Gradient based block coordinate descent algorithms for joint approximate diagonalization of matrices

Jianze Li, Konstantin Usevich, Pierre Comon

GRADIENT BASED BLOCK COORDINATE DESCENT ALGORITHMS FOR JOINT APPROXIMATE DIAGONALIZATION OF MATRICES

JIANZE LI[†], KONSTANTIN USEVICH[‡], AND PIERRE COMON[♮]

Abstract. In this paper, we propose a gradient based block coordinate descent (BCD-G) framework to solve the joint approximate diagonalization of matrices defined on the product of the complex Stiefel manifold and the special linear group. Instead of the cyclic fashion, we choose the block for optimization in a way based on the Riemannian gradient. To update the first block variable in the complex Stiefel manifold, we use the well-known line search descent method. To update the second block variable in the special linear group, based on four different kinds of elementary transformations, we construct three classes: GLU, GQU and GU, and then get three BCD-G algorithms: BCD-GLU, BCD-GQU and BCD-GU. We establish the global convergence and weak convergence of these three algorithms using the Łojasiewicz gradient inequality under the assumption that the iterates are bounded. We also propose a gradient based Jacobi-type framework to solve the joint approximate diagonalization of matrices defined on the special linear group. Similar as in the BCD-G case, using the GLU and GQU classes of elementary transformations, we focus on the Jacobi-GLU and Jacobi-GQU algorithms, and establish their global convergence and weak convergence as well. All the algorithms and convergence results in this paper also apply to the real case.

1. Introduction

1.1. **Problem formulation.** Let $1 \leq m \leq n$. For a matrix $\boldsymbol{X} \in \mathbb{C}^{n\times m}$, we denote by $\boldsymbol{X}^{\mathsf{T}}$, $\boldsymbol{X}^*$ and $\boldsymbol{X}^{\mathsf{H}}$ its *transpose, conjugate* and *conjugate transpose,* respectively. We shall also use $(\cdot)^{\blacklozenge}$ to denote either $(\cdot)^{\mathsf{T}}$ or $(\cdot)^{\mathsf{H}}$. Let $\{\boldsymbol{A}^{(\ell)}\}_{1\leq\ell\leq \mathrm{L}} \subseteq \mathbb{C}^{n\times m}$ be a set of complex matrices, and $\mu_\ell \in \mathbb{R}^+$ for $1 \leq \ell \leq L$. It is well-known that the *blind source separation* (BSS) [16, 17, 35, 42] can be formulated as finding a full column rank matrix $\boldsymbol{Z} \in \mathbb{C}^{n\times m}$ to make the matrices $\boldsymbol{W}^{(\ell)} = \boldsymbol{Z}^{\blacklozenge}\boldsymbol{A}^{(\ell)}\boldsymbol{Z} \in \mathbb{C}^{m\times m}$ simultaneously as diagonal as possible. This is to solve the *joint approximate diagonalization of matrices* (JADM) problem, which consists in minimizing the cost function

$$f(\boldsymbol{Z}) = \sum_{\ell=1}^{L} \mu_\ell \| \operatorname{offdiag}\{\boldsymbol{W}^{(\ell)}\}\|^2, \tag{1}$$

2010 *Mathematics Subject Classification.* 49M30, 65F99, 90C30, 15A23.

Key words and phrases. blind source separation, joint approximate diagonalization of matrices, block coordinate descent, Jacobi-G algorithm, convergence analysis, manifold optimization.

† Shenzhen Research Institute of Big Data, The Chinese University of Hong Kong, Shenzhen, China (lijianze@gmail.com).

‡ Université de Lorraine, CNRS, CRAN, Nancy, France (konstantin.usevich@univ-lorraine.fr).

♮ Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-Lab, France (pierre.comon@gipsa-lab.fr).

This work was supported in part by the National Natural Science Foundation of China (No. 11601371) and the GuangDong Basic and Applied Basic Research Foundation (No. 2021A1515010232).

where offdiag$\{\cdot\}$ is the *zero diagonal* operator, which sets all the diagonal elements of a square matrix in $\mathbb{C}^{m\times m}$ to zero.

Note that, if $\boldsymbol{Z} \in \mathbb{C}^{n\times m}$ is of full column rank and $\lambda \in \mathbb{C}$ is nonzero, then $\lambda\boldsymbol{Z}$ is still of full column rank and the function f in (1) satisfies $f(\lambda\boldsymbol{Z}) = |\lambda|^2 f(\boldsymbol{Z})$. Therefore, the optimization problem of minimizing f on the set of full column rank matrices is not well defined. To tackle this issue, one approach is to introduce new cost functions. For example, several cost functions with the scale and permutation invariance property are introduced in [4, 48]. In this paper, we still use the function (1), while restricting the variable $\boldsymbol{Z}$ in a smaller set $\mathbf{RSL}(m, n, \mathbb{C})$, which will be defined in (2).

Let $\mathbf{GL}_m(\mathbb{C}) \stackrel{\text{def}}{=} \{\boldsymbol{X} \in \mathbb{C}^{m\times m}, \det(\boldsymbol{X}) \neq 0\}$ be the *general linear group*, and $\mathbf{SL}_m(\mathbb{C}) \stackrel{\text{def}}{=} \{\boldsymbol{X} \in \mathbf{GL}_m(\mathbb{C}), \det(\boldsymbol{X}) = 1\}$ be the *special linear group*. We define the *rectangular special linear set* as

$$\mathbf{RSL}(m, n, \mathbb{C}) \stackrel{\text{def}}{=} \{\boldsymbol{X} \in \mathbb{C}^{n\times m}, \boldsymbol{X}^{\mathsf{H}}\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})\}. \tag{2}$$

In this paper, we mainly study the JADM problem on $\mathbf{RSL}(m, n, \mathbb{C})$, which is to minimize the cost function

$$f(\boldsymbol{Z}) = \sum_{\ell=1}^{L} \mu_\ell \| \operatorname{offdiag}\{\boldsymbol{W}^{(\ell)}\}\|^2, \tag{3}$$

where $\boldsymbol{Z} \in \mathbf{RSL}(m, n, \mathbb{C})$.

1.2. Two equivalent reformulations.

1.2.1. *First reformulation on* $\mathbf{St}(m, n, \mathbb{C}) \times \mathbf{SL}_m(\mathbb{C})$. Let $\mathbf{St}(m, n, \mathbb{C}) \stackrel{\text{def}}{=} \{\boldsymbol{Y} \in \mathbb{C}^{n\times m}, \boldsymbol{Y}^{\mathsf{H}}\boldsymbol{Y} = \boldsymbol{I}_m\}$ be the *complex Stiefel manifold*. We have the following relationship[1]:

$$\mathbf{RSL}(m, n, \mathbb{C}) = \mathbf{St}(m, n, \mathbb{C}) \times \mathbf{SL}_m(\mathbb{C}). \tag{4}$$

By equation (4), problem (3) is equivalent to minimizing the cost function

$$f : \mathbf{St}(m, n, \mathbb{C}) \times \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}^+, \quad (\boldsymbol{Y}, \boldsymbol{X}) \mapsto \sum_{\ell=1}^{L} \mu_\ell \| \operatorname{offdiag}\{\boldsymbol{W}^{(\ell)}\}\|^2, \tag{5}$$

where $\boldsymbol{W}^{(\ell)} = (\boldsymbol{Y}\boldsymbol{X})^{\blacklozenge}\boldsymbol{A}^{(\ell)}(\boldsymbol{Y}\boldsymbol{X}) \in \mathbb{C}^{m\times m}$.

1.2.2. *Second reformulation on* $\mathbf{SL}_m(\mathbb{C})$. In problem (5), it is often possible to find a solution $\boldsymbol{Y}_* \in \mathbf{St}(m, n, \mathbb{C})$ in advance by some other method, *e.g.* PCA [15, 16, 17]. This well-known procedure allows to reduce the space dimension by projection onto the dominant subspace, and at the same time to reduce the noise level. Let $\boldsymbol{B}^{(\ell)} = \boldsymbol{Y}_*^{\blacklozenge}\boldsymbol{A}^{(\ell)}\boldsymbol{Y}_*$ for $1 \leq \ell \leq L$. Then the cost function (5) becomes

$$g : \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}^+, \quad \boldsymbol{X} \mapsto g(\boldsymbol{X}) = \sum_{\ell=1}^{L} \mu_\ell \| \operatorname{offdiag}\{\boldsymbol{W}^{(\ell)}\}\|^2, \tag{6}$$

[1]This equation means that a complex matrix $\boldsymbol{Z} \in \mathbf{RSL}(m, n, \mathbb{C})$ if and only if there exist $\boldsymbol{Y} \in \mathbf{St}(m, n, \mathbb{C})$ and $\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})$ such that $\boldsymbol{Z} = \boldsymbol{Y}\boldsymbol{X}$. It is the same case with equations (11), (12), (13), (15) and (16).

where $\boldsymbol{W}^{(\ell)} = \boldsymbol{X}^{\blacklozenge}\boldsymbol{B}^{(\ell)}\boldsymbol{X} \in \mathbb{C}^{m\times m}$. A complex matrix $\boldsymbol{B} \in \mathbb{C}^{m\times m}$ is called *Hermitian* if $\boldsymbol{B}^{\mathsf{H}} = \boldsymbol{B}$. It is called *complex symmetric* if $\boldsymbol{B}^{\mathsf{T}} = \boldsymbol{B}$. Problem (6) has the following well-known special cases:

- *joint approximate diagonalization of Hermitian matrices* (JADM-H)[41, 35]: $(\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{H}}$, $\boldsymbol{B}^{(\ell)}$ is Hermitian and $\mu_\ell = 1$ for $1 \le \ell \le L$;
- *joint approximate diagonalization of complex symmetric matrices* (JADM-CS) [35]: $(\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{T}}$, $\boldsymbol{B}^{(\ell)}$ is complex symmetric and $\mu_\ell = 1$ for $1 \le \ell \le L$;
- *joint approximate diagonalization of real symmetric matrices* (JADM-RS) [4, 5]: on the real field $\mathbb{R}$, $(\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{T}}$, $\boldsymbol{B}^{(\ell)}$ is real symmetric and $\mu_\ell = 1$ for $1 \le \ell \le L$.

These three problems have been widely used in the BSS and *Independent component analysis* (ICA) [13, 17, 4, 6]. Compared with the joint approximate diagonalization by orthogonal transformations [11, 12, 16, 24, 27, 25, 26, 44], the non-orthogonal joint diagonalizer in (6) does not require prewhitening and may less suffer from noise. However, since $\mathbf{SL}_m(\mathbb{C})$ is not compact, solving this problem is much more difficult.

To solve the JADM-RS problem, Jacobi-type algorithms were introduced based on the LU and QR decompositions in [5], and based on the Givens transformations, hyperbolic transformations and diagonal transformations in [42, Eq. (9)]. To solve the JADM-H problem, Jacobi-type algorithms were proposed based on the LU decomposition in [35, 36], and based on the QL decomposition in [41]. To solve the JADM-CS problem, a Jacobi-type algorithm was proposed based on the LU decomposition in [34, 35]. However, to our knowledge, there was no theoretical result about the convergence of these Jacobi-type algorithms in the literature.

1.3. **Block coordinate descent.** Suppose that $\{\mathcal{M}_i\}_{1\le i\le d}$ is a set of smooth manifolds. To minimize the following smooth function

$$f:\ \mathcal{M}_1 \times \mathcal{M}_2 \times \cdots \times \mathcal{M}_d \longrightarrow \mathbb{R}^+, \tag{7}$$

a popular approach is the *block coordinate descent* (BCD) [8, 31, 32, 46, 47, 28]. In this method, only one block variable is updated at each iteration, while other block variables are fixed. Then problem (7) is decomposed to a sequence of lower-dimensional optimization problems, similar as the subproblems in Jacobi-type algorithms. In BCD algorithm, there are different ways to choose blocks for optimization, including the *essentially cyclic, cyclic*, *random* fashions [46, 47] and the so-called *maximum block improvement* (MBI) method [14, 29].

1.4. **BCD-G algorithm for the first reformulation on $\mathbf{St}(m,n,\mathbb{C}) \times \mathbf{SL}_m(\mathbb{C})$.** Suppose that

$$f : \mathbf{St}(m,n,\mathbb{C}) \times \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}^+ \tag{8}$$

is an abstract smooth function, which includes the cost function (5) as a special case. For $\boldsymbol{\omega} = (\boldsymbol{Y},\boldsymbol{X}) \in \mathbf{St}(m,n,\mathbb{C}) \times \mathbf{SL}_m(\mathbb{C})$, we define

$$f_{1,\boldsymbol{X}} : \boldsymbol{Y} \mapsto f(\boldsymbol{Y},\boldsymbol{X}), \quad f_{2,\boldsymbol{Y}} : \boldsymbol{X} \mapsto f(\boldsymbol{Y},\boldsymbol{X}), \tag{9}$$

as the restricted functions. For simplicity, we denote the *Riemannian gradient*[2] $\operatorname{grad} f_1(\boldsymbol{\omega}) \overset{\mathsf{def}}{=} \operatorname{grad} f_{1,\boldsymbol{X}}(\boldsymbol{Y})$ and $\operatorname{grad} f_2(\boldsymbol{\omega}) \overset{\mathsf{def}}{=} \operatorname{grad} f_{2,\boldsymbol{Y}}(\boldsymbol{X})$. Let $\boldsymbol{\omega}_k \overset{\mathsf{def}}{=} (\boldsymbol{Y}_k,\boldsymbol{X}_k)$ for $k \ge 0$. Now we propose

[2]See [3, Section 3.6] for a detailed definition.

the following *gradient based block coordinate descent* (BCD-G) algorithm to minimize the cost function (8). It is easy to see that, in Algorithm 1, we can always choose $t_k = 1$ or 2 such that the inequality (10) is satisfied[3].

Algorithm 1: BCD-G algorithm

Input: A starting point $\boldsymbol{\omega}_0$, $0 < \upsilon < \frac{\sqrt{2}}{2}$.
Output: Sequence of iterates $\boldsymbol{\omega}_k$.
for $k = 1, 2, \cdots$, **do**
  Choose $t_k = 1$ or 2 such that

$$\| \operatorname{grad} f_{t_k}(\boldsymbol{\omega}_{k-1}) \| \geq \upsilon \| \operatorname{grad} f(\boldsymbol{\omega}_{k-1}) \|; \tag{10}$$

  if $t_k = 1$ **then**
    Update $\boldsymbol{U}_k$ using the line search descent method (cf. Equation (31) in Section 3);
    Set $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}$;
  else
    Set $\boldsymbol{U}_k = \boldsymbol{U}_{k-1}$;
    Update $\boldsymbol{X}_k$ using the elementary transformations (cf. Subalgorithm 1a to 1c in Section 4);
  end if
end for

Now we need to recall some Lie groups before introducing further details about Algorithm 1. A matrix $\boldsymbol{X} \in \mathbb{C}^{m \times m}$ is said to be *upper triangular* if $\mathrm{X}_{ij} = 0$ for $i > j$. Let $\mathbf{UT}_m(\mathbb{C}) \subseteq \mathbf{GL}_m(\mathbb{C})$ be the *upper triangular subgroup*, that is

$$\mathbf{UT}_m(\mathbb{C}) \stackrel{\text{def}}{=} \{\boldsymbol{X} \in \mathbf{GL}_m(\mathbb{C}), \boldsymbol{X} \text{ is upper triangular}\}.$$

Let $\mathbf{EUT}_m(\mathbb{C}) = \mathbf{UT}_m(\mathbb{C}) \cap \mathbf{SL}_m(\mathbb{C})$, *i.e.*, the set of upper triangular matrices with determinant equal to 1. Similarly, we let $\mathbf{LT}_m(\mathbb{C}) \subseteq \mathbb{C}^{m \times m}$ be the *lower triangular subgroup* and $\mathbf{ELT}_m(\mathbb{C}) = \mathbf{LT}_m(\mathbb{C}) \cap \mathbf{SL}_m(\mathbb{C})$. Let $\mathbf{U}_m \subseteq \mathbb{C}^{m \times m}$ be the *unitary group*, and $\mathbf{SU}_m \subseteq \mathbf{U}_m$ be the *special unitary group*.

In Algorithm 1, to update $\boldsymbol{U}_k$, we choose the *line search descent* method [3], which will be presented in Section 3. To update $\boldsymbol{X}_k$, using the *Givens plane, plane upper triangular, plane lower triangular* and *plane diagonal transformations*[4], which will be introduced in Section 4, we construct three classes of elementary transformations: GLU, GQU and GU.

- Any matrix $\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})$ has the *LU decomposition* [18] $\boldsymbol{X} = \boldsymbol{L}\boldsymbol{U}$ with $\boldsymbol{L} \in \mathbf{LT}_m(\mathbb{C})$ and $\boldsymbol{U} \in \mathbf{UT}_m(\mathbb{C})$. The first class GLU is inspired by the following relationship

$$\mathbf{RSL}(m, n, \mathbb{C}) = \mathbf{St}(m, n, \mathbb{C}) \times \mathbf{ELT}_m(\mathbb{C}) \times \mathbf{EUT}_m(\mathbb{C}), \tag{11}$$

[3]The inequality (10) can be seen as a block coordinate analogue of [20, Eq. (3.3)] and [24, Eq. (10)].
[4]The reason why we use plane diagonal transformations will be shown in Section 4

which follows from equation (4) and the LU decomposition, and it includes the plane lower triangular, plane upper triangular and plane diagonal transformations (Subalgorithm 1a). In this case, we call Algorithm 1 the *BCD-GLU* algorithm.

- Any matrix $\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})$ has the the *QU decomposition* $\boldsymbol{X} = \boldsymbol{QU}$ with $\boldsymbol{Q} \in \mathbf{SU}_m$ and $\boldsymbol{U} \in \mathbf{UT}_m(\mathbb{C})$. The second class GQU is inspired by the following relationship

$$\mathbf{RSL}(m, n, \mathbb{C}) = \mathbf{St}(m, n, \mathbb{C}) \times \mathbf{SU}_m \times \mathbf{EUT}_m(\mathbb{C}), \tag{12}$$

which follows from equation (4) and the QU decomposition, and it includes the Givens plane, plane upper triangular and plane diagonal transformations (Subalgorithm 1b). In this case, we call Algorithm 1 the *BCD-GQU* algorithm.

- Note that the equation (12) can be further written as

$$\mathbf{RSL}(m, n, \mathbb{C}) = \mathbf{St}(m, n, \mathbb{C}) \times \mathbf{SU}_m \times \mathbf{EUT}_m(\mathbb{C}) = \mathbf{St}(m, n, \mathbb{C}) \times \mathbf{EUT}_m(\mathbb{C}). \tag{13}$$

We construct the third class GU, which only includes the plane upper triangular and plane diagonal transformations (Subalgorithm 1c). In this case, we call Algorithm 1 the *BCD-GU* algorithm.

1.5. **Jacobi-G algorithm for the second reformulation on $\mathbf{SL}_m(\mathbb{C})$.** Suppose that

$$g : \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}^+ \tag{14}$$

is an abstract smooth function, which includes the cost function (6) as a special case. To minimize the cost function (14), using the Givens plane, plane upper triangular, plane lower triangular and plane diagonal transformations, which will be introduced in Section 4, we will propose two types of *gradient-based Jacobi-type algorithms*: Jacobi-GLU and Jacobi-GQU, which will be detailedly formulated in Section 7.3.1. Now we only show the motivations of these two algorithms. Jacobi-GLU is inspired by the LU decomposition

$$\mathbf{SL}_m(\mathbb{C}) = \mathbf{ELT}_m(\mathbb{C}) \times \mathbf{EUT}_m(\mathbb{C}), \tag{15}$$

and it uses the GLU class. Jacobi-GQU is inspired by the QU decomposition

$$\mathbf{SL}_m(\mathbb{C}) = \mathbf{SU}_m \times \mathbf{EUT}_m(\mathbb{C}), \tag{16}$$

and it uses the GQU class.

1.6. **Contributions.** The main contributions of this paper can be summarized as follows:

- To solve problem (5), we propose the BCD-G framework (Algorithm 1), which chooses the block for optimization in a way based on the Riemannian gradient. This is similar to the gradient-based way of choosing index pairs in Jacobi-G algorithms [20, 24, 44].
- To update the first block variable $\boldsymbol{Y}$, we adopt the well-known *line search descent* method. To update the second block variable $\boldsymbol{X}$, based on four kinds of elementary transformations, we construct three classes (GLU, GQU and GU), and then get three BCD-G algorithms: BCD-GLU, BCD-GQU and BCD-GU.

- We establish the *global convergence*[5] and *weak convergence*[6] of BCD-GLU, BCD-GQU and BCD-GU algorithms using the *Łojasiewicz gradient inequality*, under the assumption that the iterates are bounded, that is, there exists $\mathrm{M}_\omega > 0$ such that
$$\|\boldsymbol{\omega}_k\| \leq \mathrm{M}_\omega \tag{17}$$
always holds.
- To solve problem (6), we propose two types of gradient based Jacobi-type algorithms (Jacobi-GLU and Jacobi-GQU) on $\mathbf{SL}_m(\mathbb{C})$, which can be seen as non-orthogonal analogues of the Jacobi-G algorithm on orthogonal or unitary group [20, 24, 44].
- We establish the global convergence and weak convergence of Jacobi-GLU and Jacobi-GQU algorithms using the Łojasiewicz gradient inequality, under the assumption that the iterates are bounded, that is, there exists $\mathrm{M}_{\boldsymbol{X}} > 0$ such that
$$\|\boldsymbol{X}_k\| \leq \mathrm{M}_{\boldsymbol{X}} \tag{18}$$
always holds. To our knowledge, this is the first time that the theoretical convergence is established for the Jacobi-type algorithms on $\mathbf{SL}_m(\mathbb{C})$.
- These algorithms and convergence results are summarized in Table 1.

TABLE 1. A summary of the proposed algorithms

Model	Proposed algorithms	Position	Elementary functions	Global convergence	Weak convergence
First reformulation (5) on $\mathbf{St}(m,n,\mathbb{C}) \times \mathbf{SL}_m(\mathbb{C})$	BCD-GLU	Algorithm 1&Subalgorithm 1a	φ, ψ, ρ	Theorem 7.1	Theorem 7.2
	BCD-GQU	Algorithm 1&Subalgorithm 1b	h, φ, ρ		
	BCD-GU	Algorithm 1&Subalgorithm 1c	φ, ρ	Theorem 7.3	Theorem 7.4
Second reformulation (6) on $\mathbf{SL}_m(\mathbb{C})$	Jacobi-GLU	Algorithm 2	φ, ψ, ρ	Theorem 7.5	Theorem 7.6
	Jacobi-GQU	Algorithm 3	h, φ, ρ		

Remark 1.1. This paper is based on the complex matrices, complex Stiefel manifold $\mathbf{St}(m,n,\mathbb{C})$ and complex special linear group $\mathbf{SL}_m(\mathbb{C})$. In fact, all the algorithms and convergence results described in this paper also apply to the real case.

1.7. **Organization.** The paper is organized as follows. In Section 2, we recall the basics of first order geometries on the Stiefel manifold $\mathbf{St}(m,n,\mathbb{C})$ and special linear group $\mathbf{SL}_m(\mathbb{C})$, as well as the Lojasiewicz gradient inequality. In Section 3, we show the details of how to use line search descent method to update the first block variable in $\mathbf{St}(m,n,\mathbb{C})$. In Section 4, using four kinds of elementary transformations, we construct three classes of elementary transformations to update the second block variable in $\mathbf{SL}_m(\mathbb{C})$, which induce three BCD-G algorithms (BCD-GLU, BCD-GQU and BCD-GU) to solve problem (5). In Section 5 and Section 6, we present the details of these four kinds of elementary transformations. In Section 7, we prove our main results about the global and weak convergence of BCD-G algorithms. We also present two

[5]For any starting point, the iterates converge as a whole sequence.
[6]Every accumulation point is a stationary point.

gradient based Jacobi-type algorithms (Jacobi-GLU and Jacobi-GQU) to solve problem (6), and prove their global and weak convergence as well. In Section 8, some experiments are conducted to show the efficiency of the proposed algorithms. Section 9 concludes this paper with some final remarks and possible future work.

2. Geometries on $\mathbf{St}(m,n,\mathbb{C})$ and $\mathbf{SL}_m(\mathbb{C})$

2.1. **Notations.** Let $1 \le m \le n$. For a complex matrix $\boldsymbol{X} \in \mathbb{C}^{n\times m}$ and a complex number $z \in \mathbb{C}$, we write the real and imaginary parts as $\boldsymbol{X} = \boldsymbol{X}^{\Re} + i\boldsymbol{X}^{\Im}$ and $z = \Re(z) + i\Im(z)$, respectively. For $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{C}^{n\times m}$, we introduce the following real-valued inner product

$$\langle \boldsymbol{X}, \boldsymbol{Y} \rangle \stackrel{\mathsf{def}}{=} \langle \boldsymbol{X}^{\Re}, \boldsymbol{Y}^{\Re} \rangle + \langle \boldsymbol{X}^{\Im}, \boldsymbol{Y}^{\Im} \rangle = \Re\left(\mathrm{tr}(\boldsymbol{X}^{\mathsf{H}}\boldsymbol{Y})\right), \tag{19}$$

which makes $\mathbb{C}^{n\times m}$ a real Euclidean space of dimension $2nm$. Let $h : \mathbb{C}^{n\times m} \to \mathbb{R}$ be a differentiable function and $\boldsymbol{X} \in \mathbb{C}^{n\times m}$. We denote by $\frac{\partial h}{\partial \boldsymbol{X}^{\Re}}, \frac{\partial h}{\partial \boldsymbol{X}^{\Im}} \in \mathbb{R}^{n\times m}$ the matrix Euclidean derivatives of h with respect to real and imaginary parts of $\boldsymbol{X}$. The *Wirtinger derivatives* [1, 10, 22] are defined as

$$\frac{\partial h}{\partial \boldsymbol{X}^*} := \frac{1}{2}\left(\frac{\partial h}{\partial \boldsymbol{X}^{\Re}} + i\frac{\partial h}{\partial \boldsymbol{X}^{\Im}}\right), \quad \frac{\partial h}{\partial \boldsymbol{X}} := \frac{1}{2}\left(\frac{\partial h}{\partial \boldsymbol{X}^{\Re}} - i\frac{\partial h}{\partial \boldsymbol{X}^{\Im}}\right).$$

Then the Euclidean gradient of h with respect to the inner product (19) becomes

$$\nabla h(\boldsymbol{X}) = \frac{\partial h}{\partial \boldsymbol{X}^{\Re}} + i\frac{\partial h}{\partial \boldsymbol{X}^{\Im}} = 2\frac{\partial h}{\partial \boldsymbol{X}^*}. \tag{20}$$

For real matrices $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{n\times m}$, we see that (19) becomes the standard inner product, and (20) becomes the standard Euclidean gradient. We denote by $\mathbb{S}_{n-1} \subseteq \mathbb{R}^n$ the unit sphere.

2.2. **Riemannian gradient on $\mathbf{St}(m,n,\mathbb{C})$.** For $\boldsymbol{X} \in \mathbb{C}^{m\times m}$, we denote

$$\mathrm{sym}\,(\boldsymbol{X}) = \frac{1}{2}\left(\boldsymbol{X} + \boldsymbol{X}^{\mathsf{H}}\right), \quad \mathrm{skew}(\boldsymbol{X}) = \frac{1}{2}\left(\boldsymbol{X} - \boldsymbol{X}^{\mathsf{H}}\right).$$

Let $\mathbf{T}_{\boldsymbol{U}}\mathbf{St}(m,n,\mathbb{C})$ be the *tangent space* to $\mathbf{St}(m,n,\mathbb{C})$ at a point $\boldsymbol{U}$. Let $\boldsymbol{U}_\perp \in \mathbb{C}^{n\times(n-m)}$ be an orthogonal complement of $\boldsymbol{U}$, that is, $[\boldsymbol{U}, \boldsymbol{U}_\perp] \in \mathbb{C}^{n\times n}$ is a unitary matrix. By [33, Definition 6], we know that

$$\mathbf{T}_{\boldsymbol{U}}\mathbf{St}(m,n,\mathbb{C}) = \{\boldsymbol{Z} \in \mathbb{C}^{n\times m}, \boldsymbol{Z} = \boldsymbol{U}\boldsymbol{A} + \boldsymbol{U}_\perp\boldsymbol{B}, \boldsymbol{A} \in \mathbb{C}^{m\times m}, \boldsymbol{A}^{\mathsf{H}} + \boldsymbol{A} = 0, \boldsymbol{B} \in \mathbb{C}^{(n-m)\times m}\},$$

which is a $(2nm - m^2)$-dimensional vector space. The orthogonal projection of $\xi \in \mathbb{C}^{n\times m}$ to $\mathbf{T}_{\boldsymbol{U}}\mathbf{St}(m,n,\mathbb{C})$ is

$$\mathrm{Proj}_{\boldsymbol{U}}\xi = (\boldsymbol{I}_n - \boldsymbol{U}\boldsymbol{U}^{\mathsf{H}})\xi + \boldsymbol{U}\,\mathrm{skew}(\boldsymbol{U}^{\mathsf{H}}\xi) = \xi - \boldsymbol{U}\,\mathrm{sym}\left(\boldsymbol{U}^{\mathsf{H}}\xi\right). \tag{21}$$

We denote $\mathrm{Proj}_{\boldsymbol{U}}^{\perp}\xi \stackrel{\mathsf{def}}{=} \xi - \mathrm{Proj}_{\boldsymbol{U}}\xi$. Note that $\mathbf{St}(m,n,\mathbb{C})$ is an embedded submanifold of the Euclidean space $\mathbb{C}^{n\times m}$. By (21), we have the Riemannian gradient of h at $\boldsymbol{U}$ as:

$$\mathrm{grad}\, h(\boldsymbol{U}) = \mathrm{Proj}_{\boldsymbol{U}}\nabla h(\boldsymbol{U}) = \nabla h(\boldsymbol{U}) - \boldsymbol{U}\,\mathrm{sym}\left(\boldsymbol{U}^{\mathsf{H}}\nabla h(\boldsymbol{U})\right). \tag{22}$$

By [3, Example 5.4.2], the exponential map at $\boldsymbol{U}$ is defined as

$$\begin{aligned}\mathrm{Exp}_{\boldsymbol{U}} : \mathbf{T}_{\boldsymbol{U}}\mathrm{St}(m,n,\mathbb{C}) &\longrightarrow \mathrm{St}(m,n,\mathbb{C}) \\ \boldsymbol{Z} &\longmapsto [\boldsymbol{U},\boldsymbol{Z}]\exp\left(\begin{bmatrix}\boldsymbol{U}^H\boldsymbol{Z} & -\boldsymbol{Z}^{\mathsf{H}}\boldsymbol{Z} \\ \boldsymbol{I}_m & \boldsymbol{U}^H\boldsymbol{Z}\end{bmatrix}\right)\begin{bmatrix}\exp\left(-\boldsymbol{U}^H\boldsymbol{Z}\right) \\ \boldsymbol{0}_{m\times m}\end{bmatrix}.\end{aligned}$$

2.3. **Matrix groups.** Let $\mathfrak{sl}_m(\mathbb{C}) \stackrel{\mathsf{def}}{=} \{\boldsymbol{X} \in \mathbb{C}^{m\times m}, \mathrm{tr}(\boldsymbol{X}) = 0\}$. Then the tangent space to $\mathbf{SL}_m(\mathbb{C})$ at a point $\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})$ can be constructed [7, Eq. (3.7),(3.8)] by $\mathbf{T}_{\boldsymbol{X}}\mathbf{SL}_m(\mathbb{C}) = \{\boldsymbol{X}\Omega, \Omega \in \mathfrak{sl}_m(\mathbb{C})\}$. Let $\mathbf{SU}_m \subseteq \mathbb{C}^{m\times m}$ be the *special unitary group*. Let

$$\mathfrak{su}_m(\mathbb{C}) \stackrel{\mathsf{def}}{=} \{\boldsymbol{X} \in \mathbb{C}^{m\times m}, \boldsymbol{X}^{\mathsf{H}} = -\boldsymbol{X}, \mathrm{tr}(\boldsymbol{X}) = 0\}.$$

Then the tangent space to $\mathbf{SU}_m$ at a point $\boldsymbol{X} \in \mathbf{SU}_m$ can be constructed [7, Eq. (3.15)] by $\mathbf{T}_{\boldsymbol{X}}\mathbf{SU}_m = \{\boldsymbol{X}\Omega, \Omega \in \mathfrak{sl}_m(\mathbb{C})\}$.

A matrix $\boldsymbol{X} \in \mathbb{C}^{m\times m}$ is said to be *upper triangular* if $\mathrm{X}_{ij} = 0$ for $i > j$. An upper triangular matrix $\boldsymbol{X}$ is said to be *unipotent* if it satisfies $\mathrm{X}_{ii} = 1$ for $1 \le i \le m$. Let $\mathbf{UT}_m(\mathbb{C}) \subseteq \mathbf{GL}_m(\mathbb{C})$ be the *upper triangular subgroup*, that is

$$\mathbf{UT}_m(\mathbb{C}) \stackrel{\mathsf{def}}{=} \{\boldsymbol{X} \in \mathbf{GL}_m(\mathbb{C}), \boldsymbol{X} \text{ is upper triangular}\}.$$

Let $\mathbf{SUT}_m(\mathbb{C}) \subseteq \mathbf{GL}_m(\mathbb{C})$ be the *upper unipotent subgroup*, that is

$$\mathbf{SUT}_m(\mathbb{C}) \stackrel{\mathsf{def}}{=} \{\boldsymbol{X} \in \mathbf{GL}_m(\mathbb{C}), \boldsymbol{X} \text{ is upper triangular and unipotent}\}.$$

A matrix $\boldsymbol{X} \in \mathbb{C}^{m\times m}$ is said to be *strictly upper triangular* if $\mathrm{X}_{ij} = 0$ for $i \ge j$. Let $\mathfrak{sut}_m(\mathbb{C}) \subseteq \mathbb{C}^{m\times m}$ be the set of strictly upper triangular matrices. Then the tangent space to $\mathbf{SUT}_m(\mathbb{C})$ at a point $\boldsymbol{X} \in \mathbf{SUT}_m(\mathbb{C})$ can be constructed [7, Eq. (3.11)], [4, Section 6.4] by $\mathbf{T}_{\boldsymbol{X}}\mathbf{SUT}_m(\mathbb{C}) = \{\boldsymbol{X}\Omega, \Omega \in \mathfrak{sut}_m(\mathbb{C})\}$. For the case $n = 2$,

$$\mathbf{SUT}_2(\mathbb{C}) = \left\{\begin{bmatrix}1 & z \\ 0 & 1\end{bmatrix}, z \in \mathbb{C}\right\}, \quad \mathfrak{sut}_2(\mathbb{C}) = \left\{\begin{bmatrix}0 & z \\ 0 & 0\end{bmatrix}, z \in \mathbb{C}\right\}.$$

Similary, we let $\mathbf{LT}_m(\mathbb{C}) \subseteq \mathbb{C}^{m\times m}$ be the *lower triangular subgroup*, $\mathbf{SLT}_m(\mathbb{C}) \subseteq \mathbb{C}^{m\times m}$ be the *lower unipotent subgroup*, and $\mathfrak{slt}_n(\mathbb{C}) \subseteq \mathbb{C}^{m\times m}$ be the set of *strictly lower triangular* matrices. Then the tangent space to $\mathbf{SLT}_m(\mathbb{C})$ at a point $\boldsymbol{X} \in \mathbf{SLT}_m(\mathbb{C})$ can be constructed by $\mathbf{T}_{\boldsymbol{X}}\mathbf{SLT}_m(\mathbb{C}) = \{\boldsymbol{X}\Omega, \Omega \in \mathfrak{slt}_m(\mathbb{C})\}$.

A diagonal matrix $\boldsymbol{X} \in \mathbb{C}^{m\times m}$ is said to be a *diagonal transformation* if the product of all the diagonal elements is equal to 1. Let $\mathbf{D}_m(\mathbb{C}) \subseteq \mathbf{GL}_m(\mathbb{C})$ be the set of diagonal transformation matrices. Let $\mathfrak{d}_m(\mathbb{C}) \subseteq \mathbb{C}^{m\times m}$ be the set of diagonal traceless matrices. Then the tangent space to $\mathbf{D}_m(\mathbb{C})$ at a point $\boldsymbol{X} \in \mathbf{D}_m(\mathbb{C})$ can be constructed by $\mathbf{T}_{\boldsymbol{X}}\mathbf{D}_m(\mathbb{C}) = \{\boldsymbol{X}\Omega, \Omega \in \mathfrak{d}_m(\mathbb{C})\}$. For the case $n = 2$,

$$\mathbf{D}_2(\mathbb{C}) = \left\{\begin{bmatrix}z & 0 \\ 0 & \frac{1}{z}\end{bmatrix}, z \in \mathbb{C}_*\right\}, \quad \mathfrak{d}_2(\mathbb{C}) = \left\{\begin{bmatrix}z & 0 \\ 0 & -z\end{bmatrix}, z \in \mathbb{C}\right\}.$$

2.4. **Riemannian gradient on $\mathbf{SL}_m(\mathbb{C})$.** For tangent vectors $\xi, \eta \in \mathbf{T}_{\boldsymbol{X}}\mathbf{SL}_m(\mathbb{C})$, we use the left invariant [3], [4, Eq. (6.2)] Riemannian metric

$$\langle \xi, \eta \rangle_{\boldsymbol{X}} \stackrel{\text{def}}{=} \langle \boldsymbol{X}^{-1}\xi, \boldsymbol{X}^{-1}\eta \rangle = \Re\left(\operatorname{tr}(\xi^{\mathsf{H}}(\boldsymbol{X}\boldsymbol{X}^{\mathsf{H}})^{-1}\eta)\right).$$

Let $g : \mathbf{SL}_m(\mathbb{C}) \longrightarrow \mathbb{R}^+$ be a differentiable cost function. Then the Riemannian gradient of g at $\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})$ is the orthogonal projection [4, Lemma 6.2] of its Euclidian gradient $\nabla g(\boldsymbol{X})$ to $\mathbf{T}_{\boldsymbol{X}}\mathbf{SL}_m(\mathbb{C})$, that is,

$$\operatorname{grad} g(\boldsymbol{X}) = \boldsymbol{X}\left(\boldsymbol{X}^{\mathsf{H}}\nabla g(\boldsymbol{X}) - \frac{\operatorname{tr}(\boldsymbol{X}^{\mathsf{H}}\nabla g(\boldsymbol{X}))}{n}\boldsymbol{I}_n\right). \tag{23}$$

We denote $\boldsymbol{\Lambda}(\boldsymbol{X}) \stackrel{\text{def}}{=} \boldsymbol{X}^{-1}\operatorname{grad} g(\boldsymbol{X}) \in \mathfrak{sl}_m(\mathbb{C})$ for $\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})$, which will be frequently used in this paper.

In what follows, we will use the following exponential map

$$\operatorname{Exp}_{\boldsymbol{X}} : \mathbf{T}_{\boldsymbol{X}}\mathbf{SL}_m(\mathbb{C}) \to \mathbf{SL}_m(\mathbb{C}), \quad \boldsymbol{X}\Omega \mapsto \boldsymbol{X}\exp(\Omega^*)\exp(\Omega - \Omega^*),$$

where $\exp(\Omega)$ is the matrix exponential function [3, 7, 19]. For any $\Delta \in \mathbf{T}_{\boldsymbol{X}}\mathbf{SL}_m(\mathbb{C})$, we have the following relationship between $\operatorname{Exp}_{\boldsymbol{X}}$ and the Riemannian gradient

$$\langle \boldsymbol{\Delta}, \operatorname{grad} g(\boldsymbol{X}) \rangle_{\boldsymbol{X}} = \left(\frac{d}{dt} g(\operatorname{Exp}_{\boldsymbol{X}}(t\boldsymbol{\Delta}))\right)\Big|_{t=0}. \tag{24}$$

2.5. **Conditions for convergence analysis.**

2.5.1. *Łojasiewicz gradient inequality.* In this subsection, we present some results about the Łojasiewicz gradient inequality [23, 30, 2, 43]. These results were used in [25, 44] to prove the global convergence of Jacobi-G algorithms on the orthogonal and unitary groups.

Definition 2.1 ([40, Definition 2.1])**.** Let $\mathcal{M} \subseteq \mathbb{R}^n$ be a Riemannian submanifold, and $f : \mathcal{M} \to \mathbb{R}$ be a differentiable function. The function $f : \mathcal{M} \to \mathbb{R}$ is said to satisfy a *Łojasiewicz gradient inequality* at $\boldsymbol{x} \in \mathcal{M}$, if there exist $\sigma > 0$, $\zeta \in (0, \frac{1}{2}]$ and a neighborhood $\mathcal{U}$ in $\mathcal{M}$ of $\boldsymbol{x}$ such that for all $\boldsymbol{y} \in \mathcal{U}$, it follows that

$$|f(\boldsymbol{y}) - f(\boldsymbol{x})|^{1-\zeta} \leq \sigma \|\operatorname{grad} f(\boldsymbol{y})\|. \tag{25}$$

Lemma 2.2 ([40, Proposition 2.2])**.** Let $\mathcal{M} \subseteq \mathbb{R}^n$ be an analytic submanifold[7] and $f : \mathcal{M} \to \mathbb{R}$ be a real analytic function. Then for any $\boldsymbol{x} \in \mathcal{M}$, f satisfies a Łojasiewicz gradient inequality (25) in the δ-neighborhood of $\boldsymbol{x}$, for some[8] $\delta, \sigma > 0$ and $\zeta \in (0, \frac{1}{2}]$.

Theorem 2.3 ([40, Theorem 2.3])**.** Let $\mathcal{M} \subseteq \mathbb{R}^n$ be an analytic submanifold and $\{\boldsymbol{x}_k\}_{k\geq 1} \subseteq \mathcal{M}$. Suppose that f is real analytic and, for large enough k,
(i) there exists $\sigma > 0$ such that

$$f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}_k) \geq \sigma \|\operatorname{grad} f(\boldsymbol{x}_k)\| \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|;$$

(ii) $\operatorname{grad} f(\boldsymbol{x}_k) = 0$ implies that $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$.
Then any accumulation point $\boldsymbol{x}_*$ of $\{\boldsymbol{x}_k\}_{k\geq 1}$ must be the only limit point.

[7] See [21, Definition 2.7.1] or [25, Definition 5.1] for a definition of an analytic submanifold.
[8] The values of δ, σ, ζ depend on the specific point in question.

2.5.2. *Conditions for global convergence.* The following result about the global convergence of Algorithm 1 is a direct consequence of Theorem 2.3 and condition (10).

Proposition 2.4. Suppose that the function f in (8) is real analytic and the sequence $\{\boldsymbol{\omega}_k\}_{k\geq 1}$ produced by Algorithm 1 satisfies that, for large enough k,
(i) there exists $\sigma > 0$ such that

$$f_{t_k}(\boldsymbol{\omega}_{k-1}) - f_{t_k}(\boldsymbol{\omega}_k) \geq \sigma \| \operatorname{grad} f_{t_k}(\boldsymbol{\omega}_{k-1})\| \|\boldsymbol{\omega}_k - \boldsymbol{\omega}_{k-1}\|; \tag{26}$$

(ii) $\operatorname{grad} f_{t_k}(\boldsymbol{\omega}_{k-1}) = 0$ implies that $\boldsymbol{\omega}_k = \boldsymbol{\omega}_{k-1}$.
Then any accumulation point $\boldsymbol{\omega}_*$ of the sequence $\{\boldsymbol{\omega}_k\}_{k\geq 1}$ must be the only limit point.

2.5.3. *Conditions for weak convergence.* Since the special linear group $\mathbf{SL}_m(\mathbb{C})$ is not compact, the iterates $\{\boldsymbol{\omega}_k\}_{k\geq 1}$ produced by Algorithm 1 may have no accumulation point. However, if there exists an accumulation point, we have the following result about its weak convergence, which can be proved easily by condition (10) and the fact that $f(\omega) \geq 0$.

Lemma 2.5. In Algorithm 1 for cost function (8), if there exists $\eta > 0$ such that

$$f(\boldsymbol{\omega}_{k-1}) - f(\boldsymbol{\omega}_k) \geq \eta \| \operatorname{grad} f_{t_k}(\boldsymbol{\omega}_{k-1})\|^2 \tag{27}$$

always holds, then $\lim_{k\to\infty} \operatorname{grad} f(\boldsymbol{\omega}_{k-1}) = 0$. In particular, if $\boldsymbol{\omega}_*$ is an accumulation point of the iterates $\{\boldsymbol{\omega}_k\}_{k\geq 1}$, then $\boldsymbol{\omega}_*$ is a stationary point of f.

3. Line search descent method on $\mathbf{St}(m, n, \mathbb{C})$

Let f be the cost function (5). Let $\boldsymbol{\omega}_{k-1} = (\boldsymbol{U}_{k-1}, \boldsymbol{X}_{k-1})$ and $p = f_{1,\boldsymbol{X}_{k-1}}$ be the first restricted function. Denote $\boldsymbol{X} = \boldsymbol{X}_{k-1}$ for simplicity. Then p can be expressed as

$$p : \mathbf{St}(m, n, \mathbb{C}) \to \mathbb{R}^+, \quad \boldsymbol{U} \mapsto \sum_{\ell=1}^{L} \mu_\ell \| \operatorname{offdiag}\{\boldsymbol{W}^{(\ell)}\}\|^2, \tag{28}$$

where $\boldsymbol{W}^{(\ell)} = \boldsymbol{X}^{\blacklozenge}\boldsymbol{U}^{\blacklozenge}\boldsymbol{A}^{(\ell)}\boldsymbol{U}\boldsymbol{X}$.

3.1. **Riemannian gradient.** We first present a lemma, which can be obtained by direct calculations. This result will help us to obtain the Riemannian gradient of p in (28).

Lemma 3.1. Let $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ and the function $\tilde{p}$ be defined as

$$\tilde{p} : \mathbb{C}^{n\times m} \to \mathbb{R}^+, \quad \boldsymbol{Y} \mapsto \| \operatorname{offdiag}\{\boldsymbol{W}\}\|^2,$$

where $\boldsymbol{W} = \boldsymbol{Y}^{\blacklozenge}\boldsymbol{A}\boldsymbol{Y}$. Let $\boldsymbol{V} = \boldsymbol{A}\boldsymbol{Y} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_m]$ and $\boldsymbol{V}' = \boldsymbol{A}^{\blacklozenge}\boldsymbol{Y} = [\boldsymbol{v}_1', \cdots, \boldsymbol{v}_m']$. Denote $\boldsymbol{Y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m]$. Then the Euclidean gradient is

$$\nabla\tilde{p}(\boldsymbol{Y}) = 2\left(\sum_{j\neq 1} \boldsymbol{v}_j\boldsymbol{v}_j^{\mathsf{H}}\boldsymbol{y}_1, \cdots, \sum_{j\neq m} \boldsymbol{v}_j\boldsymbol{v}_j^{\mathsf{H}}\boldsymbol{y}_m\right) + 2\left(\sum_{j\neq 1} \boldsymbol{v}_j'(\boldsymbol{v}_j')^{\mathsf{H}}\boldsymbol{y}_1, \cdots, \sum_{j\neq m} \boldsymbol{v}_j'(\boldsymbol{v}_j')^{\mathsf{H}}\boldsymbol{y}_m\right).$$

In particular, it satisfies

$$\boldsymbol{Y}\boldsymbol{Y}^{\mathsf{H}}\nabla\tilde{p}(\boldsymbol{Y}) = 2\boldsymbol{Y}\Upsilon(\boldsymbol{W}), \tag{29}$$

where

$$\Upsilon(\boldsymbol{W}) \stackrel{\text{def}}{=} \begin{cases} \boldsymbol{W}\,\mathrm{offdiag}\{\boldsymbol{W}\}^{\mathsf{H}} + \boldsymbol{W}^{H}\,\mathrm{offdiag}\{\boldsymbol{W}\}, & \text{if } (\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{H}}; \\ \boldsymbol{W}^{*}\,\mathrm{offdiag}\{\boldsymbol{W}\}^{\mathsf{T}} + \boldsymbol{W}^{H}\,\mathrm{offdiag}\{\boldsymbol{W}\}, & \text{if } (\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{T}}. \end{cases}$$

Lemma 3.2. Let $\boldsymbol{W}^{(\ell)}$ and the function p be as in (28). Then the Euclidean gradient satisfies

$$\boldsymbol{U}^{\mathsf{H}}\nabla p(\boldsymbol{U}) = 2(\boldsymbol{X}^{\mathsf{H}})^{-1}\sum_{\ell=1}^{L}\mu_\ell\Upsilon(\boldsymbol{W}^{(\ell)})\boldsymbol{X}^{\mathsf{H}}. \tag{30}$$

Proof. By the product rule, we see that $\nabla p(\boldsymbol{U}) = \nabla\tilde{p}(\boldsymbol{U}\boldsymbol{X})\boldsymbol{X}^{\mathsf{H}}$. Then, by (29), we have

$$\boldsymbol{X}\boldsymbol{X}^{\mathsf{H}}\boldsymbol{U}^{\mathsf{H}}\nabla p(\boldsymbol{U}) = \boldsymbol{U}^{\mathsf{H}}\boldsymbol{U}\boldsymbol{X}(\boldsymbol{U}\boldsymbol{X})^{\mathsf{H}}\nabla\tilde{p}(\boldsymbol{U}\boldsymbol{X})\boldsymbol{X}^{\mathsf{H}} = 2\boldsymbol{U}^{\mathsf{H}}\boldsymbol{U}\boldsymbol{X}\sum_{\ell=1}^{L}\mu_\ell\Upsilon(\boldsymbol{W}^{(\ell)})\boldsymbol{X}^{\mathsf{H}}.$$

Note that $\boldsymbol{X}$ is invertible. The proof is complete. □

Now, by (30) and (22), we see that the Riemannian gradient of p in (28) satisfies

$$\boldsymbol{U}^{\mathsf{H}}\,\mathrm{grad}\,p(\boldsymbol{U}) = (\boldsymbol{X}^{\mathsf{H}})^{-1}\sum_{\ell=1}^{L}\mu_\ell\Upsilon(\boldsymbol{W}^{(\ell)})\boldsymbol{X}^{\mathsf{H}} - \boldsymbol{X}\sum_{\ell=1}^{L}\mu_\ell\Upsilon(\boldsymbol{W}^{(\ell)})^{\mathsf{H}}\boldsymbol{X}^{-1}.$$

3.2. **Line search descent method.** In this paper, we adopt the *line search descent* method [2, 3, 37, 38, 39] on $\mathbf{St}(m,n,\mathbb{C})$ to find $\boldsymbol{U}_k$ for the cost function (28). More precisely, we set

$$\boldsymbol{U}_k = \mathrm{Exp}_{\boldsymbol{U}_{k-1}}(t_{k-1}\boldsymbol{Z}_{k-1}), \tag{31}$$

where $\boldsymbol{Z}_{k-1}$ is the *search direction* and t_{k-1} is the *step size*. We always choose $\boldsymbol{Z}_{k-1}$ such that

$$\langle\mathrm{grad}\,p(\boldsymbol{U}_{k-1}), \boldsymbol{Z}_{k-1}\rangle_{\boldsymbol{U}_{k-1}} \leq -\delta_s\|\mathrm{grad}\,p(\boldsymbol{U}_{k-1})\|\|\boldsymbol{Z}_{k-1}\|, \tag{32}$$

with $0 < \delta_s < 1$. We say that t_{k-1} satisfies the *Armijo condition*[9], if

$$p(\boldsymbol{U}_k) \leq p(\boldsymbol{U}_{k-1}) + \delta_w t_{k-1}\langle\mathrm{grad}\,p(\boldsymbol{U}_{k-1}), \boldsymbol{Z}_{k-1}\rangle_{\boldsymbol{U}_{k-1}}, \tag{33}$$

with $0 < \delta_w < 1$. We say that t_{k-1} satisfies the *curvature condition*, if

$$\langle\mathrm{grad}\,p(\boldsymbol{U}_k), \mathbf{D}\mathrm{Exp}_{\boldsymbol{U}_{k-1}}(t_{k-1}\boldsymbol{Z}_{k-1})[\boldsymbol{Z}_{k-1}]\rangle_{\boldsymbol{U}_k} \geq c_2\langle\mathrm{grad}\,p(\boldsymbol{U}_{k-1}), \boldsymbol{Z}_{k-1}\rangle_{\boldsymbol{U}_{k-1}}, \tag{34}$$

with $\delta_w < c_2 < 1$. The conditions (33) and (34) are known collectively as the *Wolfe conditions*. As in the Euclidean space case [37, Lemma 3.1], it was shown [38, 39] that we can always choose t_{k-1} such that the conditions (33) and (34) are both satisfied. It is not difficult to see that there exists $\mathrm{M}_e > 0$ such that

$$\|\mathrm{Exp}_{\boldsymbol{U}}(\boldsymbol{Z}_1) - \mathrm{Exp}_{\boldsymbol{U}}(\boldsymbol{Z}_2)\| \leq \mathrm{M}_e\|\boldsymbol{Z}_1 - \boldsymbol{Z}_2\|,$$

for any $\boldsymbol{U} \in \mathbf{St}(m,n,\mathbb{C})$ and $\boldsymbol{Z}_1, \boldsymbol{Z}_2 \in \mathbf{T}_{\boldsymbol{U}}\mathrm{St}(m,n,\mathbb{C})$. Then the next result follows directly.

Lemma 3.3. If we set $\boldsymbol{U}_k$ as in (31) such that the conditions (32) and (33) are both satisfied, then we have

$$p(\boldsymbol{U}_{k-1}) - p(\boldsymbol{U}_k) \geq \delta_s\delta_w\|\mathrm{grad}\,p(\boldsymbol{U}_{k-1})\|\|t_{k-1}\boldsymbol{Z}_{k-1}\| \geq \sigma_p\|\mathrm{grad}\,p(\boldsymbol{U}_{k-1})\|\|\boldsymbol{U}_k - \boldsymbol{U}_{k-1}\|,$$

where $\sigma_p = \frac{\delta_s\delta_w}{\mathrm{M}_e}$.

[9] It is also known as the *first Wolfe condition* in the literature.

Moreover, we have the next result, which is a simple corollary of the proof in [38, Theorem 2].

Lemma 3.4. If we set $\boldsymbol{U}_k$ as in (31) such that the conditions (32), (33) and (34) are all satisfied, then we have

$$p(\boldsymbol{U}_{k-1}) - p(\boldsymbol{U}_k) \geq \eta_p \| \operatorname{grad} p(\boldsymbol{U}_{k-1}) \|^2, \tag{35}$$

where $\eta_p > 0$ is fixed.

4. GLU, GQU and GU classes of transformations on $\mathbf{SL}_m(\mathbb{C})$

4.1. **Elementary functions.** Let (i, j) be a pair of indices with $1 \leq i < j \leq n$. Define a projection operator $\mathcal{P}_{i,j} : \mathbb{C}^{m \times m} \to \mathbb{C}^{2 \times 2}$ which extracts a submatrix of $\boldsymbol{X} \in \mathbb{C}^{m \times m}$ as follows:

$$\mathcal{P}_{i,j}(\boldsymbol{X}) = \begin{bmatrix} \mathrm{X}_{ii} & \mathrm{X}_{ij} \\ \mathrm{X}_{ji} & \mathrm{X}_{jj} \end{bmatrix}. \tag{36}$$

Conversely, we introduce an operator

$$\mathcal{E}_{i,j} : \mathbb{C}^{2 \times 2} \to \mathbb{C}^{m \times m}, \ \boldsymbol{\Psi} \mapsto \mathcal{E}_{i,j}(\boldsymbol{\Psi})$$

where $\mathcal{E}_{i,j}(\boldsymbol{\Psi})$ is the identity matrix $\boldsymbol{I}_m$ except that $\mathcal{P}_{i,j}(\mathcal{E}_{i,j}(\boldsymbol{\Psi})) = \boldsymbol{\Psi}$. Now we define the following four elementary transformations in $\mathbf{SL}_m(\mathbb{C})$.

- $\boldsymbol{G}^{(i,j,\boldsymbol{\Psi})} \stackrel{\text{def}}{=} \mathcal{E}_{i,j}(\boldsymbol{\Psi})$: *Givens plane transformation* for a matrix $\boldsymbol{\Psi} \in \mathbf{SU}_2$.
- $\boldsymbol{U}^{(i,j,\boldsymbol{\Psi})} \stackrel{\text{def}}{=} \mathcal{E}_{i,j}(\boldsymbol{\Psi})$: *plane upper triangular transformation* for a matrix $\boldsymbol{\Psi} \in \mathbf{SUT}_2(\mathbb{C})$.
- $\boldsymbol{L}^{(i,j,\boldsymbol{\Psi})} \stackrel{\text{def}}{=} \mathcal{E}_{i,j}(\boldsymbol{\Psi})$: *plane lower triangular transformation* for a matrix $\boldsymbol{\Psi} \in \mathbf{SLT}_2(\mathbb{C})$.
- $\boldsymbol{D}^{(i,j,\boldsymbol{\Psi})} \stackrel{\text{def}}{=} \mathcal{E}_{i,j}(\boldsymbol{\Psi})$: *plane diagonal transformation* for a matrix $\boldsymbol{\Psi} \in \mathbf{D}_2(\mathbb{C})$.

Remark 4.1. These four elementary transformations have all been used in the literature. The Givens plane transformation $\boldsymbol{G}^{(i,j,\boldsymbol{\Psi})}$ was used very often in Jacobi-type algorithms for joint approximate diagonalization of matrices or tensors by orthogonal or non-orthogonal transformations [17, 24, 44, 4, 5, 41]. Plane triangular transformations $\boldsymbol{U}^{(i,j,\boldsymbol{\Psi})}$ and $\boldsymbol{L}^{(i,j,\boldsymbol{\Psi})}$ also appeared many times in the Jacobi-type algorithms on special linear group $\mathbf{SL}_m(\mathbb{C})$ or $\mathbf{SL}_m(\mathbb{R})$ [4, 5, 34, 35, 36]. In the real case, the diagonal transform $\boldsymbol{D}^{(i,j,\boldsymbol{\Psi})}$ was used in [42]. In this paper, using these four elementary transformations, we will construct three classes (GLU, GQU and GU) in Sections 4.3 and 4.4, such that a gradient inequality (38) is always satisfied to establish the global convergence of Algorithm 1 in Section 7.

In this paper, as in [17, 44], we parameterize $\boldsymbol{\Psi} \in \mathbf{SU}_2$ as

$$\boldsymbol{\Psi} = \boldsymbol{\Psi}(c, s_1, s_2) = \begin{bmatrix} c & -s \\ s^* & c \end{bmatrix} = \begin{bmatrix} c & -(s_1 + is_2) \\ s_1 - is_2 & c \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \mathrm{e}^{i\phi} \\ \sin\theta \mathrm{e}^{-i\phi} & \cos\theta \end{bmatrix},$$

where $(c, s_1, s_2) \in \mathbb{S}_2$ and $(\theta, \phi) \in \mathbb{R}^2$. Let $g : \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}^+$ be a smooth function. Let $\boldsymbol{X} \in \mathbf{SL}_m(\mathbb{C})$ and $z = x + iy$. We define the following elementary functions:

$$\begin{aligned} h_{(i,j),\boldsymbol{X}}(c, s_1, s_2) = h_{(i,j),\boldsymbol{X}}(\theta, \phi) = h_{(i,j),\boldsymbol{X}}(\boldsymbol{\Psi}) &\stackrel{\text{def}}{=} g(\boldsymbol{X}\boldsymbol{G}^{(i,j,\boldsymbol{\Psi})}), \quad (c, s_1, s_2) \in \mathbb{S}_2; \\ \varphi_{(i,j),\boldsymbol{X}}(x, y) = \varphi_{(i,j),\boldsymbol{X}}(\boldsymbol{\Psi}) &\stackrel{\text{def}}{=} g(\boldsymbol{X}\boldsymbol{U}^{(i,j,\boldsymbol{\Psi})}), \quad z \in \mathbb{C}; \\ \psi_{(i,j),\boldsymbol{X}}(x, y) = \psi_{(i,j),\boldsymbol{X}}(\boldsymbol{\Psi}) &\stackrel{\text{def}}{=} g(\boldsymbol{X}\boldsymbol{L}^{(i,j,\boldsymbol{\Psi})}), \quad z \in \mathbb{C}; \\ \rho_{(i,j),\boldsymbol{X}}(x, y) = \rho_{(i,j),\boldsymbol{X}}(\boldsymbol{\Psi}) &\stackrel{\text{def}}{=} g(\boldsymbol{X}\boldsymbol{D}^{(i,j,\boldsymbol{\Psi})}), \quad z \in \mathbb{C}_*. \end{aligned}$$

4.2. **Derivatives of elementary functions.** Denote $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}(\boldsymbol{X})$, which is defined as in Section 2.4. Let $\mathcal{P}_{i,j}^{\mathsf{T}} : \mathbb{C}^{2\times 2} \to \mathbb{C}^{m\times m}$ be the adjoint operator of projection operator $\mathcal{P}_{i,j}$ defined in (36), *i.e.*,

$$\mathcal{P}_{i,j}^{\mathsf{T}}(\boldsymbol{\Psi}) = \begin{array}{c} \\ i \\ \\ j \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} & i & & j \end{array} \\ \begin{bmatrix} \mathbf{0} & \vdots & \vdots & \mathbf{0} \\ \cdots & \Psi_{ii} & \Psi_{ij} & \\ & & & \\ \cdots & \Psi_{ji} & \Psi_{jj} & \\ \mathbf{0} & & & \mathbf{0} \end{bmatrix} \end{array}$$

for $\boldsymbol{\Psi} \in \mathbb{C}^{2\times 2}$.

Lemma 4.2. The Riemannian gradients of the elementary functions defined in Section 4.1 at the identity matrix $\boldsymbol{I}_2$ can be expressed as follows:

- $\operatorname{grad} h_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) = \begin{bmatrix} \frac{i}{2}\Im(\boldsymbol{\Lambda}_{ii} - \boldsymbol{\Lambda}_{jj}) & \frac{1}{2}\Re(\boldsymbol{\Lambda}_{ij} - \boldsymbol{\Lambda}_{ji}) + \frac{i}{2}\Im(\boldsymbol{\Lambda}_{ij} + \boldsymbol{\Lambda}_{ji}) \\ -\frac{1}{2}\Re(\boldsymbol{\Lambda}_{ij} - \boldsymbol{\Lambda}_{ji}) + \frac{i}{2}\Im(\boldsymbol{\Lambda}_{ij} + \boldsymbol{\Lambda}_{ji}) & -\frac{i}{2}\Im(\boldsymbol{\Lambda}_{ii} - \boldsymbol{\Lambda}_{jj}) \end{bmatrix}$;
- $\operatorname{grad} \varphi_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) = \begin{bmatrix} 0 & \boldsymbol{\Lambda}_{ij} \\ 0 & 0 \end{bmatrix}$;
- $\operatorname{grad} \psi_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) = \begin{bmatrix} 0 & 0 \\ \boldsymbol{\Lambda}_{ji} & 0 \end{bmatrix}$;
- $\operatorname{grad} \rho_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) = \begin{bmatrix} \Re(\boldsymbol{\Lambda}_{ii} - \boldsymbol{\Lambda}_{jj}) & 0 \\ 0 & \Im(\boldsymbol{\Lambda}_{ii} - \boldsymbol{\Lambda}_{jj}) \end{bmatrix}$.

Proof. We only prove the case of $h_{(i,j),\boldsymbol{X}}$. Other cases are similar. For any $\boldsymbol{\Delta} \in \mathfrak{su}_2(\mathbb{C}) = \mathbf{T}_{\boldsymbol{I}_2}\mathbf{SU}_2$, by equation (24), we have

$$\begin{aligned} &\langle \boldsymbol{\Delta}, \operatorname{grad} h_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2)\rangle_{\boldsymbol{I}_2} = \left(\frac{d}{dt} h_{(i,j),\boldsymbol{X}}(\operatorname{Exp}_{\boldsymbol{I}_2}(t\boldsymbol{\Delta}))\right)\Big|_{t=0} = \left(\frac{d}{dt} f(\boldsymbol{X}\boldsymbol{G}^{(i,j,\operatorname{Exp}_{\boldsymbol{I}_2}(\boldsymbol{\Delta}t))})\right)\Big|_{t=0} \\ &= \left(\frac{d}{dt} f(\operatorname{Exp}_{\boldsymbol{X}}(\boldsymbol{X}\mathcal{P}_{i,j}^{\mathsf{T}}(\boldsymbol{\Delta})t))\right)\Big|_{t=0} = \langle \boldsymbol{X}\mathcal{P}_{i,j}^{\mathsf{T}}(\boldsymbol{\Delta}), \operatorname{grad} f(\boldsymbol{X})\rangle_{\boldsymbol{X}} = \langle \boldsymbol{\Delta}, \mathcal{P}_{i,j}(\boldsymbol{\Lambda})\rangle_{\boldsymbol{I}_2}. \end{aligned}$$

The proof is complete. □

The following lemma can be easily obtained by Lemma 4.2.

Lemma 4.3. The partial derivatives of the elementary functions defined in Section 4.1 satisfy

- $\partial h_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) \stackrel{\text{def}}{=} \partial h_{(i,j),\boldsymbol{X}}(1,0,0) = [0,\ -\Re(\boldsymbol{\Lambda}_{ij} - \boldsymbol{\Lambda}_{ji}),\ -\Im(\boldsymbol{\Lambda}_{ij} + \boldsymbol{\Lambda}_{ji})]^{\mathsf{T}}$;
- $\partial \varphi_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) \stackrel{\text{def}}{=} \partial \varphi_{(i,j),\boldsymbol{X}}(0,0) = [\Re(\boldsymbol{\Lambda}_{ij}),\ \Im(\boldsymbol{\Lambda}_{ij})]^{\mathsf{T}}$;
- $\partial \psi_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) \stackrel{\text{def}}{=} \partial \psi_{(i,j),\boldsymbol{X}}(0,0) = [\Re(\boldsymbol{\Lambda}_{ji}),\ \Im(\boldsymbol{\Lambda}_{ji})]^{\mathsf{T}}$;
- $\partial \rho_{(i,j),\boldsymbol{X}}(\boldsymbol{I}_2) \stackrel{\text{def}}{=} \partial \rho_{(i,j),\boldsymbol{X}}(0,0) = [\Re(\boldsymbol{\Lambda}_{ii} - \boldsymbol{\Lambda}_{jj}),\ \Im(\boldsymbol{\Lambda}_{ii} - \boldsymbol{\Lambda}_{jj})]^{\mathsf{T}}$.

4.3. **GLU and GQU classes of transformations.** Let $\boldsymbol{\omega}_{k-1} = (\boldsymbol{U}_{k-1}, \boldsymbol{X}_{k-1})$ be the $(k-1)$-th iterate produced by Algorithm 1, and $g : \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}$ be the restricted function $f_{2,\boldsymbol{U}_{k-1}}$ defined as in (9). Let (i_k, j_k) be a pair of indices with $1 \le i_k < j_k \le m$. We denote that

$$h_k = h_{(i_k,j_k),\boldsymbol{X}_{k-1}},\quad \varphi_k = \varphi_{(i_k,j_k),\boldsymbol{X}_{k-1}},\quad \psi_k = \psi_{(i_k,j_k),\boldsymbol{X}_{k-1}},\quad \rho_k = \rho_{(i_k,j_k),\boldsymbol{X}_{k-1}}. \tag{37}$$

Inspired by the equations (11), (12) and a similar idea as in [20, 24, 44], we now propose two classes of elementary transformations on $\mathbf{SL}_m(\mathbb{C})$. We call them the *GLU* (based on LU decomposition) and *GQU* (based on QU decomposition) transformations[10], respectively. Based on these two classes of elementary transformations, the subalgorithms to update $\boldsymbol{X}_k$ in Algorithm 1 are summarized in Subalgorithm 1a and Subalgorithm 1b, respectively.

Subalgorithm 1a: The subalgorithm to update $\boldsymbol{X}_k$ based on GLU transformations

1: **Input:** A fixed positive constant $0 < \varepsilon < \sqrt{\frac{2}{3m(m-1)}}$.
2: **Output:** New iterate $\boldsymbol{X}_k$.
3: Choose the index pair (i_k, j_k) and ν_k such that

$$\|\partial \nu_k(\boldsymbol{I}_2)\| \ge \varepsilon \|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|, \tag{38}$$

where $\nu_k = \varphi_k, \psi_k$ or ρ_k;
4: Compute $\boldsymbol{\Psi}_k^*$ that minimizes the function ν_k;
5: Update $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}\boldsymbol{V}_k$, where $\boldsymbol{V}_k = \boldsymbol{U}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}, \boldsymbol{L}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$ or $\boldsymbol{D}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$.

Subalgorithm 1b: The subalgorithm to update $\boldsymbol{X}_k$ based on GQU transformations

1: **Input:** A fixed positive constant $0 < \varepsilon < \sqrt{\frac{3+\sqrt{5}}{3m(m-1)}}$.
2: **Output:** New iterate $\boldsymbol{X}_k$.
3: Choose the index pair (i_k, j_k) and ν_k satisfying (38), where $\nu_k = h_k, \varphi_k$ or ρ_k;
4: Compute $\boldsymbol{\Psi}_k^*$ that minimizes the function ν_k;
5: Update $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}\boldsymbol{V}_k$, where $\boldsymbol{V}_k = \boldsymbol{G}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}, \boldsymbol{U}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$ or $\boldsymbol{D}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$.

Proposition 4.4. In Subalgorithm 1a and Subalgorithm 1b, we can always choose an index pair (i_k, j_k) and an elementary function $\nu_k(x)$ such that the inequality (38) is satisfied.

We need a simple lemma before the proof of Proposition 4.4.

[10]The inequality (38) can be seen as a non-orthogonal analogue of [20, Eq. (3.3)] and [24, Eq. (10)].

Lemma 4.5. (i) Let $z_1, z_2 \in \mathbb{C}$. Then

$$|z_1 - z_2|^2 + |z_2|^2 \geq \frac{3-\sqrt{5}}{2}(|z_1|^2 + |z_2|^2).$$

(ii) Let $\{z_i\}_{1\leq i\leq m} \subseteq \mathbb{C}$ satisfy that $\sum_{1\leq i\leq m} z_i = 0$. Then

$$\sum_{1\leq i<j\leq m} |z_i - z_j|^2 = m \sum_{1\leq i\leq m} |z_i|^2.$$

Proof of Proposition 4.4. We first prove the existence of such an index pair (i_k, j_k) and elementary function ν_k in Subalgorithm 1a. By Lemma 4.3, Lemma 4.5 and $\mathbf{\Lambda} = \mathbf{\Lambda}(\boldsymbol{X}_{k-1}) \in \mathfrak{sl}_m(\mathbb{C})$, we get that

$$\begin{aligned}
&\sum_{1\leq i_k<j_k\leq m} \left(\|\partial\varphi_k(\boldsymbol{I}_2)\|^2 + \|\partial\psi_k(\boldsymbol{I}_2)\|^2 + \|\partial\rho_k(\boldsymbol{I}_2)\|^2\right) \\
&= \sum_{1\leq i_k<j_k\leq m} \left(|\Lambda_{i_kj_k}|^2 + |\Lambda_{j_ki_k}|^2\right) + m \sum_{1\leq i_k\leq m} |\Lambda_{i_ki_k}|^2 \geq \|\mathbf{\Lambda}\|^2.
\end{aligned}$$

Therefore, there exist an index pair (i_k, j_k) and $\nu_k(x)$ such that

$$\frac{3}{2}m(m-1)|\|\partial\psi_k(\boldsymbol{I}_2)\|^2 \geq \|\mathbf{\Lambda}\|^2.$$

Next, we prove the existence in Subalgorithm 1b. Similarly, we get that

$$\begin{aligned}
&\sum_{1\leq i_k<j_k\leq m} \left(\|\partial h_k(\boldsymbol{I}_2)\|^2 + \|\partial\varphi_k(\boldsymbol{I}_2)\|^2 + \|\partial\rho_k(\boldsymbol{I}_2)\|^2\right) \\
&= \sum_{1\leq i_k<j_k\leq m} \left(|\Lambda^*_{i_kj_k} - \Lambda_{j_ki_k}|^2 + |\Lambda_{i_kj_k}|^2 + |\Lambda_{i_ki_k} - \Lambda_{j_kj_k}|^2\right) \\
&\geq \frac{3-\sqrt{5}}{2} \sum_{1\leq i_k<j_k\leq m} \left(|\Lambda_{i_kj_k}|^2 + |\Lambda_{j_ki_k}|^2\right) + m \sum_{1\leq i_k\leq m} |\Lambda_{i_ki_k}|^2 \geq \frac{3-\sqrt{5}}{2}\|\mathbf{\Lambda}\|^2.
\end{aligned}$$

Therefore, there exists an index pair (i_k, j_k) and $\nu_k(x)$ such that

$$\frac{3}{3+\sqrt{5}}m(m-1)\|\partial\nu_k(\boldsymbol{I}_2)\|^2 \geq \|\mathbf{\Lambda}\|^2.$$

The proof is complete. □

4.4. **GU class of transformations.** Inspired by the equation (13), we now propose the third class of elementary transformations: GU, which includes the plane upper triangular and plane diagonal transformations. In this case, we call Algorithm 1 the *BCD-GU* algorithm, in which we choose a starting point $\boldsymbol{X}_0 \in \mathbf{EUT}_m(\mathbb{C})$. Based on this class of elementary transformations, the subalgorithm to update $\boldsymbol{X}_k$ in Algorithm 1 is summarized in Subalgorithm 1c.

Let $\mathfrak{eut}_m(\mathbb{C}) \subseteq \mathbb{C}^{m\times m}$ be the set of upper triangular matrices with trace equal to 0. Then the tangent space to $\mathbf{EUT}_m(\mathbb{C})$ at a point $\boldsymbol{X} \in \mathbf{EUT}_m(\mathbb{C})$ can be constructed [7, 4] by $\mathbf{T}_{\boldsymbol{X}}\mathbf{EUT}_m(\mathbb{C}) = \{\boldsymbol{X}\Omega, \Omega \in \mathfrak{eut}_m(\mathbb{C})\}$.

Proposition 4.6. In Subalgorithm 1c, we can always choose an index pair (i_k, j_k) and an elementary function $\nu_k(x)$ such that the inequality (38) is satisfied.

Subalgorithm 1c: The subalgorithm to update $\boldsymbol{X}_k$ based on GU transformations

1: **Input:** A fixed positive constant $0 < \varepsilon < \sqrt{\frac{2}{m(m+1)}}$.
2: **Output:** New iterate $\boldsymbol{X}_k$.
3: Choose the index pair (i_k, j_k) and ν_k satisfying (38), where $\nu_k = \varphi_k$ or ρ_k;
4: Compute $\boldsymbol{\Psi}_k^*$ that minimizes the function ν_k;
5: Update $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}\boldsymbol{V}_k$, where $\boldsymbol{V}_k = \boldsymbol{U}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$ or $\boldsymbol{D}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$.

Proof. Note that $\boldsymbol{X}_0 \in \mathbf{EUT}_m(\mathbb{C})$ in Subalgorithm 1c. We see that $\boldsymbol{X}_k \in \mathbf{EUT}_m(\mathbb{C})$ for all $k \in \mathbb{N}$. By Lemma 4.3, Lemma 4.5 and $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}(\boldsymbol{X}_{k-1}) \in \mathfrak{eut}_m(\mathbb{C})$, we get that

$$\begin{aligned}\sum_{1\le i_k<j_k\le m} \left(\|\partial\varphi_k(\boldsymbol{I}_2)\|^2 + \|\partial\rho_k(\boldsymbol{I}_2)\|^2\right) &= \sum_{1\le i_k<j_k\le m} \left(|\Lambda_{i_kj_k}|^2 + |\Lambda_{i_ki_k} - \Lambda_{j_kj_k}|^2\right) \\ &= \sum_{1\le i_k<j_k\le m} |\Lambda_{i_kj_k}|^2 + m \sum_{1\le i_k\le m} |\Lambda_{i_ki_k}|^2 \ge \|\boldsymbol{\Lambda}\|^2.\end{aligned}$$

Therefore, there exists an index pair (i_k, j_k) and $\nu_k(x)$ such that

$$\frac{m(m+1)}{2}\|\partial\nu_k(\boldsymbol{I}_2)\|^2 \ge \|\boldsymbol{\Lambda}\|^2.$$

The proof is complete. □

5. Triangular and diagonal transformations for JADM problem

Let f be the cost function (5). Let $\boldsymbol{\omega}_{k-1} = (\boldsymbol{U}_{k-1}, \boldsymbol{X}_{k-1})$ and $g : \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}$ be the restricted function $f_{2,\boldsymbol{U}_{k-1}}$ as in Section 4.3. Denote $\boldsymbol{B}^{(\ell)} = \boldsymbol{U}_{k-1}^{\blacklozenge}\boldsymbol{A}^{(\ell)}\boldsymbol{U}_{k-1}$. Then g can be expressed as

$$g : \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}^+, \quad \boldsymbol{X} \mapsto \sum_{\ell=1}^{L} \mu_\ell \|\operatorname{offdiag}\{\boldsymbol{W}^{(\ell)}\}\|^2, \tag{39}$$

where $\boldsymbol{W}^{(\ell)} = \boldsymbol{X}^{\blacklozenge}\boldsymbol{B}^{(\ell)}\boldsymbol{X}$. In this section, we will first calculate the Riemannian gradient of g in (39), and the partial derivatives of elementary functions φ_k, ψ_k and ρ_k in (37). Then, we will prove that conditions (26) and (27) are both satisfied in the triangular and diagonal transformations.

5.1. Riemannian gradient.

Let $\boldsymbol{W}^{(\ell)}$ and g be as in (39). Then, by (29) and (23), we have

$$\nabla g(\boldsymbol{X}) = 2(\boldsymbol{X}^{\mathsf{H}})^{-1} \sum_{\ell=1}^{L} \mu_\ell \Upsilon(\boldsymbol{W}^{(\ell)}), \tag{40}$$

$$\operatorname{grad} g(\boldsymbol{X}) = 2\boldsymbol{X} \sum_{\ell=1}^{L} \mu_\ell \left(\Upsilon(\boldsymbol{W}^{(\ell)}) - \frac{\operatorname{tr}(\Upsilon(\boldsymbol{W}^{(\ell)}))}{n}\boldsymbol{I}_n\right). \tag{41}$$

Remark 5.1. In the real case, the Euclidean gradient in (40) was earlier derived in [4, Eq. (6.3)] and [9, Section 2.3]. In this paper, we extend it to problem (39) in the complex case, and calculate the Riemannian gradieent (41) as well.

5.2. **Elementary functions.** Let $\boldsymbol{W}^{(\ell)} = \boldsymbol{X}_{k-1}^{\blacklozenge}\boldsymbol{B}^{(\ell)}\boldsymbol{X}_{k-1}$. Let

$$\varrho = \begin{cases} 1, & \text{if } (\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{H}}; \\ -1, & \text{if } (\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{T}}. \end{cases}$$

Denote $(i,j) = (i_k, j_k)$ for simpliciy. Now we make the following notations:

- $\alpha_1 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq j}\mu_\ell\left(|\mathrm{W}_{ip}^{(\ell)}|^2 + |\mathrm{W}_{pi}^{(\ell)}|^2\right),$

  $\alpha_2 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq j}\mu_\ell\left(\mathrm{W}_{ip}^{(\ell,\Re)}\mathrm{W}_{jp}^{(\ell,\Re)} + \mathrm{W}_{ip}^{(\ell,\Im)}\mathrm{W}_{jp}^{(\ell,\Im)} + \mathrm{W}_{pi}^{(\ell,\Re)}\mathrm{W}_{pj}^{(\ell,\Re)} + \mathrm{W}_{pi}^{(\ell,\Im)}\mathrm{W}_{pj}^{(\ell,\Im)}\right),$

  $\alpha_3 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq j}\mu_\ell\left(\varrho\left(\mathrm{W}_{ip}^{(\ell,\Im)}\mathrm{W}_{jp}^{(\ell,\Re)} - \mathrm{W}_{ip}^{(\ell,\Re)}\mathrm{W}_{jp}^{(\ell,\Im)}\right) + \mathrm{W}_{pi}^{(\ell,\Re)}\mathrm{W}_{pj}^{(\ell,\Im)} - \mathrm{W}_{pi}^{(\ell,\Im)}\mathrm{W}_{pj}^{(\ell,\Re)}\right).$
- $\beta_1 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq i}\mu_\ell\left(|\mathrm{W}_{jp}^{(\ell)}|^2 + |\mathrm{W}_{pj}^{(\ell)}|^2\right),$

  $\beta_2 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq i}\mu_\ell\left(\mathrm{W}_{ip}^{(\ell,\Re)}\mathrm{W}_{jp}^{(\ell,\Re)} + \mathrm{W}_{ip}^{(\ell,\Im)}\mathrm{W}_{jp}^{(\ell,\Im)} + \mathrm{W}_{pi}^{(\ell,\Re)}\mathrm{W}_{pj}^{(\ell,\Re)} + \mathrm{W}_{pi}^{(\ell,\Im)}\mathrm{W}_{pj}^{(\ell,\Im)}\right),$

  $\beta_3 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq i}\mu_\ell\left(\varrho\left(\mathrm{W}_{ip}^{(\ell,\Re)}\mathrm{W}_{jp}^{(\ell,\Im)} - \mathrm{W}_{ip}^{(\ell,\Im)}\mathrm{W}_{jp}^{(\ell,\Re)}\right) + \mathrm{W}_{pi}^{(\ell,\Im)}\mathrm{W}_{pj}^{(\ell,\Re)} - \mathrm{W}_{pi}^{(\ell,\Re)}\mathrm{W}_{pj}^{(\ell,\Im)}\right).$
- $\gamma_1 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq i,j}\mu_\ell\left(|\mathrm{W}_{ip}^{(\ell)}|^2 + |\mathrm{W}_{pi}^{(\ell)}|^2\right), \qquad \gamma_2 \overset{\text{def}}{=} \sum_{\ell=1}^{L}\sum_{p\neq i,j}\mu_\ell\left(|\mathrm{W}_{jp}^{(\ell)}|^2 + |\mathrm{W}_{pj}^{(\ell)}|^2\right).$

Then we can get the following results by direct calculations.

Lemma 5.2. Let the function g be as in (39). Then
(i) the elementary function φ_k in (37) satisfies

$$\begin{aligned} \varphi_k(x,y) - \varphi_k(0,0) &= \alpha_1 x^2 + 2\alpha_2 x + \alpha_1 y^2 + 2\alpha_3 y, \\ \varphi_k(x_k^*, y_k^*) - \varphi_k(0,0) &= -\frac{1}{\alpha_1}\left(\alpha_2^2 + \alpha_3^2\right), \\ \partial\varphi_k(0,0) &= 2[\alpha_2,\ \alpha_3]^{\mathsf{T}}. \end{aligned} \tag{42}$$

(ii) the elementary function ψ_k in (37) satisfies

$$\begin{aligned} \psi_k(x,y) - \psi_k(0,0) &= \beta_1 x^2 + 2\beta_2 x + \beta_1 y^2 + 2\beta_3 y, \\ \psi_k(x_k^*, y_k^*) - \psi_k(0,0) &= -\frac{1}{\beta_1}\left(\beta_2^2 + \beta_3^2\right), \\ \partial\psi_k(0,0) &= 2[\beta_2,\ \beta_3]^{\mathsf{T}}. \end{aligned}$$

(iii) the elementary function ρ_k in (37) satisfies

$$\begin{aligned}\rho_k(x,y)-\rho_k(1,0)&=\gamma_1(x^2+y^2)+\gamma_2\frac{1}{x^2+y^2}-\gamma_1-\gamma_2,\\ \rho_k(x_k^*,y_k^*)-\rho_k(1,0)&=-\left(\sqrt{\gamma_1}-\sqrt{\gamma_2}\right)^2,\\ \partial\rho_k(1,0)&=2[\gamma_1-\gamma_2,\ 0]^{\mathsf{T}}.\end{aligned}$$

Remark 5.3. In the complex case, the solution $z_k^*=x_k^*+iy_k^*$ in (42) was earlier derived in [45, Eq. (8)]. In the real case, the solution x_k^* in (42) was earlier derived in [5, Eq. (7)].

Setting 5.4. In Algorithm 1 for cost function (5), when $\nu_k=\varphi_k$, we see that $x_k^*=0$ if $\alpha_1\neq 0$ and $\alpha_2=0$. It is not possible that $\alpha_1=0$ and $\alpha_2\neq 0$. If $\alpha_1=\alpha_2=0$, we set $x_k^*=0$. In the case of $\nu_k=\psi_k$, we make the similar settings for the value of y_k^*.

Setting 5.5. Let $0<\varsigma<\frac{1}{4}$ be a small positive constant. In Algorithm 1 for cost function (5), if $\nu_k=\rho_k$, we always set $y_k^*=0$. Moreover, we determine x_k^* based on the following rules.

- If $\gamma_1=\gamma_2=0$, we set $x_k^*=0$.
- Let $\varpi\overset{\mathsf{def}}{=}\frac{\gamma_2}{\gamma_1}$. If $\varpi\in[0,\varsigma)$, we set $x_k^*=\frac{1}{2}$. If $\varpi\in(\frac{1}{\varsigma},+\infty]$, we set $x_k^*=2$.
- Otherwise, if $\varpi\in[\varsigma,\frac{1}{\varsigma}]$, we set $x_k^*=\sqrt[4]{\varpi}$, which is the minimum point.

5.3. **Condition (26) for global convergence.** It will be seen that $f(\boldsymbol{\omega}_k)\leq f(\boldsymbol{\omega}_{k-1})$ always holds in Algorithm 1. In Algorithm 1 for cost function (5), we denote $\mathrm{M}_0\overset{\mathsf{def}}{=}f(\boldsymbol{\omega}_0)$. Then we have that $\gamma_1+\gamma_2\leq\mathrm{M}_0=f(\boldsymbol{\omega}_0)$.

Lemma 5.6. In Algorithm 1 for cost function (5), there exists $\iota_\rho>0$ such that

$$g(\boldsymbol{X}_{k-1})-g(\boldsymbol{X}_k)\geq\iota_\rho\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^*-\boldsymbol{I}_2\|, \tag{43}$$

whenever $\nu_k=\rho_k$.

Proof. We now prove the inequality (43) by Lemma 5.2(iii) in three different cases shown in Setting 5.5.

- If $\gamma_1=\gamma_2=0$, it is clear that the inequality (43) is satisfied for any $\iota_\rho>0$.
- If $\varpi\in[0,\varsigma)$, we get that

$$\begin{aligned}\rho_k(1,0)-\rho_k(x_k^*,y_k^*)&=\frac{3}{2}\gamma_1(1-4\varpi)=\frac{3(1-4\varpi)}{8(1-\varpi)}|\partial\rho_k(1,0)|\geq\frac{3(1-4\varpi)\varepsilon}{8(1-\varpi)}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\\ &\geq\frac{3(1-4\varpi)\varepsilon}{4\sqrt{5}(1-\varpi)}\frac{\sqrt{5}}{2}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\geq\frac{3(1-4\varpi)\varepsilon}{4\sqrt{5}(1-\varpi)}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^*-\boldsymbol{I}_2\|\\ &\geq\frac{3(1-4\varsigma)\varepsilon}{4\sqrt{5}}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^*-\boldsymbol{I}_2\|.\end{aligned}$$

- If $\varpi \in (\frac{1}{\varsigma}, +\infty]$, we similarly get that

$$\begin{aligned}\rho_k(1,0) - \rho_k(x_k^*, y_k^*) &= \frac{3}{2}\gamma_2(1-\frac{4}{\varpi}) = \frac{3(1-\frac{4}{\varpi})}{8(1-\frac{1}{\varpi})}|\partial\rho_k(1,0)| \geq \frac{3(1-\frac{4}{\varpi})\varepsilon}{8(1-\frac{1}{\varpi})}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\| \\ &\geq \frac{3(1-\frac{4}{\varpi})\varepsilon}{4\sqrt{5}(1-\frac{1}{\varpi})}\frac{\sqrt{5}}{2}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\| \geq \frac{3(1-\frac{4}{\varpi})\varepsilon}{4\sqrt{5}(1-\frac{1}{\varpi})}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\| \\ &\geq \frac{3(1-4\varsigma)\varepsilon}{4\sqrt{5}}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|.\end{aligned}$$

- If $\varpi \in [\varsigma, \frac{1}{\varsigma}]$, it is easy to verify that

$$\sqrt[4]{\gamma_1\gamma_2} \geq \frac{\sqrt[4]{\varsigma}}{2}\left(\sqrt{\gamma_1} + \sqrt{\gamma_2}\right). \tag{44}$$

Then, we get that

$$\begin{aligned}\rho_k(1,0) - \rho_k(x_k^*, y_k^*) &= 2\left(\sqrt{\gamma_1} - \sqrt{\gamma_2}\right)^2 = \frac{1}{2}|\partial\rho_k(1,0)|\frac{\left|\sqrt{\gamma_1} - \sqrt{\gamma_2}\right|}{\sqrt{\gamma_1} + \sqrt{\gamma_2}} \\ &\geq \frac{\varepsilon\sqrt[4]{\varsigma}}{4}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\frac{\left|\sqrt{\gamma_1} - \sqrt{\gamma_2}\right|}{\sqrt[4]{\gamma_1\gamma_2}} \quad \text{(by equation (44))} \\ &\geq \frac{\varepsilon\sqrt[4]{\varsigma}}{4}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\frac{\left|\sqrt[4]{\gamma_1} - \sqrt[4]{\gamma_2}\right|\left(\sqrt{\gamma_1} + \sqrt{\gamma_2}\right)^{1/2}}{\sqrt[4]{\gamma_1\gamma_2}} \\ &\geq \frac{\varepsilon\sqrt[4]{\varsigma}}{4}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\||x_k^*|\sqrt{1 + \frac{1}{(1+x_k^*)^2}} \geq \frac{\varepsilon\sqrt[4]{\varsigma}}{4}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|.\end{aligned}$$

Now we set $\iota_\rho = \min(\frac{3(1-4\varsigma)\varepsilon}{4\sqrt{5}}, \frac{\varepsilon\sqrt[4]{\varsigma}}{4})$. The proof is complete. □

Note that $\|\boldsymbol{X}_k - \boldsymbol{X}_{k-1}\| \leq \|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|\|\boldsymbol{X}_{k-1}\|$ and $\|\operatorname{grad} g(\boldsymbol{X}_{k-1})\| \leq \|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{X}_{k-1}\|$. Let ι_ρ be as in (43) and $\sigma_\rho = \frac{\iota_\rho}{\mathrm{M}_\omega^2} > 0$. Then the next result follows directly from Lemma 5.6.

Corollary 5.7. In Algorithm 1 for cost function (5), if the condition (17) is satisfied, then

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \sigma_\rho\|\operatorname{grad} g(\boldsymbol{X}_{k-1})\|\|\boldsymbol{X}_k - \boldsymbol{X}_{k-1}\|, \tag{45}$$

whenever $\nu_k = \rho_k$.

Lemma 5.8. In Algorithm 1 for cost function (5), there exists $\iota_\varphi > 0$ such that

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \iota_\varphi\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|, \tag{46}$$

whenever $\nu_k = \varphi_k$ or ψ_k.

Proof. We prove that the inequality (46) is satisfied in two cases.

- If $\nu_k = \varphi_k$, by Lemma 5.2(i), we see that

$$\begin{aligned}g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) &= \varphi_k(0,0) - \varphi_k(x_k^*, y_k^*) = 2\left(\frac{(\alpha_2)^2}{\alpha_1} + \frac{(\alpha_3)^2}{\alpha_1}\right) \\ &= \frac{1}{2}\|\partial\varphi_k(0,0)\|\|(x_k^*, y_k^*)\| \geq \frac{\varepsilon}{2}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|.\end{aligned}$$

- If $\nu_k = \psi_k$, by Lemma 5.2(ii), we see that

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) = \psi_k(0,0) - \psi_k(x_k^*, y_k^*) = 2\left(\frac{(\beta_2)^2}{\beta_1} + \frac{(\beta_3)^2}{\beta_1}\right)$$
$$= \frac{1}{2}\|\partial\psi_k(0,0)\|\|(x_k^*, y_k^*)\| \geq \frac{\varepsilon}{2}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|.$$

Now we set $\iota_\varphi = \frac{\varepsilon}{2}$. The proof is complete. □

Let ι_φ be as in (46) and $\sigma_\varphi = \frac{\iota_\varphi}{\mathrm{M}_\omega^2} > 0$. Similar as for Corollary 5.7, the next result follows directly from Lemma 5.8.

Corollary 5.9. In Algorithm 1 for cost function (5), if the condition (17) is satisfied, then

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \sigma_\varphi\|\operatorname{grad} g(\boldsymbol{X}_{k-1})\|\|\boldsymbol{X}_k - \boldsymbol{X}_{k-1}\|,$$

whenever $\nu_k = \varphi_k$ or ψ_k.

5.4. Condition (27) for weak convergence.

Lemma 5.10. In Algorithm 1 for cost function (5), if the condition (17) is satisfied, then there exists $\kappa_\rho > 0$ such that

$$\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\| \geq \kappa_\rho\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|,$$

whenever $\nu_k = \rho_k$, φ_k or ψ_k.

Proof. If $\nu_k = \varphi_k$, we have

$$\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|^2 = \frac{\alpha_2^2 + \alpha_3^2}{\alpha_1^2} \geq \frac{\|\partial\varphi_k(0,0)\|^2}{4\mathrm{M}_2^2} \geq \frac{\varepsilon^2}{4\mathrm{M}_2^2}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|^2.$$

The case $\nu_k = \psi_k$ is similar. Now we prove the $\nu_k = \rho_k$ case. If $\varpi \in [0, \varsigma)$, we have

$$\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|^2 = \frac{5}{4} \geq \frac{5}{4}\frac{1}{8(\gamma_1^2 + \gamma_2^2)}\|\partial\rho_k(1,0)\|^2 \geq \frac{5}{4}\frac{\varepsilon^2}{16\mathrm{M}_\omega^2}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|^2.$$

The case $\varpi \in (\frac{1}{\varsigma}, +\infty]$ is similar. If $\varpi \in [\varsigma, \frac{1}{\varsigma}]$, we have

$$\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|^2 \geq (1 - x_k^*)^2 \geq \frac{1}{\mathrm{M}_\omega^2\mathrm{M}_3^2}\gamma_1^2(1-\varpi)^2 = \frac{1}{4\mathrm{M}_\omega^2\mathrm{M}_3^2}\|\partial\rho_k(1,0)\|^2 \geq \frac{\varepsilon^2}{4\mathrm{M}_\omega^2\mathrm{M}_3^2}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|^2.$$

We only need to set $\kappa_\rho^2 = \min(\frac{\varepsilon^2}{4\mathrm{M}_2^2}, \frac{5}{4}\frac{\varepsilon^2}{16\mathrm{M}_\omega^2}, \frac{\varepsilon^2}{4\mathrm{M}_\omega^2\mathrm{M}_3^2})$. The proof is complete. □

By Lemma 5.6, Lemma 5.8 and Lemma 5.10, we can easily get the following results by setting $\eta_\rho = \sigma_\rho\kappa_\rho$ and $\eta_\varphi = \sigma_\varphi\kappa_\rho$.

Corollary 5.11. In Algorithm 1 for cost function (5), if the condition (17) is satisfied, then

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \eta_\rho\|\operatorname{grad} g(\boldsymbol{X}_{k-1})\|^2 \tag{47}$$

whenever $\nu_k = \rho_k$.

Corollary 5.12. In Algorithm 1 for cost function (5), if the condition (17) is satisfied, then

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \eta_\varphi\|\operatorname{grad} g(\boldsymbol{X}_{k-1})\|^2$$

whenever $\nu_k = \varphi_k$ or ψ_k.

6. Givens plane transformations for JADM problem

Let g be as in (39). Let $\boldsymbol{W}^{(\ell)}$ and ϱ be as in Section 5.2. Denote $(i,j)=(i_k,j_k)$ for simpliciy. Define

$$\boldsymbol{\Gamma}^{(i,j,\boldsymbol{X}_{k-1})} \overset{\mathsf{def}}{=} \frac{\varrho}{2}\sum_{\ell=1}^{L}\mu_\ell \Re\left(\boldsymbol{z}_{i,j}(\boldsymbol{W}^{(\ell)})\boldsymbol{z}_{i,j}^{\mathsf{H}}(\boldsymbol{W}^{(\ell)})\right),$$

where

$$\boldsymbol{z}_{i,j}(\boldsymbol{W}) \overset{\mathsf{def}}{=} \begin{cases} \begin{bmatrix} \mathrm{W}_{jj}-\mathrm{W}_{ii} & \mathrm{W}_{ij}+\mathrm{W}_{ji} & -i(\mathrm{W}_{ij}-\mathrm{W}_{ji}) \end{bmatrix}^{\mathsf{T}}, & \text{if } (\cdot)^{\blacklozenge}=(\cdot)^{\mathsf{H}};\\ \begin{bmatrix} \mathrm{W}_{ij}+\mathrm{W}_{ji} & \mathrm{W}_{ii}-\mathrm{W}_{jj} & i(\mathrm{W}_{ii}+\mathrm{W}_{jj}) \end{bmatrix}^{\mathsf{T}}, & \text{if } (\cdot)^{\blacklozenge}=(\cdot)^{\mathsf{T}}. \end{cases}$$

Let

$$c_0 = \begin{cases} \frac{1}{2}\sum_{\ell=1}^{L}\mu_\ell\left|\mathrm{W}_{jj}^{(\ell)}-\mathrm{W}_{ii}^{(\ell)}\right|^2, & \text{if } (\cdot)^{\blacklozenge}=(\cdot)^{\mathsf{H}};\\ -\frac{1}{2}\sum_{\ell=1}^{L}\mu_\ell\left|\mathrm{W}_{ij}^{(\ell)}+\mathrm{W}_{ji}^{(\ell)}\right|^2, & \text{if } (\cdot)^{\blacklozenge}=(\cdot)^{\mathsf{T}}. \end{cases}$$

6.1. Elementary function.

As in [44, Eq. (4.4)], we let

$$\boldsymbol{r}=\boldsymbol{r}(c,s_1,s_2) \overset{\mathsf{def}}{=} \begin{bmatrix} 2c^2-1, & -2cs_1, & -2cs_2 \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} \cos 2\theta, & -\sin 2\theta\cos\phi, & -\sin 2\theta\sin\phi \end{bmatrix}^{\mathsf{T}}. \tag{48}$$

Then we can get the following result[11] by direct calculations.

Lemma 6.1. In Algorithm 1 for cost function (5), the elementary function h_k satisfies

$$h_k(c,s_1,s_2)-h_k(1,0,0) = -\left(\boldsymbol{r}^{\mathsf{T}}\boldsymbol{\Gamma}^{(i,j,\boldsymbol{X}_{k-1})}\boldsymbol{r}-c_0\right). \tag{49}$$

Now we denote $\boldsymbol{\Gamma}=\boldsymbol{\Gamma}^{(i,j,\boldsymbol{X}_{k-1})}$ for simplicity. It follows by (48) and (49) that

$$h_k(c,s_1,s_2)-h_k(1,0,0) = -\left(q(\theta,\phi)-c_0\right), \tag{50}$$

where

$$\begin{aligned} q(\theta,\phi) \overset{\mathsf{def}}{=}& \frac{1}{2}\left(\Gamma_{11}-\Gamma_{22}\cos^2\phi-\Gamma_{33}\sin^2\phi-\Gamma_{23}\sin(2\phi)\right)\cos(4\theta)\\ &-\left(\Gamma_{12}\cos\phi+\Gamma_{13}\sin\phi\right)\sin(4\theta)+\frac{1}{2}\left(\Gamma_{11}+\Gamma_{22}\cos^2\phi+\Gamma_{33}\sin^2\phi+\Gamma_{23}\sin(2\phi)\right). \end{aligned} \tag{51}$$

Note that, by Lemma 6.1 and (48), we have

$$\partial h_k(\boldsymbol{I}_2) = -4[0\ \Gamma_{12}\ \Gamma_{13}]^{\mathsf{T}}. \tag{52}$$

Remark 6.2. By equation (50), we see that $h_k(\theta+\frac{\pi}{2},\phi)=h_k(\theta,\phi)$ for any $\theta,\phi\in\mathbb{R}$. Therefore, we can always choose $\theta_*\in[-\frac{\pi}{4},\frac{\pi}{4}]$.

Setting 6.3. In Algorithm 1 for cost function (5), we set a positive constant $\epsilon>0$. If $\nu_k=h_k$, we find the eigenvector $\boldsymbol{u}$ of $\boldsymbol{\Gamma}$ corresponding to the largest eigenvalue. Define two vectors $\boldsymbol{v}_{i,j}\overset{\mathsf{def}}{=}[\Gamma_{12}\ \Gamma_{13}]^{\mathsf{T}}\in\mathbb{R}^2$ and $\boldsymbol{w}_{i,j}\overset{\mathsf{def}}{=}[u_2\ u_3]^{\mathsf{T}}\in\mathbb{R}^2$.

- If it holds that
$$|\langle\boldsymbol{v}_{i,j},\boldsymbol{w}_{i,j}\rangle|\geq\epsilon\|\boldsymbol{v}_{i,j}\|\|\boldsymbol{w}_{i,j}\|, \tag{53}$$
then we find ϕ_* and θ_* by setting $\boldsymbol{r}=\boldsymbol{u}$, and $\boldsymbol{\Psi}_k^*=\boldsymbol{\Psi}(\theta_*,\phi_*)$;
- Otherwise, we set $[\cos\phi_*\ \sin\phi_*]^{\mathsf{T}}=\frac{\boldsymbol{v}_{i,j}}{\|\boldsymbol{v}_{i,j}\|}$ and then determine the best θ_* based on ϕ_*.

[11] In the $(\cdot)^{\blacklozenge}=(\cdot)^{\mathsf{H}}$ case, this expression was first formulated in [12].

6.2. **Condition (26) for global convergence.** We first present a lemma, which will help us to prove Lemma 6.5.

Lemma 6.4. Let $\alpha, \beta \in \mathbb{R}$ be two constants. For $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4}]$, we define

$$p(\theta) \stackrel{\mathsf{def}}{=} \alpha \cos(4\theta) + \beta \sin(4\theta).$$

If $p(\theta_*) = \max p(\theta)$, we have

$$p(\theta_*) - p(0) \geq 2\sqrt{2}|\beta||\sin(\frac{\theta_*}{2})|.$$

Lemma 6.5. Let the function $q(\theta, \phi)$ be as in (51). Suppose that ϕ_* and θ_* are determined as in Setting 6.3. Then we have

$$q(\theta_*, \phi_*) - q(0,0) \geq 2\sqrt{2}\epsilon|\sin(\frac{\theta_*}{2})|\|\boldsymbol{v}_{i,j}\|.$$

Proof. By Setting 6.3, we see that

$$|\langle \boldsymbol{v}_{i,j}, [\cos\phi_* \;\; \sin\phi_*]^{\mathsf{T}} \rangle| \geq \epsilon \|\boldsymbol{v}_{i,j}\| \tag{54}$$

always holds. By Lemma 6.4 and (54), we get that

$$\begin{aligned} q(\theta_*, \phi_*) - q(0,0) = q(\theta_*, \phi_*) - q(0, \phi_*) &\geq 2\sqrt{2}|\sin(\frac{\theta_*}{2})||\Gamma_{12}\cos\phi_* + \Gamma_{13}\sin\phi_*| \\ &\geq 2\sqrt{2}\epsilon|\sin(\frac{\theta_*}{2})|\|\boldsymbol{v}_{i,j}\|. \end{aligned}$$

The proof is complete. □

Lemma 6.6. In Algorithm 1 for cost function (5), there exists $\iota_h > 0$ such that

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \iota_h \|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\|, \tag{55}$$

whenever $\nu_k = h_k$.

Proof. We only prove the $(\cdot)^{\blacklozenge} = (\cdot)^{\mathsf{H}}$ case. The other case is similar. By Lemma 6.5 and (52), we get that

$$\begin{aligned} h_k(0,0) - h_k(\theta_*, \phi_*) &\geq 2\sqrt{2}\epsilon|\sin(\frac{\theta_*}{2})|\|\boldsymbol{v}_{i,j}\| = \frac{\epsilon}{4}2\sqrt{2}|\sin(\frac{\theta_*}{2})|\|\partial h_k(\boldsymbol{I}_2)\| \\ &\geq \frac{\epsilon\varepsilon}{4}\|\boldsymbol{G}^{(i,j,\boldsymbol{\Psi}_k^*)} - \boldsymbol{I}_m\|\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|. \end{aligned}$$

We can set $\iota_h = \frac{\epsilon\varepsilon}{4}$. The proof is complete. □

Let ι_h be as in (55) and $\sigma_h = \frac{\iota_h}{\mathrm{M}_\omega^2} > 0$. Similar as for Corollary 5.7, the next result follows directly from Lemma 6.6.

Corollary 6.7. In Algorithm 1 for cost function (5), if the condition (17) is always satisfied, then

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \sigma_h \|\operatorname{grad} g(\boldsymbol{X}_{k-1})\|\|\boldsymbol{X}_k - \boldsymbol{X}_{k-1}\|,$$

whenever $\nu_k = h_k$.

6.3. **Condition (27) for weak convergence.** If the condition (17) is satisfied, it is easy to see that there exists $\mathrm{M}_\Gamma > 0$ such that $\|\boldsymbol{\Gamma}^{(i,j,\boldsymbol{X}_{k-1})}\| \leq \mathrm{M}_\Gamma$ always hods.

Lemma 6.8. In Algorithm 1 for cost function (5), if the condition (17) is satisfied, then there exists $\kappa_h > 0$ such that

$$\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\| \geq \kappa_h \|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|, \tag{56}$$

whenever $\nu_k = h_k$.

Proof. If $\boldsymbol{v}_{i,j}$ and $\boldsymbol{w}_{i,j}$ satisfy the condition (53), then the inequality (56) can be proved by a similar method as for [44, Lemma 7.2]. Otherwise, if we set $[\cos\phi_* \ \sin\phi_*]^\mathsf{T} = \frac{\boldsymbol{v}_{i,j}}{\|\boldsymbol{v}_{i,j}\|}$ and find θ_* based on ϕ_*, then

$$\begin{aligned}|\sin(4\theta_*)| &= \frac{|\Gamma_{12}\cos\phi_* + \Gamma_{13}\sin\phi_*|}{\sqrt{\left(\Gamma_{12}\cos\phi_* + \Gamma_{13}\sin\phi_*\right)^2 + \frac{1}{4}\left(\Gamma_{11} - \Gamma_{22}\cos^2\phi_* - \Gamma_{33}\sin^2\phi_* - \Gamma_{23}\sin(2\phi_*)\right)^2}} \\ &\geq \frac{\sqrt{\Gamma_{12}^2 + \Gamma_{13}^2}}{2\sqrt{5}\mathrm{M}_\Gamma} = \frac{\|\partial h_k(\boldsymbol{I}_2)\|}{8\sqrt{5}\mathrm{M}_\Gamma} \geq \frac{\varepsilon}{8\sqrt{5}\mathrm{M}_\Gamma}\|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|.\end{aligned}$$

Note that

$$\|\boldsymbol{\Psi}_k^* - \boldsymbol{I}_2\| = 2\sqrt{2}|\sin(\frac{\theta_*}{2})| \geq \frac{\sqrt{2}}{4}|\sin(4\theta_*)|.$$

We only need to set $\kappa_h = \frac{\sqrt{2}\varepsilon}{32\sqrt{5}\mathrm{M}_\Gamma}$ in this case. The proof is complete. □

Then, by Lemma 6.6 and Lemma 6.8, we can easily get the following result by setting $\eta_h = \sigma_h\kappa_h$.

Corollary 6.9. In Algorithm 1 for cost function (5), if the condition (17) is satisfied, then

$$g(\boldsymbol{X}_{k-1}) - g(\boldsymbol{X}_k) \geq \eta_h \|\operatorname{grad} g(\boldsymbol{X}_{k-1})\|^2$$

whenever $\nu_k = h_k$.

7. Convergence analysis

7.1. Convergence analysis of BCD-GLU and BCD-GQU algorithms.

7.1.1. *Global convergence.* We first prove the following result about the global convergence of BCD-GLU and BCD-GQU algorithms for cost function (5).

Theorem 7.1. In BCD-GLU and BCD-GQU algorithms for cost function (5), if the condition (17) is satisfied, then the iterates $\{\boldsymbol{\omega}_k\}_{k\geq 1}$ produced by BCD-GLU and BCD-GQU algorithms converge to a point $\boldsymbol{\omega}_*$.

Proof. By Lemma 3.3, Corollary 5.7, Corollary 5.9 and Corollary 6.7, we see that the condition (26) is always satisfied in BCD-GLU and BCD-GQU algorithms for cost function (5), if we set $\sigma = \min(\sigma_p, \sigma_\rho, \sigma_\varphi, \sigma_h)$. Then the proof is complete by Proposition 2.4. □

7.1.2. *Weak convergence.* Now we prove the following result about the weak convergence of BCD-GLU and BCD-GQU algorithms for cost function (5).

Theorem 7.2. In BCD-GLU and BCD-GQU algorithms for cost function (5), if condition (17) is always satisfied, and $\boldsymbol{\omega}_*$ is an accumulation point of the iterates $\{\boldsymbol{\omega}_k\}_{k\geq 1}$, then $\boldsymbol{\omega}_*$ is a stationary point of the cost function (5).

Proof. By Equation (35), Corollary 5.11, Corollary 5.12 and Corollary 6.9, we see that the condition (27) is always satisfied in BCD-GLU and BCD-GQU algorithms for cost function (5), if we set $\eta = \min(\eta_p, \eta_\rho, \eta_\varphi, \eta_h)$. Then the proof is complete by Lemma 2.5. □

7.2. **Convergence analysis of BCD-GU algorithm.**

7.2.1. *Global convergence.* Similar as in Section 7.1, we have the following result about the global convergence of BCD-GU algorithm for cost function (5).

Theorem 7.3. In BCD-GU algorithm for cost function (5), if the condition (17) is satisfied, then the iterates $\{\boldsymbol{\omega}_k\}_{k\geq 1}$ produced by BCD-GU algorithm converge to a point $\boldsymbol{\omega}_*$.

7.2.2. *Weak convergence.* We also have the following result about the weak convergence of BCD-GU algorithm for cost function (5).

Theorem 7.4. In BCD-GU algorithm for cost function (5), if condition (17) is always satisfied, and $\boldsymbol{\omega}_*$ is an accumulation point of the iterates $\{\boldsymbol{\omega}_k\}_{k\geq 1}$, then $\boldsymbol{\omega}_*$ is a stationary point of the cost function (5).

7.3. **Jacobi-type algorithms and their convergence analysis.**

7.3.1. *Jacobi-GLU and Jacobi-GQU algorithms on* $\mathbf{SL}_m(\mathbb{C})$. Let $n = m$ in the cost function (8) and fix $\boldsymbol{U}_k = \boldsymbol{U}_0$ for $k \geq 1$ in Algorithm 1. In other words, we keep the first block variable unchanged and only update the second block variable $\boldsymbol{X}_k$ in $\mathbf{SL}_m(\mathbb{C})$ using GLU and GQU transformations[12] in Section 4.3. Then we get the *Jacobi-GLU* and *Jacobi-GQU* algorithms[13] to minimize the restricted function

$$g : \mathbf{SL}_m(\mathbb{C}) \to \mathbb{R}^+, \quad \boldsymbol{X} \mapsto f(\boldsymbol{U}_0, \boldsymbol{X}).$$

7.3.2. *Special cases: Jacobi-GLU-M and Jacobi-GQU-M algorithms.* In this subsection, we propose two natural variants of Jacobi-GLU and Jacobi-GQU algorithms, which will be called *Jacobi-GLU-M* and *Jacobi-GQU-M* algorithms, respectively. In these two algorithms, in each iteration, among all the index pairs (i_k, j_k) and elementary functions ν_k satisfying (57), we choose (i_k, j_k) and ν_k such that the cost function obtains the largest reduction.

7.3.3. *Global convergence.* Similar as in Section 7.1, we have the following result about the global convergence of Jacobi-GLU and Jacobi-GQU algorithms for cost function (6). It is clear that Theorem 7.5 also applies to Jacobi-GLU-M and Jacobi-GQU-M algorithms.

Theorem 7.5. In Algorithm 2 and Algorithm 3 for cost function (6), if condition (18) is always satisfied, then the iterates $\{\boldsymbol{X}_k\}_{k\geq 1}$ converge to a point $\boldsymbol{X}_*$.

[12] In this case, we do not consider the GU class of transformations.

[13] These algorithms are based on the similar ideas as the Jacobi-G type algorithms in [20, 24, 44].

Algorithm 2: Jacobi-GLU

1: **Input:** A starting point $\boldsymbol{X}_0$, a positive constant $0 < \varepsilon < \sqrt{\frac{2}{3m(m-1)}}$.
2: **Output:** Sequence of iterates $\{\boldsymbol{X}_k\}_{k\geq 1}$.
3: **for** $k = 1, 2, \cdots$, **do**
4: Choose the index pair (i_k, j_k) and ν_k such that

$$\|\partial \nu_k(\boldsymbol{I}_2)\| \geq \varepsilon \|\boldsymbol{\Lambda}(\boldsymbol{X}_{k-1})\|, \tag{57}$$

where $\nu_k = \varphi_k, \psi_k$ or ρ_k;
5: Compute $\boldsymbol{\Psi}_k^*$ that minimizes the function ν_k;
6: Update $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}\boldsymbol{V}_k$, where $\boldsymbol{V}_k = \boldsymbol{U}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}, \boldsymbol{L}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$ or $\boldsymbol{D}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$.
7: **end for**

Algorithm 3: Jacobi-GQU

1: **Input:** A starting point $\boldsymbol{X}_0$, a positive constant $0 < \varepsilon < \sqrt{\frac{3+\sqrt{5}}{3m(m-1)}}$.
2: **Output:** Sequence of iterates $\{\boldsymbol{X}_k\}_{k\geq 1}$.
3: **for** $k = 1, 2, \cdots$, **do**
4: Choose the index pair (i_k, j_k) and ν_k satisfying (57), where $\nu_k = h_k, \psi_k$ or ρ_k;
5: Compute $\boldsymbol{\Psi}_k^*$ that minimizes the function ν_k;
6: Update $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}\boldsymbol{V}_k$, where $\boldsymbol{V}_k = \boldsymbol{G}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}, \boldsymbol{L}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$ or $\boldsymbol{D}^{(i_k,j_k,\boldsymbol{\Psi}_k^*)}$.
7: **end for**

7.3.4. *Weak convergence.* We also have the following result about the weak convergence of Jacobi-GLU and Jacobi-GQU algorithms for cost function (6).

Theorem 7.6. In Algorithm 2 and Algorithm 3 for cost function (6), if condition (18) is always satisfied, and $\boldsymbol{X}_*$ is an accumulation point of the iterates $\{\boldsymbol{X}_k\}_{k\geq 1}$, then $\boldsymbol{X}_*$ is a stationary point of the cost function (6).

Remark 7.7. In Algorithm 2 and Algorithm 3, a more natural way of choosing the index pair (i_k, j_k) is according to a cyclic ordering. In fact, this cyclic way has been often used in the literature [35, 41, 45]. In this case, we call them the *Jacobi-CLU* (based on LU decomposition) and *Jacobi-CQU* (based on QU decomposition) algorithms, respectively. To our knowledge, there has been no theoretical result about the convergence of Jacobi-CLU and Jacobi-CQU algorithms in the literature. In this paper, we propose Jacobi-GLU and Jacobi-GQU algorithms, and establish their convergence. It may be also interesting to study how to establish the convergence of Jacobi-CLU and Jacobi-CQU algorithms.

8. Experiments

In this section, we conduct some numerical experiments to compare the performances of Jacobi-GLU in Algorithm 2, Jacobi-GLU-M in Section 7.3.2, Jacobi-CLU in Remark 7.7, Jacobi-GQU in Algorithm 3, Jacobi-GQU-M in Section 7.3.2, Jacobi-CQU in Remark 7.7, Jacobi-GQ and Jacobi-CQ algorithms. Here, we denote by Jacobi-GQ the gradient based Jacobi-type

algorithm on unitary group proposed in [44], and by Jacobi-CQ the Jacobi-type algorithm on unitary group with a cyclic ordering. Note that Jacobi-GQ and Jacobi-CQ find the iterations only in $\mathbf{U}_n$, not in $\mathbf{SL}_n(\mathbb{C})$. All the algorithms stop after 1000 iterations.

Example 8.1. For the following sets of complex matrices, we set $\mu_\ell = 1$ and $(\cdot)^\blacklozenge = (\cdot)^{\mathsf{H}}$ in the cost function (6), and run the eight Jacobi-type algorithms to minimize it. The values of cost function (6) in the iterations are shown in Figure 1.
(i) We randomly generate two complex matrices $\{\boldsymbol{A}_\ell\}_{1\le\ell\le 2} \subseteq \mathbb{C}^{5\times 5}$.
(ii) We randomly generate a complex matrix $\boldsymbol{X} \in \mathbb{C}^{10\times 10}$, and set $\boldsymbol{A}^{(\ell)} = \boldsymbol{X}^{\mathsf{H}}(\boldsymbol{I}_{10} + \boldsymbol{e}_\ell^{\mathsf{T}}\boldsymbol{e}_\ell)\boldsymbol{X}$ for $1 \le \ell \le 10$.
(iii) We randomly generate a complex upper triangular matrix $\boldsymbol{X} \in \mathbf{UT}_{10}(\mathbb{C})$, complex diagonal matrices $\{\boldsymbol{D}_\ell\}_{1\le\ell\le 10} \subseteq \mathbb{C}^{10\times 10}$, and set $\boldsymbol{A}^{(\ell)} = \boldsymbol{X}^{\mathsf{H}}\boldsymbol{D}_\ell\boldsymbol{X}$ for $1 \le \ell \le 10$.
(iv) We randomly generate a complex nonsingular matrix $\boldsymbol{X} \in \mathbf{SL}_{10}(\mathbb{C})$, complex diagonal matrices $\{\boldsymbol{D}_\ell\}_{1\le\ell\le 10} \subseteq \mathbb{C}^{10\times 10}$, and set $\boldsymbol{A}^{(\ell)} = \boldsymbol{X}^{\mathsf{H}}\boldsymbol{D}_\ell\boldsymbol{X}$ for $1 \le \ell \le 10$.

Example 8.2. Let $\{\boldsymbol{A}_\ell\}_{1\le\ell\le L} \subseteq \mathbb{C}^{n\times n}$ be randomly generated with different n and L values as shown in Table 2. Let $\mu_\ell = 1$ and $(\cdot)^\blacklozenge = (\cdot)^{\mathsf{H}}$ in the cost function (6). We run the eight Jacobi-type algorithms to minimize it, and get the final values of cost function (6) in Table 2. In this table, the numbers appearing in bold indicate that this number is the best result in the row where it lies.

Table 2. Cost function values in Example 8.2

		GLU-M	GLU	CLU	GQU-M	GQU	CQU	GQ	CQ
$n=5$	$L=1$	0.0670	0.0133	**0.0000**	0.0229	0.0921	0.0692	0.8098	0.8098
$n=5$	$L=2$	**1.1933**	1.8995	1.9069	2.0520	2.1803	2.3514	3.3850	3.3850
$n=5$	$L=5$	11.3882	11.3908	11.3967	**11.3880**	11.3883	11.4016	13.3835	13.3835
$n=10$	$L=1$	0.3070	**0.2220**	0.3229	0.6100	0.6640	1.5259	3.2546	3.2546
$n=10$	$L=5$	**47.5993**	48.5667	50.2540	47.6150	48.3432	48.1588	54.0629	53.8712
$n=10$	$L=10$	119.0841	120.8933	121.9417	**115.7748**	117.8872	119.3527	123.2703	123.2546

From these experimental results, we can see that, compared with the Jacobi-GQ and Jacobi-CQ algorithms on unitary group, the Jacobi-type algorithms on $\mathbf{SL}_m(\mathbb{C})$ considered in this paper always obtain smaller cost function values, and they need more iterations to attain the best cost functions values. Moreover, compared with the QU decomposition based Jacobi-type algorithms, the LU decomposition based ones obtain better experimental results in most cases.

9. Conclusions

In this paper, to solve JADM problem (3), which is important in blind source separation, we formulate two different equivalent formulations, *i.e.* problem (5) defined on $\mathbf{St}(m,n,\mathbb{C}) \times \mathbf{SL}_m(\mathbb{C})$, and problem (6) defined on $\mathbf{SL}_m(\mathbb{C})$. Then, for these two approaches, we propose three

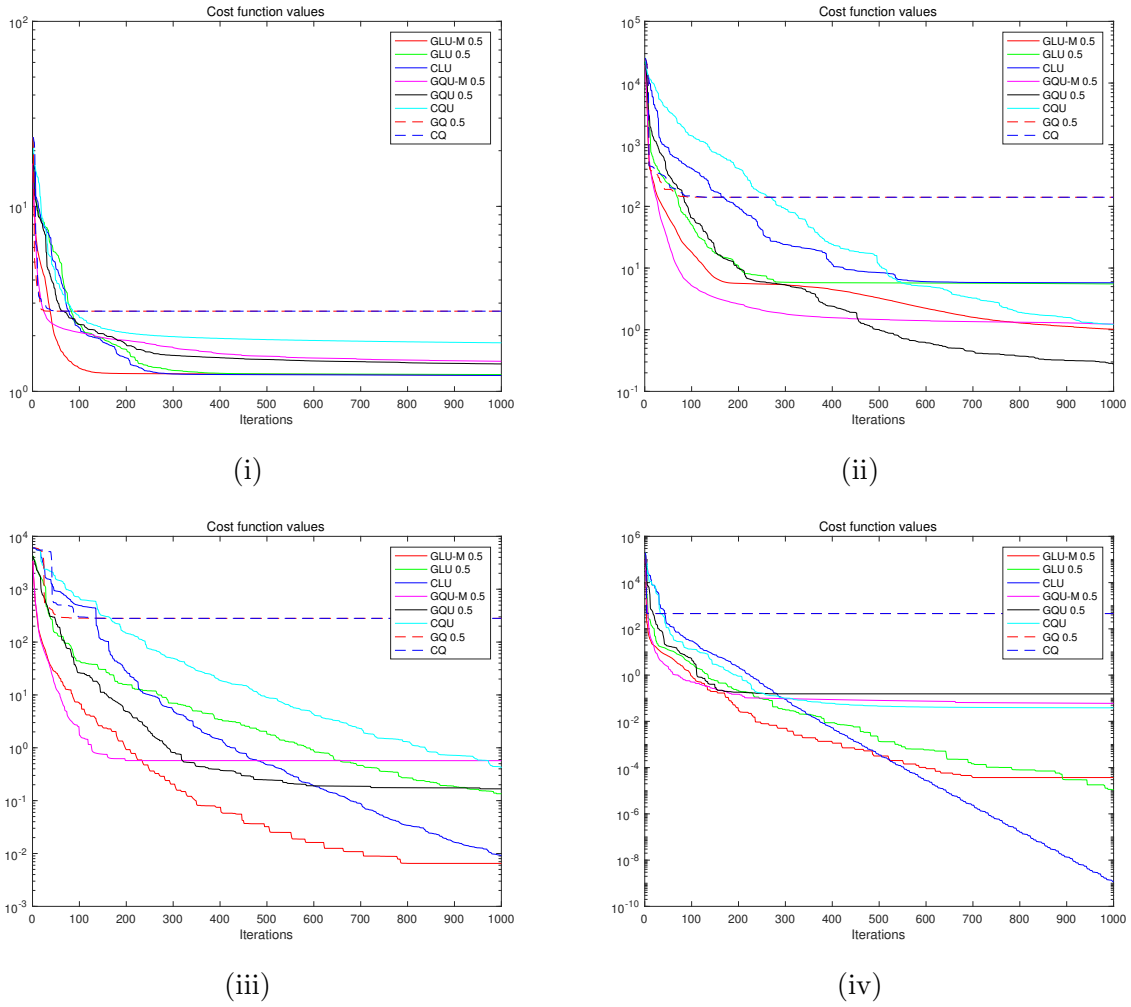

FIGURE 1. Cost function values in Example 8.1.

BCD-G algorithms and two Jacobi-type algorithms, and establish their global and weak convergence. In the future work, it may be interesting to extend these algorithms and convergence results to the higher order tensor cases.

REFERENCES

[1] T. E. ABRUDAN, J. ERIKSSON, AND V. KOIVUNEN, *Steepest descent algorithms for optimization under unitary matrix constraint*, IEEE Transactions on Signal Processing, 56 (2008), pp. 1134–1147.

[2] P.-A. ABSIL, R. MAHONY, AND B. ANDREWS, *Convergence of the iterates of descent methods for analytic cost functions*, SIAM Journal on Optimization, 16 (2005), pp. 531–547.

[3] P.-A. ABSIL, R. MAHONY, AND R. SEPULCHRE, *Optimization algorithms on matrix manifolds*, Princeton University Press, 2009.

[4] B. AFSARI, *Gradient flow-based matrix joint diagonalization for independent component analysis*, University of Maryland, College Park, 2004.

[5] B. Afsari, *Simple LU and QR based non-orthogonal matrix joint diagonalization*, in International Conference on Independent Component Analysis and Signal Separation, Springer, 2006, pp. 1–7.
[6] R. André, X. Luciani, and E. Moreau, *A new class of block coordinate algorithms for the joint eigenvalue decomposition of complex matrices*, Signal Processing, 145 (2018), pp. 78–90.
[7] A. Baker, *Matrix groups: An introduction to Lie group theory*, Springer Science & Business Media, 2012.
[8] D. P. Bertsekas, *Nonlinear programming*, Athena Scientific, second ed., 1999.
[9] F. Bouchard, B. Afsari, J. Malick, and M. Congedo, *Approximate joint diagonalization with Riemannian optimization on the general linear group*, SIAM Journal on Matrix Analysis and Applications, 41 (2020), pp. 152–170.
[10] D. Brandwood, *A complex gradient operator and its application in adaptive array theory*, IEE Proceedings H-Microwaves, Optics and Antennas, 130 (1983), pp. 11–16.
[11] J. Cardoso and A. Souloumiac, *Blind beamforming for non-gaussian signals*, IEE Proceedings F (Radar and Signal Processing), 6 (1993), pp. 362–370.
[12] J.-F. Cardoso and A. Souloumiac, *Jacobi angles for simultaneous diagonalization*, SIAM Journal on Matrix Analysis and Applications, 17 (1996), pp. 161–164.
[13] G. Chabriel, M. Kleinsteuber, E. Moreau, H. Shen, P. Tichavsky, and A. Yeredor, *Joint matrices decompositions and blind source separation: A survey of methods, identification, and applications*, IEEE Signal Processing Magazine, 31 (2014), pp. 34–43.
[14] B. Chen, S. He, Z. Li, and S. Zhang, *Maximum block improvement and polynomial optimization*, SIAM Journal on Optimization, 22 (2012), pp. 87–107.
[15] P. Comon, *Independent Component Analysis*, in Higher Order Statistics, J.-L. Lacoume, ed., Elsevier, Amsterdam, London, 1992, pp. 29–38.
[16] P. Comon, *Independent component analysis, a new concept?*, Signal Processing, 36 (1994), pp. 287–314.
[17] P. Comon and C. Jutten, eds., *Handbook of Blind Source Separation*, Academic Press, Oxford, 2010.
[18] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, third ed., 1996.
[19] B. Hall, *Lie groups, Lie algebras, and representations: an elementary introduction*, vol. 222, Springer, 2015.
[20] M. Ishteva, P.-A. Absil, and P. Van Dooren, *Jacobi algorithm for the best low multilinear rank approximation of symmetric tensors*, SIAM Journal on Matrix Analysis and Applications, 2 (2013), pp. 651–672.
[21] S. Krantz and H. Parks, *A Primer of Real Analytic Functions*, Birkhäuser Boston, 2002.
[22] S. G. Krantz, *Function theory of several complex variables*, vol. 340, American Mathematical Soc., 2001.
[23] S. law Lojasiewicz, *Ensembles semi-analytiques*, IHES notes, (1965).
[24] J. Li, K. Usevich, and P. Comon, *Globally convergent Jacobi-type algorithms for simultaneous orthogonal symmetric tensor diagonalization*, SIAM Journal on Matrix Analysis and Applications, 39 (2018), pp. 1–22.
[25] J. Li, K. Usevich, and P. Comon, *On approximate diagonalization of third order symmetric tensors by orthogonal transformations*, Linear Algebra and its Applications, 576 (2019), pp. 324–351.
[26] J. Li, K. Usevich, and P. Comon, *On the convergence of jacobi-type algorithms for independent component analysis*, in 2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM), IEEE, 2020, pp. 1–5.
[27] J. Li, K. Usevich, and P. Comon, *Jacobi-type algorithm for low rank orthogonal approximation of symmetric tensors and its convergence analysis*, Pacific Journal of Optimization, 17 (2021), pp. 357–379.
[28] J. Li and S. Zhang, *Polar decomposition based algorithms on the product of Stiefel manifolds with applications in tensor approximation*, arXiv:1912.10390v2, (2020).
[29] Z. Li, A. Uschmajew, and S. Zhang, *On convergence of the maximum block improvement method*, SIAM Journal on Optimization, 25 (2015), pp. 210–233.
[30] S. Łojasiewicz, *Sur la géométrie semi- et sous-analytique*, Annales de l'institut Fourier, 43 (1993), pp. 1575–1595.
[31] Z.-Q. Luo and P. Tseng, *On the convergence of the coordinate descent method for convex differentiable minimization*, Journal of Optimization Theory and Applications, 72 (1992), pp. 7–35.

[32] Z.-Q. Luo and P. Tseng, *Error bounds and convergence analysis of feasible descent methods: a general approach*, Annals of Operations Research, 46 (1993), pp. 157–178.

[33] J. H. Manton, *Modified steepest descent and Newton algorithms for orthogonally constrained optimisation. part i. the complex Stiefel manifold*, in Proceedings of the Sixth International Symposium on Signal Processing and its Applications, vol. 1, IEEE, 2001, pp. 80–83.

[34] V. Maurandi, C. De Luigi, and E. Moreau, *Fast jacobi like algorithms for joint diagonalization of complex symmetric matrices*, in 21st European Signal Processing Conference (EUSIPCO 2013), IEEE, 2013, pp. 1–5.

[35] V. Maurandi and E. Moreau, *A decoupled Jacobi-like algorithm for non-unitary joint diagonalization of complex-valued matrices*, IEEE Signal Processing Letters, 21 (2014), pp. 1453–1456.

[36] V. Maurandi, E. Moreau, and C. De Luigi, *Jacobi like algorithm for non-orthogonal joint diagonalization of hermitian matrices*, in 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2014, pp. 6196–6200.

[37] J. Nocedal and S. Wright, *Numerical optimization*, Springer Science & Business Media, 2006.

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

[39] H. Sato and T. Iwai, *A new, globally convergent Riemannian conjugate gradient method*, Optimization, 64 (2015), pp. 1011–1031.

[40] R. Schneider and A. Uschmajew, *Convergence results for projected line-search methods on varieties of low-rank matrices via łojasiewicz inequality*, SIAM Journal on Optimization, 25 (2015), pp. 622–646.

[41] M. Sørensen, P. Comon, S. Icart, and L. Deneire, *Approximate tensor diagonalization by invertible transforms*, in 2009 17th European Signal Processing Conference, IEEE, 2009, pp. 500–504.

[42] A. Souloumiac, *Nonorthogonal joint diagonalization by combining givens and hyperbolic rotations*, IEEE Transactions on Signal Processing, 57 (2009), pp. 2222–2231.

[43] A. Uschmajew, *A new convergence proof for the higher-order power method and generalizations*, Pacific Journal of Optimization, 11 (2015), pp. 309–321.

[44] K. Usevich, J. Li, and P. Comon, *Approximate matrix and tensor diagonalization by unitary transformations: convergence of jacobi-type algorithms*, SIAM Journal on Optimization, 30 (2020), pp. 2998–3028.

[45] K. Wang, X.-F. Gong, and Q.-H. Lin, *Complex non-orthogonal joint diagonalization based on LU and LQ decompositions*, in International Conference on Latent Variable Analysis and Signal Separation, Springer, 2012, pp. 50–57.

[46] S. J. Wright, *Coordinate descent algorithms*, Mathematical Programming, 151 (2015), pp. 3–34.

[47] Y. Xu and W. Yin, *A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion*, SIAM Journal on Imaging Sciences, 6 (2013), pp. 1758–1789.

[48] A. Yeredor, *Non-orthogonal joint diagonalization in the least-squares sense with application in blind source separation*, IEEE Transactions on Signal Processing, 50 (2002), pp. 1545–1553.