Department of Policy and Planning Sciences

Discussion Paper Series

No.1377

Completely Positive Factorization in Orthogonality Optimization via Smoothing Method

by

Zhijian LAI and Akiko YOSHISE

July 2021

UNIVERSITY OF TSUKUBA
Tsukuba, Ibaraki 305-8573
JAPAN

Completely Positive Factorization in Orthogonality Optimization via Smoothing Method*

ZHIJIAN LAI† AKIKO YOSHISE‡

July, 2021

Abstract

We examine the problem of completely positive (CP) factorization for a given completely positive matrix. We are inspired by the idea of Groetzner and Dür in 2020, wherein the CP factorization problem can be reformulated as an equivalent feasibility problem containing an orthogonality constraint. We begin by solving this feasibility problem through the use of a special case of the Riemannian smoothing steepest descent method proposed by Zhang et al. in 2021. To apply it to the CP factorization problem, we employ a smooth approximation function, named LogSumExp, as the maximum function. Numerical experiments show the efficiency of our method especially for large-scale factorizations.

Keywords: completely positive factorization, orthogonality constrained problem, nonsmooth optimization, smoothing method, Stiefel manifold

AMS: 15A23, 15B48, 90C48, 90C59

1 Introduction

The space of $n \times n$ real symmetric matrices $\mathcal{S}_n$ is endowed with the trace inner product $\langle A, B\rangle := \mathrm{trace}(AB)$. A matrix $A \in \mathcal{S}_n$ is called *completely positive* if for some $r \in \mathbb{N}$ there exists an entrywise nonnegative matrix $B \in \mathbb{R}^{n\times r}$ such that $A = BB^\top$, and we call B a *CP factorization* of A. We define $\mathcal{CP}_n$ as the set of all $n \times n$ completely positive matrices, equivalently characterized as

$$\mathcal{CP}_n = \{BB^\top \in \mathcal{S}_n \mid B \text{ is a nonnegative matrix } \} = \mathrm{conv}\{xx^\top \mid x \in \mathbb{R}^n_+\},$$

where $\mathrm{conv}(S)$ denotes the convex hull of a given set S. We also denote the set of all $n \times n$ copositive matrices by

$$\mathcal{COP}_n := \{A \in \mathcal{S}_n \mid x^\top Ax \geq 0 \text{ for all } x \in \mathbb{R}^n_+\}.$$

It is known that $\mathcal{COP}_n$ and $\mathcal{CP}_n$ are duals of each other under the trace inner product [27, Theorem 2.6]. Both $\mathcal{CP}_n$ and $\mathcal{COP}_n$ are proper convex cones [20, Chapter 5]. For any positive integer n, we have the following inclusion relationship among other important cones in conic optimization:

$$\mathcal{CP}_n \subseteq \mathcal{S}^+_n \cap \mathcal{N}_n \subseteq \mathcal{S}^+_n \subseteq \mathcal{S}^+_n + \mathcal{N}_n \subseteq \mathcal{COP}_n,$$

where $\mathcal{S}^+_n$ is the cone of $n\times n$ symmetric positive semidefinite and $\mathcal{N}_n$ is the cone of $n\times n$ symmetric nonnegative matrices. See the monograph [1] for a comprehensive description of $\mathcal{CP}_n$ and $\mathcal{COP}_n$.

*This research was supported by the Japan Society for the Promotion of Science through a Grant-in-Aid for Scientific Research ((B)19H02373) from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

†Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan. E-mail:s2130117@s.tsukuba.ac.jp

‡Corresponding author. Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan. E-mail:yoshise@sk.tsukuba.ac.jp

1.1 Applications and open problems

Conic optimization is a subfield of convex optimization that studies the minimization of linear functions over proper cones. Here, if the proper cone is $\mathcal{CP}_n$ or its dual cone $\mathcal{COP}_n$, we call the conic optimization problem a *copositive programming* problem.

The copositive programming is closely related to many nonconvex, NP-hard quadratic and combinatorial optimizations. For example, consider the so-called standard quadratic optimization, i.e.,

$$\min\{x^\top Mx \mid \mathbf{e}^\top x = 1, x \in \mathbb{R}^n_+\}, \tag{1}$$

where $M \in \mathcal{S}_n$ is possibly not positive semidefinite and $\mathbf{e}$ is the all-ones vector. Bomze et al. [8] showed that the following completely positive reformulation,

$$\min\{\langle M, X\rangle \mid \langle E, X\rangle = 1, X \in \mathcal{CP}_n\},$$

where E is the all-ones matrix, is equivalent to (1). Burer [12] reported a more general result, where any quadratic problem with binary and continuous variables can be rewritten as a linear program over $\mathcal{CP}_n$. As an application to combinatorial problems, consider the problem of computing the independence number $\alpha(G)$ of a graph G with n nodes. De Klerk and Pasechnik [18] showed that

$$\alpha(G) = \max\{\langle E, X\rangle \mid \langle A + I, X\rangle = 1, X \in \mathcal{CP}_n\},$$

where A is the adjacency matrix of G. For surveys on the applications of copositive programming, see [5, 9, 13, 24].

The difficulty of the above problems lies entirely in the completely positive conic constraint. Note that neither $\mathcal{COP}_n$ nor $\mathcal{CP}_n$ is self-dual implies that the primal-dual interior point method for conic optimization does not work as it is. Besides this, there are many fundamental open problems related to completely positive cones. A typical one is the checking membership in $\mathcal{CP}_n$, which was shown to be NP-hard by [22]. Computing or estimating the cp-rank as defined later in (3) is also an open problem. We refer the reader to [4, 24] for a more detailed discussion of those unresolved issues.

In this paper we focus on finding a CP factorization for a given $A \in \mathcal{CP}_n$, i.e., the *CP factorization problem*:

$$\text{Find } B \in \mathbb{R}^{n\times r} \text{ s.t. } A = BB^\top \text{ and } B \geq 0, \tag{CPfact}$$

which seems to be closely related to the membership problem $A \in \mathcal{CP}_n$. In fact, as we will describe later, the choice of r above has no intrinsic effect. Sometimes, a matrix has been shown to be completely positive through duality, or rather, $\langle A, X\rangle \geqslant 0$ for all $X \in \mathcal{COP}_n$, but in this case, a CP factorization will not necessarily be obtained.

1.2 Related work on CP factorization

Various methods of solving CP factorization problems have been studied. Jarre and Schmallowsky [30] stated a criterion of complete positivity, based on the augmented primal dual method to solve a particular second-order cone problem. Dickinson and Dür [21] dealt with complete positivity of matrices that possess a specific sparsity pattern, and proposed a method for finding CP factorizations of these special matrices, which can be performed in linear time. Nie [34] formulated the CP factorization problem as a case of the $\mathcal{A}$-truncated K-moment problem, for which the author developed an algorithm that solves a series of semidefinite optimization problems. Sponsel and Dür [42] considered the problem of projecting a matrix onto $\mathcal{CP}_n$ and $\mathcal{COP}_n$ by using polyhedral approximations of these cones. With the help of these projections, they devised a method to compute a CP factorization for any matrix in the interior of $\mathcal{CP}_n$. Given a known CP factorization of an $(n-1)\times(n-1)$ matrix A, Bomze [6] extended it to a CP factorization of an $n \times n$ matrix containing A as a principal submatrix. Sikirić, Schürmann and Vallentin [40] developed a simplex-like method for a rational CP factorization that works if the input matrix allows a rational CP factorization.

In 2020, Groetzner and Dür [26] applied the alternating projection method to the CP factorization problem under an equivalent feasibility problem (see (FeasCP) later). Shortly afterwards, Chen and Pong et al. [14] reformulated the split feasibility problem as a difference-of-convex

optimization problem and solved (FeasCP) as a specific application. In fact, we will solve this equivalent feasibility problem (FeasCP) by other means in this paper. In 2021, Boţ and Nguyen [11] proposed a projected gradient method with relaxation and inertia parameters for the CP factorization problem, aimed at solving

$$\min_X\{\|A - XX^\top\|^2 \mid X \in \mathbb{R}^{n\times r}_+ \cap \mathcal{B}(0, \sqrt{\text{trace}(A)})\}, \tag{2}$$

where $\mathcal{B}(0,\varepsilon) := \{X \in \mathbb{R}^{n\times r} \mid \|X\| \le \varepsilon\}$ is the closed ball centered at 0. The authors argued that its optimal value is zero if and only if $A \in \mathcal{CP}_n$.

1.3 Riemannian optimization

Minimization of a real-valued function over Riemannian manifolds, called *Riemannian optimization*, has been actively studied during the last few decades. In particular, the *Stiefel manifold* (i.e., *orthogonality constraint*)

$$\text{St}(n,p) = \{X \in \mathbb{R}^{n\times p} \mid X^\top X = I\}$$

is an important case of Riemannian manifolds where optimization over it is called *orthogonality optimization*; it is our main interest here. Various Riemannian optimization algorithms have been proposed that extend concepts and techniques used in Euclidean space to Riemannian manifolds; see [36, 29, 44, 31, 48, 38], for example. In particular, the CP factorization problem can amount to a minimization of a nonsmooth function over the Stiefel manifold (i.e., *nonsmooth orthogonality optimization*), for which variants of subgradient methods [10, 23], proximal gradient methods [17], alternating direction method of multipliers (ADMM) [43, 32, 33] have been studied on Riemannian manifolds.

Smoothing methods [15], using a parameterized smoothing function to approximate the objective function, are effective on a class of nonsmooth unconstrained optimizations. Recently, Zhang et al. [46] constructed a smoothing method for the Riemannian submanifolds of $\mathbb{R}^n$, named Riemannian smoothing steepest descent method, to minimize nonsmooth functions on submanifolds. In this paper, we apply this method to the CP factorization problem.

1.4 Our contribution

Inspired by the idea of Groetzner and Dür [26], wherein (CPfact) is equivalent to a feasibility problem containing an orthogonality constraint, we treat the latter as a nonsmooth orthogonality optimization and solve it through the use of a combination of the curvilinear search method [44] and the smoothing method [15]. Actually, that combination is precisely a special case of the Riemannian smoothing steepest descent method [46]. Without resorting to the complicated notation and terminology of Riemannian geometry such as Riemannian gradients and retractions, we elaborate the smoothing method for nonsmooth orthogonality optimization in the view of the usual Euclidean optimization, which allows us to build our toolbox rapidly and sufficiently to handle the CP factorization problem. Our contributions are summarized as follows:

1. To the best of our knowledge, this study is the first to handle CP factorization via orthogonality optimization, or rather, Riemannian optimization, for which various techniques have been developed.

2. Numerical experiments clarify that our method is competitive with other efficient CP factorization methods, especially for large-scale matrices.

1.5 Organization of the paper

In section 2, we review the method to reconstruct (CPfact) into another feasibility problem and we will use take a different approach to this problem. In section 3, the smoothing method is adapted to the nonsmooth orthogonality optimization in the view of the usual Euclidean optimization. To apply it for to the CP factorization problem, we employ a smoothing function named *LogSumExp* in section 4. Finally, section 5 is a collection of our numerical experiments.

2 Preliminaries

2.1 cp-rank and cp-plus-rank

First, let us recall some basic properties of completely positive matrices. Generally, one can have many CP factorizations for a given A even if the numbers of columns are distinct, which gives rise to the following definition. Denote by $\mathrm{cp}(A)$ the cp-rank of $A \in \mathcal{S}_n$; it is defined as

$$\mathrm{cp}(A) := \min_B \{r \in \mathbb{N} | A = BB^\top, B \in \mathbb{R}^{n\times r}, B \geq 0\}, \tag{3}$$

and $\mathrm{cp}(A) = \infty$ if $A \notin \mathcal{CP}_n$. Similarly, we can define the cp-plus-rank:

$$\mathrm{cp}^+(A) := \min_B \{r \in \mathbb{N} | A = BB^\top, B \in \mathbb{R}^{n\times r}, B > 0\}.$$

Immediately, for all $A \in \mathcal{S}_n$ we have

$$\mathrm{rank}(A) \leq \mathrm{cp}(A) \leq \mathrm{cp}^+(A). \tag{4}$$

Every CP factorization B of A is of the same rank as A since $\mathrm{rank}(XX^\top) = \mathrm{rank}(X)$ holds for any matrix X. The first inequality of (4) comes from that for any CP factorization B,

$$\mathrm{rank}(A) = \mathrm{rank}(B) \leq \text{ the number of columns of } B.$$

The second is trivial by definition.

Note that computing or estimating the cp-rank of any given $A \in \mathcal{CP}_n$ is still an open problem. The following result gives a tight upper bound of the cp-rank for $A \in \mathcal{CP}_n$ in terms of the order n.

Theorem 2.1 (Bomze, Dickinson, and Still [7, Theorem 4.1]). *For all $A \in \mathcal{CP}_n$, we have*

$$\mathrm{cp}(A) \leq \mathrm{cp}_n := \begin{cases} n & \text{for } n \in \{2,3,4\} \\ \frac{1}{2}n(n+1) - 4 & \text{for } n \geq 5. \end{cases}$$

The following result is useful for distinguishing completely positive matrices in either the interior or on the boundary of $\mathcal{CP}_n$.

Theorem 2.2 (Dickinson [19, Theorem 3.8]). *We have*

$$\begin{aligned} \mathrm{int}(\mathcal{CP}_n) &= \{A \in \mathcal{S}_n \mid \mathrm{rank}(A) = n, \mathrm{cp}^+(A) < \infty\} \\ &= \{A \in \mathcal{S}_n \mid \mathrm{rank}(A) = n, A = BB^\top, B \in \mathbb{R}^{n\times r}, B \geq 0, \\ &\quad b_j > 0 \text{ for at least one column } b_j \text{ of } B\}. \end{aligned}$$

2.2 CP factorization as an equivalent feasibility problem

Groetzner and Dür [26] reformulated the CP factorization problem as a equivalent feasibility problem containing an orthogonality constraint.

Given $A \in \mathcal{CP}_n$, if we have had a CP factorization B with r columns, then we can easily get another CP factorization $\widehat{B}$ with r' columns for every integer $r' \geq r$. The simplest way to construct such an $n \times r'$ matrix $\widehat{B}$ is to append $k := r' - r$ zero columns to B, i.e., $\widehat{B} := [B, 0_{n\times k}] \geq 0$. Another way is called *column replication*, i.e.,

$$\widehat{B} := [b_1, \ldots, b_{n-1}, \underbrace{\frac{1}{\sqrt{m}}b_n, \ldots, \frac{1}{\sqrt{m}}b_n}_{m:=r'-n+1 \text{ columns}}], \tag{5}$$

where b_i denotes the i-th column of B. It is easy to see that $\widehat{B}\widehat{B}^\top = BB^\top = A$. The next lemma is easily derived from the previous discussion, and it implies that there always exists an $n \times \mathrm{cp}_n$ CP factorization for any $A \in \mathcal{CP}_n$.

Lemma 2.3. *Suppose that $A \in \mathcal{S}_n$, $r \in \mathbb{N}$. Then, $r \geq \mathrm{cp}(A)$ if and only if A has a CP factorization B with r columns.*

The following lemma is very essential in our study. Many authors have proved the existence of such an orthogonal matrix X (see, e.g., [45, Lemma 1] and [26, Lemma 2.6]).

Lemma 2.4. *Let $\mathcal{O}_r$ denote the set of $r \times r$ orthogonal matrices and $B, C \in \mathbb{R}^{n\times r}$. Then, $BB^\top = CC^\top$ if and only if there exists $X \in \mathcal{O}_r$ such that $BX = C$.*

The next proposition just puts the previous two lemmas together.

Proposition 2.5. *Let $A \in \mathcal{CP}_n, r \geq \mathrm{cp}(A), A = \bar{B}\bar{B}^\top$, where $B \in \mathbb{R}^{n\times r}$ may be not nonnegative. Then there exists an orthogonal matrix $X \in \mathcal{O}_r$ such that $\bar{B}X \geq 0$ and $A = (\bar{B}X)(\bar{B}X)^\top$.*

This proposition tells us that one can find an orthogonal matrix X which can take a "bad" factorization $\bar{B}$ into a "good" factorization $\bar{B}X$. Thus, the task of finding a CP factorization of A can be formulated as the following feasibility problem:

$$\text{Find } X \text{ s.t. } \bar{B}X \geq 0 \text{ and } X \in \mathcal{O}_r, \tag{FeasCP}$$

where $r \geq \mathrm{cp}(A)$, $\bar{B} \in \mathbb{R}^{n\times r}$ is an arbitrary initial factorization $A = \bar{B}\bar{B}^\top$ and may be not nonnegative. We should notice that the condition $r \geq \mathrm{cp}(A)$ is necessary; otherwise, (FeasCP) has no solution even if $A \in \mathcal{CP}_n$. Regardless of the exact $\mathrm{cp}(A)$ which is often unknown, one can use cp_n. Note that finding an initial matrix $\bar{B}$ is not difficult. Since a completely positive matrix is necessarily positive semidefinite, one can use Cholesky decomposition or spectral decomposition and then extend its columns to r columns by using (5). The following corollary summarizes that the feasibility of (FeasCP) is precisely a criterion for complete positivity.

Corollary 2.5.1. *Set $r \geq \mathrm{cp}(A)$, $\bar{B} \in \mathbb{R}^{n\times r}$ an arbitrary initial factorization of A. Then $A \in \mathcal{CP}_n$ if and only if (FeasCP) is feasible. In this case, for any feasible solution X, $\bar{B}X$ is a CP factorization of A.*

2.3 Approaches to (FeasCP)

In this study, solving (FeasCP) is the key to finding a CP factorization, but it is still a hard problem because $\mathcal{O}_r$ is nonconvex. For example, $\frac{1}{2}X + \frac{1}{2}(-X) = 0 \notin \mathcal{O}_r$.

Groetzner and Dür [26] applied the so-called alternating projections method to (FeasCP). They defined the polyhedral cone, $\mathcal{P} := \{X \in \mathbb{R}^{r\times r} | \bar{B}X \geq 0\}$, and rewrote (FeasCP) as

$$\text{Find } X \text{ s.t. } X \in \mathcal{P} \cap \mathcal{O}_r.$$

The alternating projections method is as follows: choose a starting point $X_0 \in \mathcal{O}_r$; then compute $P_0 = \mathrm{proj}_{\mathcal{P}}(X_0)$ and $X_1 = \mathrm{proj}_{\mathcal{O}_r}(P_0)$, and iterate this process. Computing the projection onto $\mathcal{P}$ amounts to solving a second-order cone problem (SOCP), and computing the projection onto $\mathcal{O}_r$ amounts to a singular value decomposition. Note that we need to solve an SOCP alternately at every iteration, which is still expensive in practice. A modified version involves calculating an approximation of $\mathrm{proj}_{\mathcal{P}}(X_k)$ by using the Moore-Penrose inverse of B; for details, see [26, Algorithm 2].

Our way of solving (FeasCP) is to use orthogonality optimization. Here, we denote by $\max(\cdot)$ (resp. $\min(\cdot)$) the *maximum function* (resp. *minimum function*)) that selects the largest (resp. smallest) entry of a vector or matrix. Notice that $-\min(\cdot) = \max(-(\cdot))$. We associate (FeasCP) with the following optimization problem:

$$\max_{X\in\mathcal{O}_r} \{\min(\bar{B}X)\}.$$

For consistency of notation, we turn the maximization into a minimization:

$$\min_{X\in\mathcal{O}_r} \{\max(-\bar{B}X)\}. \tag{OptCP}$$

The feasible set $\mathcal{O}_r$ is known to be compact [28, Observation 2.1.7]. In accordance with the extreme value theorem [37, Theorem 4.16], (OptCP) can obtain the global minimum, say t. Summarizing these observations together with Corollary 2.5.1 yields the following proposition.

Proposition 2.6. *Set $r \geq \mathrm{cp}(A)$, $\bar{B} \in \mathbb{R}^{n\times r}$ as an arbitrary initial factorization of A. Then the following statements are equivalent:*

1. *$A \in \mathcal{CP}_n$.*
2. *(FeasCP) is feasible.*
3. *In (OptCP), there exists a feasible solution X such that $\max(-\bar{B}X) \leq 0$; alternatively, $\min(\bar{B}X) \geq 0$.*
4. *In (OptCP), the global minimum $t \leq 0$.*

3 Smoothing method for nonsmooth orthogonality optimization

In this section, the smoothing method is adapted to nonsmooth orthogonality optimization. Consider the *nonsmooth* unconstrained optimization:

$$\min_{x\in\mathbb{R}^n} f(x), \tag{UnOpt}$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is locally Lipschitz continuous and bounded below in $\mathbb{R}^n$. The theory and methods related to (UnOpt) have been developed over the course of several decades. Recall that the *Clarke subdifferential* [16] is characterized by

$$\partial f(x) = \mathrm{conv}\{v \mid \nabla f(x^k) \to v \text{ for } x^k \to x, f \text{ is differentiable at } x^k\}.$$

For a matrix argument, $\partial f(X)$ has the same form. It is known that the first-order optimality condition of (UnOpt) is

$$\partial f(x) = 0,$$

and we call such x a *Clarke stationary point*.

Suppose that $f(X) : \mathbb{R}^{n\times p} \mapsto \mathbb{R}$ is locally Lipschitz continuous on $\mathbb{R}^{n\times p}$. Now let us consider the *nonsmooth* orthogonality optimization,

$$\min_{X\in\mathrm{St}(n,p)} f(X), \tag{StOpt}$$

where the feasible set

$$\mathrm{St}(n,p) = \{X \in \mathbb{R}^{n\times p} \mid X^\top X = I\}$$

is the Stiefel manifold (i.e., orthogonality constraint), on which optimization problems have a variety of applications [29]. The CP factorization problem has also become one of the applications, through (OptCP) with $\mathcal{O}_r = \mathrm{St}(r,r)$. We will also give an optimality condition of (StOpt) using the Clarke subdifferential. For convenience, we call (StOpt) (resp. (UnOpt)) *smooth* if $f(\cdot)$ is continuously differentiable on $\mathbb{R}^{n\times p}$ (resp. $\mathbb{R}^n$).

Lemma 3.1. *Suppose that X is a local minimizer of (StOpt). Then X satisfies the first-order optimality condition,*

$$0 \in \partial f(X) - X\partial f(X)^\top X, \tag{6}$$

and we call any X a Clarke stationary point *of (StOpt) if X satisfies the above. In particular, if (StOpt) is smooth, it reduces to*

$$0 = \nabla f(X) - X\nabla f(X)^\top X,$$

where $\nabla f(X) = (\frac{\partial f(X)}{\partial X_{ij}})$.

Proof. Suppose that X is a local minimizer. It follows from $X^\top X = I$ that the Cottle constraint qualification [2, Definition 4.9] is satisfied at any feasible point X. Hence, from the generalized Karush-Kuhn-Tucker optimality conditions for nonsmooth optimization [2, Theorem 4.11], we have a first-order optimality condition

$$0 \in \partial f(X) - X\Lambda$$

with the associated symmetric Lagrangian multiplier Λ. After performing similar algebra operations to those in the proof of [44, Lemma 1], we obtain $0 \in \partial f(X) - X\partial f(X)^\top X$. Finally, the last statement holds by noting that $\partial f(X) = \{\nabla f(X)\}$, see [2, Theorem 3.7]. □

3.1 Curvilinear search

Wen and Yin [44] proposed the *curvilinear search method* for solving the *smooth* (StOpt). Here, we briefly review this method as preparation for solving the *nonsmooth* (StOpt) in the next subsection.

Starting at a point $X \in \mathrm{St}(n,p)$, we construct a smooth curve $Y(\tau)$ on $\mathrm{St}(n,p)$ starting from X. This means that the curve goes through X, and we obtain X at zero step size; in addition, the curve maintains orthogonality with an arbitrary step size; i.e., the image of $Y(\tau)$ is contained in $\mathrm{St}(n,p)$. Simultaneously, as long as the point X is not a local minimizer of smooth (StOpt), the objective value will become smaller along this curve at a certain step size; i.e., we are able to find a suitable $\bar{\tau}$ such that

$$f(Y(\bar{\tau})) < f(Y(0)) = f(X).$$

It is sufficient to show that the composition function $(f \circ Y)(\tau) = f(Y(\tau))$ from reals to reals satisfies

$$(f \circ Y)'(0) := \left.\frac{df(Y(\tau))}{d\tau}\right|_{\tau=0} < 0.$$

The above framework is precisely the classical gradient descent method for smooth (UnOpt). In this method, we often generate a sequence $x^0, x^1, x^2, \ldots$, where x^{k+1} is generated from x^k by using the rule $x^{k+1} = x^k + \alpha_k d_k$ with the descent direction d_k and the step size α_k. In general, α_k is chosen so that $f(x^{k+1}) < f(x^k)$.

An intuitive difference between the classical gradient descent and the curvilinear search is that one searches along a straight line in the classical gradient method while one does so along a curve in the curvilinear search. Both methods search within the feasible region, and each iteration finds a better point. A merit of employing the curve search is that it can recast the constrained problem as an unconstrained one. Next, we give the mathematical details of the curvilinear search method.

Suppose that $X \in \mathrm{St}(n,p)$. Lemma 3.1 has indicated the first-order optimality condition for smooth (StOpt). So, if we define

$$\nabla F(X) := \nabla f(X) - X\nabla f(X)^\top X$$

and

$$A := \nabla f(X)X^\top - X\nabla f(X)^\top, \tag{7}$$

then $\nabla F(X) = AX$. Thus, $\nabla F(X) = 0$ if and only if $A = 0$. The next lemma gives a nice way to construct such a curve on $\mathrm{St}(n,p)$.

Lemma 3.2 (Update scheme [44, Lemma 3]). *1. Let X be a feasible point. Given any skew-symmetric matrix $W \in \mathbb{R}^{n\times n}$, $Y(\tau) : \mathbb{R} \mapsto \mathbb{R}^{n\times p}$ defined below satisfies $Y(\tau)^\top Y(\tau) = X^\top X$ for any τ and $Y(0) = X$.*

$$Y(\tau) := (I + \frac{\tau}{2}W)^{-1}(I - \frac{\tau}{2}W)X. \tag{8}$$

2. Let $W = A$ in (7). Then, $Y(\tau)$ is a descent curve at $\tau = 0$, that is,

$$(f \circ Y)'(0) = \left.\frac{df(Y(\tau))}{d\tau}\right|_{\tau=0} = -\frac{1}{2}\|A\|^2.$$

Hence, $(f \circ Y)'(0) < 0$ holds, provided that X is not yet a local minimizer.

At iteration k, one can find τ_k by the Armijo-Wolfe rules (10a) and (10b) in preparation for convergence. The proof of existence of τ_k is exactly the same as in the classical line search, cf. [35, Lemma 3.1]. Now, every iteration of Algorithm 1 is well defined. Note that the generated sequence $\{f(X^k)\}$ is monotonically decreasing. In particular, the global convergence is guaranteed from [44, Theorem 2], that is,

$$\lim_{k\to\infty} \|\nabla F(X^k)\| = 0. \tag{9}$$

Algorithm 1: Monotone Curvilinear Search for Smooth (StOpt)

Set $0 < c_1 < c_2 < 1, \epsilon > 0, X^0 \in \mathcal{O}_r, k \leftarrow 0$.
while $\|\nabla F(X^k)\| > \epsilon$ **do**
Generate $A_k \leftarrow \nabla f(X^k)X^{k\top} - X^k \nabla f(X^k)^\top, W_k \leftarrow A_k$;
Find a step size $\tau_k > 0$ that satisfies the Armijo-Wolfe conditions:

$$(f \circ Y_k)(\tau_k) \leq (f \circ Y_k)(0) + c_1 \tau_k (f \circ Y_k)'(0), \tag{10a}$$

$$(f \circ Y_k)'(\tau_k) \geq c_2 (f \circ Y_k)'(0); \tag{10b}$$

Set $X_{k+1} \leftarrow Y_k(\tau_k)$ in (8);
$k \leftarrow k + 1$ and continue;
end while

3.2 Smoothing method

Now, let us pay attention to a class of *smoothing methods* proposed by Chen [15] for solving (UnOpt), where $f(\cdot)$ is not assumed to be continuously differentiable. We call $\tilde{f} : \mathbb{R}^n \times (0, \infty) \mapsto \mathbb{R}$ a *smoothing function* of f, if (i) $\tilde{f}(\cdot, \mu)$ is continuously differentiable on $\mathbb{R}^n$ for any fixed $\mu > 0$; (ii) and for any $x \in \mathbb{R}^n$

$$\lim_{z \to x, \mu \downarrow 0} \tilde{f}(z, \mu) = f(x).$$

Given this definition, the parametric smoothing function $\tilde{f}$ is a smoothing approximation to f.

A smoothing method can be constructed simply by using $\tilde{f}$ and $\nabla_x \tilde{f}$. However, the above requirements are not sufficient to establish the convergence of the smoothing method. In particular, we need *gradient consistency* for f and $\tilde{f}$, i.e., for any x

$$\partial f(x) = G_{\tilde{f}}(x), \tag{11}$$

where the *subdifferential associated with* $\tilde{f}$ is given by

$$G_{\tilde{f}}(x) := \text{conv}\{v \mid \nabla_x \tilde{f}(x^k, \mu_k) \to v \text{ for } x^k \to x, \mu_k \downarrow 0\}.$$

By extending this form to $G_f(X)$ of the matrix argument X, the smoothing method can be stated as Algorithm 2.

Algorithm 2: Smoothing Method for (UnOpt)

Initial step:

1. Find a smoothing function $\tilde{f}$ of f such that (11) holds.
2. Select an algorithm satisfying the *weak* global convergence condition,

$$\liminf_{k \to \infty} \|\nabla f(x^k)\| = 0 \tag{12}$$

for smooth (UnOpt).

3. Choose constants $\sigma \in (0, 1), \gamma, \mu_0 > 0$ and $x^0 \in \mathbb{R}^n$. Set $k = 0$.

Inner iteration: Generate x^{k+1} from x^k by using the above algorithm to solve

$$\min_{x \in \mathbb{R}^n} \tilde{f}(x, \mu_k) \tag{13}$$

with a fixed $\mu_k > 0$.
Outer iteration: If

$$\|\nabla_x \tilde{f}(x^{k+1}, \mu_k)\| < \gamma \mu_k, \tag{14}$$

then set $\mu_{k+1} = \sigma \mu_k$; otherwise, set $\mu_{k+1} = \mu_k$.

As stated in [15], the efficiency (also, convergence) of the smoothing method depends on (i) the smoothing function $\tilde{f}$, (ii) the solution method for the smooth optimization problem (13) in the inner iteration, and (iii) the update scheme for the smoothing parameter μ_k in the outer iteration.

Theorem 3.3. *[15, Theorem 3] Any accumulation point generated by the smoothing method for (UnOpt) is a Clarke stationary point of (UnOpt).*

We will sketch the proof of convergence. Suppose that the solution method in the inner iteration has the convergence property (12). In combination with the update scheme (14), we eventually obtain

$$\liminf_{k\to\infty} \|\nabla_x \tilde{f}(x^{k+1}, \mu_k)\| = 0.$$

If $\bar{x}$ is an accumulation point of $\{x^k\}$, then by (11), we have $0 \in \partial f(\bar{x})$.

Note that there are many powerful methods (e.g., steepest descent method, Newton or quasi-Newton method, conjugate gradient method, etc.) for performing smooth unconstrained optimization such that the weak global convergence (11) easily holds. Thus, under the update scheme (14), we have various options for solving (13) without having to worry about loss of final convergence.

After analyzing the smoothing method, we find that this method can be easily extended to a nonsmooth orthogonality optimization: Algorithm 3.

Algorithm 3: Smoothing Method for (StOpt)

Initial step:

1. Find a smoothing function $\tilde{f}$ of f such that (11) holds.
2. Select an algorithm satisfying the *weak* global convergence condition,

$$\liminf_{k\to\infty} \|\nabla F(X^k)\| = 0 \tag{15}$$

   for smooth (StOpt).
3. Choose constants $\sigma \in (0,1), \gamma, \mu_0 > 0$ and $X^0 \in \mathrm{St}(n,p)$. Set $k = 0$.

Inner iteration: Generate X^{k+1} from X^k by using the above algorithm to solve

$$\min_{X\in\mathrm{St}(n,p)} \tilde{f}(X, \mu_k) \tag{16}$$

with a fixed $\mu_k > 0$.

Outer iteration: If

$$\|\nabla_X \tilde{F}(X^{k+1}, \mu_k)\| < \gamma\mu_k, \tag{17}$$

then set $\mu_{k+1} = \sigma\mu_k$; otherwise, set $\mu_{k+1} = \mu_k$.

In a sense, the convergence (9) of the curvilinear search method is similar to the strong convergence result $\lim_{k\to\infty} \|\nabla f(x^k)\| = 0$ that holds for many methods of unconstrained optimization. Note that

$$\nabla_X \tilde{F}(X, \mu_k) := \nabla_X \tilde{f}(X, \mu_k) - X\nabla_X \tilde{f}(X, \mu_k)^\top X.$$

Theorem 3.4. *Any accumulation point generated by the smoothing method for (StOpt) is a Clarke stationary point of (StOpt).*

This theorem is proved in a similar way as (3.3) in [15, Theorem 3]. For completeness, we give a proof in Appendix A.

4 Application for CP factorization

In this section, we apply the smoothing method to the CP factorization problem (OptCP). While it is reasonable to choose the curvilinear search method in subsection 3.1 as the inner iteration algorithm, we will make the following improvements.

4.1 Nonmonotone curvilinear search

Instead of the Armijo-Wolfe rules, the Barzilai-Borwein (BB) step size can usually speed up the gradient method without any extra cost for unconstrained optimizations on $\mathbb{R}^n$ (cf.[3]). Likewise for orthogonality optimization, we can set τ_{k+1} to either

$$\tau_{k+1,1} = \frac{\langle S_k, S_k\rangle}{|\langle S_k, Y_k\rangle|} \text{ or } \tau_{k+1,2} = \frac{|\langle S_k, Y_k\rangle|}{\langle Y_k, Y_k\rangle},$$

where $S_k = X^{k+1} - X^k$ and $Y_k = \nabla F(X^{k+1}) - \nabla F(X^k)$. To ensure global convergence (15), Wen and Yin suggested a nonmonotone line search technique based on a strategy in [47] to adjust the BB step size occasionally. The convergence of that adaption to orthogonality optimization was proved in [29, Theorem 3]. The Barzilai–Borwein method with a nonmonotone line search is summarized as Algorithm 4. The only difference from Algorithm 1 is how the convergence-guaranteed step size is chosen.

Algorithm 4: Nonmonotone Curvilinear Search for Smooth (StOpt)

Set $\tau > 0, \rho, \delta, \eta \in (0,1), \epsilon, \tau_M, \tau_m > 0, c_0 \leftarrow f(X^0), q_0 \leftarrow 1, k \leftarrow 0$.
while $\|\nabla F(X^k)\| > \epsilon$ **do**
  Generate $A_k \leftarrow \nabla f(X^k)X^{k\top} - X^k\nabla f(X^k)^\top, W_k \leftarrow A_k$;
  while $(f \circ Y_k)(\tau) \geq c_k + \rho\tau(f \circ Y_k)'(0)$ **do**
    $\tau \leftarrow \delta\tau$
  end while
  Set $X^{k+1} \leftarrow Y_k(\tau)$;
  $q_{k+1} \leftarrow \eta q_k + 1$, $c_{k+1} \leftarrow (\eta q_k c_k + f(X_{k+1}))/q_{k+1}$;
  $\tau \leftarrow \max(\min(\tau_{k+1,1} \text{ or } \tau_{k+1,2}, \tau_M), \tau_m)$;
  $k \leftarrow k+1$ and continue;
end while

The remaining problem is to select an appropriate smoothing function of the maximum function in (OptCP).

4.2 LogSumExp—smoothing function of maximum function

Now let us introduce the *LogSumExp* function, $\operatorname{lse}(x,\mu) : \mathbb{R}^n \times (0,\infty) \mapsto \mathbb{R}$, given by

$$\operatorname{lse}(x,\mu) = \mu \log(\textstyle\sum_{i=1}^n \exp(x_i/\mu)).$$

Similar to the vector argument, $\operatorname{lse}(X,\mu)$ of the matrix argument can be simply derived from entrywise operations. For simplicity, we will employ the vector argument to build the crucial properties.

Theorems 4.1 and 4.2 show that $\operatorname{lse}(x,\mu)$ is a smoothing function of $\max(x)$ such that gradient consistency (11) holds. Then, from the properties of compositions of smoothing functions [15, Proposition 1 (c)], we find that $\operatorname{lse}(-\bar{B}X,\mu)$ is a smoothing function of $\max(-\bar{B}X)$ with gradient consistency for (OptCP). Notice that $\max(\cdot)$ is convex and regular at any point [16, Prosition 2.3.6].

Theorem 4.1. *The function* $\operatorname{lse}(x,\mu)$ *has the following properties.*

(i) $\operatorname{lse}(\cdot,\mu)$ *is continuously differentiable on* $\mathbb{R}^n$ *for any fixed* $\mu > 0$. *In particular,* $\nabla_x \operatorname{lse}(x,\mu)$ *is the so-called softmax function, given by* $\sigma(\cdot,\mu) : \mathbb{R}^n \mapsto \Delta^{n-1}$,

$$\nabla_x \operatorname{lse}(x,\mu) = \sigma(x,\mu) := \frac{1}{\sum_{l=1}^n \exp(x_l/\mu)} \begin{bmatrix} \exp(x_1/\mu) \\ \vdots \\ \exp(x_n/\mu) \end{bmatrix}, \tag{18}$$

where $\Delta^{n-1} := \{x \in \mathbb{R}^n \mid \sum_{i=1} x_i = 1, x_i \geq 0\}$ *is the unit simplex. Note that under the equality,*

$$\sum_{l=1}^n \exp(x_l/\mu) = \exp\{\operatorname{lse}(x,\mu)/\mu\},$$

Table 1. Example of approximation effect with different parameters μ.

$n = 4$	$\mu = 1$	$\mu = 1/2$	$\mu = 1/4$	$\mu = 1/8$
$x_1 = (2, 5, -1, 3)$	5.1719	5.0103	5.0001	5.0000
$x_2 = (5, 5, 5, 5)$	6.3863	5.6931	5.3466	5.1733
$\epsilon_\mu = \mu \log(n)$	1.3863	0.6931	0.3466	0.1733

the i-th component of $\sigma(x, \mu)$ *can be rewritten as*

$$\sigma_i(x, \mu) = \exp\{(x_i - \mathrm{lse}(x, \mu))/\mu\}. \tag{19}$$

(ii) For all $x \in \mathbb{R}^n$ *and* $\mu > 0$*, we have*

$$\max(x) < \mathrm{lse}(x, \mu) \leq \max(x) + \mu \log(n).$$

The above inequalities imply that for any $x \in \mathbb{R}^n$*,*

$$\lim_{z \to x, \mu \downarrow 0} \mathrm{lse}(z, \mu) = \max(x). \tag{20}$$

(iii) If $0 < \mu_2 < \mu_1$*, then for all* $x \in \mathbb{R}^n$*, we have*

$$\mathrm{lse}(x, \mu_2) < \mathrm{lse}(x, \mu_1).$$

Proof. We will only prove (ii) and (iii), since (i) is trivial.

(ii) Let $x_j := \max(x)$, then it is easy to show

$$\mathrm{lse}(x, \mu) = \mu \log(1 + \textstyle\sum_{i \neq j}^n \exp((x_i - x_j)/\mu)) + x_j. \tag{21}$$

For every $i \neq j$, $\mu(x_i - x_j) \leq 0$ implies $1 < 1 + \sum_{i \neq j}^n \exp((x_i - x_j)/\mu) \leq n$. Then, taking the logarithm and multiplying by μ gives

$$0 < \mu \log(1 + \textstyle\sum_{i \neq j}^n \exp((x_i - x_j)/\mu)) \leq \mu \log(n).$$

It follows that $0 < \mathrm{lse}(x, \mu) - x_j \leq \mu \log(n)$ by (21).

(iii) This property is illustrated in Figure 1. For any fixed $x \in \mathbb{R}^n$, consider the derivative of a real function $\mu \mapsto \mathrm{lse}(x, \cdot) : (0, \infty) \to \mathbb{R}$. Then, by (19),

$$\begin{aligned} \nabla_\mu \mathrm{lse}(x, \mu) = \mathrm{lse}\,/\mu - \frac{\sum_{i=1}^n x_i \exp(x_i/\mu)}{\mu \exp(\mathrm{lse}\,/\mu)} &= (\mathrm{lse} - \textstyle\sum_{i=1}^n x_i \exp\{(x_i - \mathrm{lse})/\mu\})/\mu \\ &= (\mathrm{lse} - \textstyle\sum_{i=1}^n x_i \sigma_i)/\mu \leq 0, \end{aligned}$$

where "lse, σ" are shorthand for $\mathrm{lse}(x, \mu)$ and $\sigma(x, \mu)$. For the last inequality above, we observe that $\sigma \in \Delta^{n-1}$; hence, the term $\sum_{i=1}^n x_i \sigma_i$ is a convex combination of all entries of x, which implies that $\sum_{i=1}^n x_i \sigma_i \leq \max(x) < \mathrm{lse}$. This completes our proof. □

An example of the approximation effect with different parameters μ is shown in Table 1. Rows 2 and 3 show the values of $\mathrm{lse}(x, \mu)$ corresponding to x and μ. The upper bound of the error $\epsilon_\mu := \mu \log(n)$, which is completely determined by μ, vanishes as $\mu \to 0$. If all entries are the same, the worst approximation will appear, but a small enough μ can eliminate this concern.

Theorem 4.2. *The gradient consistency*

$$\partial \max(x) = G_{\mathrm{lse}}(x)$$

holds for any $x \in \mathbb{R}^n$*. In other words,*

$$\mathrm{conv}\{e_i \mid i \in \mathcal{I}(x)\} = \mathrm{conv}\{\lim_{x^k \to x, \mu_k \downarrow 0} \sigma(x^k, \mu_k)\},$$

where e_i *is a standard unit vector and* $\mathcal{I}(x) = \{i \mid i \in \{1, \cdots, n\}, x_i = \max(x)\}$.

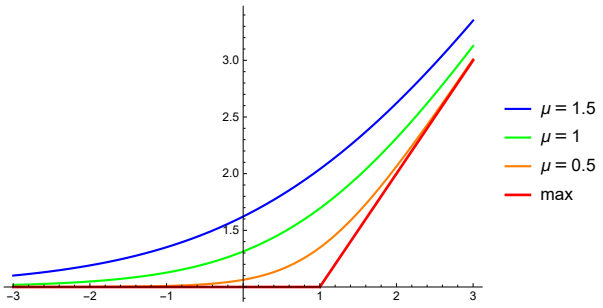

Figure 1. Slices through the line $y = 1$ of $\max(x, y)$ and $\mu \log(e^{x/\mu} + e^{y/\mu})$ on $\mathbb{R}^2$.

Proof. By [44, Theorem 3.23], we immediately get

$$\partial \max(x) = \operatorname{conv}\{e_i \mid i \in \mathcal{I}(x)\}.$$

Let $S := \{\lim_{x^k \to x, \mu_k \downarrow 0} \sigma(x^k, \mu_k)\}$, then $G_{\mathrm{lse}}(x) = \operatorname{conv} S$. We will prove the assertion in two steps; i.e., (1) show that $\operatorname{conv}(S) \subseteq \operatorname{conv}\{e_i \mid i \in \mathcal{I}(x)\}$ and (2) show that $\operatorname{conv}\{e_i \mid i \in \mathcal{I}(x)\} \subseteq \operatorname{conv}(S)$.

1. It is sufficient to show that $S \subseteq \operatorname{conv}\{e_i \mid i \in \mathcal{I}(x)\}$. Taking any two sequences $x^k \to x$ and $\mu_k \downarrow 0$, we have

$$\lim_{x^k \to x, \mu_k \downarrow 0} \sigma(x^k, \mu_k) \in \operatorname{conv}\{e_i \mid i \in \{1, \cdots, n\}\} = \Delta^{n-1}, \tag{22}$$

since for any x and $\mu > 0$, $\sigma(x, \mu) \in \Delta^{n-1}$ and Δ^{n-1} is closed. From the equalities (19) and (20), for the i-th component of the limit with $i \notin \mathcal{I}(x)$, we have

$$\begin{aligned} \lim_{x^k \to x, \mu_k \downarrow 0} \sigma_i(x^k, \mu_k) &= \lim_{x^k \to x, \mu_k \downarrow 0} \exp\{(x_i^k - \operatorname{lse}(x^k, \mu_k))/\mu_k\} \\ &= \lim_{x^k \to x, \mu_k \downarrow 0} \exp\{(x_i - \max(x))/\mu_k\} = 0. \end{aligned} \tag{23}$$

It follows from (22) that

$$\lim_{x^k \to x, \mu_k \downarrow 0} \sigma(x^k, \mu_k) \in \operatorname{conv}\{e_i \mid i \in \mathcal{I}(x)\}.$$

2. Conversely, it is sufficient to show that $\{e_i \mid i \in \mathcal{I}(x)\} \subseteq S$ in order to obtain $\operatorname{conv}\{e_i \mid i \in \mathcal{I}(x)\} \subseteq \operatorname{conv}(S)$.

(a) Suppose the case where $|\mathcal{I}(x)| = 1$. Say $\mathcal{I}(x) = \{j\}$, i.e., $x_j = \max(x)$ is the unique maximal value. Taking two sequences $x^k = x (k = 1, 2, \ldots)$ and $\mu_k \downarrow 0$, from the expression in (18), for the j-th component of the limit, we have

$$\begin{aligned} \lim_{x^k \to x, \mu_k \downarrow 0} \sigma_j(x^k, \mu_k) &= \lim_{\mu_k \downarrow 0} \frac{\exp((x_j - \max(x))/\mu_k)}{\sum_{l=1}^n \exp((x_l - \max(x))/\mu_k)} \\ &= \lim_{\mu_k \downarrow 0} \frac{1}{1 + \sum_{l \notin \mathcal{I}(x)} \exp((x_l - \max(x))/\mu_k)} = 1. \end{aligned}$$

For any $i \neq j$, by (23), we have $\lim_{x^k \to x, \mu_k \downarrow 0} \sigma_i(x^k, \mu_k) = 0$. And we are done.

(b) Suppose the case where $|\mathcal{I}(x)| \geq 2$. Select $j \in \mathcal{I}(x)$. Taking a sequence $x^k \to x$ such that $x_j^k = x_j (k = 1, 2, \ldots)$, but $x_i^k \uparrow x_i$ for any $i \neq j$, and taking

$$\mu_k := \min_{i \in \mathcal{I}(x) \setminus \{j\}} \{(x_i^k - \max(x))^2\},$$

we have

$$\exp((x_i^k - \max(x))/\mu_k) \leq \exp(\frac{1}{x_i^k - \max(x)}) \to 0, \tag{24}$$

as $k \to \infty$ for all $i \in \mathcal{I}(x), i \neq j$. For $i \in \{1, \cdots, n\}$, we have

$$\begin{aligned}
&\lim_{x^k \to x, \mu_k \downarrow 0} \sigma_i(x^k, \mu_k) \\
&= \lim_{x^k \to x, \mu_k \downarrow 0} \frac{\exp((x_i^k - \max(x))/\mu_k)}{1 + \sum_{l \in \mathcal{I}(x) \setminus \{j\}} \exp((x_l^k - \max(x))/\mu_k) + \sum_{l \notin \mathcal{I}(x)} \exp((x_l^k - \max(x))/\mu_k)} \\
&= \lim_{x^k \to x, \mu_k \downarrow 0} \exp((x_i^k - \max(x))/\mu_k).
\end{aligned}$$

The last equality comes from (24). Thus, for $i = j$ we get $\lim_{x^k \to x, \mu_k \downarrow 0} \sigma_j(x^k, \mu_k) = 1$, while for $i \in \mathcal{I}(x), i \neq j$, again from (24), we obtain 0. For $i \notin \mathcal{I}(x)$, again by (23), we have $\lim_{x^k \to x, \mu_k \downarrow 0} \sigma_i(x^k, \mu_k) = 0$. Thus, $e_j \in S$ if $j \in \mathcal{I}(x)$.

This completes the proof. □

5 Numerical results

We conducted numerical experiments to solve (OptCP) in the framework of Algorithm 3 where a curvilinear search is employed with the BB step (Algorithm 4) as the inner iteration algorithm and $\mathrm{lse}(-\bar{B}X, \mu)$ as the smoothing function. Besides the dependently (to X^k) decreasing rule of the smoothing parameter μ_k as in outer iteration (17), we also considered an independently (to X^k) decreasing rule—"Set the fixed values $\mu_0, \mu_d > 0$; and $\mu_k \leftarrow \mu_0/(1 + k\mu_d)$ in place of (17)", which gives better results empirically. In what follows, we denote the former algorithm by "*StOpt_SM_dd*" and the latter by "*StOpt_SM_id*". We compared our algorithms with the following numerical algorithms for CP factorization, mentioned in subsection 1.2:

- *SpFeasDC_ls* [14]: A difference-of-convex fiunctions approach for solving the split feasibility problem, which can be applied to CP factorization. All the implementation details about the parameters are the same as in numerical experiments in [14, Section 6.1].
- *RIPG_mod* [11]: A projected gradient method with relaxation and inertia parameters, for solving (2). As shown in numerical experiments in [11, Section 4.2], RIPG_mod is the best version among many strategies of choosing parameters.
- *APM_mod* [26]: A modified alternating projection method for CP factorization; for details see [26, Algorithm 2].

We followed the settings used by the authors in their papers. The numerical experiments were performed on a computer equipped with an Intel Core i7-10700 @ 2.90GHz 2.90GHz and 16GB RAM. The algorithms were implemented in MatlabR2021a. The details of the experiments are as follows.

If $A \in \mathcal{CP}_n$ is of full rank, for accuracy reasons, we obtained an initial $\bar{B}$ by using Cholesky decomposition. Otherwise, $\bar{B}$ was obtained by spectral decomposition. Then, we extended $\bar{B}$ to r columns by column replication (5). $r = \mathrm{cp}(A)$ if $\mathrm{cp}(A)$ is known, or r is sufficiently larger, otherwise. We used `RandOrthMat.m` [39] to generate a random starting point X^0 on the basis of the Gram–Schmidt process.

For both StOpt_SM_dd and StOpt_SM_id, we set $\tau = 0.5, \rho = 10^{-4}, \delta = 0.5, \eta = 0.5, \tau_M = 10^3, \tau_m = 10^{-7}$ and chose $\tau_{k+1,2}$ in the inner iteration (4). For the outer iteration, we set $\mu_0 = 100, \sigma = 0.8, \gamma = 0.5$ in StOpt_SM_dd, and $\mu_0 = 10, \mu_{\mathrm{setp}} = 1$ in StOpt_SM_id.

Except for RIPG_mod, all the algorithms terminated successfully at iteration k, where $\min(\bar{B}X^k) \geq -10^{-15}$ is attained before the maximum number of iterations (5,000) was reached. In addition, SpFeasDC_ls failed when $\bar{L}_k > 10^{10}$. Regarding RIPG_mod, it terminated successfully when $\|A - X_k X_k^\top\|^2/\|A\|^2 < 10^{-15}$ was attained before at most 10,000 iterations for $n < 100$, and before at most 50,000 iterations, otherwise.

Table 2. CP factorization of random completely positive matrices.

Method		StOpt_SM_id			StOpt_SM_dd			SpFeasDC_ls			RIPG_mod			APM_mod		
n	$r = 1.5n$	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s
20	30	1.00	**0.0080**	38	1.00	0.0139	60	1.00	**0.0027**	24	1.00	0.0081	1229	0.32	0.3502	2318
30	45	1.00	**0.0217**	43	1.00	0.0425	72	1.00	**0.0075**	24	1.00	0.0231	1481	0.04	1.0075	2467
40	60	1.00	**0.0374**	51	1.00	0.0641	75	1.00	**0.0216**	46	1.00	0.0574	1990	0.00	-	-
100	150	1.00	**0.2585**	69	1.00	0.4087	92	1.00	**0.2831**	109	1.00	0.8169	4912	0.00	-	-
200	300	1.00	**1.2510**	98	1.00	**1.7768**	116	1.00	2.2504	212	1.00	5.2908	9616	0.00	-	-
400	600	1.00	**14.1671**	155	1.00	**15.7512**	147	1.00	36.9650	636	1.00	90.6752	17987	0.00	-	-
600	900	1.00	**50.8520**	213	1.00	**50.3576**	177	1.00	140.0720	882	1.00	344.7035	26146	0.00	-	-
800	1200	1.00	**142.9719**	289	1.00	**114.5538**	190	1.00	413.3798	1225	1.00	891.1210	34022	0.00	-	-
Method		StOpt_SM_id			StOpt_SM_dd			SpFeasDC_ls			RIPG_mod			APM_mod		
n	$r = 3n$	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s
20	60	1.00	0.0244	43	1.00	0.0399	64	1.00	**0.0057**	15	1.00	**0.0105**	1062	0.30	0.7267	2198
30	90	1.00	0.0569	53	1.00	0.0853	69	1.00	**0.0128**	17	1.00	**0.0336**	1127	0.00	-	-
40	120	1.00	0.1031	59	1.00	0.1518	74	1.00	**0.0256**	19	1.00	**0.0822**	1460	0.00	-	-
100	300	1.00	**0.8597**	78	1.00	1.1297	87	1.00	**0.8115**	86	1.00	1.1909	4753	0.00	-	-
200	600	1.00	**6.6653**	107	1.00	**8.0160**	110	1.00	8.1517	184	1.00	9.2248	9402	0.00	-	-
400	1200	1.00	**58.9898**	165	1.00	**64.1486**	149	1.00	124.3410	453	1.00	156.6019	17563	0.00	-	-
600	1800	1.00	**235.9950**	196	1.00	**260.2481**	187	1.00	981.8537	795	1.00	616.7851	25336	0.00	-	-
800	2400	1.00	**588.2687**	254	1.00	**574.6292**	216	1.00	4027.4278	1070	1.00	1289.3736	26820	0.00	-	-

Table 3. CP factorization of a family of specifically structured instances.

Method	StOpt_SM_id			StOpt_SM_dd			SpFeasDC_ls			RIPG_mod			APM_mod		
$n = r$	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s	Rate	Time_s	Iter_s
10	1.00	**0.0049**	71	1.00	0.0052	69	1.00	**0.0043**	149	1.00	0.0074	2085	0.80	0.0174	616
20	1.00	0.0209	150	1.00	**0.0168**	107	0.98	**0.0139**	201	0.74	0.0212	3478	0.90	0.0591	864
50	1.00	**0.1765**	233	1.00	**0.1090**	125	0.98	0.3389	770	0.00	-	-	0.76	0.6948	1416
75	1.00	**0.4408**	309	1.00	**0.2314**	139	0.98	1.0706	1186	0.00	-	-	0.64	1.4809	1510
100	1.00	**0.9224**	385	1.00	**0.5118**	185	0.80	1.6653	1083	0.00	-	-	0.60	2.8150	1690
150	1.00	**2.8308**	560	1.00	**1.5551**	265	0.70	3.7652	1170	0.00	-	-	0.35	9.9930	2959

5.1 Randomly generated instances

We examined the case of randomly generated matrices to see how those methods are affected by the order. The instances were generated in the same way as in [26, Section 7.7]. We computed C by setting $C_{ij} := |B_{ij}|$ for all i, j, where B is a random $n \times 2n$ matrix based on the Matlab command `randn`, and we took $A = CC^\top$ to be factorized. In Table 2, we set $r = 1.5n$ and $r = 3n$ for the values $n \in \{20, 30, 40, 100, 200, 400, 600, 800\}$. For each pair of n and r, we generated 50 instances if $n \leq 100$ and 10 instances otherwise. For each instance, we initialized all the algorithms at the same random starting point X^0 and initial decomposition $\bar{B}$, except for RIPG_mod. Note that each instance A was assigned only one starting point. Table 2 lists the average time in seconds (Time_s) and the average number of iterations (Iter_s) among the successful instances. It also lists the rounded success rate (Rate) relative to the total number of instances. Boldface highlights the two best results for each pair of n and r.

As shown in Table 2, except for APM_mod, each method had a success rate of 1 for all pairs of n and r. StOpt_SM_dd and StOpt_SM_id outperformed the other methods on the large-scale matrices. In particular, StOpt_SM_id with the independently decreasing rule gave the best results.

5.2 A specifically structured instance

Let $\mathbf{e}_n$ denote the all-ones vector in $\mathbb{R}^n$ and consider the matrix [26, Example 7.1],

$$A_n = \begin{pmatrix} 0 & \mathbf{e}_{n-1}^\top \\ \mathbf{e}_{n-1} & I_{n-1} \end{pmatrix}^\top \begin{pmatrix} 0 & \mathbf{e}_{n-1}^\top \\ \mathbf{e}_{n-1} & I_{n-1} \end{pmatrix} \in \mathcal{CP}_n.$$

It has been shown by Theorem 2.2 that $A_n \in \text{int}(\mathcal{CP}_n)$ for every $n \geq 2$. By construction, it is obvious that $\text{cp}(A_n) = n$. We tried to factorize A_n for the values $n \in \{10, 20, 50, 75, 100, 150\}$ in Table 3. For each A_n, using $r := \text{cp}(A_n) = n$ and the same initial decomposition $\bar{B}$, we tested all the algorithms on the same 50 randomly generated starting points, except for RIPG_mod. Here, each instance was assigned 50 starting points. Table 3 lists the average time in seconds (Time_s) and the average number of iterations (Iter_s) for the successful starting points. It also lists the rounded successful rate (Rate) relative to the total number of starting points. Boldface highlights the two best results for each n. We can see from Table 3 that the success rates of StOpt_SM_dd and StOpt_SM_id were always 1, but the success rates of the other methods decreased as n increased.

5.3 Factorization whose smallest entry is as large as possible

Consider the following matrix from [41, Example 2.7]:

$$A = \begin{pmatrix} 41 & 43 & 80 & 56 & 50 \\ 43 & 62 & 89 & 78 & 51 \\ 80 & 89 & 162 & 120 & 93 \\ 56 & 78 & 120 & 104 & 62 \\ 50 & 51 & 93 & 62 & 65 \end{pmatrix}.$$

The sufficient condition from [41, Theorem 2.5] ensures that this matrix is completely positive and $\text{cp}(A) = \text{rank}(A) = 3$. Theorem 2.2 tells us that $A \in \text{bd}(\mathcal{CP}_5)$, since $\text{rank}(A) \neq 5$. We found that all the algorithms could easily factorize this matrix. However, our method returned a CP factorization B whose smallest entry was as large as possible. When we did not terminate as soon as $\min(\bar{B}X^k) \geq -10^{-15}$, for example, after 1000 iterations, StOpt_SM_dd and StOpt_SM_id gave the following CP factorization whose the smallest entry is around $2.8573 \gg -10^{-15}$:

$$A = BB^\top \text{ , where } B \approx \begin{pmatrix} 3.5771 & 4.4766 & \mathbf{2.8573} \\ 2.8574 & 3.0682 & 6.6650 \\ 8.3822 & 7.0001 & 6.5374 \\ 5.7515 & 2.8574 & 7.9219 \\ 2.8574 & 6.7741 & 3.3085 \end{pmatrix}.$$

In fact, our methods also maximized the smallest entry in the $n \times r$ symmetric factorization of A, since (OptCP) is equivalent to

$$\max_{A = XX^\top, X \in \mathbb{R}^{n \times r}} \{\min(X)\}.$$

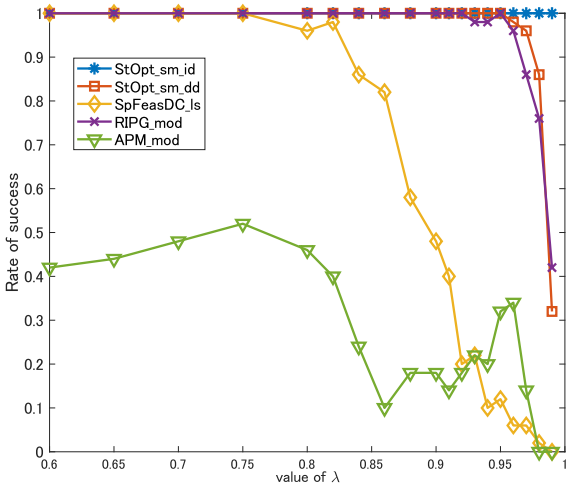

Figure 2. Success rate of CP factorization of A_λ for values of λ from 0.6 to 0.99.

5.4 A hard instance on the boundary of $\mathcal{CP}_n$

Next, we examined how well these methods work for a hard matrix on the boundary of $\mathcal{CP}_n$. Consider the following matrix on the boundary taken from [25]:

$$A = \begin{pmatrix} 8 & 5 & 1 & 1 & 5 \\ 5 & 8 & 5 & 1 & 1 \\ 1 & 5 & 8 & 5 & 1 \\ 1 & 1 & 5 & 8 & 5 \\ 5 & 1 & 1 & 5 & 8 \end{pmatrix} \in \mathrm{bd}(\mathcal{CP}_5).$$

Since A is of full rank, then by Theorem 2.2 $\mathrm{cp}^+(A) = \infty$; i.e., there is no strictly positive CP factorization for A. Hence, the global minimum of (OptCP), $t = 0$ is clear. None of the algorithms could decompose this matrix under our tolerance, 10^{-15}, in the stopping criteria. Just as was done in [26, Example 7.3], we investigated slight perturbations of this matrix. Given

$$MM^\top =: C \in \mathrm{int}(\mathcal{CP}_5) \text{ with } M = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

we factorized $A_\lambda := \lambda A + (1- \lambda)C$ for different values of $\lambda \in [0, 1)$ using $r = 12 > \mathrm{cp}_5 = 11$. Note that $A_\lambda \in \mathrm{int}(C\mathcal{P}_5)$ provided $0 \le \lambda < 1$ and A_λ approaches the boundary as $\lambda \to 1$. We chose the largest $\lambda = 0.99$. For each A_λ, we tested all the algorithms on 50 randomly generated starting points and computed the success rate relative to the total number of starting points. Figure 2 shows how the success rate of each algorithm changes as A_λ approaches the boundary. Except for StOpt_SM_id whose success rate is always 1, the success rates of all the other algorithms significantly decrease as λ increases to 0.99. But we can deduce that the success rate of StOpt_SM_id always decreases to zero on the interval $(0.99, 1]$ since it fails to factorize $A_1 = A$.

6 Conclusions

We examined the problem of finding a CP factorization of a given completely positive matrix. We treated it as a nonsmooth orthogonality optimization and applied the Riemannian smoothing steepest descent method. To the best of our knowledge, ours is the first study to handle the CP factorization problem by Riemannian optimization, for which various techniques have been developed in recent years. The numerical experiments clarify that our method can compete with other efficient CP factorization methods, in particular for large-scale matrices.

A Proof of Theorem 3.4

Proof. Define $K = \{k \mid \mu_{k+1} = \sigma\mu_k\}$. We show that K must be infinite and $\lim_{k\to\infty} \mu_k = 0$. Suppose that K is finite, then there exists an integer $\bar{k}$ such that for all $k > \bar{k}$,

$$\|\nabla_X \tilde{F}(X^{k+1}, \mu_k)\| \geq \gamma\mu_k, \tag{25}$$

and $\bar{\mu} := \mu_k$ for all $k \geq \bar{k}$. This means that the outer iteration will not be executed anymore after the $\bar{k}$-th iteration. The algorithm only continues to iterate for solving

$$\min_{X\in \mathrm{St}(n,p)} \tilde{f}(X, \bar{\mu})$$

with fixed $\bar{\mu}$ in the inner iteration. Thus, it satisfies

$$\liminf_{k\to\infty} \|\nabla_X \tilde{F}(X^k, \bar{\mu})\| = 0,$$

which contradicts (25).

Suppose that $K = \{k_0, k_1, \ldots\} \subseteq \mathbb{N}$. Then, we have

$$\lim_{i\to\infty} \|\nabla_X \tilde{F}(X^{k_i+1}, \mu_{k_i})\| \leq \gamma \lim_{i\to\infty} \mu_{k_i} = 0,$$

i.e.,

$$\liminf_{k\to\infty} \|\nabla_X \tilde{F}(X^{k+1}, \mu_k)\| = 0.$$

Let $\bar{X}$ be an accumulation point of $\{X^k\}$. Then, we see that

$$0 = \lim_{k\to\infty} \|\nabla_X \tilde{F}(X^{k+1}, \mu_k)\| = \lim_{k\to\infty} \|\nabla_X \tilde{f}(X^{k+1}, \mu_k) - X^k \nabla_X \tilde{f}(X^{k+1}, \mu_k)^\top X^k\|.$$

This implies that

$$\begin{aligned} 0 = \lim_{k\to\infty} \nabla_X \tilde{f}(X^{k+1}, \mu_k) - \bar{X} \lim_{k\to\infty} \nabla_X \tilde{f}(X^{k+1}, \mu_k)^\top \bar{X} &\in G_{\tilde{f}}(\bar{X}) - \bar{X} G_{\tilde{f}}(\bar{X})^\top \bar{X} \\ &= \partial f(\bar{X}) - \bar{X}\partial f(\bar{X})^\top \bar{X}, \end{aligned}$$

which completes the proof. □

References

[1] B. Abraham and S.-m. Naomi, *Completely Positive Matrices*, World Scientific, 2003.

[2] A. Bagirov, N. Karmitsa, and M. M. Mäkelä, *Introduction to Nonsmooth Optimization: theory, practice and software*, Springer, 2014.

[3] J. Barzilai and J. M. Borwein, *Two-point step size gradient methods*, IMA journal of numerical analysis, 8 (1988), pp. 141–148.

[4] A. Berman, M. Dür, and N. Shaked-Monderer, *Open problems in the theory of completely positive and copositive matrices*, Electronic Journal of Linear Algebra, 29 (2015), pp. 46–58.

[5] I. M. Bomze, *Copositive optimization–recent developments and applications*, European Journal of Operational Research, 216 (2012), pp. 509–520.

[6] ——, *Building a completely positive factorization*, Central European journal of operations research, 26 (2018), pp. 287–305.

[7] I. M. Bomze, P. J. Dickinson, and G. Still, *The structure of completely positive matrices according to their cp-rank and cp-plus-rank*, Linear algebra and its applications, 482 (2015), pp. 191–206.

[8] I. M. Bomze, M. Dür, E. De Klerk, C. Roos, A. J. Quist, and T. Terlaky, *On copositive programming and standard quadratic optimization problems*, Journal of Global Optimization, 18 (2000), pp. 301–320.

[9] I. M. Bomze, W. Schachinger, and G. Uchida, *Think co (mpletely) positive! matrix properties, examples and a clustered bibliography on copositive optimization*, Journal of Global Optimization, 52 (2012), pp. 423–445.

[10] P. B. Borckmans, S. E. Selvan, N. Boumal, and P.-A. Absil, *A riemannian subgradient algorithm for economic dispatch with valve-point effect*, Journal of Computational and Applied Mathematics, 255 (2014), pp. 848–866.

[11] R. I. Boţ and D.-K. Nguyen, *Factorization of completely positive matrices using iterative projected gradient steps*, Numerical Linear Algebra with Applications, (2021), p. e2391.

[12] S. Burer, *On the copositive representation of binary and continuous nonconvex quadratic programs*, Mathematical Programming, 120 (2009), pp. 479–495.

[13] ——, *A gentle, geometric introduction to copositive optimization*, Mathematical Programming, 151 (2015), pp. 89–116.

[14] C. Chen, T. K. Pong, L. Tan, and L. Zeng, *A difference-of-convex approach for split feasibility with applications to matrix factorizations and outlier detection*, Journal of Global Optimization, (2020), pp. 1–30.

[15] X. Chen, *Smoothing methods for nonsmooth, nonconvex minimization*, Mathematical programming, 134 (2012), pp. 71–99.

[16] F. H. Clarke, *Optimization and nonsmooth analysis*, SIAM, 1990.

[17] G. de Carvalho Bento, J. X. da Cruz Neto, and P. R. Oliveira, *A new approach to the proximal point method: convergence on general riemannian manifolds*, Journal of Optimization Theory and Applications, 168 (2016), pp. 743–755.

[18] E. De Klerk and D. V. Pasechnik, *Approximation of the stability number of a graph via copositive programming*, SIAM Journal on Optimization, 12 (2002), pp. 875–892.

[19] P. J. Dickinson, *An improved characterisation of the interior of the completely positive cone*, Electronic Journal of Linear Algebra, 20 (2010), pp. 723–729.

[20] ——, *The Copositive Cone, the Completely Positive Cone and Their Generalisations*, PhD thesis, University of Groningen, 2013.

[21] P. J. Dickinson and M. Dür, *Linear-time complete positivity detection and decomposition of sparse matrices*, SIAM Journal on Matrix Analysis and Applications, 33 (2012), pp. 701–720.

[22] P. J. Dickinson and L. Gijben, *On the computational complexity of membership problems for the completely positive cone and its dual*, Computational optimization and applications, 57 (2014), pp. 403–415.

[23] G. Dirr, U. Helmke, and C. Lageman, *Nonsmooth riemannian optimization with applications to sphere packing and grasping*, in Lagrangian and Hamiltonian methods for nonlinear control 2006, Springer, 2007, pp. 29–45.

[24] M. Dür, *Copositive programming–a survey*, in Recent advances in optimization and its applications in engineering, Springer, 2010, pp. 3–20.

[25] M. Dür and G. Still, *Interior points of the completely positive cone*, The Electronic Journal of Linear Algebra, 17 (2008).

[26] P. Groetzner and M. Dür, *A factorization method for completely positive matrices*, Linear Algebra and its Applications, 591 (2020), pp. 1–24.

[27] P. H. Groetzner, *A Method for Completely Positive and Nonnegative Matrix Factorization*, PhD thesis, University of Trier, 2018.

[28] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge university press, 2012.

[29] J. Hu, X. Liu, Z.-W. Wen, and Y.-X. Yuan, *A brief introduction to manifold optimization*, Journal of the Operations Research Society of China, 8 (2020), pp. 199–248.

[30] F. Jarre and K. Schmallowsky, *On the computation of C^* certificates*, Journal of Global Optimization, 45 (2009), p. 281.

[31] B. Jiang and Y.-H. Dai, *A framework of constraint preserving update schemes for optimization on stiefel manifold*, Mathematical Programming, 153 (2015), pp. 535–575.

[32] A. Kovnatsky, K. Glashoff, and M. M. Bronstein, *MADMM: a generic algorithm for non-smooth optimization on manifolds*, in European Conference on Computer Vision, Springer, 2016, pp. 680–696.

[33] R. Lai and S. Osher, *A splitting method for orthogonality constrained problems*, Journal of Scientific Computing, 58 (2014), pp. 431–449.

[34] J. Nie, *The $\mathcal{A}$-truncated K-moment problem*, Foundations of Computational Mathematics, 14 (2014), pp. 1243–1276.

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

[36] M. Obara, T. Okuno, and A. Takeda, *Sequential quadratic optimization for nonlinear optimization problems on riemannian manifolds*, arXiv preprint arXiv:2009.07153, (2020).

[37] W. Rudin et al., *Principles of mathematical analysis*, vol. 3, McGraw-hill New York, 1976.

[38] H. Sato and K. Aihara, *Cholesky qr-based retraction on the generalized stiefel manifold*, Computational Optimization and Applications, 72 (2019), pp. 293–308.

[39] O. Shilon, *RandOrthMat*, 2020.

[40] M. D. Sikirić, A. Schürmann, and F. Vallentin, *A simplex algorithm for rational cp-factorization*, Mathematical Programming, (2020), pp. 1–21.

[41] W. So and C. Xu, *A simple sufficient condition for complete positivity*, Operators and Matrices, 9 (2015), pp. 233–239.

[42] J. Sponsel and M. Dür, *Factorization and cutting planes for completely positive matrices by copositive projection*, Mathematical Programming, 143 (2014), pp. 211–229.

[43] Y. Wang, W. Yin, and J. Zeng, *Global convergence of admm in nonconvex nonsmooth optimization*, Journal of Scientific Computing, 78 (2019), pp. 29–63.

[44] Z. Wen and W. Yin, *A feasible method for optimization with orthogonality constraints*, Mathematical Programming, 142 (2013), pp. 397–434.

[45] C. Xu, *Completely positive matrices*, Linear algebra and its applications, 379 (2004), pp. 319–327.

[46] C. Zhang, X. Chen, and S. Ma, *A riemannian smoothing steepest descent method for non-lipschitz optimization on submanifolds*, arXiv preprint arXiv:2104.04199, (2021).

[47] H. Zhang and W. W. Hager, *A nonmonotone line search technique and its application to unconstrained optimization*, SIAM journal on Optimization, 14 (2004), pp. 1043–1056.

[48] X. Zhu, *A riemannian conjugate gradient method for optimization on the stiefel manifold*, Computational optimization and Applications, 67 (2017), pp. 73–110.