arXiv:2009.01451v1 [math.OC] 3 Sep 2020

Sufficient Descent Riemannian Conjugate Gradient Method *

Hiroyuki Sakai Hideaki Iiduka

September 4, 2020

Abstract

This paper considers sufficient descent Riemannian conjugate gradient methods with line search algorithms. We propose two kinds of sufficient descent nonlinear conjugate gradient methods and prove these methods satisfy the sufficient descent condition even on Riemannian manifolds. One is the hybrid method combining the Fletcher-Reeves-type method with the Polak-Ribiere-Polyak-type method, and the other is the Hager-Zhang-type method, both of which are generalizations of those used in Euclidean space. Also, we generalize two kinds of line search algorithms that are widely used in Euclidean space. In addition, we numerically compare our generalized methods by solving several Riemannian optimization problems. The results show that the performance of the proposed hybrid method greatly depends regardless of the type of line search used. Meanwhile, the Hager-Zhang-type method has the fast convergence property regardless of the type of line search used.

1 Introduction

In Euclidean space, nonlinear conjugate gradient methods aim to solve unconstrained optimization problems. Conjugate gradient methods have been developed by Hestenes and Stiefel [12] for solving linear systems whose coefficient matrix is symmetric positive-definite. Fletcher and Reeves [8] extended the conjugate gradient method to unconstrained nonlinear optimization problems. Theirs is the first nonlinear conjugate gradient method in Euclidean space. Since then, various nonlinear conjugate gradient methods have been proposed (see [3, 5, 17, 18]); these have summarized by Hager and Zhang in [11]. A sufficient descent condition is used to analyze the global convergence of conjugate gradient methods with inexact line searches. Hager and Zhang [10] proposed a conjugate gradient method whose search direction satisfies the sufficient descent condition regardless of whether a line search is used or not. In addition, Dai [4] proposed a nonlinear conjugate gradient methods that are the generalizations

*This work was supported by JSPS KAKENHI Grant Number JP18K11184.

of the Hager-Zhang method. His method also satisfies the sufficient descent condition regardless of whether a line search is used or not. A nonlinear conjugate gradient method that satisfies the sufficient descent condition is called a sufficient descent nonlinear conjugate gradient method. Narushima and Yabe summarized the sufficient descent nonlinear conjugate gradient methods in [15].

The conjugate gradient method in Euclidean space is applicable to a Riemannian manifold. In [23], Smith introduced the notion of Riemannian optimization. He used the exponential map and parallel transport to generalize the optimization method on a Riemannian manifold. However, in general, using the exponential map or parallel transport on a Riemannian manifold is not computationally efficient. Absil, Mahony, and Sepulchre [2] proposed to use a mapping called a retraction that approximates the exponential map. Moreover, they introduced the notion of vector transport, which approximates parallel transport. Retractions and vector transports on Stiefel manifolds have been summarized and numerically compared by Zhu [28].

Ring and Wirth [19] proposed the Fletcher-Reeves type of nonlinear conjugate gradient method on Riemannian manifolds with retraction and vector transport. They indicated that the Fletcher-Reeves methods converges globally when each step size satisfies the strong Wolfe conditions [26, 27]. However, their convergence analysis assumed that vector transport satisfies the Ring-Wirth nonexpansive condition (see (1) for the definition of the Ring-Wirth nonexpansive condition). Vector tansports that do not satisfy this condition have also been used (see [22, Section 5]). In [22], Sato and Iwai introduced the notion of scaled vector transport [22, Definition 2.2] to remove this impractical assumption from the convergence analysis. They proved that by using scaled vector transport, the Fletcher-Reeves method on a Riemannian manifold generates a descent direction at every iteration and converges globally without the Ring-Wirth nonexpansive condition. Similarly, Sato [21] used scaled vector transport in a convergence analysis. He indicated that the Dai-Yuan-type Riemannian conjugate gradient method generates a descent direction at every iteration and converges globally under the Wolfe conditions. In [20], Sakai and Iiduka proposed the hybrid Riemannian conjugate gradient method, which combines the Hestenes-Stiefel and Dai-Yuan methods. They proved that by using scaled vector transport, this hybrid method generates a descent direction at every iteration and converges globally under the strong Wolfe conditions.

In this paper, we focus on the sufficient descent condition and sufficient descent conjugate gradient method on Riemannian manifolds. We propose two kinds of sufficient descent nonlinear conjugate methods to Riemannian manifold. One is a hybrid formula combining the Fletcher-Reeves method with the Polak-Ribière-Polyak method, and we prove that using scaled vector transport, this hybrid method has the global convergence property under the strong Wolfe conditions. The other is a formula that satisfies the sufficient descent condition regardless of whether a line search is used or not, and we prove that this method has this property even on Riemannian manifolds. This formula is a generalization of the Hager-Zhang method defined on Euclidean space. Moreover, we generalize two typical line search algorithms in Euclidean space, i.e., the back-

tracking line search and line search algorithm with a *zoom* phase. In numerical experiments, we compare the sufficient descent Riemannian conjugate gradient methods with the above two line search algorithms. The numerical results show that the proposed hybrid method should use the step sizes satisfying the strong Wolfe conditions, which guarantee its convergence (Theorem 3.3). Moreover, the results show that the benefit of the Hager-Zhang-type method is its fast convergence property regardless of the type of line search used, as promised by its sufficient descent property (Theorem 3.4). The intellectual contribution of this paper is to show the fast convergence property of the sufficient descent Riemannian conjugate gradient method regardless of the type of line search used.

This paper is organized as follows. Section 2 reviews the Riemannian conjugate gradient methods and some useful concepts. Moreover, two Riemannian conjugate gradient methods are proposed in this section. Section 3 proves that several Riemannian conjugate gradient methods satisfy the sufficient descent condition. Section 4 generalizes two kinds of typical line search algorithms on Riemannian manifolds. Section 5 provides the numerical experiments on several Riemannian optimization problems. Section 6 concludes the paper.

2 Riemannian Conjugate Gradient Methods

Let M be a Riemannian manifold and T_xM be a tangent space at a point $x \in M$. $\langle \cdot, \cdot \rangle_x : T_xM \times T_xM \to \mathbb{R}$ denotes a Riemannian metric at a point $x \in M$. The Riemannian gradient of a smooth function $f : M \to \mathbb{R}$ at $x \in M$ is denoted by $\operatorname{grad} f(x)$. Let TM be the tangent bundle of M and $\oplus$ be the Whitney sum. For a smooth mapping $F : M \to N$ between two manifolds M and N, $\mathrm{D}F(x) : T_xM \to T_{F(x)}N$ denotes the differential of F at $x \in M$ (see [2, Section 3]). An unconstrained optimization problem on a Riemannian manifold M is expressed as follows:

Problem 2.1. *Let $f : M \to \mathbb{R}$ be smooth. Then, we would like to*

$$\begin{aligned} &\textit{minimize} && f(x), \\ &\textit{subject to} && x \in M. \end{aligned}$$

In order to generalize line search optimization algorithms to Riemannian manifolds, we will use the notions of a retraction [2, Chapter 4, Definition 4.1.1] and vector transport [2, Chapter 8, Definition 8.1.1], which are defined as follows:

Definition 2.1 (Retraction)**.** *Any smooth map $R : TM \to M$ is called a retraction on M if it has the following properties.*

- *$R_x(0_x) = x$, where 0_x denotes the zero element of T_xM;*
- *With the canonical identification $T_{0_x}T_xM \simeq T_xM$, R_x satisfies*

$$\mathrm{D}R_x(0_x)[\xi] = \xi$$

for all $\xi \in T_xM$,

where R_x *denotes the restriction of* R *to* T_xM.

Definition 2.2 (Vector transport)**.** *Any smooth map* $\mathcal{T} : TM \oplus TM \to TM : (\eta, \xi) \mapsto \mathcal{T}_\eta(\xi)$ *is called a vector transport on* M *if it has the following properties.*

- *There exists a retraction* R*, called the retraction associated with* $\mathcal{T}$*, such that* $\mathcal{T}_\eta(\xi) \in T_{R_x(\eta)}M$ *for all* $x \in M$*, and for all* $\eta, \xi \in T_xM$*;*
- $\mathcal{T}_{0_x}(\xi) = \xi$ *for all* $\xi \in T_xM$*;*
- $\mathcal{T}_\eta(a\xi + b\zeta) = a\mathcal{T}_\eta(\xi) + b\mathcal{T}_\eta(\zeta)$ *for all* $a, b \in \mathbb{R}$*, and for all* $\eta, \xi, \zeta \in T_xM$.

Retraction and vector transport are generalizations of the exponential map and parallel transport, respectively. We will use the Ring-Wirth nonexpansive condition [19], that is a vector transport $\mathcal{T}$ satisfying

$$\left\|\mathcal{T}_\eta(\xi)\right\|_{R_x(\eta)} \leq \|\xi\|_x , \tag{1}$$

to establish global convergence for the Fletcher-Reeves type Riemannian conjugate gradient method. In this paper, we will focus on the differentiated retraction $\mathcal{T}^R$ of R as a vector transport, defined by

$$\mathcal{T}^R_\eta(\xi) := \mathrm{D}R_x(\eta)[\xi],$$

where $x \in M$ and $\eta, \xi \in T_xM$. Then, the retraction R is associated with $\mathcal{T}^R$. However, the differentiated retraction $\mathcal{T}^R$ does not always satisfy the Ring-Wirth nonexpansive condition (1). To overcome this difficulty, Sato and Iwai [22] introduced the notion of scaled vector transport. Scaled vector transport $\mathcal{T}^S$ respect to a retraction R is defined as

$$\mathcal{T}^S_\eta(\xi) := \begin{cases} \mathcal{T}^R_\eta(\xi), & \text{if } \left\|\mathcal{T}^R_\eta(\xi)\right\|_{R_x(\xi)} \leq \|\eta\|_x , \\ \dfrac{\|\eta\|_x}{\left\|\mathcal{T}^R_\eta(\xi)\right\|_{R_x(\xi)}}\mathcal{T}^R_\eta(\xi), & \text{otherwise.} \end{cases} \tag{2}$$

The general framework of Riemannian conjugate gradient methods is described in Algorithm 1.

In this paper, we say that the search direction $\eta_k \in T_{x_k}M$ is a descent direction if $\langle g_k, \eta_k \rangle < 0$ holds. In addition, η_k is a sufficient descent direction (see [15]) if the sufficient descent condition,

$$\langle g_k, \eta_k \rangle \leq -\kappa \left\|g_k\right\|^2 , \tag{5}$$

holds for some constant $\kappa > 0$. In (3), for a given descent direction $\eta_k \in T_xM$ at $x \in M$, one often chooses a step size $\alpha_k > 0$ to satisfy the Armijo condition [13, Definition 2.3], [19, (1a)], namely,

$$f(R_{x_k}(\alpha_k\eta_k)) \leq f(x_k) + c_1\alpha_k \left\langle \operatorname{grad} f(x_k), \eta_k \right\rangle_{x_k} , \tag{6}$$

Algorithm 1 General framework of Riemannian conjugate gradient method with scaled vector transport for solving Problem 2.1 [2, 19, 21, 22].

Input: A Riemann manifold M, a retraction R, a smooth function $f : M \to \mathbb{R}$, an initial point $x_0 \in M$, convergence tolerance $\epsilon > 0$.

Output: Sequence $\{x_k\}_{k=0,1,\cdots} \subset M$.

1: Set $\eta_0 = -g_0 := -\operatorname{grad} f(x_0)$

2: $k \leftarrow 0$.

3: **while** $\|g_k\|_{x_k} > \epsilon$ **do**

4: Determine the positive step size $\alpha_k > 0$ and set

$$x_{k+1} = R_{x_k}(\alpha_k \eta_k). \tag{3}$$

5: Compute $g_{k+1} = -\operatorname{grad} f(x_{k+1})$.

6: Compute the parameter β_{k+1} (e.g., Hestenes-Stiefel formula (9)).

7: Set the search direction

$$\eta_{k+1} = -g_{k+1} + \beta_{k+1} \mathcal{T}^S_{\alpha_k \eta_k}(\eta_k), \tag{4}$$

where $\mathcal{T}^S$ is the scaled vector transport (2) respect to R.

8: $k \leftarrow k + 1$.

9: **end while**

where $0 < c_1 < 1$. The following condition is called the curvature condition [13, Definition 2.5]:

$$\left\langle \operatorname{grad} f(R_{x_k}(\alpha_k \eta_k)), \mathcal{T}^R_{\alpha_k \eta_k}(\eta_k) \right\rangle_{R_{x_k}(\alpha_k \eta_k)} \geq c_2 \left\langle \operatorname{grad} f(x_k), \eta_k \right\rangle_{x_k}, \tag{7}$$

where $0 < c_1 < c_2 < 1$. Conditions (6) and (7) are called the Wolfe conditions [13, Definition 2.7], [19, (1a), (1b)]. If condition (7) is replaced by

$$\left| \left\langle \operatorname{grad} f(R_{x_k}(\alpha_k \eta_k)), \mathcal{T}^R_{\alpha_k \eta_k}(\eta_k) \right\rangle_{R_{x_k}(\alpha_k \eta_k)} \right| \leq c_2 \left| \left\langle \operatorname{grad} f(x_k), \eta_k \right\rangle_{x_k} \right|, \tag{8}$$

then (6) and (8) are called the *strong* Wolfe conditions [19, (1a), (2)].

In (4), β_{k+1} is given by generalizations of the formulas in Euclidean space

(see [5, 8, 12, 17, 18]), e.g.,

$$\beta_{k+1}^{\mathrm{HS}} = \frac{\langle g_{k+1}, y_{k+1}\rangle_{x_{k+1}}}{\left\langle g_{k+1}, \mathcal{T}^S_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}} - \langle g_k, \eta_k\rangle_{x_k}}, \tag{9}$$

$$\beta_{k+1}^{\mathrm{FR}} = \frac{\|g_{k+1}\|^2_{x_{k+1}}}{\|g_k\|^2_{x_k}}, \tag{10}$$

$$\beta_{k+1}^{\mathrm{PRP}} = \frac{\langle g_{k+1}, y_{k+1}\rangle_{x_{k+1}}}{\|g_k\|^2_{x_k}}, \tag{11}$$

$$\beta_{k+1}^{\mathrm{DY}} = \frac{\|g_{k+1}\|^2_{x_{k+1}}}{\left\langle g_{k+1}, \mathcal{T}^S_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}} - \langle g_k, \eta_k\rangle_{x_k}}, \tag{12}$$

where $y_{k+1} := g_{k+1} - \mathcal{T}^S_{\alpha_k\eta_k}(g_k)$. Formulas (9), (10), (11), and (12) are called the Hestenes-Stiefel (HS), Fletcher-Reeves (FR), Polak-Ribière-Polyak (PRP), and Dai-Yuan (DY) formulas, respectively. In [22], Sato and Iwai indicated that by using scaled vector transport, the FR method converges globally under the *strong* Wolfe conditions (6) and (8). In [21], Sato proved that the DY method converges globally under the Wolfe conditions (6) and (7). The HS and PRP methods have good numerical performance; however, no useful convergence analyses have been presented for them. To make up for these shortcomings, hybrid-type formulas, such as

$$\beta_{k+1}^{\mathrm{Hyb1}} = \max\{0, \min\{\beta_{k+1}^{\mathrm{HS}}, \beta_{k+1}^{\mathrm{DY}}\}\}, \tag{13}$$

$$\beta_{k+1}^{\mathrm{Hyb2}} = \max\{0, \min\{\beta_{k+1}^{\mathrm{FR}}, \beta_{k+1}^{\mathrm{PRP}}\}\}, \tag{14}$$

have been developed in Euclidean space (see [6, 14]). Below, we call the hybrid methods using (13) and (14), Hybrid1 and Hybrid2, respectively. The Hybrid1 method was proposed by Dai and Yuan [6], and the Hybrid2 method was suggested by Hu and Storey [14]. In [20], Sakai and Iiduka generalized the Hybrid1 method on Riemannian manifolds and proved that it converges globally under the strong Wolfe conditions (6) and (8). They also showed that the numerical performance of the Hybrid1 method is better than that of the PRP method [20, Section 4]. In the next section (Theorem 3.1), we generalize the Hybrid2 method on a Riemannian manifold and prove that it satisfies the sufficient descent condition under the strong Wolfe conditions. Moreover, we give its convergence analysis (Theorem 3.3).

We consider the nonlinear conjugate gradient methods that can guarantee the sufficient descent condition (5) regardless of the type of line search used. We generalize the Hager-Zhang method [10] to Riemannian manifolds, as

$$\beta_{k+1}^{\mathrm{HZ}} = \beta_{k+1}^{\mathrm{HS}} - \mu\frac{\|y_{k+1}\|^2_{x_{k+1}}\left\langle g_{k+1}, \mathcal{T}^S_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}}}{\left(\left\langle g_{k+1}, \mathcal{T}^S_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}} - \langle g_k, \eta_k\rangle_{x_k}\right)^2}, \tag{15}$$

where $y_{k+1} := g_{k+1} - \mathcal{T}^S_{\alpha_k\eta_k}(g_k)$ and $\mu > 1/4$. Moreover, we modify β_{k+1} of the form $\beta_{k+1} = \langle g_{k+1}, \xi_{k+1}\rangle_{x_{k+1}}$ by

$$\beta^{\mathrm{SD}}_{k+1} = \beta_{k+1} - \mu \|\xi_{k+1}\|^2_{x_{k+1}} \left\langle g_{k+1}, \mathcal{T}^S_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}}, \tag{16}$$

where $\xi_{k+1} \in T_{x_{k+1}}M$ is any tangent vector and $\mu > 1/4$ (see [4, 15]). For instance, if we set $\xi_{k+1} = y_{k+1}/\left(\left\langle g_{k+1}, \mathcal{T}^S_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}} - \langle g_k, \eta_k\rangle_{x_k}\right)$, we have $\beta^{\mathrm{SD}}_{k+1} = \beta^{\mathrm{HZ}}_{k+1}$. We will show that the SD method always satisfies the sufficient descent condition (5) with $\kappa = 1 - (1/4\mu)$ (Theorem 3.4).

3 Sufficient Descent Properties of the Riemannian Conjugate Gradient Methods

In this section, we recall the properties of the FR (10), DY (12) and Hybrid1 (13) methods (see [20–22]).

Proposition 3.1. *The following statements hold:*

(P1) *If $\beta_{k+1} = \beta^{\mathrm{FR}}_{k+1}$ and α_k satisfies the strong Wolfe conditions (6) and (8) with $0 < c_1 < c_2 < 1/2$, then*

$$-\frac{1}{1-c_2}\|g_k\|^2_{x_k} \le \langle g_k, \eta_k\rangle_{x_k} \le -\frac{1-2c_2}{1-c_2}\|g_k\|^2_{x_k},$$

for all $k = 0, 1, \cdots$. Thus, the FR method satisfies the sufficient descent condition (5) with $\kappa = (1-2c_2)/(1-c_2) > 0$.

(P2) *If $\beta_{k+1} = \beta^{\mathrm{DY}}_{k+1}$ and α_k satisfies the Wolfe conditions (6) and (7), then*

$$-\frac{1}{1-c_2}\|g_k\|^2_{x_k} \le \langle g_k, \eta_k\rangle_{x_k} \le -\frac{1}{1+c_2}\|g_k\|^2_{x_k},$$

for all $k = 0, 1, \cdots$. Thus, the DY method satisfies the sufficient descent condition (5) with $\kappa = 1/(1+c_2) > 0$.

(P3) *If $\beta_{k+1} = \beta^{\mathrm{Hyb1}}_{k+1}$ and α_k satisfies the strong Wolfe conditions (6) and (8), then*

$$-\frac{1+c_2}{1-c_2}\|g_k\|^2_{x_k} \le \langle g_k, \eta_k\rangle_{x_k} \le -\frac{1-c_2}{1+c_2}\|g_k\|^2_{x_k},$$

for all $k = 0, 1, \cdots$. Thus, the Hybrid1 method satisfies the sufficient descent condition (5) with $\kappa = (1-c_2)/(1+c_2) > 0$.

Proposition 3.1 implies that the FR, DY and Hybrid1 methods satisfy the sufficient descent condition (5), dependent of the type of line search used. Here, (P1) is the result in [22, Lemma 4.1], and (P2) and (P3) are easily shown from [20, (35)].

3.1 A Sufficient Descent Property of the Hybrid2 method

In this section, we show that the Hybrid2 method generates a sufficient descent direction (5) at every iteration. This result is a simple extension of the result in Proposition 3.1 (P1).

Theorem 3.1. *Let $f : M \to \mathbb{R}$ be a smooth function. If each $\alpha_k > 0$ satisfies the strong Wolfe conditions* (6) *and* (8)*, with $0 < c_1 < c_2 < 1/2$, and β_{k+1} satisfies*[1] *$|\beta_{k+1}| \leq \beta^{\mathrm{FR}}_{k+1}$, then any sequence $\{x_k\}_{k=0,1\cdots}$ generated by Algorithm 1 satisfies*

$$-\frac{1}{1-c_2}\left\|g_k\right\|^2_{x_k} \leq \left\langle g_k, \eta_k\right\rangle_{x_k} \leq -\frac{1-2c_2}{1-c_2}\left\|g_k\right\|^2_{x_k}, \tag{17}$$

for all $k = 0, 1, \cdots$.

Proof. The proof is by induction. If $k = 0$, (17) clearly holds. Assume that (17) holds for some $k \geq 0$. By $c_2 < 1/2$, we obtain $\left\langle g_k, \eta_k\right\rangle_{x_k} < 0$. From the search direction (4), we have

$$\frac{\left\langle g_{k+1}, \eta_{k+1}\right\rangle_{x_{k+1}}}{\left\|g_{k+1}\right\|^2_{x_{k+1}}} = -1 + \beta_{k+1}\frac{\left\langle g_{k+1}, \mathcal{T}^S_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}}}{\left\|g_{k+1}\right\|^2_{x_{k+1}}},$$

which implies

$$\frac{\left\langle g_{k+1}, \eta_{k+1}\right\rangle_{x_{k+1}}}{\left\|g_{k+1}\right\|^2_{x_{k+1}}} = -1 + \frac{\beta_{k+1}}{\beta^{\mathrm{FR}}_{k+1}}\frac{s_k\left\langle g_{k+1}, \mathcal{T}^R_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}}}{\left\|g_k\right\|^2_{x_k}}, \tag{18}$$

where

$$s_k := \min\left\{1, \frac{\left\|\eta_k\right\|_{x_k}}{\left\|\mathcal{T}^R_{\alpha_k\eta_k}(\eta_k)\right\|_{x_{k+1}}}\right\} \in [0, 1].$$

From the second condition of the strong Wolfe conditions (8) and $\left\langle g_k, \eta_k\right\rangle_{x_k} < 0$, we obtain

$$\left|\beta_{k+1}\left\langle g_{k+1}, \mathcal{T}^R_{\alpha_k\eta_k}(\eta_k)\right\rangle_{x_{k+1}}\right| \leq -c_2\left|\beta_{k+1}\right|\left\langle g_k, \eta_k\right\rangle_{x_k},$$

which together with (18) implies

$$-1 + c_2 s_k\frac{\left|\beta_{k+1}\right|}{\beta^{\mathrm{FR}}_{k+1}}\frac{\left\langle g_k, \eta_k\right\rangle_{x_k}}{\left\|g_k\right\|^2_{x_k}} \leq \frac{\left\langle g_{k+1}, \eta_{k+1}\right\rangle_{x_{k+1}}}{\left\|g_{k+1}\right\|^2_{x_{k+1}}} \leq -1 - c_2 s_k\frac{\left|\beta_{k+1}\right|}{\beta^{\mathrm{FR}}_{k+1}}\frac{\left\langle g_k, \eta_k\right\rangle_{x_k}}{\left\|g_k\right\|^2_{x_k}}.$$

From the left-hand side of the induction hypothesis (17), we have

$$-1 - c_2 s_k\frac{\left|\beta_{k+1}\right|}{\beta^{\mathrm{FR}}_{k+1}}\frac{1}{1-c_2} \leq \frac{\left\langle g_{k+1}, \eta_{k+1}\right\rangle_{x_{k+1}}}{\left\|g_{k+1}\right\|^2_{x_{k+1}}} \leq -1 + c_2 s_k\frac{\left|\beta_{k+1}\right|}{\beta^{\mathrm{FR}}_{k+1}}\frac{1}{1-c_2}.$$

[1] The formulas defined by (10) and (14) satisfy $|\beta_{k+1}| \leq \beta^{\mathrm{FR}}_{k+1}$.

Utilizing the assumption $|\beta_{k+1}| \leq \beta_{k+1}^{\mathrm{FR}}$ and $0 \leq s_k \leq 1$, we obtain

$$-1 - \frac{c_2}{1-c_2} \leq \frac{\langle g_{k+1}, \eta_{k+1} \rangle_{x_{k+1}}}{\|g_{k+1}\|_{x_{k+1}}^2} \leq -1 + \frac{c_2}{1-c_2}.$$

This implies that (17) holds for $k+1$. □ □

Moreover, we prove the global convergence of the Hybrid2 method under the strong Wolfe conditions and the following assumption.

Assumption 3.1. *Let M be a Riemannian manifold and R be a retraction on M. Let $f : M \to \mathbb{R}$ be a smooth, bounded below function. Then, we assume that there exists $L > 0$ such that*

$$|\mathrm{D}(f \circ R_x)(t\eta)[\eta] - \mathrm{D}(f \circ R_x)(0_x)[\eta]| \leq Lt,$$

where $x \in M$, $\eta \in T_x M$, $\|\eta\|_x = 1$ and $t \geq 0$.

This is the assumption for Zoutendijk's theorem (Theorem 3.2) on Riemannian manifolds. Zoutendijk's theorem is described on Riemannian manifolds as follows:

Theorem 3.2 (Zoutendijk [19])**.** *Let (M, g) be a Riemannian manifold and R be a retraction on M. Suppose $f : M \to \mathbb{R}$ satisfies Assumption 3.1. Suppose further that in Algorithm 1, each step size $\alpha_k > 0$ satisfies the strong Wolfe conditions* (6) *and* (8)*. Then the following series converges:*

$$\sum_{k=0}^{\infty} \frac{\langle g_k, \eta_k \rangle_{x_k}^2}{\|\eta_k\|_{x_k}^2} < \infty. \tag{19}$$

The proof of this theorem is along the lines of Zoutendijk's theorem in Euclidean space (see [19, Theorem 3.3]). Global convergence proofs for Riemannian conjugate gradient methods are often based on Zoutendijk's theorem. Theorem 3.3 guarantees global convergence of the Hybrid2 method (14). It is a generalization of the convergence theorem of the Hybrid2 method in Euclidean space [9].

Theorem 3.3. *Let $f : M \to \mathbb{R}$ be a function satisfying Assumption 3.1. If each $\alpha_k > 0$ satisfies the strong Wolfe conditions* (6) *and* (8)*, with $0 < c_1 < c_2 < 1/2$, and β_{k+1} satisfies $|\beta_{k+1}| \leq \beta_{k+1}^{\mathrm{FR}}$, then any sequence $\{x_k\}_{k=0,1\cdots}$ generated by Algorithm 1 satisfies*

$$\liminf_{k \to \infty} \|g_k\|_{x_k} = 0. \tag{20}$$

Proof. We prove (20) by contradiction. If $g_{k_0} = 0$ for some k_0, then (20) follows. Assume that

$$\liminf_{k \to \infty} \|g_k\|_{x_k} > 0.$$

Then, noting $\|g_k\|_{x_k} \neq 0$ for all k, there exists $\gamma > 0$ such that

$$\|g_k\|_{x_k} \geq \gamma > 0,$$

for all k. From (8) and (17), we have

$$\begin{aligned}\left|\left\langle g_k, \mathcal{T}^R_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1})\right\rangle_{x_k}\right| &\leq -c_2 \langle g_{k-1}, \eta_{k-1}\rangle_{x_{k-1}} \\ &\leq \frac{c_2}{1-c_2}\|g_{k-1}\|^2_{x_{k-1}}.\end{aligned}$$

Thus, from (4) and (17), and using the condition $|\beta_k| \leq \beta_k^{\mathrm{FR}} \leq \|g_k\|^2_{x_k} / \|g_{k-1}\|^2_{x_{k-1}}$, we have

$$\begin{aligned}\|\eta_k\|^2_{x_k} &\leq \|g_k\|^2_{x_k} + 2s_k\left|\beta_k\left\langle g_k, \mathcal{T}^R_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1})\right\rangle_{x_k}\right| + \left\|\beta_k \mathcal{T}^S_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1})\right\|^2_{x_k} \\ &\leq \|g_k\|^2_{x_k} + \frac{2c_2}{1-c_2}|\beta_k|\,\|g_{k-1}\|^2_{x_{k-1}} + \beta_k^2\|\eta_{k-1}\|^2_{x_{k-1}} \\ &\leq \hat{c}\,\|g_k\|^2_{x_k} + \beta_k^2\|\eta_{k-1}\|^2_{x_{k-1}},\end{aligned}$$

where $\hat{c} := (1+c_2)/(1-c_2) > 1$. Applying this equation repeatedly, we obtain

$$\begin{aligned}\|\eta_k\|^2_{x_k} &\leq \hat{c}\,\|g_k\|^2_{x_k} + \beta_k^2\left(\hat{c}\,\|g_{k-1}\|^2_{x_{k-1}} + \beta_{k-1}^2\|\eta_{k-2}\|^2_{x_{k-2}}\right) \\ &\leq \hat{c}\left(\|g_k\|^2_{x_k} + \beta_k^2\|g_{k-1}\|^2_{x_{k-1}} + \cdots + \beta_k^2\beta_{k-1}^2\cdots\beta_2^2\|g_1\|^2_{x_1}\right) + \beta_k^2\beta_{k-1}^2\cdots\beta_1^2\|\eta_0\|^2_{x_0} \\ &\leq \hat{c}\,\|g_k\|^4_{x_k}\left(\frac{1}{\|g_k\|^2_{x_k}} + \frac{1}{\|g_{k-1}\|^2_{x_{k-1}}} + \cdots + \frac{1}{\|g_1\|^2_{x_1}}\right) + \frac{\|g_k\|^4_{x_k}}{\|g_0\|^2_{x_0}} \\ &< \hat{c}\,\|g_k\|^4_{x_k}\sum_{j=0}^{k}\frac{1}{\|g_j\|^2_{x_j}} \\ &\leq \frac{\hat{c}}{\gamma^2}\|g_k\|^4_{x_k}(k+1).\end{aligned}$$

This implies that

$$\frac{\|g_k\|^4_{x_k}}{\|\eta_k\|^2_{x_k}} \geq \frac{\gamma^2}{\hat{c}(k+1)},$$

which together with (17), gives

$$\begin{aligned}\sum_{k=0}^{\infty}\frac{\langle g_k, \eta_k\rangle^2_{x_k}}{\|\eta_k\|^2_{x_k}} &= \sum_{k=0}^{\infty}\frac{\|g_k\|^4_{x_k}}{\|\eta_k\|^2_{x_k}}\frac{\langle g_k, \eta_k\rangle^2_{x_k}}{\|g_k\|^4_{x_k}} \\ &\geq \left(\frac{2c_2-1}{1-c_1}\right)^2\sum_{k=0}^{\infty}\frac{\gamma^2}{\hat{c}(k+1)} \\ &= \infty.\end{aligned}$$

This contradicts (19) in Zoutendijk's theorem (Theorem 3.2) and completes the proof. □ □

3.2 Sufficient Descent Property of the SD method

Theorem 3.4 asserts that the SD method (16) produces sufficient descent directions (5) regardless of the choice of the step size α_k.

Theorem 3.4. *Let $f : M \to \mathbb{R}$ be a smooth function. If $\beta_{k+1} = \beta_{k+1}^{\mathrm{SD}}$, then any sequence $\{x_k\}_{k=0,1\cdots}$ generated by Algorithm 1 satisfies*

$$\langle g_k, \eta_k \rangle_{x_k} \leq -\left(1 - \frac{1}{4\mu}\right) \|g_k\|_{x_k}^2 . \tag{21}$$

Proof. From (16), we obtain

$$\begin{aligned} \langle g_k, \eta_k \rangle_{x_k} = &- \|g_k\|_{x_k}^2 + \langle g_k, \xi_k \rangle_{x_k} \left\langle g_k, \mathcal{T}^S_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1}) \right\rangle_{x_k} \\ &- \mu \|\xi_k\|_{x_k}^2 \left\langle g_k, \mathcal{T}^S_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1}) \right\rangle_{x_k}^2 . \end{aligned} \tag{22}$$

An upper bound for the middle term in (22) is obtained using the inequality,

$$\langle u_k, v_k \rangle_{x_k} \leq \frac{\|u_k\|_{x_k}^2 + \|v_k\|_{x_k}^2}{2}$$

with the choice

$$u_k := \frac{1}{\sqrt{2\mu}} g_k \quad \textit{and} \quad v_k := \sqrt{2\mu} \left\langle g_k, \mathcal{T}^S_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1}) \right\rangle \xi_k .$$

Then, we have

$$\begin{aligned} &\langle g_k, \xi_k \rangle_{x_k} \left\langle g_k, \mathcal{T}^S_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1}) \right\rangle_{x_k} \\ &\quad \leq \frac{1}{4\mu} \|g_k\|_{x_k}^2 + \mu \|\xi_k\|_{x_k}^2 \left\langle g_k, \mathcal{T}^S_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1}) \right\rangle_{x_k}^2 . \end{aligned}$$

Combining this with (22), we obtain (21). □ □

4 Line Search Algorithm on Riemannian Manifolds

In Algorithm 1, we need to use a line search algorithm to determine the step size α_k. A backtracking line search algorithm (see [16, Chapter 3, Algorithm 3.1]) is widely used in optimization algorithms in Euclidean space to find a step size that satisfies the Armijo condition (6). Algorithm 2 is a backtracking line search on Riemannian manifold. This algorithm multiplies a positive constant $\rho > 0$ until a step size α satisfying the Armijo condition is found.

However, a backtracking line search algorithm cannot be used for the Wolfe or the strong Wolf conditions. To find a step size satisfying the *strong* Wolfe conditions, we present Algorithm 3, a generalization of the algorithm in [16, Chapter

Algorithm 2 Backtracking line search on Riemannian manifold M.

Input: A smooth function $f : M \to \mathbb{R}$, a point $x \in M$, a descent direction $\eta \in T_x M$, scalars $0 < \alpha_{\text{hi}}$, $\rho \in (0, 1)$.

Output: A positive step size $\alpha > 0$ satisfying the Armijo condition (6).

$\alpha \leftarrow \alpha_{\text{hi}}$
while $f(R_x(\alpha\eta)) > f(x) + c_1\alpha \langle \operatorname{grad} f(x), \eta \rangle_x$ **do**
  $\alpha \leftarrow \rho\alpha$
end while
return α

3, Algorithm 3.5] for strong Wolfe conditions in Euclidean space. Algorithm 3 calls the *zoom* function (Algorithm 4), which successively decreases the size of the interval until an acceptable step size is found (see [16, Chapter 3, Algorithm 3.6]). The parameter α_{hi} is a user-supplied bound on the maximum step size. Algorithm 3 returns a positive step size, $\alpha_\star > 0$, that satisfies the strong Wolfe conditions.

Algorithm 3 Line search algorithm on Riemannian manifold M.

Input: A smooth function $f : M \to \mathbb{R}$, a point $x \in M$, a descent direction $\eta \in T_x M$, scalars $0 < c_1 < c_2 < 1$, $0 < \alpha_{\text{hi}}$ and $\alpha_0 \in (0, \alpha_{\text{hi}})$.

Output: A positive step size $\alpha > 0$ satisfying the *strong* Wolfe conditions (6) and (8).

Set $\phi(\alpha) = f(R_x(\alpha\eta))$.
$i \leftarrow 0$.
loop
  if $\phi(\alpha_i) > \alpha(0) + \alpha c_1 \phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{i-1})$ **and** $i \geq 1]$ **then**
    Set $\alpha_\star = \text{Zoom}(\alpha_{i-1}, \alpha_i)$ and stop.
  else if $|\phi'(\alpha_i)| \leq -c_2\phi'(0)$ **then**
    Set $\alpha_\star = \alpha_i$ and stop.
  else if $\phi'(0) \geq 0$ **then**
    Set $\alpha_\star = \text{Zoom}(\alpha_i, \alpha_{i-1})$ and stop.
  end if
  Choose $\alpha_{i+1} \in (\alpha_i, \alpha_{\text{hi}})$.
  $i \leftarrow i + 1$.
end loop
return $\alpha_\star$

5 Numerical Experiments

Our experiments used the source code based on `pymanopt`[2] (see [24]). In addition, Algorithm 3 was based on an implementation by SciPy[3] in Euclidean

[2] `https://www.pymanopt.org/`
[3] `https://docs.scipy.org/doc/scipy/reference/`

Algorithm 4 Zoom. [16, Chapter 3, Algorithm 3.6]

Input: Scalars $\alpha_{\min}, \alpha_{\max} > 0$, and $\phi(\alpha) = f(R_x(\alpha\eta))$.
Output: $\alpha = \mathrm{Zoom}(\alpha_{\min}, \alpha_{\max})$.

loop
  Interpolate (using quadratic, cubic, or bisection) to find a trial step length $\alpha_j \in (\alpha_{\min}, \alpha_{\max})$.
  if $\phi(\alpha_j) > \phi(0) + c_1\alpha_j\phi'(0)$ **or** $\phi(\alpha_j) \geq \phi(\alpha_{\min})$ **then**
    $\alpha_{\max} \leftarrow \alpha_j$
  else
    if $|\phi'(\alpha_j)| \leq -c_2\phi'(0)$ **then**
      Set $\alpha_\star = \alpha_j$ and stop.
    else if $\phi'(\alpha_j)(\alpha_{\max} - \alpha_{\min}) \geq 0$ **then**
      $\alpha_{\max} \leftarrow \alpha_{\min}$.
    end if
    $\alpha_{\min} \leftarrow \alpha_j$.
  end if
end loop
return $\alpha_\star$

space. We solved four kinds of Riemannian optimization problems (Problem 5.1–5.4) on several Riemannian manifolds and objective functions.

Problem 5.1 is the Rayleigh-quotient minimization problem on the unit sphere (see [2, Chapter 4.6]).

Problem 5.1. *For $A \in \mathcal{S}^n_{++}$,*

$$\begin{aligned} &\textit{minimize} && f(x) = x^\top Ax, \\ &\textit{subject to} && x \in \mathbb{S}^{n-1} := \{x \in \mathbb{R}^n : \|x\| = 1\}, \end{aligned}$$

where $\|\cdot\|$ denotes the Euclidean norm and $\mathcal{S}^n_{++}$ denotes the set of all $n \times n$ symmetric positive-definite matrices.

In the experiments, we set $n = 100$ and generated a matrix $A \in \mathcal{S}^n_{++}$ with randomly chosen elements by using `sklearn.datasets.make_spd_matrix`.

Problem 5.2 is the Brockett-cost-function minimization problem on a Stiefel manifold (see [2, Chapter 4.8]).

Problem 5.2. *For $A \in \mathcal{S}^n_{++}$ and $N = \mathrm{diag}(\mu_0, \cdots, \mu_p)$ $(0 \leq \mu_0 \leq \cdots \leq \mu_p)$,*

$$\begin{aligned} &\textit{minimize} && f(X) = \mathrm{tr}(X^\top AXN) \\ &\textit{subject to} && X \in \mathrm{St}(p, n) := \{X \in \mathbb{R}^{n\times p} : X^\top X = I_p\}. \end{aligned}$$

In the experiments, we set $p = 5$, $n = 20$ and $N := \mathrm{diag}(1, 2, 3, 4, 5)$ and generated a matrix $A \in \mathcal{S}^n_{++}$ with randomly chosen elements by using `sklearn.datasets.make_spd_matrix`.

In [25], Vandereycken discussed the following robust matrix completion problem (Problem 5.3).

Problem 5.3. *For $A \in \mathbb{R}^{m\times n}$, and a subset Ω of the complete set of entries $\{1,\cdots,m\}\times\{1,\cdots,n\}$,*

$$\begin{aligned} &\textit{minimize} \quad f(X) = \|P_\Omega(X-A)\|_F^2, \\ &\textit{subject to} \quad X \in M_k := \{X \in \mathbb{R}^{m\times n} : \mathrm{rank}(X) = k\}, \end{aligned}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and

$$P_\Omega : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}, X_{ij} \mapsto \begin{cases} X_{ij} & (i,j) \in \Omega \\ 0 & (i,j) \notin \Omega \end{cases}.$$

In the experiments, we set $m = n = 100$ and $k = 4$, and Ω contained each pair $(i,j) \in \{1,\cdots,m\}\times\{1,\cdots,n\}$ with probability $1/2$. Moreover, we used a matrix $A \in \mathbb{R}^{m\times n}$ that was generated with randomly chosen elements by using `numpy.random.randn`.

In [1], Absil and Gallivan introduced the following off-diagonal cost function minimization problem on oblique manifolds (Problem 5.4).

Problem 5.4. *For $C_i \in \mathcal{S}^n$ $(i = 1,\cdots,N)$,*

$$\begin{aligned} &\textit{minimize} \quad f(X) = \sum_{i=1}^{N} \left\| X^\top C_i X - \mathrm{ddiag}(X^\top C_i X) \right\|_F^2 \\ &\textit{subject to} \quad X \in \mathcal{OB}(n,p) := \{X \in \mathbb{R}^{n\times p} : \mathrm{ddiag}(X^{\mathrm{T}}X) = I_p\}, \end{aligned}$$

where $\mathcal{S}^n$ denotes the set of all $n\times n$ symmetric matrices and $\mathrm{ddiag}(X)$ denotes a diagonal matrix whose diagonal elements are those of X.

In the experiments, we set $N = 10$, $n = 100$ and $p = 5$ and generated ten matrices $B_i \in \mathbb{R}^{n\times n}$ $(i = 1,2,\cdots,10)$ with randomly chosen elements by using `numpy.random.randn`. Then, we set symmetric matrices $C_i \in \mathcal{S}^n$ as $C_i := (B_i + B_i^\top)/2$ $(i = 1,2,\cdots,10)$.

The experiments used a MacBook Air (2017) with a 1.8 GHz Intel Core i5, 8 GB 1600 MHz DDR3 memory, and the macOS Mojave version 10.14.5 operating system. The algorithms were written in Python 3.7.6 with the NumPy 1.19.0 package and the Matplotlib 3.2.2 package. We solved the above four problems 100 times with each algorithm, that is, 400 times in total. If the stopping condition,

$$\|\mathrm{grad} f(x_k)\|_{x_k} < 10^{-6}$$

was satisfied, we determined that a sequence had converged to an optimal solution. We compared seven Riemannian conjugate gradient methods, i.e., FR, DY, PRP, HS, HZ, Hybrid1, and Hybrid2 methods, and two line search algorithms, i.e., Algorithm 2 and 3. In the HZ method, we set $\mu = 2$. In the Armijo condition (6) and the second condition of the strong Wolfe conditions (8), we set $c_1 = 10^{-4}$ and $c_2 = 0.9$. In Algorithm 2, we set the scalars as $\alpha_{\mathrm{hi}} = 1$ and

$\rho = 0.5$. In Algorithm 3, we set the scalar as $\alpha_0 = 1$, and in step 11, we set $\alpha_i = 2\alpha_{i-1}$ (see `scipy.optimize.line_search`).

For comparison, we calculated the performance profile [7]. The performance profile $P_s : \mathbb{R} \to [0, 1]$ is defined as follows: let $\mathcal{P}$ and $\mathcal{S}$ be the set of problems and solvers, respectively. For each $p \in \mathcal{P}$ and $s \in \mathcal{S}$, we define

$$t_{p,s} := (\text{iterations or time required to solve problem } p \text{ by solver } s).$$

Furthermore, we defined the performance ratio $r_{p,s}$ as

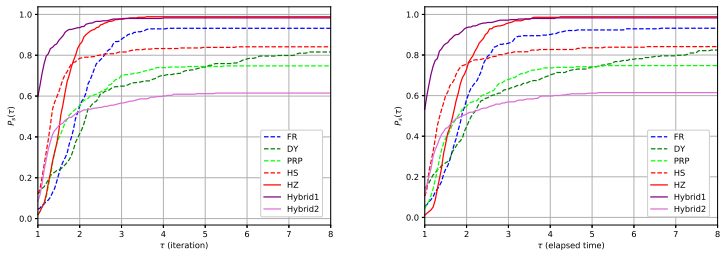
$$r_{p,s} := \frac{t_{p,s}}{\min_{s' \in \mathcal{S}} t_{p,s'}}$$

and defined the performance profile, for all $\tau \in \mathbb{R}$, as

$$P_s(\tau) := \frac{\#\{p \in \mathcal{P} : r_{p,s} \leq \tau\}}{\#\mathcal{P}},$$

where $\#S$ denotes the number of elements of a set S.

Figure 1 plots the performance profiles of each algorithm by using Algorithm 2 to determine the step size. In particular, Figure 1 (a) and (b) plot the performance profiles versus the number of iterations and the elapsed time, respectively. They show that the HZ method solved the most problems, which is about the same number as the Hybrid1 method solved. The HZ and Hybrid1 methods substantially outperformed the other methods. In particular, Hybrid1 boasted outstanding performance in terms of the number of solved problems, number of iterations, and elapsed time. It can also be seen that Hybrid2 is not compatible with Algorithm 2.

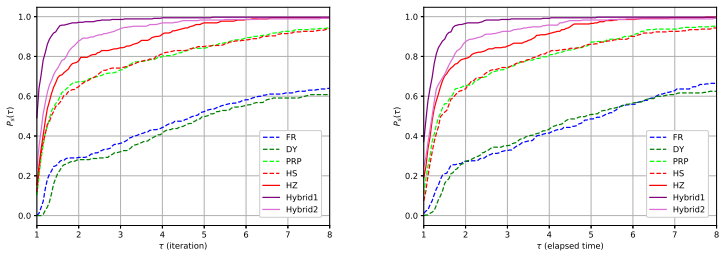


(a) iteration (b) elapsed time

Figure 1: Performance profiles of each algorithm versus the number of iterations (a) and the elapsed time (b) by using Algorithm 2 to determine the step size.

Figure 2 plots the performance profiles of each algorithm by using Algorithm 3 to determine the step size. In particular, Figure 2 (a) and (b) plot the performance profiles versus the number of iterations and the elapsed time, respectively. It shows that Hybrid1 solved the most problems, and Hybrid2

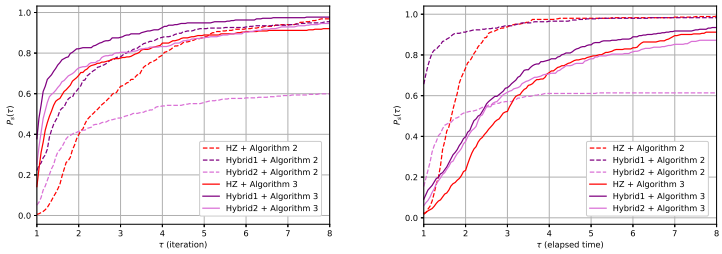
solved the second-most problems. Unlike the case of using Algorithm 2, Hybrid2 performed well when using Algorithm 3. It can be seen that the PRP and HS methods have about the same performance. Similarly, the FR and DY methods have about the same performance.



(a) iteration (b) elapsed time

Figure 2: Performance profiles of each algorithm versus the number of iterations (a) and the elapsed time (b) by using Algorithm 3 to determine the step size.

Figure 3 plots the performance profiles of HZ, Hybrid1 and Hybrid2 by using Algorithm 2 and 3 to determine the step size. In particular, Figure 3 (a) and (b) plots the performance profile versus the number of iterations and elapsed time, respectively. Figure 3 (a) shows that when Algorithm 3 is used, all methods solve the problem in fewer iterations than in the case of using Algorithm 2. It can be seen from Figure 3 (b) that Algorithm 3 often takes a long time to execute.



(a) iteration (b) elapsed time

Figure 3: Performance profiles of the HZ, Hybrid1 and Hybrid2 methods versus the number of iterations (a) and the elapsed time (b) by using Algorithm 2 and 3 to determine the step size.

It can be seen from Figure 1–3 that Hybrid1 performed the best in all cases, but the HZ method also performed well. It can also be seen that the performances of the Riemannian conjugate gradient methods depend greatly the type

of line search used. In particular, Hybrid2 may perform poorly when certain line searches are used.

6 Conclusion

We generalized two nonlinear conjugate gradient methods, i.e., the HZ and Hybrid2 method. We proved that the Hybrid2 method (14) satisfies the sufficient descent condition and converges globally under the strong Wolfe conditions. In addition, we proved that the HZ method (15) satisfies the sufficient descent condition regardless of the type of line search used. We also generalized two kinds of line search algorithms, i.e., Algorithm 2 and 3. In numerical experiments, we showed that the HZ and Hybrid1 methods perform well. Moreover, we showed that the performance of the Riemannian conjugate gradient method depends on the type of line search used. In particular, if we use Algorithm 2 to determine the step size, Hybrid2 performs poorly. On the other hand, it performs better with a step size computed by Algorithm 3, as the convergence analysis guarantees. Meanwhile, the numerical results showed that the HZ method converges quickly without depending on line search conditions. Hence, the HZ method is good for solving Riemannian optimization problems from the viewpoints of both theory and practice.

acknowledgements

This work was supported by a JSPS KAKENHI Grant, Number JP18K11184.

References

[1] P.-A. Absil and K. A. Gallivan. Joint diagonalization on the oblique manifold for independent component analysis. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 5, 2006.

[2] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[3] M. Al-Baali. Descent property and global convergence of the Fletcher-Reeves method with inexact line search. *IMA J. Numer. Anal.*, 5(1):121–124, 1985.

[4] Y.-H. Dai. Nonlinear conjugate gradient methods. *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

[5] Y.-H. Dai and Y. Yuan. A nonlinear conjugate gradient method with a strong global convergence property. *SIAM J. Optim.*, 10(1):177–182, 1999.

[6] Y.-H. Dai and Y. Yuan. An efficient hybrid conjugate gradient method for unconstrained optimization. *Ann. Oper. Res.*, 103(1-4):33–47, 2001.

[7] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math Program*, 91(2):201–213, 2002.

[8] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *Comput J*, 7(2):149–154, 1964.

[9] J. C. Gilbert and J. Nocedal. Global convergence properties of conjugate gradient methods for optimization. *SIAM J. Optim.*, 2(1):21–42, 1992.

[10] W. W. Hager and H. Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM J. Optim.*, 16(1):170–192, 2005.

[11] W. W. Hager and H. Zhang. A survey of nonlinear conjugate gradient methods. *Pacific J. Optim.*, 2(1):35–58, 2006.

[12] M. R. Hestenes and E. Stiefel. *Methods of conjugate gradients for solving linear systems*. NBS Washington, DC, 1952.

[13] S. Hosseini, W. Huang, and R. Yousefpour. Line search algorithms for locally Lipschitz functions on Riemannian manifolds. *SIAM J. Optim.*, 28(1):596–619, 2018.

[14] Y. Hu and C. Storey. Global convergence result for conjugate gradient methods. *J Optim Theory Appl*, 71(2):399–405, 1991.

[15] Y. Narushima and H. Yabe. A survey of sufficient descent conjugate gradient methods for unconstrained optimization. *SUT J. Math.*, 50(2):167–203, 2014.

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

[17] E. Polak and G. Ribière. Note sur la convergence de méthodes de directions conjuguées. *Esaim Math Model Numer Anal*, 3(R1):35–43, 1969.

[18] B. T. Polyak. The conjugate gradient method in extremal problems. *USSR Comput. Math. & Math. Phys.*, 9(4):94–112, 1969.

[19] W. Ring and B. Wirth. Optimization methods on Riemannian manifolds and their application to shape space. *SIAM J. Optim.*, 22(2):596–627, 2012.

[20] H. Sakai and H. Iiduka. Hybrid Riemannian conjugate gradient methods with global convergence properties. *Computational Optimization and Applications, accepted*, 2020.

[21] H. Sato. A Dai-Yuan-type Riemannian conjugate gradient method with the weak Wolfe conditions. *Comput Optim Appl*, 64(1):101–118, 2016.

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

[23] S. T. Smith. Optimization techniques on Riemannian manifolds. *Fields Inst. Commun.*, 3(3):113–135, 1994.

[24] J. Townsend, N. Koep, and S. Weichwald. Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation. *J Mach Learn Res*, 17(1):4755–4759, 2016.

[25] B. Vandereycken. Low-rank matrix completion by Riemannian optimization. *SIAM J. Optim.*, 23(2):1214–1236, 2013.

[26] P. Wolfe. Convergence conditions for ascent methods. *SIAM Rev Soc Ind Appl Math*, 11(2):226–235, 1969.

[27] P. Wolfe. Convergence conditions for ascent methods. ii: Some corrections. *SIAM Rev Soc Ind Appl Math*, 13(2):185–188, 1971.

[28] X. Zhu. A Riemannian conjugate gradient method for optimization on the Stiefel manifold. *Comput Optim Appl*, 67(1):73–110, 2017.