Wright, *Numerical optimization*, 2nd, Springer, 2006.

[6] H. Oviedo, H. Lara Urdaneta and O. Dalmau, A non-monotone linear search algorithm with mixed direction on Stiefel manifold, *Optimization Methods and Software*, Taylor and Francis, 1–21, 2018, DOI: 10.1080/10556788.2017.1415337.

[7] H. Oviedo and H. Lara Urdaneta. A Riemannian conjugate gradient algorithm with implicit vector transport for optimization in the Stiefel manifold. Technical report. UFSC-Blumenau, CIMAT, 2018.

[8] W. Ring, and B. Wirth, Optimization methods on Riemannian manifolds and their application to shape space, *SIAM Journal on Optimization*, SIAM, 22:596–627, 2012.

[9] H. Sato, Riemannian Newton's method for joint diagonalization on the Stiefel manifold with application to ICA, arXiv preprint, arXiv:1403.8064, 2014.

[10] H. Sato, and T. Iwai, A new, globally convergent Riemannian conjugate gradient method, *Optimization*, Taylor & Francis, 64:1011–1031, 2015.

[11] F. J. Theis, T.P Cason, and P. A. Absil, Soft Dimension Reduction for ICA by Joint Diagonalization on the Stiefel Manifold., *ICA*, Springer, 354–361, 2009.

[12] M. Wax, and J. J. Sheinvald, A least-squares approach to joint diagonalization, *IEEE Signal Processing Letters*, IEEE, 4:52-53 1997.

[13] Z. Wen, and W. Yin, A feasible method for optimization with orthogonality constraints, *Mathematical Programming*, Springer, 142:397–434, 2013.

[14] X. Zhu, A Riemannian conjugate gradient method for optimization on the Stiefel manifold, *Computational Optimization and Applications*, Springer, 67:73–110, 2017.