A Riemannian rank-adaptive method for low-rank optimization*

Guifang Zhou† Wen Huang‡ Kyle A. Gallivan† Paul Van Dooren‡
P.-A. Absil‡§

February 5, 2016

Abstract

This paper presents an algorithm that solves optimization problems on a matrix manifold $\mathcal{M} \subseteq \mathbb{R}^{m\times n}$ with an additional rank inequality constraint. The algorithm resorts to well-known Riemannian optimization schemes on fixed-rank manifolds, combined with new mechanisms to increase or decrease the rank. The convergence of the algorithm is analyzed and a weighted low-rank approximation problem is used to illustrate the efficiency and effectiveness of the algorithm.

Keywords: low-rank optimization; rank-constrained optimization; Riemannian manifold; fixed-rank manifold; low-rank approximation

1 Introduction

We consider low-rank optimization problems of the following form:

$$\min_{X\in\mathcal{M}_{\leq k}} f(X), \tag{1}$$

where $\mathcal{M}$ is a submanifold of $\mathbb{R}^{m\times n}$,

$$\mathcal{M}_{\leq k} := \{X \in \mathcal{M} \mid \operatorname{rank}(X) \leq k\}$$

with $k \leq \min(m, n)$, and f is a real-valued function on $\mathcal{M}_{\leq k}$. The notation

$$\mathcal{M}_r := \{X \in \mathcal{M} \mid \operatorname{rank}(X) = r\} \tag{2}$$

will also be used frequently. Typical choices for $\mathcal{M}$ are $\mathbb{R}^{m\times n}$ itself and the Frobenius sphere, i.e., the set of all $m \times n$ matrices of fixed Frobenius norm.

*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 the National Science Foundation under grant NSF-1262476 and by FNRS under grant PDR T.0173.13.

†Department of Mathematics, Florida State University, 1017 Academic Way, Tallahassee, FL 32306-4510, USA

‡ICTEAM Institute, Université catholique de Louvain, Avenue G. Lemaître 4, 1348 Louvain-la-Neuve, Belgium

§Corresponding author

Applications of (1) appear notably in machine learning, e.g., for collaborative filtering [Van13, CA15], multi-class classification [AFSU07], multi-response regression [MMBS13a, YYS15], learning a function over pairs of points [ABEV09], and learning a low-rank similarity measure [SWC12]. Applications of low-rank optimization are also found in other areas such as systems and control [Mar12, FPST13] and computer vision [LLY+13, SHK13].

An increasingly popular way to approach problem 1 is to consider the related but simpler problem $\min_{X \in \mathbb{R}_k^{m\times n}} f(X)$, where $\mathbb{R}_k^{m\times n} = \{X \in \mathbb{R}^{m\times n} \mid \text{rank}(X) = k\}$ in view of the notation (2); see, e.g.,[MMBS13b, AAM14, SWC12, MMBS14]. Since $\mathbb{R}_k^{m\times n}$ is a submanifold of $\mathbb{R}^{m\times n}$ of dimension $(m+n-k)k$ (see [HM94, Ch. 5, Prop. 1.14]), this simpler problem can be solved using Riemannian optimization techniques such as those presented in [AMS08, RW12, HAG14, HGA15, Sat14]. However, a disadvantage is that the manifold $\mathbb{R}_k^{m\times n}$ is not closed in $\mathbb{R}^{m\times n}$, which jeopardizes the well-posedness of the optimization problem and complicates the convergence analysis of optimization methods if the iterates cannot be assumed to stay safely away from $\mathbb{R}_{\leq k-1}^{m\times n}$.

Very recently a more global view of a projected line-search method on $\mathbb{R}_{\leq k}^{m\times n} = \{X \in \mathbb{R}^{m\times n} \mid \text{rank}(X) \leq k\}$ along with a convergence analysis has been developed in [SU15]. In [UV14], the results of [SU15] have been exploited to propose an algorithm that successively increases the rank by a given constant. Its convergence to critical points can be deduced from [SU15, Th. 3.9]; it relies on the assumption, often satisfied in practice, that the limit points have rank k. Under this assumption, a line-search method on $\mathbb{R}_{\leq k}^{m\times n}$ is ultimately the same as a line-search method on $\mathbb{R}_k^{m\times n}$.

In this paper, we develop a Riemannian rank-adaptive algorithm for the optimization problem (1). Its main features are as follows. First, the feasible set $\mathcal{M}_{\leq k}$ is more general than the set $\mathbb{R}_{\leq k}^{m\times n}$ considered in [SU15, UV14]. Second, the proposed algorithm increases or decreases the rank by an adaptively-chosen amount as the iteration proceeds. The rank update mechanism is governed by parameters that the user can adjust to strike a balance between the goals of (i) saving on space and time complexity by reducing the rank and (ii) achieving higher accuracy by increasing the rank. Finally, theoretical convergence results are given, and the proposed method is shown on numerical experiments to outperform state-of-the-art methods on a weighted low-rank approximation problem.

The rest of this paper is organized as follows. Standing assumptions are gathered in the next section. The proposed method is presented in Section 3 and analyzed in Section 4. Implementation practicalities are discussed in Section 5. Numerical experiments are reported in Section 6, and conclusions are drawn in Section 7.

A preliminary version of this work can be found in [Zho15].

2 Notation, Definitions, and Standing Assumptions

The notation $\mathcal{M}_r$ and $\mathcal{M}_{\leq r}$ defined above will be used frequently. The notation f_{F} stands for an extension of f on $\mathcal{M}$ (see Assumption 3 below) and f_r denotes the restriction of f to $\mathcal{M}_r$.

Throughout the paper, the following assumptions are in force.

Assumption 1. *$\overline{\mathcal{M}_r} \subseteq \mathcal{M}_{\leq r}$ for all positive integers $r \leq k$, where $\overline{\mathcal{M}_r}$ stands for the closure of $\mathcal{M}_r$.*

Observe that, since the closure of an intersection is a subset of the intersection of the closures and since $\overline{\mathbb{R}_r^{m\times n}} = \mathbb{R}_{\leq r}^{m\times n}$, it follows that the above assumption holds whenever the submanifold $\mathcal{M}$ is a closed subset of $\mathbb{R}^{m\times n}$. It is useful to bear in mind that a sequence of rank-r matrices can converge to a lower-rank matrix but not to a larger-rank matrix.

The next assumption is crucial to the Riemannian aspect of the proposed Riemannian rank-adaptive method:

Assumption 2. *$\mathcal{M}_r$ is a submanifold of $\mathbb{R}^{m\times n}$, for all positive integers $r \leq k$.*

We need the cost function f to be sufficiently smooth for gradient-descent techniques to be applicable:

Assumption 3. *The cost function f admits a continuously differentiable extension f_{F} on a neighborhood of $\mathcal{M}_{\leq k}$ in $\mathcal{M}$.*

The reader will observe that neither the size of the neighborhood nor the choice of the extension will have an impact on the proposed method.

The tangent cone to a set $S \subseteq \mathbb{R}^{m\times n}$ at $X \in \mathbb{R}^{m\times n}$ is the set

$$\mathrm{T}_X S := \{\dot{\gamma}(0) \mid \gamma \in C^1, \gamma(0) = X, \exists \delta > 0 : \forall t \in (0, \delta) : \gamma(t) \in S\},$$

where $\dot{\gamma}(0)$ denotes the derivative of curve γ at 0. This definition of $\mathrm{T}_X S$ is motivated by the goal of conducting line searches along smooth (i.e., C^1) curves. Observe that $\mathrm{T}_X S = \emptyset$ when $X \notin \overline{S}$.

We point out that, for any $X \in \mathcal{M}_r$, the tangent cones are nested as follows: $\mathrm{T}_X\mathcal{M}_{\leq 0} \subseteq \mathrm{T}_X\mathcal{M}_{\leq 1} \subseteq \cdots \subseteq \mathrm{T}_X\mathcal{M}$. The tangent cones $\mathrm{T}_X\mathcal{M}_{\leq r}$ and $\mathrm{T}_X\mathcal{M}$ are actually linear spaces since $\mathcal{M}$ and $\mathcal{M}_r$ are manifolds and $\mathcal{M}_{\leq r}$ is identical to $\mathcal{M}_r$ locally around $X \in \mathcal{M}_r$. Moreover, we have $\mathrm{T}_X\mathcal{M}_{\leq s} = \emptyset$ for all $s < r$. This justifies the following definition.

Definition 1 (update-rank)**.** *Let $X \in \mathcal{M}$ and $\eta_X \in \mathrm{T}_X\mathcal{M}$. The* update-rank *of η_X is the unique integer r such that $\eta_X \in \mathrm{T}_X\mathcal{M}_{\leq r} \setminus \mathrm{T}_X\mathcal{M}_{\leq r-1}$, with $A \setminus B$ denoting the set difference $\{x \in A \mid x \notin B\}$.*

For the purpose of conducting line searches along given directions while keeping the rank under control, we will need $\mathcal{M}$ to be endowed with a curve-selection mechanism defined as follows, where $\mathrm{T}\mathcal{M} := \bigsqcup_{X\in\mathcal{M}} \mathrm{T}_X\mathcal{M}$ denotes the tangent bundle of $\mathcal{M}$.

Definition 2 (Rank-related retraction)**.** *In the context of problem* (1)*, a mapping $\tilde{R} : \mathrm{T}\mathcal{M} \to \mathcal{M}$ is a* rank-related retraction *if, for all $X_* \in \mathcal{M}_{\leq k}$, there exists $\delta_{X_*} > 0$ and a neighborhood $\mathcal{U}$ of X_* in $\mathcal{M}_{\leq k}$ such that, for all $X \in \mathcal{U}$ and all $\xi_X \in \mathrm{T}_X\mathcal{M}_{\leq k}$ with $\|\xi_X\| = 1$, it holds that (i) $\tilde{R}_X(0) = X$, where $\tilde{R}_X$ denotes the restriction of $\tilde{R}$ to $\mathrm{T}_X\mathcal{M}$ and 0 stands for the zero vector in $\mathrm{T}_X\mathcal{M}$, (ii) $[0, \delta_{X_*}) \ni t \mapsto \tilde{R}_X(t\xi_X)$ is smooth and $\tilde{R}_X(t\xi_X) \in \mathcal{M}_{\leq \tilde{r}}$ for all $t \in [0, \delta_{X_*})$, where $\tilde{r}$ is the update-rank of ξ_X, (iii) $\frac{d}{dt}\tilde{R}_X(t\xi_X)|_{t=0} = \xi_X$.*

Note that $\tilde{R}_X$ is not necessarily a retraction on $\mathcal{M}$ in the sense given in [ADM+02, AMS08], since it may not be smooth on the tangent bundle $\mathrm{T}\mathcal{M}$. A specific rank-related retraction is given in Section 5.

Observe that in point (ii) of Definition 2, we require $\tilde{R}_X(t\xi_X)$ to belong to $\mathcal{M}_{\leq \tilde{r}}$ but not necessarily to $\mathcal{M}_{\tilde{r}}$. Indeed we found that the condition $\tilde{R}_X(t\xi_X) \in \mathcal{M}_{\tilde{r}}$ would be cumbersome to enforce while being unnecessary for the convergence analysis.

We let $\operatorname{grad} f_{\mathrm{F}}(X)$ denote the Riemannian gradient of f_{F} at $X \in \mathcal{M}$. It can be obtained by considering any smooth extension of f_{F} around X in $\mathbb{R}^{m\times n}$ and taking the projection to $\mathrm{T}_X\mathcal{M}$ of its Euclidean gradient at X; see [AMS08, (3.37)]. Likewise, $\operatorname{grad} f_r(X)$ denotes the Riemannian gradient of f_r at $X \in \mathcal{M}_r$, and it is obtained by projecting $\operatorname{grad} f_{\mathrm{F}}(X)$ onto the tangent space $\mathrm{T}_X\mathcal{M}_r$.

Throughout the paper, $\|\cdot\|$ denotes the Frobenius norm and $\langle\cdot,\cdot\rangle$ the Frobenius inner product.

Consider $X \in \mathcal{M}$, $\xi \in \mathrm{T}_X\mathcal{M}$, and a positive integer r. The set of best approximations of ξ in $\mathrm{T}_X\mathcal{M}_{\leq r}$ is denoted by $\mathrm{P}_{\mathrm{T}_X\mathcal{M}_{\leq r}}(\xi)$. Note that this set may contain more than one point. (In the case where $\mathcal{M} = \mathbb{R}^{m\times n}$, this follows directly from (12) and the non-uniqueness of a best low-rank approximation.) However, as indicated in [SU15, §2.1] (or see Lemma 1 below), all its elements have the same norm, hence $\|\mathrm{P}_{\mathrm{T}_X\mathcal{M}_{\leq r}}(\xi)\|$ is well defined. We say that X is a *critical point* of f if $\|\mathrm{P}_{\mathrm{T}_X\mathcal{M}_{\leq k}}(\operatorname{grad} f_{\mathrm{F}}(X))\| = 0$. (It can be seen that this notion does not depend on the chosen extension f_{F} of f.)

3 A Riemannian Rank-Adaptive Algorithm

The proposed method is listed in Algorithm 3, but we invite the reader to first read the more reader-friendly description in Section 3.1 and to refer to the pseudocode in Algorithm 3 when needed.

3.1 Algorithm description

We first discuss the two subprograms, Algorithms 1 and 2, called by Algorithm 3.

Algorithm 1 rank reduction with threshold Δ

Require: (X, Δ), where $X \in \mathbb{R}^{m\times n}$ and $\Delta > 0$.

1: Find the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_{\min\{m,n\}} \geq 0$ of matrix X;
2: Set r to be the largest integer r such that $\sigma_r/\sigma_1 \geq \Delta$;
3: Choose $\hat{X} \in \arg\min_{Y\in\mathcal{M}_{\leq r}} \|Y - X\|$;
4: Return $(\hat{X}, r)$.

The output $\hat{X}$ of Algorithm 1 is a best approximation of X in $\mathcal{M}_{\leq r}$, where r is the number (counting multiplicities) of singular values of X that are larger than $\sigma_1\Delta$, with σ_1 the largest singular value of X. Observe that $\hat{X}$ is simply X in the plausible case where X is already in $\mathcal{M}_{\leq r}$. In the case $\mathcal{M} = \mathbb{R}^{m\times n}$, Algorithm 1 consists in setting to zero the singular values of X that are smaller than $\sigma_1\Delta$.

Algorithm 2 Rank-related Armijo backtracking

1: Inherit $\tilde{R}, X_n, \beta, \bar{\alpha}, \eta^*, \mathcal{M}_{\tilde{r}}, f, \sigma$ from Algorithm 3 where Algorithm 2 is called;
2: Compute the smallest nonnegative integer m such that
(i) $\tilde{R}_{X_n}(\beta^m\bar{\alpha}\eta^*)$ belongs to $\mathcal{M}_{\leq\tilde{r}}$, and
(ii) $f(X_n) - f(\tilde{R}_{X_n}(\beta^m\bar{\alpha}\eta^*)) \geq \sigma\langle -\operatorname{grad} f_{\mathrm{F}}(X_n), \beta^m\bar{\alpha}\eta^*\rangle_{X_n}$;
3: Return $t^* \leftarrow \beta^m\bar{\alpha}$.

Algorithm 2 differs from the Riemannian Armijo backtracking [AMS08, Definition 4.2.2] by the safeguard (i). This safeguard is used because, in view of its δ_{X_*}, Definition 2 does not ensure that $\tilde{R}_{X_n}(\beta^m \bar{\alpha} \eta^*)$ is in $\mathcal{M}_{\leq \tilde{r}}$ unless m is sufficiently large.

Algorithm 3 Riemannian Rank-Adaptive Method (RRAM)

Require: initial iterate $X_0 \in \mathcal{M}_{\leq k}$, $\epsilon_1, \epsilon_3 > 0$, $\epsilon_2 \geq 0$, $c_{\mathrm{A}}, c_{\mathrm{R}}, \beta, \sigma, \tau_1, \tau_2, \Delta_0 \in (0,1)$, $\bar{\alpha} > 0$;
Ensure: Sequence of iterates $\{X_n\}$.

1: $\Delta \leftarrow \Delta_0$; $(\tilde{X}_0, r) \leftarrow$ Algorithm 1(X_0, Δ);
2: **for** $n = 0, 1, 2, \ldots$ **do**
3: Apply a Riemannian optimization method to minimize f_r over $\mathcal{M}_r$ with initial point $\tilde{X}_n$ and stop at $X_n \in \mathcal{M}_r$ where $\sigma_r(X_n)/\sigma_1(X_n) < \Delta$ ($\texttt{flag} \leftarrow 0$) or $\|\mathrm{grad}\, f_r(X_n)\| < \epsilon_3$ ($\texttt{flag} \leftarrow 1$); if $\nexists \delta_{\mathrm{ref}}$ then $\delta_{\mathrm{ref}} \leftarrow f(\tilde{X}_n) - f(X_n)$, $f_{\mathrm{ref}} \leftarrow f(\tilde{X}_n)$, $r_{\mathrm{ref}} \leftarrow r$;
4: **if** $\texttt{flag} = 1$ **then**
5: **if** $\|\mathrm{grad}\, f_{\mathrm{F}}(X_n) - \mathrm{grad}\, f_r(X_n)\| > \max\{\epsilon_1 \|\mathrm{grad}\, f_r(X_n)\|, \epsilon_2\}$ and $r < k$ **then**
6: $\tilde{r} \leftarrow r$; $\eta^* \leftarrow -\mathrm{grad}\, f_r(X_n)$; choose $\epsilon_4 < \epsilon_1$;
7: **while** $\| - \mathrm{grad}\, f_{\mathrm{F}}(X_n) - \eta^*\| > \epsilon_4 \|\eta^*\|$ and $\tilde{r} < k$ **do**
8: $\tilde{r} \leftarrow \tilde{r} + 1$; choose $\eta^* \in \arg\min_{\eta \in \mathrm{T}_X \mathcal{M}_{\leq \tilde{r}}} \| - \mathrm{grad}\, f_{\mathrm{F}}(X_n) - \eta\|$;
9: **end while**
10: Select $\tilde{X}_{n+1} \in \mathcal{M}_{\leq \tilde{r}}$ such that $f(X_n) - f(\tilde{X}_{n+1}) \geq c_{\mathrm{A}}(f(X_n) - f(\tilde{R}_{X_n}(t^* \eta^*)))$, where t^* is the rank-related Armijo step size returned by Algorithm 2;
11: $r \leftarrow \mathrm{rank}(\tilde{X}_{n+1})$; $\delta_{\mathrm{ref}} \leftarrow f(X_n) - f(\tilde{X}_{n+1})$; $f_{\mathrm{ref}} \leftarrow f(X_n)$; $r_{\mathrm{ref}} \leftarrow r$; $\Delta \leftarrow \Delta_0$;
12: **else**
13: If ϵ_3 is small enough, stop. Otherwise $\epsilon_3 \leftarrow \tau_1 \epsilon_3$ and $\tilde{X}_{n+1} \leftarrow X_n$.
14: **end if**
15: **else** $\{\texttt{flag} = 0\}$
16: $(\hat{X}_n, \tilde{r}) \leftarrow$ Algorithm 1(X_n, Δ);
17: **while** $f_{\mathrm{ref}} - f(\hat{X}_n) \leq c_{\mathrm{R}} \delta_{\mathrm{ref}}$ **do**
18: $\Delta \leftarrow \tau_2 \Delta$;
19: $(\hat{X}_n, \tilde{r}) \leftarrow$ Algorithm 1(X_n, Δ);
20: **end while**
21: $r \leftarrow \tilde{r}$; $\tilde{X}_{n+1} \leftarrow \hat{X}_n$;
22: **end if**
23: **end for**

Let us now turn to the main part (Algorithm 3) of the proposed Riemannian rank-adaptive method. The underlying idea is to alternate between the following two tasks.

1. *Riemannian update* (line 3 of Algorithm 3): given an iterate $\tilde{X}_n$ in $\mathcal{M}_r$ with $r \leq k$, run a Riemannian optimization method on the manifold $\mathcal{M}_r$, which returns a point X_n in $\mathcal{M}_r$ when a suitably chosen stopping criterion is satisfied.

2. *Rank-related update*: generate a new iterate $\tilde{X}_{n+1}$ in $\mathcal{M}_{\leq k}$ (line 10, line 13, or line 21), update the tolerances in the suitably chosen stopping criterion (line 11, line 13, line 18), increment n, and return to the Riemannian update.

We assume that the Riemannian optimization method invoked in the Riemannian update (line 3) is a descent iteration enjoying a global convergence property shared by all respectable such algorithms:

Assumption 4 (globally convergent Riemannian optimization method)**.** *Let* $\{Z_j\}$ *denote an infinite sequence generated by the Riemannian optimization method of line 3 of Algorithm 3. Then* $f(Z_{j+1}) < f(Z_j)$ *whenever* $Z_{j+1} \neq Z_j$*. If* Z_* *is a limit point of* $\{Z_j\}$ *in* $\mathcal{M}_r$ *(i.e., if there is no drop of rank at the limit point), then* $\operatorname{grad} f_r(Z_*) = 0$*.*

Well-understood general-purpose Riemannian optimization methods abound that, when applied to $\mathcal{M}_r$, satisfy Assumption 4; see, e.g., [RW12, HAG14, Sat14, HGA15] for recent points of entry to the literature. Several suitable implementations on $\mathcal{M}_r$ are available in Matlab [BMAS14] and C++.[1]

The challenge is thus to suitably choose the stopping criterion of the Riemannian update and to handle the rank-related updates so as to achieve the features mentioned in the introduction.

The stopping criterion for the Riemannian update is given in line 3 of Algorithm 3. The Riemannian update returns with $\texttt{flag} = 0$ if X_n is found to be dangerously close to the lower-rank set $\mathcal{M}_{\leq r-1}$; otherwise it returns with $\texttt{flag} = 1$ when $\|\operatorname{grad} f_r(X_n)\|$ becomes sufficiently small. (The "or" in line 3 is thus a "short-circuit or".) The danger announced by $\texttt{flag} = 0$ comes from Assumption 4, which offers no guarantee on the limit points of the Riemannian update that have rank lower than r. Hence, when the iterates of the Riemannian optimization method come too close to $\mathcal{M}_{\leq r-1}$, one needs to take action. The $\texttt{flag} = 0$ branch of the decision tree will be discussed in a moment.

Let us for now consider the case $\texttt{flag} = 1$, i.e., the Riemannian update (line 3) has returned $X_n \in \mathcal{M}_r$ sufficiently far away from $\mathcal{M}_{\leq r-1}$ and satisfying $\|\operatorname{grad} f_r(X_n)\| < \epsilon_3$. This indicates that X_n is an approximate critical point of f restricted to $\mathcal{M}_r$. However, it may still be possible to considerably reduce the value of f if we let the rank of the iterate get larger than r. We thus resort to readily available information—namely the first-order information provided by $\operatorname{grad} f_{\mathrm{F}}(X_n)$ and $\operatorname{grad} f_r(X_n)$, the latter being the projection of the former onto the tangent space $\mathrm{T}_{X_n}\mathcal{M}_r$—to decide if a rank increase looks promising (line 5). Specifically, we check if the angle between $\operatorname{grad} f_{\mathrm{F}}(X_n)$ and $\operatorname{grad} f_r(X_n)$ is large, as measured by the condition

$$\tan(\angle(\operatorname{grad} f_{\mathrm{F}}(X_n), \operatorname{grad} f_r(X_n))) > \epsilon_1, \tag{3}$$

and if moreover their difference is large, as measured by the condition

$$\|\operatorname{grad} f_{\mathrm{F}}(X_n) - \operatorname{grad} f_r(X_n)\| > \epsilon_2. \tag{4}$$

The conditions are illustrated in Figure 1. If both these conditions are satisfied, and moreover the maximal rank k is not reached (i.e., $r < k$), then we decide that we are in a favorable situation to attempt a rank increase. We now explore this branch of the decision tree.

The next issue is to decide by how much we allow the rank to increase. To this end, we increment $\tilde{r}$ from r to at most k until the tangent of the angle between $-\operatorname{grad} f_{\mathrm{F}}(X_n)$ and its projection η^* onto $\mathrm{T}_{X_n}\mathcal{M}_{\leq \tilde{r}}$ is smaller than some ϵ_4 chosen smaller than ϵ_1 (lines 6–9). At the end of this procedure, assuming that $\tilde{r}$ has not reached k, we can conclude that the update-rank of η^* (Definition 1) is $\tilde{r} > r$. Note that if $\mathcal{M} = \mathbb{R}^{m\times n}$ and $\tilde{r} = k$, then the choice of η^* (line 8) is equivalent to the definition in [SU15, Corollary 3.3].

We then perform an update along η^* by means of the rank-related retraction (Definition 2) that $\mathcal{M}$ needs to be endowed with, and we choose the step size by an Armijo-type backtracking

[1] `http://www.math.fsu.edu/~whuang2/ROPTLIB.htm`

procedure (line 10). This yields the next iterate, $\tilde{X}_{n+1}$, whose rank is less than or equal to $\tilde{r}$ but not necessarily equal to $\tilde{r}$.

Note that this Armijo procedure (line 10) is the only place in the proposed method where a rank increase can possibly occur. Choosing ϵ_1 and ϵ_2 small makes the algorithm more prone to executing the Armijo step, hence to increasing the rank. Choosing $\epsilon_2 > 0$ may result in blocking the rank at a small value for which the optimization problem (1) does not admit critical points, but the forthcoming Theorem 3 gives an upper bound on the "lack of criticality" of the output of the algorithm.

We now discuss the other branches of the decision tree. Still assume that $\texttt{flag} = 1$ but that we decide that we are *not* in a favorable situation to attempt a rank increase. Then we set $\tilde{X}_{n+1}$ to X_n and we return to the Riemannian step, now with a more stringent tolerance ϵ_3 (line 13).

Now consider the case $\texttt{flag} = 0$, where we know that we need to take action because the Riemannian update over $\mathcal{M}_r$ has returned X_n dangerously close to the lower-rank set $\mathcal{M}_{\leq r-1}$. The principle of the action is to perform a rank reduction by setting $\tilde{X}_{n+1}$ as the projection of X_n to $\mathcal{M}_{\leq \tilde{r}}$, where $\tilde{r}$ is the Δ-numerical rank of X_n returned by Algorithm 1(X_n, Δ). However, since the forthcoming convergence analysis relies crucially on the effect of the Armijo steps, we keep decreasing Δ (hence making the rank reduction less drastic) until the decrease of f achieved by the latest Armijo step (or by the initial Riemannian update if no Armijo step has been performed yet) is not too much unraveled by the rank reduction. The details are spelled out in line 16 and beyond.

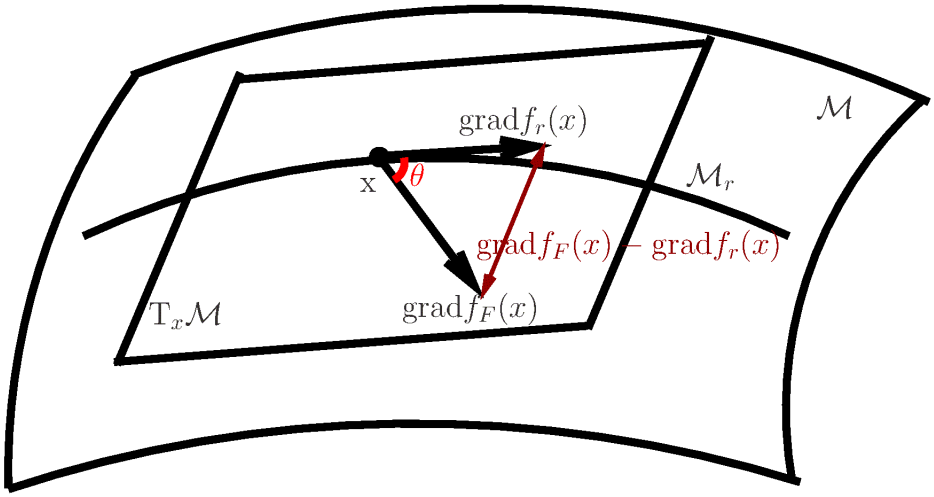

Figure 1: Illustration of conditions (3) and (4). θ is the angle between $\mathrm{grad}\, f_{\mathrm{F}}(X)$ and $\mathrm{grad}\, f_r(X)$ and the length of the red arrow represents $\|\mathrm{grad}\, f_{\mathrm{F}}(X) - \mathrm{grad}\, f_r(X)\|$.

3.2 Termination analysis

Algorithm 3 is meant to generate an infinite sequence $\{X_n\}_{n=0,1,\ldots}$, whose asymptotic behavior is analyzed in Section 4. In practice, a termination criterion can be based on the norm of the update vector η^* and on various context-dependent considerations. Our purpose in this

section is to make sure that all the steps of Algorithm 3 are well defined and terminate.

In view of Assumption 4, the Riemannian update in line 3 is guaranteed to terminate if the following assumption holds:

Assumption 5. *The sublevel set* $\{X \in \mathcal{M}_{\leq k} : f(X) \leq f(\tilde{X}_0)\}$ *is compact.*

Indeed, in view of Assumption 1 and the Bolzano–Weierstrass theorem, the sequence of iterates generated by the Riemannian optimization method has then at least one limit point, which is in $\mathcal{M}_{\leq r}$. Either this limit point is in $\mathcal{M}_r$ and thus, by Assumption 4, $\operatorname{grad} f_r$ gets arbitrarily small; or this limit point is in $\mathcal{M}_{\leq r-1}$ and the Riemannian optimization method returns with $\texttt{flag} = 0$.

The while loop starting in line 7 obviously terminates in view of the condition $\tilde{r} < k$.

The Armijo backtracking procedure (Algorithm 2) called in line 10 terminates (i.e., the smallest nonnegative integer m exists) in view of a classical argument since the function $\mathbb{R} \ni t \mapsto f(\tilde{R}_{X_n}(t\eta^*)) \in \mathbb{R}$ is differentiable around $t = 0$, a consequence of Assumption 3 and Definition 2.

In line 11, the fact that the rank may be numerically uncertain after the Armijo step is not an issue: if this is the case, then the next Riemannian update (line 3) will return immediately with $\texttt{flag} = 0$ and a rank decrease will take place.

It can be shown as follows that the while loop in line 17 is guaranteed to terminate. First consider the case where line 17 is reached for the first time from the moment where the latest assignment of δ_{ref} occurred. Then, invoking in particular Assumption 4, we obtain that $f_{\mathrm{ref}} - f(X_n) \geq \delta_{\mathrm{ref}} > 0$; and thus $f_{\mathrm{ref}} - f(X_n) > c_{\mathrm{R}}\delta_{\mathrm{ref}}$. If the while loop keeps being executed, then eventually no truncation occurs (i.e., $\hat{X}_n = X_n$), implying $f_{\mathrm{ref}} - f(\hat{X}_n) = f_{\mathrm{ref}} - f(X_n) > c_{\mathrm{R}}\delta_{\mathrm{ref}}$. The claim is then established by induction.

4 Convergence Analysis

We now proceed to the convergence analysis of Algorithm 3 under the standing assumptions stated in Section 2.

In Section 4.1 we consider the specific case where ϵ_2 is set to zero, then in Section 4.2 we exploit our findings to handle the general case $\epsilon_2 \geq 0$. Observe that ϵ_2 appears only in line 5 of Algorithm 3. When it is set to zero, the first condition in line 5—which must be satisfied to execute Armijo (line 10)—reduces to the angle condition (3). The motivation behind a choice of $\epsilon_2 > 0$ is that it makes the algorithm less inclined to execute Armijo, hence more inclined to keep the rank low, thus gaining in spatial complexity. The findings of our convergence analysis indicate that the price to pay is the residual error that may subsist, in the sense that $\liminf_{n\to\infty} \|\mathrm{P}_{\mathrm{T}_{X_n}\mathcal{M}_{\leq k}}(-\operatorname{grad} f_{\mathrm{F}}(X_n))\|$ may not be zero; however, the residual error remains under control due to an upper bound proportional to ϵ_2 guaranteed by Theorem 3.

We need some preliminary work before stating and proving the convergence results. Lemma 1 concerns the vector η^* obtained in line 8 of Algorithm 3. The first claim of Lemma 1 will be invoked in the proof of Theorem 2, while its second claim is an easy result that confirms a property announced in Section 2.

Lemma 1. *Let* $X \in \mathcal{M}_r$ *and* $\tilde{r} \geq r$. *If* $\eta^* \in \arg\min_{\eta \in \mathrm{T}_X\mathcal{M}_{\leq \tilde{r}}} \| - \operatorname{grad} f_{\mathrm{F}}(X) - \eta\|$ *then*

$$\langle \eta^*, -\operatorname{grad} f_{\mathrm{F}}(X)\rangle = \|\eta^*\|^2.$$

We also have $\|\eta^*\|^2 = \|\operatorname{grad} f_{\mathrm{F}}(X)\|^2 - \min_{\eta \in \mathrm{T}_X\mathcal{M}_{\leq \tilde{r}}} \| - \operatorname{grad} f_{\mathrm{F}}(X) - \eta\|^2$.

Proof. Let $\eta^* \in \arg\min_{\eta \in \mathrm{T}_X\mathcal{M}_{\leq \tilde{r}}} \| - \operatorname{grad} f_{\mathrm{F}}(X) - \eta\|$. Since the tangent cone $\mathrm{T}_X\mathcal{M}_{\leq \tilde{r}}$ is indeed a cone, $t\eta^* \in \mathrm{T}_X\mathcal{M}_{\leq \tilde{r}}$ for all $t > 0$. Therefore

$$\frac{d}{dt}\| - \operatorname{grad} f_{\mathrm{F}}(X) - t\eta^*\|^2|_{t=1} = 0,$$

which implies

$$\langle \eta^*, -\operatorname{grad} f_{\mathrm{F}}(X)\rangle = \langle \eta^*, \eta^*\rangle = \|\eta^*\|^2,$$

which is the first claim. We thus have the orthogonality condition $\langle \eta^*, -\operatorname{grad} f_{\mathrm{F}}(X) - \eta^*\rangle = 0$, from which it follows that $\|\operatorname{grad} f_{\mathrm{F}}(X)\|^2 = \| - \operatorname{grad} f_{\mathrm{F}}(X) - \eta^*\|^2 + \|\eta^*\|^2$, yielding the second claim. □

The analysis of the Armijo step makes use of the following assumption, which is satisfied in particular when the rank-related retraction $\tilde{R}$ is the one proposed in Section 5.

Assumption 6 (locally radially L-C^1)**.** *The lifted function*

$$\hat{f} : \mathrm{T}\mathcal{M} \to \mathbb{R} : \xi \mapsto f \circ \tilde{R}(\xi).$$

is locally radially Lipschitz continuously differentiable (locally radially L-C^1)*, that is, for all $X_* \in \mathcal{M}_{\leq k}$, there exists $\beta_{RL} > 0$, $\delta_{RL} > 0$, and a neighborhood $\mathcal{U}$ of X_* in $\mathcal{M}_{\leq k}$ such that, for all $X \in \mathcal{U}$, for all $\xi \in \mathrm{T}_X\mathcal{M}_{\leq k}$ with $\|\xi\| = 1$, and for all $t < \delta_{RL}$, it holds that*

$$\left| \frac{d}{d\tau}\hat{f}_X(\tau\xi)|_{\tau=t} - \frac{d}{d\tau}\hat{f}_X(\tau\xi)|_{\tau=0} \right| \leq \beta_{RL} t. \tag{5}$$

4.1 Convergence Analysis with $\epsilon_2 = 0$

The main global convergence properties of Algorithm 3 for $\epsilon_2 = 0$ are stated in Theorem 2 below. In a nutshell, it shows that the best approximation of $\operatorname{grad} f_{\mathrm{F}}(X_n)$ in $\mathrm{T}_{X_n}\mathcal{M}_{\leq k}$ gets arbitrarily small for some n large enough. Its proof can be viewed as an extension to the rank-adaptive setting of the well-known fact that an occasional steepest descent step is sufficient to guarantee global convergence in a Euclidean setting [NW06, p. 41] and more generally in the Riemannian setting [AG09]. As we will see, however, spelling out the proof details is not straightforward.

Theorem 2. *Under the standing assumptions (Section 2) and Assumptions 4–6, let $\{X_n\}$ be an infinite sequence of iterates generated by Algorithm 3 with $\epsilon_2 = 0$. Then*

$$\liminf_{n\to\infty} \|\mathrm{P}_{\mathrm{T}_{X_n}\mathcal{M}_{\leq k}}(-\operatorname{grad} f_{\mathrm{F}}(X_n))\| = 0$$

.

Proof. We will distinguish two cases.

Case 1: Armijo (line 10) is executed *infinitely* many times. In this case, the claim follows from a fairly standard Armijo-type analysis invoking Assumption 6. Details are given below.

Case 2: Armijo (line 10) is executed *finitely* many times. Since the Armijo step is the only step of Algorithm 3 where the rank can increase, it follows that X_n stays in a fixed-rank manifold $\mathcal{M}_r$ for all n large enough. The rank reduction mechanism of Algorithm 3 does not allow the iterates to approach $\mathcal{M}_{\leq r-1}$ (see details below) and the stronger claim

$\liminf_{n\to\infty} \|\operatorname{grad} f(X_n)\| = 0$ (when $r = k$) or even $\liminf_{n\to\infty} \|\operatorname{grad} f_{\mathrm{F}}(X_n)\| = 0$ (when $r < k$) follows from Assumption 4.

Case 1 (details): Let $\{X_{n_j}\}$ be the infinite subsequence of iterates at which Armijo (line 10) is executed. In view of line 10 of Algorithm 3,

$$f(X_{n_j}) - f(\tilde{X}_{n_j+1}) \geq c_{\mathrm{A}}\sigma\alpha_{n_j}\langle -\operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \eta^*_{n_j}\rangle_{X_{n_j}}, \tag{6}$$

where $\sigma \in (0,1)$ is a parameter of Algorithm 3 and $\eta^*_{n_j}$, resp. α_{n_j}, denotes the η^*, resp. t^*, produced by line 10 at iteration n_j. Note that Algorithm 3 is not a descent iteration (the value of the cost function may increase during the rank reduction steps), hence we cannot immediately conclude that $f(X_{n_j}) - f(X_{n_{j+1}})$ goes to zero. It does though, in view of the following argument.

Since the Riemannian optimization method (line 3) is a descent iteration (Assumption 4) and in view of line 17, we can deduce that $f(X_{n_j}) - f(\tilde{X}_{n_{j+1}}) \geq c_{\mathrm{R}}\left(f(X_{n_j}) - f(\tilde{X}_{n_j+1})\right)$ (observe the difference between $\tilde{X}_{n_{j+1}}$ and $\tilde{X}_{n_j+1}$). We thus have

$$0 \leq c_{\mathrm{R}}\left(f(X_{n_j}) - f(\tilde{X}_{n_j+1})\right) \leq f(X_{n_j}) - f(\tilde{X}_{n_{j+1}}) \leq f(X_{n_j}) - f(X_{n_{j+1}}), \tag{7}$$

where the second inequality has just been shown, the first one follows from (6) and Lemma 1, and the third one follows from $f(\tilde{X}_{n_{j+1}}) \geq f(X_{n_{j+1}})$ (Assumption 4).

Therefore $\{f(X_{n_j})\}$ is nonincreasing. Furthermore, $\{f(X_{n_j})\}$ is bounded below since $\{X_n\}$ is bounded (Assumption 5) and f is continuous (Assumption 3). Thus the sequence of differences $f(X_{n_j}) - f(X_{n_{j+1}})$ must go to zero. So does $f(X_{n_j}) - f(\tilde{X}_{n_j+1})$ in view of (7).

Contradiction is used to show that $\langle \operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \eta^*_{n_j}\rangle_{X_{n_j}} \to 0$. Suppose not. Then, since $\{X_n\}$ is bounded, there exist a convergent subsequence $\{X_{n_j}\}_{j\in\mathcal{J}}$ and $\mu > 0$ such that $\langle \operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \eta^*_{n_j}\rangle_{X_{n_j}} < -\mu$ for all $j \in \mathcal{J}$. Let X_* denote the limit of $\{X_{n_j}\}_{j\in\mathcal{J}}$. Since $f(X_{n_j}) - f(\tilde{X}_{n_j+1})$ goes to zero, it follows from (6) that $\{\alpha_{n_j}\}_{j\in\mathcal{J}} \to 0$. Since $\{\alpha_{n_j}\}_{j\in\mathcal{J}} \to 0$ and since the α_{n_j}'s are determined by Armijo backtracking (Algorithm 2), it follows that when j gets sufficiently large, at least one step of backtracking is applied to get α_{n_j}. Restrict the index set $\mathcal{J}$ to those (infinitely many) sufficiently large j's. Furthermore, again for all j sufficiently large, in view of Definition 2 and since $\lim_{j\in\mathcal{J}\to\infty} X_{n_j} = X_*$, the safeguard (i) in the Armijo backtracking is always satisfied. Restrict further the index set $\mathcal{J}$ to those (infinitely many) sufficiently large j's. Then for all $j \in \mathcal{J}$, since α_{n_j} results from at least one step of backtracking, it must be that the update $\frac{\alpha_{n_j}}{\beta}\eta^*_{n_j}$ did not satisfy the Armijo condition; that is,

$$f(X_{n_j}) - f(\tilde{R}_{X_{n_j}}(\frac{\alpha_{n_j}}{\beta}\eta^*_{n_j})) < \sigma\frac{\alpha_{n_j}}{\beta}\langle -\operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \eta^*_{n_j}\rangle_{X_{n_j}}, \quad \forall j \in \mathcal{J}.$$

Denoting

$$\tilde{\eta}_{n_j} = \frac{\eta^*_{n_j}}{\|\eta^*_{n_j}\|} \text{ and } \tilde{\alpha}_{n_j} = \frac{\alpha_{n_j}\|\eta^*_{n_j}\|}{\beta},$$

the inequality above reads

$$\frac{\tilde{f}_{\tilde{\eta}_{n_j}}(0) - \tilde{f}_{\tilde{\eta}_{n_j}}(\tilde{\alpha}_{n_j})}{\tilde{\alpha}_{n_j}} < \sigma\langle -\operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \tilde{\eta}_{n_j}\rangle_{X_{n_j}}, \quad \forall j \in \mathcal{J},$$

where $\tilde{f}_\eta(t) := f(\tilde{R}_X(t\eta))$. If necessary, remove from $\mathcal{J}$ finitely many elements to ensure that, for all $j \in \mathcal{J}$, X_{n_j} is in the neighborhood $\mathcal{U}$ of X_* and $\tilde{\alpha}_{n_j} < \delta_{\mathrm{RL}}$, where $\mathcal{U}$ and δ_{RL} are those of Assumption 6. The mean value theorem then ensures that, for all $j \in \mathcal{J}$, there exists $t_{n_j} \in [0, \tilde{\alpha}_{n_j}]$ such that

$$-\frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=t_{n_j}} < \sigma\langle -\operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \tilde{\eta}_{n_j}\rangle_{X_{n_j}}, \quad \forall j \in \mathcal{J}. \tag{8}$$

Since $\|\eta^*_{n_j}\| \leq \|\operatorname{grad} f_{\mathrm{F}}(X_{n_j})\|$ which remains bounded since $f \in C^1$ and $\{X_{n_j}\}$ is bounded, it follows that $\eta^*_{n_j}$ is bounded. Thus $\{\tilde{\alpha}_{n_j}\}_{j\in\mathcal{J}} \to 0$ since $\tilde{\alpha}_{n_j} = \frac{\alpha_{n_j}\|\eta^*_{n_j}\|}{\beta}$ with $\{\alpha_{n_j}\}_{j\in\mathcal{J}} \to 0$. In view of property (iii) of rank-related retractions (Definition 2), we have that $\frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=0} = \langle \operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \tilde{\eta}_{n_j}\rangle_{X_{n_j}}$. This and (8) yield

$$\begin{aligned}\frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=0} - \frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=t_{n_j}} &< (1-\sigma)\langle \operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \tilde{\eta}_{n_j}\rangle_{X_{n_j}} \\ &< -(1-\sigma)\mu.\end{aligned}$$

We have $\left|\frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=0} - \frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=t_{n_j}}\right| \leq \beta_{\mathrm{RL}} t_{n_j}$ in view of Assumption 6. As j goes to infinity in $\mathcal{J}$, the left-hand side thus goes to zero since $t_{n_j} \in [0, \tilde{\alpha}_{n_j}]$ must go to zero. Hence $0 = \lim_{j\in\mathcal{J}} \left(\frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=0} - \frac{d}{dt}\tilde{f}_{\tilde{\eta}_{n_j}}(t)|_{t=t_{n_j}}\right) \leq -(1-\sigma)\mu < 0$, a contradiction.

The contradiction argument is thus complete, which shows that

$$\langle \operatorname{grad} f_{\mathrm{F}}(X_{n_j}), \eta^*_{n_j}\rangle_{X_n} \to 0.$$

In view of Lemma 1, it follows that $\eta^*_{n_j} \to 0$. We distinguish two subcases:
Case 1.1: All (except finitely many) $\eta^*_{n_j}$ were obtained in line 8 with $\tilde{r} = k$. Then the claim $\liminf_{n\to\infty} \|\mathrm{P}_{\mathrm{T}_{X_n}\mathcal{M}_{\leq k}}(-\operatorname{grad} f_{\mathrm{F}}(X_n))\| = 0$ follows.
Case 1.2: The other case is where infinitely many $\eta^*_{n_j}$ were obtained with $\tilde{r} < k$, hence with $\|-\operatorname{grad} f_{\mathrm{F}}(\tilde{X}_{n_j}) - \eta^*_{n_j}\| \leq \epsilon_4\|\eta^*_{n_j}\|$ in view of line 7. The stronger claim $\liminf_{n\to\infty} \|\operatorname{grad} f_{\mathrm{F}}(X_n)\| = 0$ follows.

Case 2 (details): As we already pointed out, we know in Case 2 that X_n stays in a fixed-rank manifold $\mathcal{M}_r$ for all n large enough. Let X_* be a limit point of $\{X_n\}$, and let $\{X_{n_j}\}$ be a subsequence that converges to X_*. Then X_* is in the closure $\overline{\mathcal{M}_r}$, hence in $\mathcal{M}_{\leq r}$ by Assumption 1. But X_* cannot be in $\mathcal{M}_{\leq r-1}$, otherwise the rank reduction mechanism would not allow the rank to remain r. (This can be seen as follows. In Case 2, δ_{ref} remains constant for all n large enough. Moreover, $r \leq r_{\mathrm{ref}}$ and $f_{\mathrm{ref}} - f(X_n) > c_{\mathrm{R}}\delta_{\mathrm{ref}}$ hold throughout the execution of the Algorithm 3 after the last Armijo is executed. Suppose for contradiction that $X_* \in \mathcal{M}_s$ with $s < r$. Then X_{n_j} gets arbitrarily close to $\mathcal{M}_s$. For j large enough, the truncation in line 16 or 19 will produce $\hat{X}_{n_j} \in \mathcal{M}_s \cup \mathcal{M}_{s+1} \cdots \cup \mathcal{M}_{r-1}$ closest to X_{n_j} and thus also arbitrarily close to X_{n_j}. Since f, being C^1, is uniformly continuous in any bounded domain in which the iteration stays, it follows that $|f(\hat{X}_{n_j}) - f(X_{n_j})|$ becomes arbitrarily small, hence small enough for $f_{\mathrm{ref}} - f(\hat{X}_{n_j}) > c_{\mathrm{R}}\delta_{\mathrm{ref}}$ to hold. Hence, for some j large enough, the while loop starting in line 17 will terminate with $\hat{X}_{n_j} \in \mathcal{M}_s \cup \mathcal{M}_{s+1} \cdots \cup \mathcal{M}_{r-1}$, meaning that the rank drops below r, a contradiction.) We thus conclude that X_* is in $\mathcal{M}_r$. It follows from Assumption 4 that $\operatorname{grad} f_r(X_*) = 0$. We are in one of two subcases:
Case 2.1: $r = k$. Then $-\operatorname{grad} f_r(X_*) = \lim_{j\to\infty} -\operatorname{grad} f_r(X_{n_j}) = \lim_{j\to\infty} \mathrm{P}_{\mathrm{T}_{X_{n_j}}\mathcal{M}_k}(-\operatorname{grad} f_{\mathrm{F}}(X_{n_j})) =$

$\lim_{j\to\infty} \mathrm{P}_{\mathrm{T}_{X_{n_j}}\mathcal{M}_{\leq k}}(-\mathrm{grad}\, f_{\mathrm{F}}(X_{n_j}))$ and the claim follows.
Case 2.2: $r < k$. Then $\mathrm{grad}\, f_{\mathrm{F}}(X_*) = 0$, otherwise (recall $\epsilon_2 = 0$) the condition in line 5 would have been satisfied and Armijo would have been executed infinitely many times, a contradiction with Case 2. The stronger claim $\liminf_{n\to\infty} \|\mathrm{grad}\, f_{\mathrm{F}}(X_n)\| = 0$ follows. □

4.2 Convergence Analysis with $\epsilon_2 \geq 0$

Recall that Theorem 2 considers the case where $\epsilon_2 = 0$. Its proof led us to consider Case 1 where Armijo (line 10) is executed infinitely many times. Figure 2 shows a situation where this case can indeed occur. An Armijo step is taken from $X_n \in \mathcal{M}_r$ because the gradient angle condition (3) is satisfied (due to the narrowness of the valley in the cost function landscape), the next iterate is not far away since it must produce a decrease in f, a rank reduction to r occurs at a subsequent iterate, and the process repeats over and over again. This "hem stitching" phenomenon results in a slow convergence. It is tempting to remedy this phenomenon by making the iteration less inclined to leave $\mathcal{M}_r$.

This is the purpose of the ϵ_2 parameter of Algorithm 3. Indeed, in the example of Figure 2, when $\epsilon_2 > 0$, the gradient distance condition (4) will finally no longer hold (because $\mathrm{grad}\, f_{\mathrm{F}}(X_n)$ goes to zero as $n \to \infty$) and thus line 13 will be executed instead of Armijo. The iterates thus remain on $\mathcal{M}_r$ and the Riemannian optimization method of line 3 keeps being applied, potentially offering a faster convergence.

The downside with $\epsilon_2 > 0$ is that if the minimizer X_* of f_{F} is now slightly off $\mathcal{M}_r$ instead of being in $\mathcal{M}_r$, then the gradient distance condition (4) will never be satisfied, hence Armijo will never be executed, thus the rank will never increase beyond r, ruling out convergence to X_*. The situation, however, is not as unfavorable as it may look, much to the contrary. First, $\epsilon_2 > 0$ achieves the goal of making Algorithm 3 less inclined to increase the rank, hence obtaining a lower complexity in exchange for yielding some ground on accuracy. Second, the accuracy remains under control due to the bound on $\liminf_{n\to\infty} \|\mathrm{P}_{\mathrm{T}_{X_n}\mathcal{M}_{\leq k}}(-\mathrm{grad}\, f_{\mathrm{F}}(X_n))\|$ given in Theorem 3 below.

Theorem 3. *Under the standing assumptions (Section 2) and Assumptions 4–6, let $\{X_n\}$ be a bounded infinite sequence of iterates generated by Algorithm 3, now with $\epsilon_2 \geq 0$. Then*

$$\liminf_{n\to\infty} \|\mathrm{P}_{\mathrm{T}_{X_n}\mathcal{M}_{\leq k}}(-\mathrm{grad}\, f_{\mathrm{F}}(X_n))\| \leq \left(\sqrt{1 + \frac{1}{\epsilon_1^2}}\right) \epsilon_2.$$

Proof. Observe that ϵ_2 appears only in line 5 of Algorithm 3. We will distinguish two cases, according to whether ϵ_2 influences or not the asymptotic behavior of the iteration. To this end, let $\{X_{n_j}\}$ be the subsequence of iterates for which the requirement in line 5 does not hold solely because the gradient distance condition (4) does not hold. In other words, we have $\|\mathrm{grad}\, f_{\mathrm{F}}(X_{n_j}) - \mathrm{grad}\, f_r(X_{n_j})\| > \epsilon_1 \|\mathrm{grad}\, f_r(X_{n_j})\|$ and $r < k$ but $\|\mathrm{grad}\, f_{\mathrm{F}}(X_{n_j}) - \mathrm{grad}\, f_r(X_{n_j})\| \leq \epsilon_2$. Let θ denote the angle between $\mathrm{grad}\, f_{\mathrm{F}}(X_{n_j})$ and $\mathrm{grad}\, f_r(X_{n_j})$. We then have

$$\sin(\theta)\|\mathrm{grad}\, f_{\mathrm{F}}(X_{n_j})\| = \|\mathrm{grad}\, f_{\mathrm{F}}(X_{n_j}) - \mathrm{grad}\, f_r(X_{n_j})\| \leq \epsilon_2$$

and $\tan(\theta) > \epsilon_1$. Thus

$$\|\mathrm{grad}\, f_{\mathrm{F}}(X_{n_j})\| \leq \frac{1}{\sin(\theta)}\epsilon_2 = \left(\sqrt{1 + \frac{1}{\tan^2(\theta)}}\right) \epsilon_2 \leq \left(\sqrt{1 + \frac{1}{\epsilon_1^2}}\right) \epsilon_2.$$

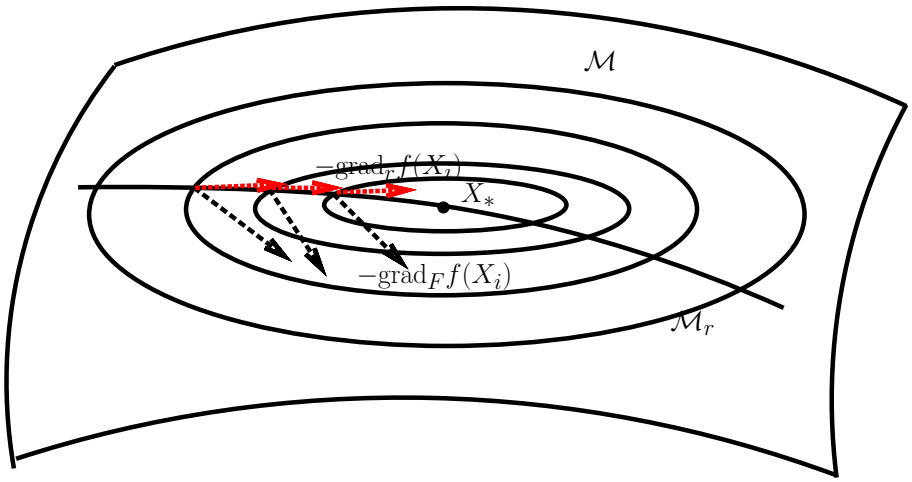

Figure 2: Illustration of a situation where the Armijo step (line 10 of Algorithm 3) would be executed infinitely many times. The black dotted arrows represent $-\mathrm{grad}\, f_{\mathrm{F}}$ and the red dotted arrows represent $-\mathrm{grad}\, f_r$. The circles are level sets of f_{F}, and X_* is a local minimum of f_{F} that belongs to $\mathcal{M}_r$.

Case 1: The subsequence $\{X_{n_j}\}$ is infinite. Then $\liminf_{n\to\infty} \|\mathrm{grad}\, f_{\mathrm{F}}(X_n)\| \leq \left(\sqrt{1+\frac{1}{\epsilon_1^2}}\right)\epsilon_2$, and the conclusion follows.

Case 2: The subsequence $\{X_{n_j}\}$ is finite. Let X_K be its last element. Then the exact same sequence $\{X_n\}$ would be generated by setting ϵ_2 to zero from iteration $K+1$ onward. One can then conclude as in Theorem 2 that $\liminf_{n\to\infty} \|\mathrm{P}_{\mathrm{T}_{X_n}\mathcal{M}_{\leq k}}(-\mathrm{grad}\, f_{\mathrm{F}}(X_n))\| = 0$. □

5 Implementation Details

A practical implementation of Algorithm 3 requires an adequate way to store the low-rank iterates X_n and the various tangent vectors, to compute the projection required in line 8, and to choose the rank-related retraction $\tilde{R}$ required in line 10. We present those implementation details for the frequently encountered case where

$$\mathcal{M} = \mathbb{R}^{m\times n}.$$

Let $X \in \mathbb{R}_r^{m\times n}$. Then X can be decomposed as $X = U_r D_r V_r^T$ where U_r and V_r are orthonormal matrices of size $m\times r$ and $n\times r$ respectively. We have (see [SU15, Theorem 3.2]):

$$\mathrm{T}_X\mathbb{R}_{\leq\tilde{r}}^{m\times n} = \left\{ \begin{array}{c} U_r A V_r^T + U_r B V_{r\perp}^T + U_{r\perp} C V_r^T + U_{r\perp} E V_{r\perp}^T : \\ A, B, C, E \ \text{ arbitrary matrices with } \mathrm{rank}(E) \leq \tilde{r} - r, \end{array} \right\},$$

where $U_{r\perp}$ is chosen such that $\begin{bmatrix} U_r & U_{r\perp}\end{bmatrix}$ is an orthogonal matrix, and likewise for $V_{r\perp}$. (An expression for the case where $\mathcal{M}$ is the Frobenius sphere can also be found in [CAV13].)

Thus the elements η of $\mathrm{T}_X\mathbb{R}_{\leq\tilde{r}}^{m\times n}$ are the matrices of the form

$$\eta = \begin{bmatrix} U_r & U_{r\perp}\end{bmatrix} \begin{bmatrix} A & B \\ C & E \end{bmatrix} \begin{bmatrix} V_r^T \\ V_{r\perp}^T \end{bmatrix}, \quad \mathrm{rank}(E) \leq \tilde{r} - r. \tag{9}$$

Since E has rank at most $\tilde{r}-r =: \Delta r$, we can write $E = \begin{bmatrix}\tilde{U}_{\Delta r} & \tilde{U}_{\Delta r\perp}\end{bmatrix}\begin{bmatrix}E_{\Delta r} & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}\tilde{V}_{\Delta r} & \tilde{V}_{\Delta r\perp}\end{bmatrix}^T$ with $E_{\Delta r}$ of size $\Delta r \times \Delta r$, and (9) becomes

$$\begin{aligned}\eta &= \begin{bmatrix}U_r & U_{\Delta r} & U_{(r+\Delta r)\perp}\end{bmatrix}\begin{bmatrix}A & B_1 & B_2\\ C_1 & E_{\Delta r} & 0_{\Delta r\times(n-\tilde{r})}\\ C_2 & 0_{(m-\tilde{r})\times\Delta r} & 0_{(m-\tilde{r})\times(n-\tilde{r})}\end{bmatrix}\begin{bmatrix}V_r^T\\ V_{\Delta r}^T\\ V_{(r+\Delta r)\perp}^T\end{bmatrix}\\ &= \begin{bmatrix}U_{\tilde{r}} & U_{\tilde{r}\perp}\end{bmatrix}\begin{bmatrix}\dot{D}_{\tilde{r}} & B_2\\ C_2 & 0_{(m-\tilde{r})\times(n-\tilde{r})}\end{bmatrix}\begin{bmatrix}V_{\tilde{r}}^T\\ V_{\tilde{r}\perp}^T\end{bmatrix}\end{aligned} \tag{10}$$

with $U_{\Delta r} = U_{r\perp}\tilde{U}_{\Delta r}$, $U_{(r+\Delta t)\perp} = U_{r\perp}\tilde{U}_{\Delta r\perp}$ and likewise for V. Since $X = U_r D_r V_r^T$, it can be written in the same block structure, yielding

$$\begin{aligned}X &= \begin{bmatrix}U_r & U_{\Delta r} & U_{(r+\Delta r)\perp}\end{bmatrix}\begin{bmatrix}D_r & 0_{r\times\Delta r} & 0_{r\times(n-\tilde{r})}\\ 0_{\Delta r\times r} & 0_{\Delta r\times\Delta r} & 0_{\Delta r\times(n-\tilde{r})}\\ 0_{(m-\tilde{r})\times r} & 0_{(m-\tilde{r})\times\Delta r} & 0_{(m-\tilde{r})\times(n-\tilde{r})}\end{bmatrix}\begin{bmatrix}V_r^T\\ V_{\Delta r}^T\\ V_{(r+\Delta r)\perp}^T\end{bmatrix}\\ &= \begin{bmatrix}U_{\tilde{r}} & U_{\tilde{r}\perp}\end{bmatrix}\begin{bmatrix}D_{\tilde{r}} & 0_{\tilde{r}\times(n-\tilde{r})}\\ 0_{(m-\tilde{r})\times\tilde{r}} & 0_{(m-\tilde{r})\times(n-\tilde{r})}\end{bmatrix}\begin{bmatrix}V_{\tilde{r}}^T\\ V_{\tilde{r}\perp}^T\end{bmatrix}.\end{aligned}$$

Expression (10) can be rewritten as

$$\eta = \dot{U}_{\tilde{r}} D_{\tilde{r}} V_{\tilde{r}}^T + U_{\tilde{r}}\dot{D}_{\tilde{r}} V_{\tilde{r}}^T + U_{\tilde{r}} D_{\tilde{r}}\dot{V}_{\tilde{r}}^T, \tag{11}$$

with $U_{\tilde{r}}^T\dot{U}_{\tilde{r}} = 0$, $V_{\tilde{r}}^T\dot{V}_{\tilde{r}} = 0$, imposing that the last Δr columns of $\dot{U}_{\tilde{r}}$ and $\dot{V}_{\tilde{r}}$ are zero, and imposing that the first r columns of $U_{\tilde{r}}$ (resp. $V_{\tilde{r}}$) are U_r (resp. V_r).

The tangent space to $\mathbb{R}^{m\times n}$ is a copy of $\mathbb{R}^{m\times n}$. Let $\xi \in \mathbb{R}^{m\times n}$. Then ξ admits a unique decomposition

$$\xi = \begin{bmatrix}U_r & U_{r\perp}\end{bmatrix}\begin{bmatrix}A & B\\ C & E\end{bmatrix}\begin{bmatrix}V_r^T\\ V_{r\perp}^T\end{bmatrix},$$

and $\eta^* \in \arg\min_{\eta\in \mathrm{T}_X\mathcal{M}_{\leq\tilde{r}}} \|\xi - \eta\|$ if and only if

$$\eta = \begin{bmatrix}U_r & U_{r\perp}\end{bmatrix}\begin{bmatrix}A & B\\ C & \tilde{E}\end{bmatrix}\begin{bmatrix}V_r^T\\ V_{r\perp}^T\end{bmatrix} \tag{12}$$

where $\tilde{E}$ is a best (in the Frobenius norm) rank-$(\tilde{r}-r)$ approximation of E; see [SU15, Corollary 3.3].

One possible choice for the rank-related retraction (Definition 2) is the following one, which relies on the decomposition (11):

$$\tilde{R}_X(\eta) = \tilde{U}_+\tilde{D}_+\tilde{V}_+, \tag{13}$$

where

$$\begin{aligned}\tilde{U}_+ &= (U_{\tilde{r}} + \dot{U}_{\tilde{r}})S_U^{-1},\\ \tilde{V}_+ &= (V_{\tilde{r}} + \dot{V}_{\tilde{r}})S_V^{-1},\\ \tilde{D}_+ &= S_U(D_{\tilde{r}} + \dot{D}_{\tilde{r}})S_V^T,\end{aligned}$$

where S_U and S_V can be chosen freely, e.g., to orthonormalize the factors $\tilde{U}_+$ and $\tilde{V}_+$. This rank-related retraction reduces to the `RRR` retraction of [AO14, §3.4] when $\eta \in \mathrm{T}_X\mathbb{R}_r^{m\times n}$. Other rank-related retractions are given in [Zho15, §4.3.5].

6 Application

In order to illustrate the potential of the rank-adaptive mechanism introduced in Algorithm 3, we present numerical experiments for the well-known weighted low-rank approximation problem:

$$\min_{X \in \mathcal{M}_{\leq k}} \|A - X\|_W^2, \quad f(X) = \|A - X\|_W^2 = \mathrm{vec}\{A - X\}^T W \mathrm{vec}\{A - X\}, \tag{14}$$

where $\mathcal{M} = \mathbb{R}^{m \times n}$, A is given, $W \in \mathbb{R}^{mn \times mn}$ is a positive definite symmetric weighting matrix, and $\mathrm{vec}\{A\}$ denotes the vectorized form of A, i.e., a vector constructed by stacking the consecutive columns of A in one vector. This problem has several applications, notably in machine learning [SJ03].

In our experiments, the matrix A is generated as $A_1 A_2^T \in \mathbb{R}^{m \times n}$, where $A_1 \in \mathbb{R}^{m \times r}$ and $A_2 \in \mathbb{R}^{n \times r}$ are drawn from the standard Gaussian distribution. The weighting matrix W is generated as $W = U \Sigma U^T$, where U is obtained by orthonormalizing a matrix drawn from the standard Gaussian distribution and Σ is a diagonal matrix whose diagonal is a vector of logarithmically spaced points between 10^{-2} and 1 multiplied element-wise by a vector drawn from the uniform distribution on $[0.5, 1.5]$. We take $m = 100$, $n = 15$, $r = 5$. Three values are considered for the rank bound k, one less than the true rank r, one equal to the true rank, and one greater than the true rank.

Four algorithms are compared: DMM [BM06], SULS [SU15], APM [LPW97], and RRAM (Algorithm 3). We use the publicly available Matlab implementation of APM[2] and our own implementation of the other algorithms.[3] In RRAM, the Riemannian update (line 3 of Algorithm 3) is performed by means of the same Riemannian steepest-descent scheme as in SULS.

The initial iterate of RRAM and SULS is taken as $U_0 \Sigma_0 V_0^T$, where U_0, resp. V_0, is obtained by orthonormalizing a matrix of size $m \times k$, resp. $n \times k$, drawn from the standard Gaussian distribution, and Σ_0 is a diagonal matrix with entries drawn from the uniform distribution on $[0, 1]$. The initial points of DMM and APM are randomly generated n-by-$(n - k)$ and m-by-$(m - k)$ matrices respectively. RRAM (Algorithm 3) is run with $\epsilon_1 = \sqrt{3}$, $\epsilon_2 = 10^{-4}$, $\epsilon_4 = \frac{\epsilon_1}{2}$, $\Delta_0 = 10^{-2}$.

The results shown in Table 1 are the average of 10 runs for different data matrices A, weighting matrices W, and initial points. All the results are obtained with Matlab version 8.3.0 (R2014a) for Linux on a platform with Intel(R) Core(TM) i7-4770 CPU at 3.4 GHz with 16GB memory.

For $k = 3 < r$ and $k = 5 = r$, the rank of the iterates of RRAM remains equal to k, hence it performs very similarly to SULS. Observe that these two algorithms clearly outperform the two others.

For $k = 10 > r$, the rank update mechanism of RRAM reduces the rank of the iterates from k to r. In view of the chosen stopping criterion, SULS and RRAM terminate with a comparable accuracy, but RRAM is considerably faster. The effect of the rank-adaptive mechanism as the iteration proceeds is visible on Figure 3.

[2] `ftp://ftp.esat.kuleuven.be/sista/markovsky/abstracts/04-220.html`

[3] `http://sites.uclouvain.be/absil/2015.05`

k	method	rank	f	Relative Error	time(sec)
k = 3	RRAM	3.0	1.833_{+02}	3.410_{-01}	9.271_{-01}
	SULS	3.0	1.833_{+02}	3.410_{-01}	9.161_{-01}
	DMM	3.0	1.833_{+02}	3.410_{-01}	2.566_{+00}
	APM	3.0	1.822_{+02}	3.401_{-01}	1.836_{+00}
k = 5	RRAM	5.0	6.752_{-12}	6.751_{-08}	3.665_{-01}
	SULS	5.0	6.752_{-12}	6.751_{-08}	3.507_{-01}
	DMM	5.0	4.432_{-11}	9.460_{-08}	2.262_{+00}
	APM	5.0	3.740_{-09}	1.439_{-06}	1.166_{+00}
k = 10	RRAM	5.0 (10/10)	6.434_{-12}	6.345_{-08}	4.733_{-01}
	SULS	10.0 (0/10)	9.483_{-12}	7.704_{-08}	9.264_{-01}
	DMM	10.0 (0/10)	3.798_{-11}	8.657_{-08}	1.034_{+00}
	APM	10.0 (0/10)	9.623_{-09}	2.270_{-06}	2.946_{+00}

Table 1: Method comparisons. The number in the parenthesis indicates the fraction of experiments where the numerical rank (number of singular values greater than 10^{-8}) found by the algorithm equals the true rank. The subscript $\pm n$ indicates a scale of $10^{\pm n}$. RRAM (Algorithm 3) and SULS are stopped when the norm of the final gradient on the fixed-rank manifold over the norm of initial full gradient is less than 10^{-7} while DMM and APM are stopped when the norm of final gradient over the norm of initial gradient is less than 10^{-7}.

7 Conclusions and perspectives

In this paper, we have proposed a new algorithm (Algorithm 3) for minimizing a real-valued function on a manifold $\mathcal{M}$ under an additional rank inequality constraint. Rank update mechanisms, based on certain rank-related objects, have been defined to facilitate efficiently finding a suitable rank. Instances of those objects have been provided for the case where $\mathcal{M}$ is the matrix space $\mathbb{R}^{m\times n}$. They can readily be adapted to the case where $\mathcal{M}$ is the Frobenius sphere, and the way is open for handling other cases. The convergence properties of the algorithm have been analyzed, and it has been shown to outperform state-of-the-art methods on a weighted low-rank approximation problem.

This work opens several avenues for further research, e.g., to improve the efficiency of Algorithm 3, set its parameters, strengthen its convergence analysis, investigate the use of other Riemannian optimization algorithms for the Riemannian update (line 3), and test the method on other low-rank optimization problems.

References

[AAM14] P.-A. Absil, Luca Amodei, and Gilles Meyer. Two Newton methods on the manifold of fixed-rank matrices endowed with Riemannian quotient geometries. *Computational Statistics*, 29(3-4):569–590, 2014. doi:10.1007/s00180-013-0441-6.

[ABEV09] Jacob Abernethy, Francis Bach, Theodoros Evgeniou, and Jean-Philippe Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10:803–826, 2009. URL: http://www.jmlr.org/papers/v10/abernethy09a.html.

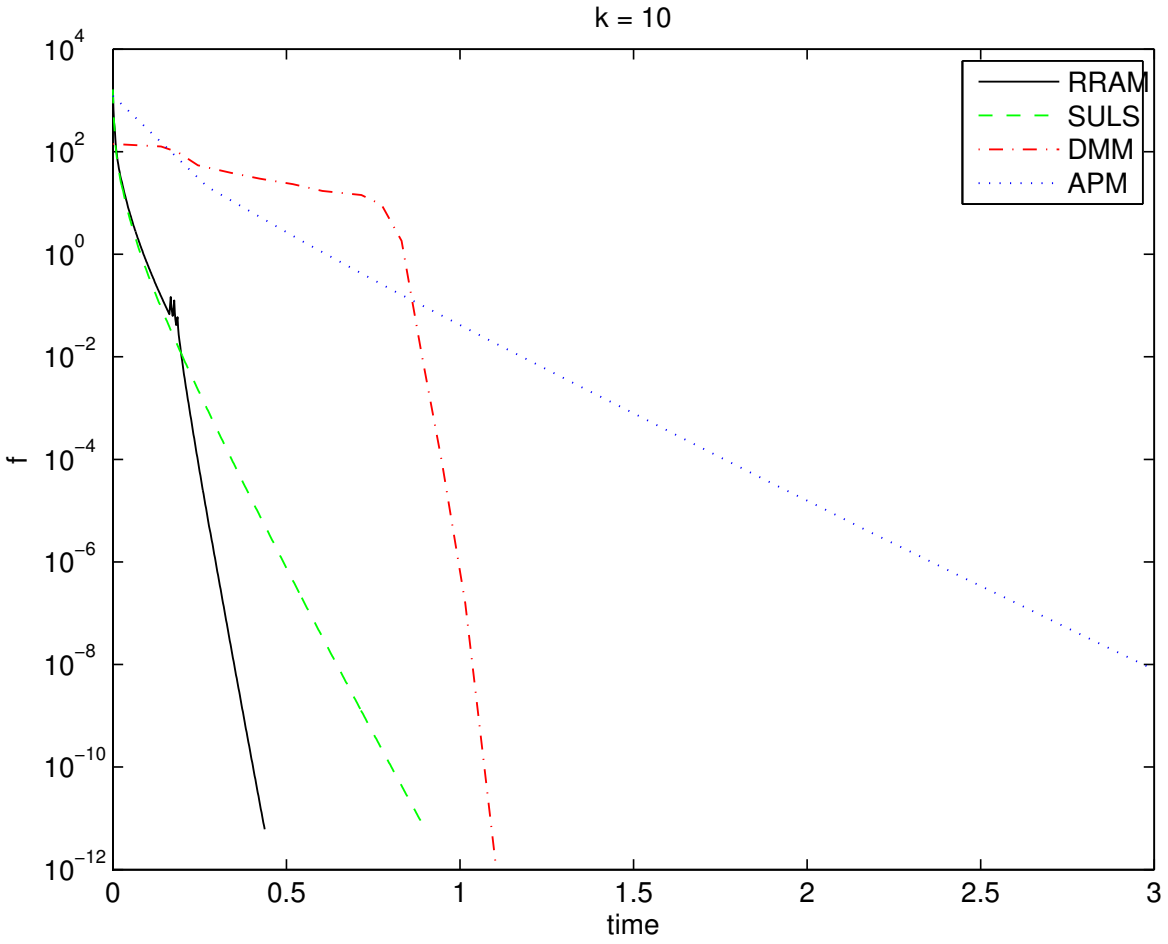

Figure 3: Objective function value against computational time with k is greater than true rank.

[ADM+02] Roy L. Adler, Jean-Pierre Dedieu, Joseph Y. Margulies, Marco Martens, and Michael Shub. Newton's method on Riemannian manifolds and a geometric model for the human spine. *IMA Journal of Numerical Analysis*, 22(3):359–390, 2002.

[AFSU07] Yonatan Amit, Michael Fink, Nathan Srebro, and Shimon Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 17–24, New York, NY, USA, 2007. ACM. URL: `http://doi.acm.org/10.1145/1273496.1273499`, `doi:10.1145/1273496.1273499`.

[AG09] P.-A. Absil and K. A. Gallivan. Accelerated line-search and trust-region methods. *SIAM Journal on Numerical Analysis*, 47(2):997–1018, 2009.

[AMS08] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, 2008.

[AO14] P.-A. Absil and I. V. Oseledets. Low-rank retractions: a survey and new results. *Computational Optimization and Applications*, 62(1):5–29, 2014. URL: http://sites.uclouvain.be/absil/2013.04, doi:10.1007/s10589-014-9714-4.

[BM06] I. Brace and J. H. Manton. An improved BFGS-on-manifold algorithm for computing low-rank weighted approximations. In *Proceedings of 17th International Symposium on Mathematical Theory of Networks and Systems*, pages 1735–1738, 2006.

[BMAS14] Nicolas Boumal, Bamdev Mishra, P.-A. Absil, and Rodolphe Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15:1455–1459, 2014. URL: http://jmlr.org/papers/v15/boumal14a.html.

[CA15] Léopold Cambier and P.-A. Absil. Robust low-rank matrix completion by Riemannian optimization. Technical Report UCL-INMA-2015.04, UCLouvain, 2015. URL: http://sites.uclouvain.be/absil/2015.04.

[CAV13] T. P. Cason, P.-A. Absil, and P. Van Dooren. Iterative methods for low rank approximation of graph similarity matrices. *Linear Algebra and its Applications*, 438(4):1863 – 1882, 2013. 16th ILAS Conference Proceedings, Pisa 2010. doi:10.1016/j.laa.2011.12.004.

[FPST13] Maryam Fazel, Ting Kei Pong, Defeng Sun, and Paul Tseng. Hankel matrix rank minimization with applications to system identification and realization. *SIAM Journal on Matrix Analysis and Applications*, 34(3):946–977, 2013. doi:10.1137/110853996.

[HAG14] Wen Huang, P.-A. Absil, and K. A. Gallivan. A Riemannian symmetric rank-one trust-region method. *Mathematical Programming, Series A*, 150(2):179–216, 2014. URL: http://sites.uclouvain.be/absil/2013.03, doi:10.1007/s10107-014-0765-1.

[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. URL: http://sites.uclouvain.be/absil/2014.01, doi:10.1137/140955483.

[HM94] Uwe Helmke and John B. Moore. *Optimization and Dynamical Systems*. Communications and Control Engineering Series. Springer-Verlag London Ltd., London, 1994. With a foreword by R. Brockett.

[LLY+13] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, Jan 2013. doi:10.1109/TPAMI.2012.88.

[LPW97] W.-S. Lu, S.-C. Pei, and P.-H. Wang. Weighted low-rank approximation of general complex matrices and its application in the design of 2-d digital filters. *IEEE Transactions on Circuits and Systems I*, 44:650–655, 1997.

[Mar12] I. Markovsky. *Low Rank Approximation: Algorithms, Implementation, Applications*. Springer, 2012. URL: http://homepages.vub.ac.be/~imarkovs/book.html, doi:10.1007/978-1-4471-2227-2.

[MMBS13a] B. Mishra, G. Meyer, F. Bach, and R. Sepulchre. Low-rank optimization with trace norm penalty. *SIAM Journal on Optimization*, 23(4):2124–2149, 2013. URL: http://dx.doi.org/10.1137/110859646, arXiv:http://dx.doi.org/10.1137/110859646, doi:10.1137/110859646.

[MMBS13b] B. Mishra, G. Meyer, F. Bach, and R. Sepulchre. Low-rank optimization with trace norm penalty. *SIAM Journal on Optimization*, 23(4):2124–2149, 2013. doi:10.1137/110859646.

[MMBS14] Bamdev Mishra, Gilles Meyer, Silvère Bonnabel, and Rodolphe Sepulchre. Fixed-rank matrix factorizations and Riemannian low-rank optimization. *Computational Statistics*, 29(3-4):591–621, 2014. URL: http://dx.doi.org/10.1007/s00180-013-0464-z, doi:10.1007/s00180-013-0464-z.

[NW06] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer series in operations research and financial engineering. Springer, New York, NY, 2. ed. edition, 2006.

[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.

[Sat14] Hiroyuki Sato. A Dai-Yuan-type Riemannian conjugate gradient method with the weak Wolfe conditions, 2014. arXiv:1405.4371.

[SHK13] Florian Seidel, Clemens Hage, and Martin Kleinsteuber. pROST: a smoothed lp-norm robust online subspace tracking method for background subtraction in video. *Machine Vision and Applications*, 25(5):1227–1240, 2013. doi:10.1007/s00138-013-0555-4.

[SJ03] Nathan Srebro and Tommi Jaakkola. Weighted low-rank approximations. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, 2003.

[SU15] Reinhold Schneider and André Uschmajew. Convergence results for projected line-search methods on varieties of low-rank matrices via Łojasiewicz inequality. *SIAM Journal on Optimization*, 25(1):622–646, 2015. arXiv:1402.5284, doi:10.1137/140957822.

[SWC12] Uri Shalit, Daphna Weinshall, and Gal Chechik. Online learning in the embedded manifold of low-rank matrices. *Journal of Machine Learning Research*, 13(1):429–458, February 2012. URL: http://dl.acm.org/citation.cfm?id=2503308.2188399.

[UV14] A. Uschmajew and B. Vandereycken. Line-search methods and rank increase on low-rank matrix varieties. In *Proceedings of the 2014 International Symposium on Nonlinear Theory and its Applications (NOLTA2014)*, 2014.

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

[YYS15] Qi Yan, Jieping Ye, and Xiaotong Shen. Simultaneous pursuit of sparseness and rank structures for matrix decomposition. *Journal of Machine Learning Research*, 16:47–75, 2015. URL: `http://jmlr.org/papers/v16/yan15a.html`.

[Zho15] Guifang Zhou. *Rank-contrained optimization: A Riemannian manifold approach.* PhD thesis, Florida State University, 2015.