Tech. report UCL-INMA-2016.08

Intrinsic Representation of Tangent Vectors and Vector Transports on Matrix Manifolds

Wen Huang · P.-A. Absil · K. A. Gallivan

Received: date / Accepted: date

Abstract The quasi-Newton methods on Riemannian manifolds proposed thus far do not appear to lend themselves to satisfactory convergence analyses unless they resort to an isometric vector transport. This prompts us to propose a computationally tractable isometric vector transport on the Stiefel manifold of orthonormal p-frames in $\mathbb{R}^n$. Specifically, it requires $O(np^2)$ flops, which is considerably less expensive than existing alternatives in the frequently encountered case where $n \gg p$. We then build on this result to also propose computationally tractable isometric vector transports on other manifolds, namely the Grassmann manifold, the fixed-rank manifold, and the positive-semidefinite fixed-rank manifold. In the process, we also propose a convenient way to represent tangent vectors to these manifolds as elements of $\mathbb{R}^d$, where d is the dimension of the manifold. We call this an "intrinsic" representation, as opposed to "extrinsic" representations as elements of $\mathbb{R}^w$, where w is the dimension of the embedding space. Finally, we demonstrate the performance of the proposed isometric vector transport in the context of a Riemannian quasi-Newton method applied to minimizing the Brockett cost function.

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office. This work was supported by grant FNRS PDR T.0173.13.

Wen Huang
Department of Mathematical Engineering, ICTEAM Institute, Université catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium. Tel.: +32-10-478005
Fax: +32-10-472180
E-mail: huwst08@gmail.com

P.-A. Absil
Department of Mathematical Engineering, ICTEAM Institute, Université catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium.

K. A. Gallivan
Department of Mathematics, 208 Love Building, 1017 Academic Way, Florida State University, Tallahassee FL 32306-4510, USA.

Keywords Vector Transport · Riemannian quasi-Newton methods · Intrinsic representation · Matrix manifold

1 Introduction

Riemannian optimization concerns solving the problem

$$\min_{x \in \mathcal{M}} f(x),$$

where $\mathcal{M}$ is a d-dimensional Riemannian manifold. It has been a topic of much interest over the past few years due to many important applications that include but are not limited to matrix completion problems [BA11, MMS11, DKM12, Van13], image segmentation and recognition [RW12, TVSC11], matrix mean computation [BI13, ATV13], blind source separation [KS12, SAGQ12], finite-element discretization of Cosserat rods [San10], and elastic shape analysis of curves [HGSA15].

Various Riemannian optimization algorithms have been proposed, see e.g., [AMS08]. Among them, the trust-region Newton method [ABG07] is popular due to its local quadratic convergence rate. However, the Newton method requires the evaluation of the action of the Hessian, which may not be available to users or may be expensive. Therefore, there is a growing interest for the nonlinear conjugate gradient method and quasi-Newton methods since it is well-known that in the Euclidean setting, those methods do not require the action of Hessian, achieve faster local convergence rate, and can be faster than Newton methods in many important applications. Recently, multiple Riemannian versions of those methods have been proposed with convergence analyses, e.g., [SI15, Sat15] for Riemannian nonlinear conjugate methods and [RW12, HAG15, HGA15] for Riemannian quasi-Newton methods.

Nonlinear conjugate gradient method and quasi-Newton methods need to combine information at different iterates. For example, the search direction in a Euclidean quasi-Newton method is given by

$$\eta_k = -\mathcal{B}_k^{-1} \operatorname{grad} f(x_k), \tag{1.1}$$

where the Hessian approximation $\mathcal{B}_k$ is a linear operator, which is updated during iterations according to $\mathcal{B}_{k+1} = \kappa(\mathcal{B}_k, \mathfrak{s}_k, \mathfrak{y}_k)$, where $\mathfrak{s}_k = x_{k+1} - x_k$, $\mathfrak{y}_k = \operatorname{grad} f(x_{k+1}) - \operatorname{grad} f(x_k)$, and the κ is a function defining the update, e.g., [NW06, (6.13), (6.19), (6.24)]. We use the famous BFGS update, defined in the Euclidean space $\mathbb{R}^n$ by

$$\mathcal{B}_{k+1} = \kappa_{\mathrm{BFGS}}(\mathcal{B}_k) = \mathcal{B}_k - \frac{\mathcal{B}_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, \tag{1.2}$$

as an example to show the difficulties of Riemannian quasi-Newton methods. In the Riemannian setting, it is known that $\operatorname{grad} f(x_k)$ is in $\mathrm{T}_{x_k} \mathcal{M}$ and η_k in (1.1) must be in $\mathrm{T}_{x_k} \mathcal{M}$, where $\mathrm{T}_{x_k} \mathcal{M}$ denotes the tangent space at x_k. Therefore, the linear operator $\mathcal{B}_k$ is usually defined to be a mapping from

$\mathrm{T}_{x_k}\mathcal{M}$ to $\mathrm{T}_{x_k}\mathcal{M}$. Additionally, the $\mathfrak{s}_k$ and $\mathfrak{y}_k$ can be defined in $\mathrm{T}_{x_k}\mathcal{M}$ (see details in [RW12, HGA15]), which implies that the Riemannian generalization of (1.2)

$$\kappa_{\mathrm{BFGS}}(\mathcal{B}_k) = \mathcal{B}_k - \frac{\mathcal{B}_k s_k (\mathcal{B}_k s_k)^\flat}{(\mathcal{B}_k s_k)^\flat s_k} + \frac{y_k y_k^\flat}{y_k^\flat s_k}$$

is a linear operator in $\mathrm{T}_{x_k}\mathcal{M}$, where $a^\flat$ represents the flat of a, i.e., $a^\flat : \mathrm{T}_x\mathcal{M} \to \mathbb{R} : v \mapsto g(a,v)$. However, we need $\mathcal{B}_{k+1}$ to be a linear operator in $\mathrm{T}_{x_{k+1}}\mathcal{M}$, not in $\mathrm{T}_{x_k}\mathcal{M}$. A solution is to resort to the notion of vector transport or transporter defined in [ADM02, AMS08, HGA15] (or see Section 2 for the definitions), which yields an adequate Riemannian generalization of (1.2):

$$\mathcal{B}_{k+1} = \mathcal{T}_k \circ \kappa_{\mathrm{BFGS}}(\mathcal{B}_k) \circ \mathcal{T}_k^{-1}. \tag{1.3}$$

The presence of the vector transport $\mathcal{T}_k$ in (1.3) usually requires extra matrix-vector or matrix-matrix multiplications, which can be expensive and slow down the entire method for some applications. For instance, if an application has cheap cost function and gradient evaluations, then the cost of an inefficient vector transport may dominate the method.

In [HAG15, Section 2.2], a d-dimensional representation of tangent vectors has been proposed. Specifically, if the d-dimensional manifold $\mathcal{M}$ is either a submanifold of a w-dimensional Euclidean space or a quotient manifold whose total space is a submanifold of a w-dimensional Euclidean space, then a tangent vector in the tangent space at $x \in \mathcal{M}$ can be represented by a w-dimensional vector, called w-dimensional representation, or by a d-dimensional vector which gathers the coefficients of the tangent vector in a basis B_x of the tangent space at x, called d-dimensional representation. Using the d-dimensional representation brings many computational benefits (see Section 3): (i) manipulating smaller dimensional vectors reduces time and spatial complexity; (ii) the d-dimensional representation of the *vector transport by parallelization* [HAG15, Section 2.3.1] induced by the basis field B is the identity, which is the cheapest one can expect; and (iii) if the basis B_x of the tangent space at x is orthonormal, then the Riemannian metric at x reduces to the Euclidean metric in the d-dimensional representation, which is cheap, and moreover the vector transport by parallelization becomes isometric. However, [HAG15] did not propose ways to efficiently compute d-dimensional representations. In particular, in [HAG15, Section 5], the proposed method to compute a d-dimensional representation on the Stiefel manifold of orthonormal p-frames in $\mathbb{R}^n$ requires building an orthogonal matrix $\begin{bmatrix} X & X_\perp \end{bmatrix}$ which takes $O(n^3)$ flops. This is why the experiments in [HAG15] were limited to fairly low values of n.

In this paper, we give detailed implementations to compute d-dimensional representation given w-dimensional representation and to compute w-dimensional representation given d-dimensional representation for some commonly encountered matrix manifolds, i.e., the compact Stiefel manifold, the Grassmann manifold, the fixed-rank manifold, and the manifold of positive semidefinite matrices with rank fixed. The basis field B that underlies the proposed d-dimensional representation is orthonormal, which ensures that the vector transport by parallelization is isometric. In the case of the Stiefel manifold of orthonormal p-frames in $\mathbb{R}^n$, this isometric vector transport requires only $O(np^2)$ flops, to be compared with previous solutions such as the one of [HAG15, Section 5] which requires $O(n^3)$. The proposed isometric vector transport thus brings a significant speedup in the frequently encountered situation where $n \gg p$. This improvement is especially welcome since the vector transport used in several Riemannian optimization methods needs to be isometric in order to allow for a suitable convergence analysis; see, e.g., [RW12, HAG15, HGA15]. We illustrate these beneficial consequences in the context of a Riemannian BFGS method applied to the Brockett objective function on the Stiefel manifold.

This paper is organized as follows. Section 2 presents the preliminaries and notation of Riemannian manifolds. Section 3 describes computational benefits of d-dimensional representation. Section 4 demonstrates the algorithms for computing the d-dimensional representation from the w-dimensional representation and vise versa on the Stiefel manifold. Due to the availability of the d-dimensional representation and w-dimensional representation on the Stiefel manifold, the implementations of the conversions of the two representations for the Grassmann manifold, the fixed-rank manifold and the manifold with symmetric positive semidefinite matrices with rank fixed can be computed and demonstrated in Section 5. By applying a Riemannian quasi-Newton method for optimizing a function defined on the Stiefel manifold, Section 6 shows the advantage of intrinsic representation and vector transport by parallelization. Numerical experiments are reported in Section 7. Finally, the conclusion is given in Section 8.

2 Preliminaries and notation

The Riemannian concepts follow from the literature, e.g., [Boo86, AMS08]. The notation of this paper follows from [AMS08]. Let $\mathcal{M}$ denote a d-dimensional Riemannian manifold with the Riemannian metric $g : (\eta_x, \xi_x) \mapsto g_x(\eta_x, \xi_x) \in \mathbb{R}$, $\mathrm{T}_x \mathcal{M}$ denote the tangent space of $\mathcal{M}$ at x, and $\mathrm{T}\mathcal{M}$ denote the tangent bundle, i.e., the set of all tangent spaces.

2.1 Riemannian submanifold and quotient manifold

In this paper, we only consider the following two kinds of manifolds $\mathcal{M}$: (i) $\mathcal{M}$ is an embedded submanifold of a w-dimensional Euclidean space $\mathcal{E}$ (see Figure 1), and (ii) $\mathcal{M}$ is a quotient manifold $\bar{\mathcal{M}}/\mathcal{G} = \{[x] | x \in \bar{\mathcal{M}}\}$, where $\bar{\mathcal{M}}$ is a submanifold of a w-dimensional Euclidean space $\mathcal{E}$, $\mathcal{G}$ is a group acting on $\bar{\mathcal{M}}$ (see details in [Lee11, Theorem 21.10]) and $[x] = \{gx | g \in \mathcal{G}\}$ (see Figure 1). Every element of the quotient manifold $\bar{\mathcal{M}}/\mathcal{G}$ is a submanifold of $\bar{\mathcal{M}}$. In practice, a point x in $[x]$ is used to represent the submanifold $[x]$. The

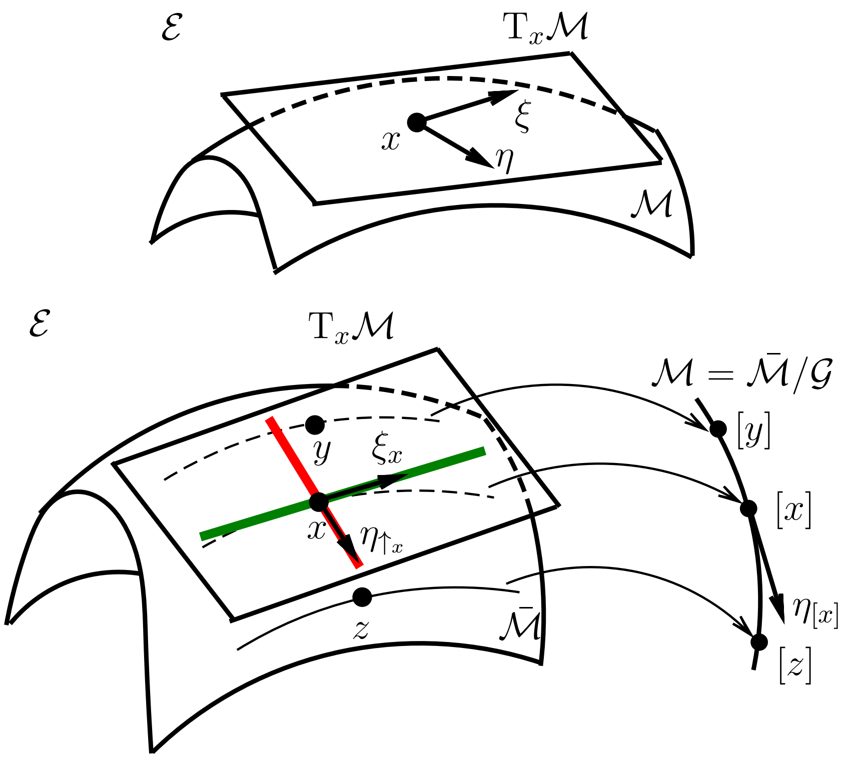

Fig. 1 Top: An embedded submanifold of a Euclidean space $\mathcal{E}$. Bottom: Notation of objects of a quotient manifold of $\mathcal{M}$. The green line denotes the vertical space $\mathcal{V}_x$ at x and the red line denotes the horizontal space $\mathcal{H}_x$ at x.

tangent space $\mathrm{T}_x[x]$ is called the vertical space $\mathcal{V}_x$ at x. The horizontal space is defined to be the perpendicular space of $\mathcal{V}_x$, i.e., $\mathcal{H}_x \oplus \mathcal{V}_x = \mathrm{T}_x \bar{\mathcal{M}}$, where the orthogonality is defined by the Riemannian metric of $\bar{\mathcal{M}}$. For any tangent vector $\eta_{[x]} \in \mathrm{T}_{[x]} \mathcal{M}$, the unique representation in $\mathcal{H}_x$ is called the horizontal lift of $\eta_{[x]}$ at x, denoted by $\eta_{\uparrow_x}$.

2.2 Retraction, vector transport, and transporter

The concepts of retraction and vector transport can be found in [AMS08] or [QGA10]. A retraction R is a smooth mapping from the tangent bundle $\mathrm{T}\mathcal{M}$ onto $\mathcal{M}$ such that (i) $R(0_x) = x$ for all $x \in \mathcal{M}$ (where 0_x denotes the origin of $\mathrm{T}_x \mathcal{M}$) and (ii) $\frac{d}{dt} R(t\xi_x)|_{t=0} = \xi_x$ for all $\xi_x \in \mathrm{T}_x \mathcal{M}$. The restriction of R to $\mathrm{T}_x \mathcal{M}$ is denoted by R_x.

A vector transport $\mathcal{T} : \mathrm{T}\mathcal{M} \oplus \mathrm{T}\mathcal{M} \to \mathrm{T}\mathcal{M}, (\eta_x, \xi_x) \mapsto \mathcal{T}_{\eta_x} \xi_x$ with associated retraction R is a smooth mapping such that, for all (x, η_x) in the domain of R and all $\xi_x, \zeta_x \in \mathrm{T}_x \mathcal{M}$, it holds that (i) $\mathcal{T}_{\eta_x} \xi_x \in \mathrm{T}_{R(\eta_x)} \mathcal{M}$, (ii) $\mathcal{T}_{0_x} \xi_x = \xi_x$,

(iii) $\mathcal{T}_{\eta_x}$ is a linear map. An isometric vector transport additionally satisfies (iv)

$$g_{R(\eta_x)}(\mathcal{T}_{\eta_x}\xi_x, \mathcal{T}_{\eta_x}\zeta_x) = g_x(\xi_x, \zeta_x). \tag{2.1}$$

The vector transport by differentiated retraction R is defined by

$$\mathcal{T}_{R_{\eta_x}}\xi_x = \frac{d}{dt}R_x(\eta_x + t\xi_x)|_{t=0}, \tag{2.2}$$

which is in general not isometric.

Note that a vector transport requires an associated retraction R. Recently, a transporter $\mathcal{L}$ was defined in [HGA15] that does not require a retraction, i.e., $\mathcal{L}(x,y)$ is a linear operator from $\mathrm{T}_x\,\mathcal{M}$ to $\mathrm{T}_y\,\mathcal{M}$ whose dependence on x and y is jointly smooth and such that $\mathcal{L}(x,x)$ is identity for all x. Given a retraction R, it can be shown that $\mathcal{T}$ defined by

$$\mathcal{T}_{\eta_x}\xi_x = \mathcal{L}(x, R_x(\eta_x))\,\xi_x \tag{2.3}$$

is a vector transport with associated retraction R. Moreover, if $\mathcal{L}(x,y)$ is isometric from $\mathrm{T}_x\,\mathcal{M}$ to $\mathrm{T}_y\,\mathcal{M}$, then the vector transport (2.3) is isometric.

If $\mathcal{M}$ is an embedded submanifold of a w-dimensional Euclidean space $\mathcal{E}$, then the transporter by projection [AMS08, Section 8.1.3] is defined by

$$\mathcal{L}^{\mathrm{Pj}}(x,y)\xi_x = P_y\xi_x,$$

where P_y denotes the orthogonal projection to $\mathrm{T}_y\,\mathcal{M}$. The transporter by parallelization [HGA15, Section 4.3] is defined by

$$\mathcal{L}^{\mathrm{Pl}}(x,y)\xi_x = B_yB_x^{\dagger}\xi_x, \tag{2.4}$$

where $B : \mathcal{V} \to \mathbb{R}^{w\times d} : z \mapsto B_z$ is a smooth tangent basis field defined on an open set $\mathcal{V}$ of $\mathcal{M}$, $B_z^{\dagger}$ denotes the pseudo-inverse of B_z, and $B_x^{\dagger}\xi_x$ and B_yv denote matrix vector multiplications. Note that it may not be possible to have $\mathcal{V} = \mathcal{M}$; for example, when $\mathcal{M}$ is an even-dimensional sphere, it would contradict the hairy ball theorem [Lee11, 16-6]. However, given any $x \in \mathcal{M}$, there always exists a basis field B which is smooth on a neighborhood of x [HAG15]. It has been shown that if B_z forms an orthonormal basis of $\mathrm{T}_z\,\mathcal{M}$, then $\mathcal{L}(x,y)$ is isometric, which implies that the vector transport by parallelization $\mathcal{T}_{\eta_x}\xi_x = B_{R_x(\eta_x)}B_x^{\dagger}\xi_x$ is isometric [HAG15, (6)].

If $\mathcal{M}$ is a quotient manifold, then one can similarly define

$$\mathcal{L}^{\mathrm{Pj}}(x,y)\xi_x = P_y^h\xi_x \text{ and } \mathcal{L}^{\mathrm{Pl}}(x,y)\xi_x = B_y^h(B_x^h)^{\dagger}\xi_x, \tag{2.5}$$

where P_y^h denotes the projection to the horizontal space $\mathcal{H}_y$ and the columns of B_z^h form an orthonormal basis of $\mathcal{H}_z$. If (2.5) is independent of the representation chosen in $[x]$ and $[y]$, then they define transporters by projection and by parallelization for the quotient manifold $\mathcal{M}$. This is summarized in Lemma 21.

Lemma 21 *Let $\mathcal{L}^{\bar{\mathcal{M}}}(x, y)$ denote a transporter on $\bar{\mathcal{M}}$. Recall the notation $\eta_{\uparrow_x}$ for the horizontal lift at x of $\eta_{[x]} \in \mathrm{T}_{[x]}\,\mathcal{M}$. If $\mathcal{L}^{\bar{\mathcal{M}}}(x_1, y_1)\eta_{\uparrow_{x_1}}$ and $\mathcal{L}^{\bar{\mathcal{M}}}(x_2, y_2)\eta_{\uparrow_{x_2}}$ are horizontal lifts of a unique tangent vector in $\mathrm{T}_{[y]}\,\mathcal{M}$ for all $x_1, x_2 \in [x]$, $y_1, y_2 \in [y]$, $[x], [y] \in \mathcal{M}$ and $\eta_{[x]} \in \mathrm{T}_{[x]}\,\mathcal{M}$, then*

$$(\mathcal{L}^{\mathcal{M}}([x], [y])\eta_{[x]})_{\uparrow_y} = \mathcal{L}^{\bar{\mathcal{M}}}(x, y)\eta_{\uparrow_x}$$

defines a transporter on $\mathcal{M}$.

Proof This can be easily verified by the definition.

3 Intrinsic representation of tangent vectors and computational benefits

Throughout this paper, d-dimensional and w-dimensional representation of a tangent vector are also called intrinsic and extrinsic representation, respectively. In this section, we present the potential computational benefits that the intrinsic representation may bring. The potential benefits are then made concrete in the following sections for several specific manifolds.

Let a tangent vector η_x in $\mathrm{T}_x\,\mathcal{M}$ be represented by a d-dimensional vector, denoted by v_x, of coordinates in a given basis B_x of $\mathrm{T}_x\,\mathcal{M}$. Let functions

$$\mathrm{D2E}_x^{\mathcal{M}} : v_x \mapsto \eta_x = B_x v_x \quad \text{and} \quad \mathrm{E2D}_x^{\mathcal{M}} : \eta_x \mapsto v_x = B_x^{\dagger}\eta_x$$

denote the maps converting from one representation to the other representation. The intrinsic representation $\mathrm{E2D}_x^{\mathcal{M}} \circ \mathcal{L}^{\mathrm{Pl}}(x, y) \circ \mathrm{D2E}_x^{\mathcal{M}}$ of the transporter by parallelization is readily seen to be the identity; indeed

$$\mathrm{E2D}_x^{\mathcal{M}} \circ \mathcal{L}^{\mathrm{Pl}}(x, y) \circ \mathrm{D2E}_x^{\mathcal{M}} v_x = B_y^{\dagger}(B_y B_x^{\dagger}(B_x v_x)) = v_x.$$

Moreover, if the columns of B_x form an orthonormal basis of $\mathrm{T}_x\,\mathcal{M}$, then the Riemannian metric reduces to the Euclidean metric for the intrinsic representations, i.e.,

$$g(\eta_x, \xi_x) = g(B_x v_x, B_x u_x) = v_x^T u_x,$$

where $\eta_x = B_x v_x$ and $\xi_x = B_x u_x \in \mathrm{T}_x\,\mathcal{M}$.

If one can compute v_x cheaply given η_x and vice versa, which is true for the manifolds discussed later, then the operations on the tangent space only require manipulating d-dimensional vectors rather than w-dimensional vectors. For example, solving a linear system $\mathcal{B}\eta = -\operatorname{grad} f$ or optimizing a local model $g(\operatorname{grad} f, \eta) + \frac{1}{2}g(\mathcal{B}\eta, \eta)$ in quasi-Newton methods needs $O(d^3)$ rather than $O(w^3)$. Besides, the implementation of transporter by parallelization (2.4) and Riemannian metric are simplified.

4 The compact Stiefel manifold

The compact Stiefel manifold or the Stiefel manifold for short is the set of orthonormal matrices, i.e., $\mathrm{St}(p,n) = \{X \in \mathbb{R}^{n\times p} | X^T X = I\}$. The Stiefel manifold can be viewed as a submanifold of $\mathbb{R}^{n\times p}$. Its dimension d is $pn - \frac{p(p+1)}{2}$ and the dimension w of the embedding space $\mathbb{R}^{n\times p}$ is pn. The tangent space of the Stiefel manifold is $\mathrm{T}_X \mathrm{St}(p,n) = \{X\Omega + X_\perp K | \Omega = -\Omega^T, K \in \mathbb{R}^{(n-p)\times p}\}$. Throughout this paper, given $M \in \mathbb{R}^{n\times p}$, $M_\perp$ denotes an n-by-$(n-p)$ matrix whose columns form an orthonormal basis of the orthogonal complement of the column space of M. The proposed d-dimensional representation of $X\Omega + X_\perp K \in \mathrm{T}_X \mathrm{St}(p,n)$ keeps the non-redundant information in Ω and K. The key result, introduced next, is a conversion method in $O(np^2)$ flops between the d-dimensional and w-dimensional representations that does *not* require choosing and building the cumbersome $n \times (n-p)$ matrix $X_\perp$. Due to this result, the d-dimensional representation becomes tractable in the important case where $n \gg p$. The d-dimensional representation proposed next is associated to a basis field B that is orthonormal in the sense of the g_c metric [EAS98, (2.22)], which is one of the two "natural" metrics commonly used on the Stiefel manifolds. Hence the vector transport by parallelization—which reduces to the identity in the d-dimensional representation—is isometric in the sense of the g_c metric. As far as we are aware, no cheap $O(np^2)$ isometric vector transport on the Stiefel manifold has been proposed before. At the end of this section, we will briefly comment on how the representation can be easily adapted to rely on a basis field B that is orthonormal with respect to the g_e metric [EAS98, (2.2)], which is the other "natural" metric on the Stiefel manifold.

The proposed d-dimensional representation exploits the fact that any tangent vector $U \in \mathrm{T}_X \mathrm{St}(p,n)$ can be written as

$$U = X \begin{bmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1p} \\ -a_{12} & 0 & a_{23} & \cdots & a_{2p} \\ -a_{13} & -a_{23} & 0 & \cdots & a_{3p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_{1p} & -a_{2p} & -a_{3p} & \cdots & 0 \end{bmatrix} + X_\perp \begin{bmatrix} b_{11} & b_{12} & b_{13} & \cdots & b_{1p} \\ b_{21} & b_{22} & b_{23} & \cdots & b_{2p} \\ b_{31} & b_{32} & b_{33} & \cdots & b_{3p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{(n-p)1} & b_{(n-p)2} & b_{(n-p)3} & \cdots & b_{(n-p)p} \end{bmatrix} \tag{4.1}$$

One way to define a basis of $\mathrm{T}_X \mathrm{St}(p,n)$ is thus

$$\begin{aligned} &\{X(e_i e_j^T - e_j e_i^T) : i = 1, \ldots, p, j = i+1, \ldots, p\} \bigcup \\ &\qquad \{X_\perp \tilde{e}_i e_j^T, i = 1, \ldots, n-p, j = 1, \ldots, p\}, \end{aligned} \tag{4.2}$$

where $(e_1, \ldots, e_p)$ is the canonical basis of $\mathbb{R}^p$ and $(\tilde{e}_1, \ldots, \tilde{e}_{n-p})$ is the canonical basis of $\mathbb{R}^{n-p}$. Proposition 41 shows an important property of the basis (4.2).

Proposition 41 *The basis of* $\mathrm{T}_X \mathrm{St}(p,n)$ *defined in* (4.2) *is orthonormal with respect to the canonical metric [EAS98, (2.22)]*

$$g_c(U,V) = \mathrm{trace}(U^T (I_n - \frac{1}{2} X X^T) V), \tag{4.3}$$

where $U, V \in \mathrm{T}_X \operatorname{St}(p,n)$.

Proof This can be easily verified.

Using the basis defined in (4.2), the intrinsic representation of U is given by

$$\mathrm{E2D}_X^{\mathrm{St}(p,n)}(U) = \big(a_{12}, a_{13}, a_{23}, \ldots, a_{1p}, a_{2p}, a_{3p}, \ldots, a_{(p-1)p},\\ b_{11}, b_{21}, \ldots, b_{(n-p)1}, \ldots, b_{1p}, \ldots, b_{(n-p)p}\big)^T.$$

Proposition 41 implies that the vector transport by parallelization using the basis (4.2) is isometric and the Riemannian metric (4.3) reduces to the Euclidean metric using the intrinsic representation.

The functions E2D and D2E using basis (4.2) are summarized in Algorithms 1 and 2 respectively. Ingredients of those algorithms are explained in the paragraphs that follow.

Note that Step 2 of Algorithm 1 computes Ω by $(\tilde{\Omega} - \tilde{\Omega}^T)/2$ to make sure Ω is skew symmetric numerically even though $\tilde{\Omega}$ is a skew symmetric matrix theoretically.

Throughout this paper, the computational complexity is measured by flop counts. A flop is a floating point operation [GV96, Section 1.2.4]. The number of flops for each step is stated on the right of each algorithm.

Algorithm 1 Compute $\mathrm{E2D}_X^{\mathrm{St}(p,n)}(U)$

```
Require: $X \in \operatorname{St}(p,n)$, $U \in \mathrm{T}_X \operatorname{St}(p,n)$, a function $\alpha_X : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : A \mapsto \begin{bmatrix} X & X_\perp \end{bmatrix}^T A$
    (see Algorithm 4).
 1: $\begin{bmatrix} \Omega \\ K \end{bmatrix} = \alpha_X(U)$, where $\Omega \in \mathbb{R}^{p\times p}$ and $K \in \mathbb{R}^{(n-p)\times p}$;      ▷ See flops in Algorithm 4
 2: Set $\Omega = (\tilde{\Omega} - \tilde{\Omega}^T)/2$ and $k = 1$;                      # $2p^2$
 3: for $j = 2, \ldots, p$, $i = 1, \ldots j-1$ do                               ▷ # $p(p-1)$
 4:     $v_X(k) = \Omega_{ij}$, where $\Omega_{ij}$ is the $i$-th row $j$-th column entry of $\Omega$;
 5:     $k \leftarrow k + 1$;
 6: end for
 7: for $i = 1, \ldots, (n-p)$, $j = 1, \ldots, p$ do                           ▷ # $p(n-p)$
 8:     $v_X(k) = K_{ij}$ and $k \leftarrow k + 1$;
 9: end for
10: return vector $v_X \in \mathbb{R}^{np - p(p+1)/2}$;
```

Algorithms 1 and 2 rely on the functions α and β. Since we want the transporter by parallelization—namely $\mathcal{L}^{\mathrm{Pl}}(x,y) = \mathrm{D2E}_Y^{\mathrm{St}(p,n)} \circ \mathrm{E2D}_X^{\mathrm{St}(p,n)}$—to be smooth, we need α and β to be smooth. One approach to construct smooth α_X and β_X is to compute $X_\perp$ explicitly, e.g., as in [HAG15, Section 5]. However, the complexity of this approach is $O(n(n-p)^2)$, which is expensive especially when $p \ll n$. The suggested approach is stated in Algorithm 4 for α_X and Algorithm 5 for β_X. They both rely on Algorithm 3 which computes the QR decomposition for almost any matrix $Z \in \mathbb{R}^{n\times p}$ by Householder transformations, which can be done efficiently by the linear algebra package [ABB+99]

Algorithm 2 Compute $\mathrm{D2E}_X^{\mathrm{St}(p,n)}(v_X)$

```
Require: $X \in \mathrm{St}(p,n)$, $v_x \in \mathbb{R}^{np-p(p+1)/2}$, a function $\beta_X : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : A \mapsto \begin{bmatrix} X & X_\perp \end{bmatrix} A$ (see Algorithm 5).
1: $k = 1$;;
2: for $j = 2, \dots, p$, $i = 1, \dots j-1$ do                $\triangleright$ # $p(p-1)$
3:     $\Omega_{ij} = v_X(k)$ and $\Omega_{ji} = -v_X(k)$;
4:     $k \leftarrow k+1$;
5: end for
6: for $i = 1, \dots, (n-p)$, $j = 1, \dots, p$ do          $\triangleright$ # $p(n-p)$
7:     $K_{ij} = v_X(k)$ and $k \leftarrow k+1$;
8: end for
9: return $\beta_X \begin{bmatrix} \Omega \\ K \end{bmatrix}$;            $\triangleright$ # See flops in Algorithm 5;
```

function *?geqrf* or *?geqp3*, where the question mark stands for different choices for precisions and number types, e.g., sgeqrf is for single precision real number, dgeqrf is for double precision real number. Algorithm 3 uses the product of the Householder matrices defined by the unit vectors $(v_1, \dots, v_p)$ and sign scalars $(s_1, \dots, s_p)$, to represent the orthonormal matrix $\begin{bmatrix} X & X_\perp \end{bmatrix}$. The details are given in Lemma 41. Note that it has been shown in [Lee11, 16-6] that not all manifolds have a continuous non-vanishing vector field globally. Therefore, we do not expect $\phi(Z)$ to be a continuous function for all $Z \in \mathbb{R}^{n\times p}$. [1]

Lemma 41 *Let the function $\phi : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times n} : Z \mapsto \phi(Z)$ be defined by*

$$\phi(Z) = Q_1 Q_2 \cdots Q_p \,\mathrm{diag}(s_1, s_2, \dots, s_p, I_{n-p}), \tag{4.4}$$

where Q_i is $\begin{bmatrix} I_{i-1} & 0 \\ 0 & I_{n-i+1} - 2v_i v_i^T \end{bmatrix}$, $\mathrm{diag}(a_1, a_2, \dots, a_t)$ denotes a block diagonal matrix whose diagonal blocks are a_1, a_2 and a_t, $\{v_1, v_2, \dots, v_p\}$ and $\{s_1, s_2, \dots, s_p\}$ are given by Algorithm 3 with Z the input matrix. Then $\phi(Z)$ is smooth at Z satisfying that $\tilde{z}_{i1}$ in Step 2 of Algorithm 3 is nonzero for all i. Moreover, if $Z \in \mathrm{St}(p,n)$, then $\phi(Z) = \begin{bmatrix} Z & Z_\perp \end{bmatrix}$.

Proof If $\tilde{z}_{i1}$ is nonzero for all i, then a small enough perturbation on Z does not change $\mathrm{sgn}(\tilde{z}_{i1})$, which implies ϕ is smooth at Z.

By definition of $\phi(Z)$, we have $Z = \phi(Z) \begin{bmatrix} R \\ 0_{(n-p)\times p} \end{bmatrix}$, and $\phi^T(Z)\phi(Z) = I_n$, where $R \in \mathbb{R}^{p\times p}$ is an upper triangular matrix with positive diagonal entries. If $Z \in \mathrm{St}(p,n)$, then $R = I_p$. Therefore, $\phi(Z) = \begin{bmatrix} Z & Z_\perp \end{bmatrix}$.

With a slight abuse of notation, we use $\alpha_{(V_X, S_X)}$ and $\beta_{(V_X, S_X)}$ to denote Algorithms 4 and 5 with input $V_X = (v_1, v_2, \dots, v_p)$ and $S_X = (s_1, s_2, \dots, s_p)$, which are from Algorithm 3 with input X. It follows from Lemma 41 that

[1] It should be noted that for those Z such that $\phi(Z)$ is nonsmooth, there always exists a permutation matrix P, which is usually obtained by row pivoting QR decomposition, such that $\phi(PZ)$ is smooth. Therefore, one can define $\tilde{\phi}(Z) = P^T \phi(PZ)$, which is also smooth and satisfies $\tilde{\phi}(Z) = \begin{bmatrix} Z & Z_\perp \end{bmatrix}$.

$\alpha_{(V_Z,S_Z)}(A)$ and $\beta_{(V_Z,S_Z)}(A)$ are $\phi(Z)^T A$ and $\phi(Z)A$ respectively, which yield Algorithms 4 and 5. Both Algorithms 4 and 5 have been efficiently implemented in the linear algebra package [ABB+99] by function *?ormqr*.

The actions of the Householder matrices or their inverses are only a few rank one updates, which are cheap. The complexities of Algorithms 3, 4 and 5 are $2np^2 - 2p^3/3$, $4np^2 - 2p^3$ and $4np^2 - 2p^3$ respectively. Note that Algorithm 3 does not necessarily require extra $2np^2 - 2p^3/3$ flops. The computations can be done in the evaluation of retraction. For instance, the unit vectors $(v_1, v_2, \dots, v_p)$ and scalars $(s_1, s_2, \dots, s_p)$ can be obtained without extra cost when Householder reflections are used to compute the qf retraction [AMS08, (4.8)]

$$R_X(U) = \mathrm{qf}(X + U), \tag{4.5}$$

where $\mathrm{qf}(A)$ denote the Q factor of the QR decomposition with nonnegative elements on the diagonal of the upper triangle matrix. The details can be found in Section 6.

Algorithm 3 Compute unit vectors in Householder matrices $(v_1, v_2, \dots, v_p)$ and sign scalars $(s_1, s_2, \dots, s_p)$

```
Require: $Z = [z_1\ z_2\ \dots\ z_p] \in \mathbb{R}^{n\times p}$;
1: for $i = 1, \dots, p$ do                                  ▷ # $2np^2 - 2p^3/3$
2:     Let $a$ denote $-\mathrm{sgn}(\tilde{z}_{i1})\|\tilde{z}_i\|_2$ and define $v_i = (\tilde{z}_i - a e_1)/\|\tilde{z}_i - a e_1\|_2$ and $s_i = -\mathrm{sgn}(\tilde{z}_{i1})$, where $\tilde{z}_i$ is the vector formed by last $n - i + 1$ entries of $z_i$, $\tilde{z}_{i1}$ is the first entry of $\tilde{z}_i$ and $e_1$ denotes the first canonical basis of $\mathbb{R}^{n-i+1}$;
3:     $Z = [z_1\ z_2\ \dots\ z_p] \leftarrow Q_i Z$, where $Q_i = \begin{bmatrix} I_{i-1} & 0 \\ 0 & I_{n-i+1} - 2v_i v_i^T \end{bmatrix}$.
4: end for
5: return $(v_1, v_2, \dots, v_p)$ and $(s_1, s_2, \dots, s_p)$;
```

Algorithm 4 Compute $\alpha_X(A)$

```
Require: $A \in \mathbb{R}^{n\times p}$, and $V_X = (v_1, v_2, \dots, v_p)$ and $S_X = (s_1, s_2, \dots, s_p)$ generated by Algorithm 3 with input $X$;
1: for $i = 1, \dots, p$ do                                  ▷ # $4np^2 - 2p^3$
2:     $A \leftarrow Q_i A$, where $Q_i = \begin{bmatrix} I_{i-1} & 0 \\ 0 & I_{n-i+1} - 2v_i v_i^T \end{bmatrix}$;
3: end for
4: return $\mathrm{diag}(s_1, s_2, \dots, s_p, I_{n-p})A$;             ▷ # $p^2$
```

Another basis of $\mathrm{T}_X \mathrm{St}(p, n)$ is

$$\{\frac{1}{\sqrt{2}} X(e_i e_j^T - e_j e_i^T) : i = 1, \dots, p, j = i + 1, \dots, p\} \bigcup \{X_\perp \tilde{e}_i e_j^T, i = 1, \dots, n - p, j = 1, \dots, p\} \tag{4.6}$$

Algorithm 5 Compute $\beta_X(A)$

```
Require: $A \in \mathbb{R}^{n\times p}$, and $V_X = (v_1, v_2, \dots, v_p)$ and $S_X = (s_1, s_2, \dots, s_p)$ generated by
    Algorithm 3 with input $X$;
1: $A \leftarrow \mathrm{diag}(s_1, s_2, \dots, s_p, I_{n-p})A$                      ▷ # $p^2$
2: for $i = p, (p-1) \dots, 1$ do                                                  ▷ # $4np^2 - 2p^3$
3:     $A \leftarrow Q_i A$, where $Q_i = \begin{bmatrix} I_{i-1} & 0 \\ 0 & I_{n-i+1} - 2v_i v_i^T \end{bmatrix}$.
4: end for
5: Return $A$;
```

which can be shown to be an orthonormal basis with respect to the metric $g_e(U, V) := \mathrm{trace}(U^T V)$ defined in [EAS98, (2.2)]. It follows that the intrinsic representation of (4.1) is

$$\mathrm{E2D}_X^{\mathrm{St}(p,n)}(U) = \Big(\sqrt{2}a_{12}, \sqrt{2}a_{13}, \sqrt{2}a_{23}, \dots, \sqrt{2}a_{1p}, \sqrt{2}a_{2p}, \sqrt{2}a_{3p}, \dots, \sqrt{2}a_{(p-1)p}, \\ b_{11}, b_{21}, \dots, b_{(n-p)1}, \dots, b_{1p}, \dots, b_{(n-p)p}\Big)^T.$$

Algorithms 1 and 2 therefore can be modified to match the bases (4.6), i.e., Step 4 of Algorithm 1 is replaced by $v_X(k) = \sqrt{2}\Omega_{ij}$ and Step 3 of Algorithm 2 is replaced by $\Omega_{ij} = v_X(k)/\sqrt{2}$.

5 More matrix manifolds

In this section, the efficient implementations of D2E and E2D for the Grassmann manifold, the fixed-rank manifold, and the manifold of positive semidefinite matrices with rank fixed are given using the same idea presented in Section 4. We propose a tangent basis field for each manifold. In addition, the tangent basis B_x is orthonormal with respect to a Riemannian metric of each manifold. Therefore, the vector transport by parallelization, which has identity implementation, is isometric for the Riemannian metric. We emphasize that as far as we know, there is no cheap isometric vector transport proposed before for the fixed-rank manifold, and the manifold of positive semidefinite matrices with rank fixed. As shown in Section 3, the considered Riemannian metric reduces to the Euclidean metric when the intrinsic representation is used.

5.1 The Grassmann manifold

The Grassmann Manifold $\mathrm{Gr}(p, n)$ is the set of all p dimensional linear subspace of an n dimensional linear space. In this paper, we consider the representation $\mathrm{Gr}(p, n) = \mathrm{St}(p, n)/\mathcal{O}_p$, where $\mathcal{O}_p$ denotes the p-by-p orthogonal group. Specifically, an element in $\mathrm{Gr}(p, n)$ is an orbit in $\mathrm{St}(p, n)$, i.e., $\mathrm{Gr}(p, n) = \{[X] | X \in \mathrm{St}(p, n)\}$, where $[X] = \{XO | O \in \mathcal{O}_p\}$. This representation uses the set of orthonormal bases of a linear space as a representation of the linear space. The detailed discussions can be found in e.g., [EAS98, Section 2.5]. The metric of the Grassmann manifold considered in this paper is endowed from the Euclidean space

$$g(U,V) = \mathrm{trace}(U^T V). \tag{5.1}$$

The horizontal space at X is

$$\mathcal{H}_X = \{X_\perp K : K \in \mathbb{R}^{(n-p)\times p}\}.$$

An orthonormal basis of the horizontal space at X with respect to the metric (5.1) is given by

$$\{X_\perp \tilde{e}_i e_j^T, i = 1, \dots, n-p, j = 1, \dots, p\},$$

where $(e_1, \dots, e_p)$ is the canonical basis of $\mathbb{R}^p$ and $(\tilde{e}_1, \dots, \tilde{e}_{n-p})$ is the canonical basis of $\mathbb{R}^{n-p}$. This basis is the second half of the basis of $\mathrm{T}_X \mathrm{St}(p,n)$ in (4.2). Therefore, the $\mathrm{E2D}^{\mathrm{Gr}(p,n)}$ and $\mathrm{D2E}^{\mathrm{Gr}(p,n)}$ can be easily generated by using the ideas of Algorithm 1 and Algorithm 2.

5.2 The fixed-rank manifold

The fixed-rank manifold is the set of matrices with rank fixed, i.e., $\mathcal{M}_r = \{X \in \mathbb{R}^{m\times n} | \mathrm{rank}(X) = r\}$, where $r < \min(m,n)$. The discussions about its geometric structure can be found in e.g., [Van13] and [AO15]. Let $X \in \mathcal{M}_r$ and X be represented by $X = USV^T$, where $U \in \mathrm{St}(r,m)$, $V \in \mathrm{St}(r,n)$ and $S \in \mathbb{R}^{r\times r}$ is not necessary diagonal. It follows that the tangent space of $\mathcal{M}_r$ at X is

$$\mathrm{T}_X \mathcal{M}_r = \Big\{ U\dot{S}V^T + \dot{U}SV^T + US\dot{V}^T : \\ \dot{S} \in \mathbb{R}^{r\times r}, \dot{U} \in \mathbb{R}^{m\times r}, U^T\dot{U} = 0, \dot{V} \in \mathbb{R}^{n\times r}, V^T\dot{V} = 0 \Big\},$$

or equivalently

$$\mathrm{T}_X \mathcal{M}_r = \Big\{ U\dot{S}V^T + U_\perp K V^T + U W^T V_\perp^T : \\ \dot{S} \in \mathbb{R}^{r\times r}, K \in \mathbb{R}^{(m-r)\times r}, W \in \mathbb{R}^{(n-r)\times r} \Big\}.$$

The Euclidean metric is considered

$$g(\eta_X, \xi_X) = \mathrm{trace}(\eta_X^T \xi_X), \tag{5.2}$$

where $\eta_X, \xi_X \in \mathrm{T}_X \mathcal{M}_r$.

The tangent space $\mathrm{T}_X \mathcal{M}_r$ can be written as

$$\left\{ \begin{bmatrix} U & U_\perp \end{bmatrix} \begin{bmatrix} \dot{S} & W^T \\ K & 0 \end{bmatrix} \begin{bmatrix} V & V_\perp \end{bmatrix}^T \mid \dot{S} \in \mathbb{R}^{r\times r}, K \in \mathbb{R}^{(m-r)\times r}, W \in \mathbb{R}^{(n-r)\times r} \right\}.$$

Since the orthonormal matrices $\begin{bmatrix}U \ U_\perp\end{bmatrix}$ and $\begin{bmatrix}V \ V_\perp\end{bmatrix}$ do not influence the metric (5.2), an orthonormal basis is given by

$$\{Ue_ie_jV^T, i = 1, \dots r, j = 1, \dots, r\} \cup \{U_\perp \tilde{e}_i e_j V^T, i = 1, \dots, m-r, j = 1, \dots, r\} \bigcup \{Ue_i\bar{e}_jV_\perp^T, i = 1, \dots, r, j = 1, \dots, n-r\},$$

where $(e_1, \dots, e_r)$ is the canonical basis of $\mathbb{R}^r$, $(\tilde{e}_1, \dots, \tilde{e}_{m-r})$ is the canonical basis of $\mathbb{R}^{m-r}$ and $(\bar{e}_1, \dots, \bar{e}_{n-r})$ is the canonical basis of $\mathbb{R}^{n-r}$. In practice, the extrinsic representation of a tangent vector $\eta_X = U\dot{S}V^T + \dot{U}SV^T + US\dot{V}^T$ is usually represented by $(\dot{U}, \dot{S}, \dot{V})$, see [AO15], which avoids the dense m-by-n matrix and reduce the computational cost. The E2D and D2E functions using representation $(\dot{U}, \dot{S}, \dot{V})$ are given in Algorithms 6 and 7 respectively.

Algorithm 6 Compute $\mathrm{E2D}_X^{\mathcal{M}_r}(U)$

```
Require: $X = USV^T \in \mathcal{M}_r$, $\dot{X} = U\dot{S}V^T + \dot{U}SV^T + US\dot{V}^T \in \mathrm{T}_X\,\mathcal{M}_r$ represented by $(\dot{U}, \dot{S}, \dot{V})$, a function $\alpha_U : \mathbb{R}^{m\times r} \to \mathbb{R}^{m\times r} : A \mapsto \begin{bmatrix}U \ U_\perp\end{bmatrix}^T A$ and a function $\alpha_V : \mathbb{R}^{n\times r} \to \mathbb{R}^{n\times r} : A \mapsto \begin{bmatrix}V \ V_\perp\end{bmatrix}^T A$.
1: $K = (\alpha_U(\dot{U}S))_{(p+1:m,:)}$ and $W = \alpha_V(\dot{V}S^T)_{(p+1:n,:)}$, where $M_{(a:b,:)}$ denotes the submatrix forming by $a$-th row to $b$-th row of the matrix $M$; ▷ # $2(m+n)r^2$ and see flops in Algorithm 4
2: Reshape $\dot{S}$, $K$ and $W$ to be column vectors; Stack them to make a vector $v_X \in \mathbb{R}^{mr+nr-r^2}$
3: return vector $v_X$;
```

Algorithm 7 Compute $\mathrm{D2E}_X^{\mathcal{M}_r}(v_X)$

```
Require: $X = USV^T \in \mathcal{M}_r$, $v_X \in \mathbb{R}^{mr+nr-r^2}$, a function $\beta_U : \mathbb{R}^{m\times r} \to \mathbb{R}^{m\times r} : A \mapsto \begin{bmatrix}U \ U_\perp\end{bmatrix} A$ and a function $\beta_V : \mathbb{R}^{n\times r} \to \mathbb{R}^{n\times r} : A \mapsto \begin{bmatrix}V \ V_\perp\end{bmatrix} A$;
1: Reshape the first $r^2$ entries of $v_X$ to be $\dot{S} \in \mathbb{R}^{r\times r}$, the second $(m-r)r$ entries to be $K \in \mathbb{R}^{(m-r)\times r}$, and last $(n-r)r$ entries to be $W \in \mathbb{R}^{(n-r)\times r}$;
2: if $m \le n$ then
3:     Compute $M_1 = \beta_U \begin{bmatrix}\dot{S} \\ K\end{bmatrix}$ and $M_2 = \beta_V \begin{bmatrix}0 \\ W\end{bmatrix}$;    ▷ See flops in Algorithm 5
4:     $\dot{S} = U^T M_1$, $\dot{U} = (M_1 - U\dot{S})S^{-1}$, and $\dot{V} = M_2 S^{-T}$; ▷ # $2(m+n)r^2 + 4mr^2 + 2r^3/3$
5: else
6:     Compute $M_1 = \beta_U \begin{bmatrix}0 \\ K\end{bmatrix}$ and $M_2 = \beta_V \begin{bmatrix}\dot{S}^T \\ W\end{bmatrix}$;    ▷ See flops in Algorithm 5
7:     $\dot{S} = M_2^T V$, $\dot{U} = M_1 S^{-1}$, and $\dot{V} = (M_2 - V\dot{S}^T)S^{-T}$;    ▷ # $2(m+n)r^2 + 4nr^2 + 2r^3/3$
8: end if
9: return $(\dot{U}, \dot{S}, \dot{V})$ which represents $\dot{X} = U\dot{S}V^T + \dot{U}SV^T + US\dot{V}^T$;
```

5.3 The manifold of positive semidefinite matrices with rank fixed

Let $\mathrm{S}_+(p,n) = \{YY^T : Y \in \mathbb{R}_*^{n\times p}\}$ denote the manifold of positive semidefinite matrices with rank fixed, where $R_*^{n\times p}$ denotes all full-rank real $n\times p$ matrices. A discussion about this manifold can be found in e.g., [VAV09]. The tangent space at $X = YY^T$ is given by

$$\mathrm{T}_{YY^T}\,\mathrm{S}_+(p,n) = \left\{ \begin{bmatrix} Y & Y_\perp \end{bmatrix} \begin{bmatrix} 2S & K^T \\ K & 0 \end{bmatrix} \begin{bmatrix} Y^T \\ Y_\perp^T \end{bmatrix} \mid S = S^T \in \mathbb{R}^{p\times p}, K \in \mathbb{R}^{(n-p)\times p} \right\},$$

or equivalently

$$\mathrm{T}_{YY^T}\,\mathrm{S}_+(p,n) = \left\{ Y\dot{Y}^T + \dot{Y}Y^T \mid \dot{Y} \in \mathbb{R}^{n\times p}, \dot{Y} = YS + Y_\perp K \right\}$$

The Riemannian metric endow from $\mathbb{R}^{n\times n}$ is

$$g(Z_1, Z_2) = \mathrm{trace}(Z_1^T Z_2) = \mathrm{trace}(Y^TY(4S_1Y^TYS_2 + 2K_1^TK_2)), \qquad (5.3)$$

where $Z_1, Z_2 \in \mathrm{T}_X\,\mathrm{S}_+(p,n)$.

The tangent space can be expressed as

$$\mathrm{T}_{YY^T}\,\mathrm{S}_+(p,n) = \left\{ \begin{bmatrix} YL^{-T} & Y_\perp \end{bmatrix} \begin{bmatrix} 2L^TSL & L^TK^T \\ KL & 0 \end{bmatrix} \begin{bmatrix} L^{-1}Y^T \\ Y_\perp^T \end{bmatrix} \mid \right.$$
$$\left. S = S^T \in \mathbb{R}^{p\times p}, K \in \mathbb{R}^{(n-p)\times p} \right\},$$

where $Y^TY = LL^T$ is the Cholesky decomposition such that the diagonal entries of L are positive. Since the matrix $\begin{bmatrix} YL^{-T} & Y_\perp \end{bmatrix}$ is an orthonormal matrix, the orthogonality of tangent vectors depends on the middle term, i.e., $\begin{bmatrix} 2L^TSL & L^TK^T \\ KL & 0 \end{bmatrix}$. Thus, an orthonormal basis of $\mathrm{T}_{YY^T}\,\mathrm{S}_+(p,n)$ with respect to the Euclidean metric (5.3) is given by

$$\{YL^{-T}e_ie_i^TL^{-1}Y^T, i = 1, \dots p\} \bigcup$$
$$\{\frac{1}{\sqrt{2}}YL^{-T}(e_ie_j^T + e_je_i^T)L^{-1}Y^T, i = 1, \dots p, j = i+1, \dots, p\} \bigcup$$
$$\{\frac{1}{\sqrt{2}}Y_\perp\tilde{e}_ie_j^TL^{-1}Y^T + \frac{1}{\sqrt{2}}YL^{-T}e_j\tilde{e}_i^TY_\perp^T, i = 1, \dots, n-p, j = 1, \dots, p\}.$$

The E2D and D2E functions are given in Algorithms 8 and 9 respectively. Both Algorithms use $\dot{Y}$ to represent the extrinsic representation of the tangent vector $Y\dot{Y}^T + \dot{Y}Y^T$ at $YY^T \in \mathrm{S}_+(p,n)$.

Algorithm 8 Compute $\mathrm{E2D}_X^{\mathrm{S}_+(p,n)}(U)$

```
Require: $X = YY^T \in \mathcal{M}_r$, $\dot{X} = Y\dot{Y}^T + \dot{Y}Y^T \in \mathrm{T}_X\, \mathrm{S}_+(p,n)$ represented by $\dot{Y}$, a function $\alpha_Y : \mathbb{R}^{n\times p} \to \mathbb{R}^{(n-p)\times p} : A \mapsto \begin{bmatrix} YL^{-T} & Y_\perp \end{bmatrix}^T A$;   ▷ Note that $YL^{-T}$ is an orthonormal matrix.
 1: Compute the Cholesky decomposition $Y^TY = LL^T$;   ▷ # $2np^2 + p^3/3$
 2: $\begin{bmatrix} M \\ K \end{bmatrix} = \alpha_Y(\dot{Y})$;   ▷ # See flops in Algorithm 4 and note that $M = L^TS$
 3: $H = 2ML$ and $W = KL$ and set $k = 1$;   ▷ # $2np^2$
 4: for $i = 1, \dots, p$ do   ▷ # $p$
 5:     $v_X(k) = H_{ii}$ and $k = k + 1$;
 6: end for
 7: for $i = 1, \dots, p, j = i + 1, \dots, p$ do   ▷ # $p(p-1)/2$
 8:     $v_X(k) = (H_{ij} + H_{ji})/\sqrt{2}$ and $k = k + 1$;
 9: end for
10: for $i = 1, \dots, n - p, j = 1, \dots, p$ do   ▷ # $(n-p)p$
11:     $v_X(k) = \sqrt{2}W_{ij}$ and $k = k + 1$;
12: end for
13: return vector $v_X \in \mathbb{R}^{np-p(p-1)/2}$;
```

Algorithm 9 Compute $\mathrm{D2E}_X^{\mathrm{S}_+(p,n)}(v_X)$

```
Require: $X = YY^T \in \mathcal{M}_r$, $v_X \in \mathbb{R}^{np-p(p-1)/2}$, a function $\beta_Y : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : A \mapsto \begin{bmatrix} YL^{-T} & Y_\perp \end{bmatrix} A$;
 1: Set $k = 1$;
 2: for $i = 1, \dots, p$ do   ▷ # $p$
 3:     $H_{ii} = v_X(k)$ and $k = k + 1$;
 4: end for
 5: for $i = 1, \dots, p, j = i + 1, \dots, p$ do   ▷ # $p(p-1)/2$
 6:     $H_{ij} = v_X(k)/\sqrt{2}$, $H_{ji} = H_{ij}$ and $k = k + 1$;
 7: end for
 8: for $i = 1, \dots, n - p, j = 1, \dots, p$ do   ▷ # $(n-p)p$
 9:     $W_{ij} = v_X(k)/\sqrt{2}$ and $k = k + 1$;
10: end for
11: Return $\dot{Y} = \beta_Y \begin{bmatrix} H/2 \\ W \end{bmatrix} L^{-1}$;   ▷ # $np^2$ plus flops in Algorithm 5 and note $L$ has been computed in Algorithm 8
```

6 Limited-memory Riemannian BFGS method for problems on the Stiefel manifold

In this section, we use a limited-memory Riemannian BFGS method (LRBFGS) as the representative algorithm to show the benefits of using the intrinsic representation and the vector transport by parallelization for problems on the Stiefel manifold. This idea can be easily generalized to other Riemannian optimization algorithms and problems on other manifolds. We exploit the version of LRBFGS developed in [HGA15, Algorithm 2] and modified the version to use an alternate update defined in [HAG16] which allows the line search using the Wolfe conditions to be replaced by the Armijo line search.

Algorithms 10 and 11 state the LRBFGS for optimizing a function

$$f : \mathrm{St}(p, n) \to \mathbb{R} : X \to f(X),$$

using extrinsic representation and intrinsic representation respectively. We use the qf retraction (4.5) and Riemannian metric g_e in Algorithms 10 and 11. The number of flops for each step—except problem-related operations, i.e., function, gradient evaluations and line search—is given on the right-hand side of the algorithms. By summing them up and noting that $d = np - p(p+1)/2$ and $w = np$, the known number of flops per iteration for Algorithms 10 and 11 are $8lnp + 10np^2 - 8p^3/3 + 2l\lambda + o(np^2) + o(p^3)$ and $8l(np - p(p+1)/2) + 18np^2 - 26p^3/3 + o(np^2) + o(p^3)$ respectively, where λ denotes the flops in a vector transport evaluation.

As far as we are aware, the cheapest vector transport on the Stiefel manifold using extrinsic representation is the vector transport by projection, whose cost is $\lambda = 4np^2$. In this case, even when the limited-memory size l is chosen to be the smallest positive integer 1, which punishes Algorithm 10 least, Algorithm 11 still shows the benefit that the number of flops required is smaller. The advantage of Algorithm 11 is more significant when another vector transport or a larger l is used. It can be seen that in this case, the computational efficiency mainly comes from the identity implementation of the vector transport by parallelization.

Moreover, it is pointed out that the existing convergence analyses for Riemannian quasi-Newton methods given in e.g., [RW12, HAG15, HGA15, HAG16] all rely on an isometric vector transport. Unlike the proposed vector transport by parallelization, the vector transport by projection is not isometric, which implies that the existing convergence analyses no longer apply.

7 Experiments

To compare the performance of Algorithm 10 and 11, we consider the Brockett cost function

$$f : \mathrm{St}(p, n) \to \mathbb{R} : X \mapsto \mathrm{trace}(X^T AXN),$$

where $N = \mathrm{diag}(\mu_1, \mu_2, \ldots, \mu_p)$ with $\mu_1 > \cdots > \mu_p > 0$, $A \in \mathbb{R}^{n\times n}$ and $A = A^T$. It is shown in [AMS08, §4.8] that the columns of any global minimizer, $X^* e_i$, are eigenvectors for the p smallest eigenvalues, λ_i, ordered so that $\lambda_1 \leq \cdots \leq \lambda_p$.

If the matrix A is dense and $n \gg p$, then the number of flops in a function evaluation is $2n^2p$, which dominates the complexities of Algorithm 10 and Algorithm 11. Therefore, it would not be easy to observe the influence of intrinsic representation and vector transport by parallelization. Thus, we choose a sparse matrix A and $p \ll n$, i.e., $n = 1000$ and $p = 8$ in the experiments. Note that the situation $p \ll n$ is commonly encountered in many applications such as the matrix completion problem [Van13], the max-cut SDP relaxation [GW95], and the phase retrieval problem [HGZ16].

[2] Note that $Alg1(X, Z) = Alg1(X, P_X(Z))$, where $X \in \mathrm{St}(p, n)$ and $Alg1 : (\mathrm{St}(p, n), \mathbb{R}^{n\times p}) \to \mathbb{R}^{np-p(p+1)/2}$ is the function defined by Algorithm 1

Algorithm 10 LRBFGS for problems on the Stiefel manifold with extrinsic representation and general vector transport

```
Require: Initial iterate $x_0 \in \mathcal{M}$; an integer $m > 0$; line search constant $\varrho \in (0,1)$.
1: $k = 0$, $\gamma_0 = 1$, $l = 0$.
2: Compute $\mathrm{grad}\,\tilde{f}(x_k)$, where $\tilde{f} : \mathbb{R}^{n\times p} \to \mathbb{R} : X \to f(X)$;
3: Compute the Riemannian gradient $\mathrm{grad}\, f(x_k) = P_{x_k}\, \mathrm{grad}\,\tilde{f}(x_k)$;   ▷ # $4np^2$
4: $\mathcal{H}_k^0 = \gamma_k\, \mathrm{id}$. Obtain $\eta_k \in \mathrm{T}_{x_k}\mathcal{M}$ by the following algorithm, Step 5 to Step 15:
5: $q \leftarrow \mathrm{grad}\, f(x_k)$
6: for $i = k-1, k-2, \dots, k-l$ do   ▷ # $4lw$
7:     $\xi \leftarrow \rho_i g(s_i^{(k)}, q)$;
8:     $q \leftarrow q - \xi_i y_i^{(k)}$;
9: end for
10: $r \leftarrow \mathcal{H}_k^0 q$;   ▷ # $w$
11: for $i = k-l, k-l+1, \dots, k-1$ do   ▷ # $4lw$
12:     $\omega \leftarrow \rho_i g(y_i^{(k)}, r)$;
13:     $r \leftarrow r + s_i^{(k)}(\xi_i - \omega)$;
14: end for
15: set $\eta_k = -r$;   ▷ # $w$
16: find the largest $\alpha_k \in \{1, \varrho, \varrho^2, \dots\}$ satisfying
        $$f(x_{k+1}) \leq f(x_k) + \delta\alpha_k g(\mathrm{grad}\, f(x_k), \eta_k),$$
17: Apply Algorithm 3 with $Z = x_k + \alpha_k\eta_k$ and obtain unit vectors $V = \{v_1, v_2, \dots, v_p\}$
    and sign scalars $S = \{s_1, s_2, \dots, s_p\}$.   ▷ # $2np^2 - 2p^3/3$
18: Set $x_{k+1} = \beta_{(V,S)}(I_p)$ by Algorithm 5;   ▷ # $4np^2 - 2p^3$
19: Define $\mathfrak{s}_k^{(k+1)} = \mathcal{T}_{\alpha_k\eta_k}\alpha_k\eta_k$ and $\mathfrak{y}_k^{(k+1)} = \mathrm{grad}\, f(x_{k+1}) - \mathcal{T}_{\alpha_k\eta_k}\,\mathrm{grad}\, f(x_k)$;   ▷ #
    $2\lambda + 2w$
20: Compute $a = g(\mathfrak{y}_k^{(k+1)}, \mathfrak{s}_k^{(k+1)})$ and $b = \|\mathfrak{s}_k^{(k+1)}\|^2$;   ▷ # $4w$
21: if $\frac{a}{b} \geq 10^{-4}\|\mathrm{grad}\, f(x_k)\|$ then   ▷ # $2w$
22:     Compute $c = \|\mathfrak{y}_k^{(k+1)}\|^2$ and define $\rho_k = 1/a$ and $\gamma_{k+1} = a/c$;   ▷ # $2w$
23:     Add $\mathfrak{s}_k^{(k+1)}$, $\mathfrak{y}_k^{(k+1)}$ and $\rho_k$ into storage and if $l \geq m$, then discard vec-
        tor pair $\{\mathfrak{s}_{k-l}^{(k)}, \mathfrak{y}_{k-l}^{(k)}\}$ and scalar $\rho_{k-l}$ from storage, else $l \leftarrow l+1$; Transport
        $\mathfrak{s}_{k-l+1}^{(k)}, \mathfrak{s}_{k-l+2}^{(k)}, \dots, \mathfrak{s}_{k-1}^{(k)}$ and $\mathfrak{y}_{k-l+1}^{(k)}, \mathfrak{y}_{k-l+2}^{(k)}, \dots, \mathfrak{y}_{k-1}^{(k)}$ from $\mathrm{T}_{x_k}\mathcal{M}$ to $\mathrm{T}_{x_{k+1}}\mathcal{M}$ by $\mathcal{T}$,
        then get $\mathfrak{s}_{k-l+1}^{(k+1)}, \mathfrak{s}_{k-l+2}^{(k+1)}, \dots, \mathfrak{s}_{k-1}^{(k+1)}$ and $\mathfrak{y}_{k-l+1}^{(k+1)}, \mathfrak{y}_{k-l+2}^{(k+1)}, \dots, \mathfrak{y}_{k-1}^{(k+1)}$;   ▷ # $2(l-1)\lambda$
24: else
25:     Set $\gamma_{k+1} \leftarrow \gamma_k$, $\{\rho_k, \dots, \rho_{k-l+1}\} \leftarrow \{\rho_{k-1}, \dots, \rho_{k-l}\}$,
        $\{\mathfrak{s}_k^{(k+1)}, \dots, \mathfrak{s}_{k-l+1}^{(k+1)}\} \leftarrow \{\mathcal{T}_{\alpha_k\eta_k}\mathfrak{s}_{k-1}^{(k)}, \dots, \mathcal{T}_{\alpha_k\eta_k}\mathfrak{s}_{k-l}^{(k)}\}$ and $\{\mathfrak{y}_k^{(k+1)}, \dots, \mathfrak{y}_{k-l+1}^{(k+1)}\} \leftarrow$
        $\{\mathcal{T}_{\alpha_k\eta_k}\mathfrak{y}_{k-1}^{(k)}, \dots, \mathcal{T}_{\alpha_k\eta_k}\mathfrak{y}_{k-l}^{(k)}\}$;   ▷ # $2l\lambda$
26: end if
27: $k = k+1$, goto Step 4.
```

The vector transport in Algorithm 10 is the vector transport by projection. The initial iterate X_0 is given by applying Matlab's function "orth" to a matrix whose entries are drawn from the standard normal distribution. A is set to be $\mathrm{diag}(1, 2, \dots n) + B + B^T$, where the entries of B have probability $1/n$ to be nonzero, i.e., drawn from the standard normal distribution. N is chosen to be $\mathrm{diag}(p, p-1, \cdots, 1)$. ϱ is set to be 0.5. The stopping criterion requires the norm of the initial gradient $\|\mathrm{grad}\, f(X_0)\|$ over the norm of the gradient $\|\mathrm{grad}\, f(X_k)\|$ to be less than 10^{-6}. All the experiments are performed in

Algorithm 11 LRBFGS for problems on the Stiefel manifold using intrinsic representation and vector transport by parallelization

```
Require: Initial iterate $x_0 \in \mathcal{M}$; an integer $m > 0$; line search constant $\varrho \in (0, 1)$.
 1: $k = 0$, $\gamma_0 = 1$, $l = 0$.
 2: Compute grad $\tilde{f}(x_k)$, where $\tilde{f} : \mathbb{R}^{n\times p} \to \mathbb{R} : X \to f(X)$;
 3: Compute the intrinsic representation $\mathrm{gf}_k^d$ of grad $f(x_k)$ by Algorithm 1;²  ▷ # $4np^2 - 2p^3$
 4: $\mathcal{H}_k^0 = \gamma_k$ id. Obtain $\eta_k \in \mathrm{T}_{x_k}\mathcal{M}$ by the following algorithm, Step 5 to Step 15:
 5: $q \leftarrow \mathrm{gf}_k^d$;
 6: for $i = k-1, k-2, \dots, k-l$ do                                  ▷ # $4ld$
 7:     $\xi \leftarrow \rho_i q^T s_i$;
 8:     $q \leftarrow q - \xi_i y_i$;
 9: end for
10: $r \leftarrow \mathcal{H}_k^0 q$;                                   ▷ # $d$
11: for $i = k-l, k-l+1, \dots, k-1$ do                                 ▷ # $4ld$
12:     $\omega \leftarrow \rho_i r^T y_i$;
13:     $r \leftarrow r + s_i(\xi_i - \omega)$;
14: end for
15: set $\eta_k = -r$;                                                  ▷ # $d$
16: find the largest $\alpha_k \in \{1, \varrho, \varrho^2, \dots\}$ satisfying
        $$f(x_{k+1}) \leq f(x_k) + \delta\alpha_k \eta_k^T \mathrm{gf}_k^d,$$
17: Compute $\eta_k^w = \mathrm{D2E}_{x_k}^{\mathrm{St}(p,n)}(\eta_k)$ by Algorithm 2;   ▷ # $4np^2 - 2p^3$
18: Apply Algorithm 3 with $Z = x_k + \alpha_k \eta_k^w$ and obtain unit vectors $V = \{v_1, v_2, \dots, v_p\}$
    and sign scalars $S = \{s_1, s_2, \dots, s_p\}$.                    ▷ # $2np^2 - 2p^3/3$
19: Set $x_{k+1} = \beta_{(V,S)}(I_p)$ by Algorithm 5;                   ▷ # $4np^2 - 2p^3$
20: Compute grad $\tilde{f}(x_{k+1})$;
21: Compute the intrinsic representation $\mathrm{gf}_{k+1}^d$ of grad $f(x_{k+1})$;   ▷ # $4np^2 - 2p^3$
22: Define $\mathfrak{s}_k = \alpha_k \eta_k$ and $\mathfrak{y} = \mathrm{gf}_{k+1}^d - \mathrm{gf}_k^d$;   ▷ # $2d$
23: Compute $a = \mathfrak{y}_k^T \mathfrak{s}_k$ and $b = \|\mathfrak{s}_k\|_2^2$;   ▷ # $4d$
24: if $\frac{a}{b} \geq 10^{-4}\|\mathrm{gf}_k^d\|_2$ then               ▷ # $2d$
25:     Compute $c = \|\mathfrak{y}_k^{(k+1)}\|_2^2$ and define $\rho_k = 1/a$ and $\gamma_{k+1} = a/c$;   ▷ # $2d$
26:     Add $\mathfrak{s}_k$, $\mathfrak{y}_k$ and $\rho_k$ into storage and if $l \geq m$, then discard vector pair $\{\mathfrak{s}_{k-l}, \mathfrak{y}_{k-l}\}$
        and scalar $\rho_{k-l}$ from storage, else $l \leftarrow l + 1$;
27: else
28:     Set $\gamma_{k+1} \leftarrow \gamma_k$, $\{\rho_k, \dots, \rho_{k-l+1}\} \leftarrow \{\rho_{k-1}, \dots, \rho_{k-l}\}$, $\{\mathfrak{s}_k, \dots, \mathfrak{s}_{k-l+1}\} \leftarrow$
        $\{\mathfrak{s}_{k-1}, \dots, \mathfrak{s}_{k-l}\}$ and $\{\mathfrak{y}_k, \dots, \mathfrak{y}_{k-l+1}\} \leftarrow \{\mathfrak{y}_{k-1}, \dots, \mathfrak{y}_{k-l}\}$
29: end if
30: $k = k + 1$, goto Step 4.
```

C++ with compiler g++-4.7 on a 64 Ubuntu platform with 3.6GHz CPU (Intel(R) Core(TM) i7-4790). The code is available at `http://www.math.fsu.edu/~whuang2/papers/IRTVVTMM.htm`.

An average of 100 random runs of Algorithms 10 and 11 with various choices of m is reported in Table 1. Note that m is the upper bound of the limited-memory size l. The average computational time per iteration of Algorithm 11 is smaller than that of Algorithm 10. Moreover, the larger m is, the faster Algorithm 11 is in the sense of the computational time per iteration. In addition, the number of iteration of Algorithm 11 is also slightly smaller than Algorithm 10. Therefore, Algorithm 11 still needs less computational time in total.

Table 1 An average of 100 random runs of Algorithms 10 and 11 with various choices of m. Note that m is the upper bound of the limited-memory size l. The subscript k indicates a scale of 10^k. $iter$, nf, ng, nR, and nV denote the number of iterations, function evaluations, gradient evaluations, retractions and vector transport evaluations, respectively. $gf/gf0$ denotes the norm of initial gradient over the norm of the final gradient. t denote the computational time in seconds. $t/iter$ denotes the average computational time per iteration.

m	2		8		32	
	Algo. 10	Algo. 11	Algo. 10	Algo. 11	Algo. 10	Algo. 11
iter	1027	915	933	830	877	745
nf	1052	937	941	837	883	751
ng	1028	916	934	831	878	746
nR	1051	936	940	836	882	750
nV	1027	915	933	830	877	745
gf/gf0	9.00_{-7}	9.11_{-7}	9.24_{-7}	9.25_{-7}	9.52_{-7}	9.49_{-7}
t	2.94_{-1}	2.50_{-1}	4.84_{-1}	2.74_{-1}	1.27	4.31_{-1}
t/iter	2.86_{-4}	2.73_{-4}	5.18_{-4}	3.31_{-4}	1.45_{-3}	5.79_{-4}

Table 2 The complexities of E2D and D2E for $\mathrm{S}_+(p,n)$, $\mathrm{Gr}(p,n)$, $\mathcal{M}_r$ and $\mathrm{S}_+(p,n)$.

	$\mathrm{St}(p,n)$	$\mathrm{Gr}(p,n)$	$\mathcal{M}_r$	$\mathrm{S}_+(p,n)$
E2D	$4np^2-2p^3$	$4np^2-2p^3$	$6(m+n)r^2-4r^3$	$8np^2-5p^3/3$
D2E	$4np^2-2p^3$	$4np^2-2p^3$	$6(m+n)r^2+4\min(m,n)r^2-10r^3/3$	$5np^2-2p^3$

8 Conclusion

We have presented tractable implementations of computing a d-dimensional representation given a w-dimensional representation and vice versa for the Stiefel manifold $\mathrm{St}(p,n)$, the Grassmann manifold $\mathrm{Gr}(p,n)$, the fixed-rank manifold $\mathcal{M}_r$ and the manifold of positive semidefinite matrices with rank fixed $\mathrm{S}_+(p,n)$. Their complexities are summarized in Table 2. The resulting vector transport by parallelization is isometric and has the identity implementation. It is demonstrated theoretically and empirically that the intrinsic representation and vector transport by parallelization can outperform extrinsic representation and other vector transports in the sense of efficiency.

References

ABB$^+$99. E. Anderson, Z. Bai, C. Bischof, L. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide.* Society for Industrial and Applied Mathematics, Third edition, 1999.

ABG07. P.-A. Absil, C. G. Baker, and K. A. Gallivan. Trust-region methods on Riemannian manifolds. *Foundations of Computational Mathematics*, 7(3):303–330, 2007.

ADM02. R. L. Adler, J.-P. Dedieu, and J. Y. Margulies. Newton's method on Riemannian manifolds and a geometric model for the human spine. *IMA Journal of Numerical Analysis*, 22(3):359–390, 2002.

AMS08. P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds.* Princeton University Press, Princeton, NJ, 2008.

AO15. P.-A. Absil and I. V. Oseledets. Low-rank retractions: a survey and new results. *Computational Optimization and Applications*, 62(1):5–29, 2015.

ATV13. B. Afsari, R. Tron, and R. Vidal. On the convergence of gradient descent for finding the Riemannian center of mass. *SIAM Journal on Control and Optimization*, 51(3):2230–2260, 2013. arXiv:1201.0925v1.

BA11. N. Boumal and P.-A. Absil. RTRMC: A Riemannian trust-region method for low-rank matrix completion. *Advances in Neural Information Processing Systems 24 (NIPS)*, pages 406–414, 2011.

BI13. D. A. Bini and B. Iannazzo. Computing the Karcher mean of symmetric positive definite matrices. *Linear Algebra and its Applications*, 438(4):1700–1710, February 2013. doi:10.1016/j.laa.2011.08.052.

Boo86. W. M. Boothby. *An introduction to differentiable manifolds and Riemannian geometry.* Academic Press, second edition, 1986.

DKM12. W. Dai, E. Kerman, and O. Milenkovic. A geometric approach to low-rank matrix completion. *IEEE Transactions on Information Theory*, 58(1):237–247, 2012. arXiv:1006.2086v1.

EAS98. A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, January 1998. doi:10.1137/S0895479895290954.

GV96. G. H. Golub and C. F. Van Loan. *Matrix computations.* Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, third edition, 1996.

GW95. M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

HAG15. W. Huang, P.-A. Absil, and K. A. Gallivan. A Riemannian symmetric rank-one trust-region method. *Mathematical Programming*, 150(2):179–216, February 2015.

HAG16. Wen Huang, P.-A. Absil, and K. A. Gallivan. A Riemannian BFGS method for nonconvex optimization problems. *Lecture Notes in Computational Science and Engineering, To appear*, 2016.

HGA15. Wen Huang, K. A. Gallivan, and P.-A. Absil. A Broyden Class of Quasi-Newton Methods for Riemannian Optimization. *SIAM Journal on Optimization*, 25(3):1660–1685, 2015.

HGSA15. Wen Huang, K. A. Gallivan, Anuj Srivastava, and P.-A. Absil. Riemannian optimization for registration of curves in elastic shape analysis. *Journal of Mathematical Imaging and Vision*, 2015. DOI:10.1007/s10851-015-0606-8.

HGZ16. Wen Huang, K. A. Gallivan, and Xiangxiong Zhang. Solving phaselift by low rank riemannian optimization methods. In *Proceedings of of the International Conference on Computational Science (ICCS2016), accepted*, 2016.

KS12. M. Kleinsteuber and H. Shen. Blind source separation with compressively sensed linear mixtures. *IEEE Signal Processing Letters*, 19(2):107–110, 2012. arXiv:1110.2593v1.

Lee11. J. M. Lee. *Introduction to smooth manifolds*, volume 36 of *Graduate Texts in Mathematics.* Springer New York, New York, NY, January 2011. doi:10.1016/B978-0-12-387667-6.00013-0.

MMS11. B. Mishra, G. Meyer, and R. Sepulchre. Low-rank optimization for distance matrix completion. *In Proceeding of 50th IEEE Conference on Decision and Control and European Control Conference*, pages 4455–4460, December 2011. doi:10.1109/CDC.2011.6160810.

NW06. J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, second edition, 2006.

QGA10. C. Qi, K. A. Gallivan, and P.-A. Absil. Riemannian BFGS algorithm with applications. *Recent Advances in Optimization and its Applications in Engineering*, pages 183–192, 2010.

RW12. W. Ring and B. Wirth. Optimization methods on Riemannian manifolds and their application to shape space. *SIAM Journal on Optimization*, 22(2):596–627, January 2012. doi:10.1137/11082885X.

SAGQ12. S. E. Selvan, U. Amato, K. A. Gallivan, and C. Qi. Descent algorithms on oblique manifold for source-adaptive ICA contrast. *IEEE Transactions on Neural Networks and Learning Systems*, 23(12):1930–1947, 2012.

San10. O. Sander. Geodesic finite elements for Cosserat rods. *International Journal for Numerical Methods in Engineering*, 82(13):1645–1670, 2010. doi:10.1002/nme.2814.

Sat15. H. Sato. A Dai-Yuan-type Riemannian conjugate gradient method with the weak Wolfe conditions. *Computational Optimization and Applications*, 2015. to appear.

SI15. H. Sato and T. Iwai. A new, globally convergent Riemannian conjugate gradient method. *Optimization*, 64(4):1011–1031, February 2015.

TVSC11. P. Turaga, A. Veeraraghavan, A. Srivastava, and R. Chellappa. Statistical computations on Grassmann and Stiefel manifolds for image and video-based recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2273–86, November 2011. doi:10.1109/TPAMI.2011.52.

Van13. B. Vandereycken. Low-rank matrix completion by Riemannian optimization—extended version. *SIAM Journal on Optimization*, 23(2):1214–1236, 2013.

VAV09. B. Vandereycken, P.-A. Absil, and S. Vandewalle. Embedded geometry of the set of symmetric positive semidefinite matrices of fixed rank. *Proceedings of the IEEE 15th Workshop on Statistical Signal Processing*, pages 389–392, 2009.