Journal of Machine Learning Research 25 (2024) 1-52 Submitted 1/23; Revised 10/24; Published 12/24

Accelerating Nuclear-norm Regularized Low-rank Matrix Optimization Through Burer-Monteiro Decomposition

Ching-pei Lee CHINGPEI@ISM.AC.JP
Institute of Statistical Mathematics
10-3 Midori-cho, Tachikawa
Tokyo 190-8562, Japan

Ling Liang LIANG.LING@U.NUS.EDU
Department of Mathematics
University of Maryland at College Park
4176 Campus Drive, College Park, MD, USA 20742

Tianyun Tang TTANG@U.NUS.EDU
Institute of Operations Research and Analytics
National University of Singapore
10 Lower Kent Ridge Road, Singapore 119076

Kim-Chuan Toh MATTOHKC@NUS.EDU.SG
Department of Mathematics and Institute of Operations Research and Analytics
National University of Singapore
10 Lower Kent Ridge Road, Singapore 119076

Editor: Prateek Jain

Abstract

This work proposes a rapid algorithm, BM-Global, for nuclear-norm-regularized convex and low-rank matrix optimization problems. BM-Global efficiently decreases the objective value via low-cost steps leveraging the nonconvex but smooth Burer-Monteiro (BM) decomposition, while effectively escapes saddle points and spurious local minima ubiquitous in the BM form to obtain guarantees of fast convergence rates to the global optima of the original nuclear-norm-regularized problem through aperiodic inexact proximal gradient steps on it. The proposed approach adaptively adjusts the rank for the BM decomposition and can provably identify an optimal rank for the BM decomposition problem automatically in the course of optimization through tools of manifold identification. BM-Global hence also spends significantly less time on parameter tuning than existing matrix-factorization methods, which require an exhaustive search for finding this optimal rank. Extensive experiments on real-world large-scale problems of recommendation systems, regularized kernel estimation, and molecular conformation confirm that BM-Global can indeed effectively escapes spurious local minima at which existing BM approaches are stuck, and is a magnitude faster than state-of-the-art algorithms for low-rank matrix optimization problems involving a nuclear-norm regularizer. Based on this research, we have released an open-source package of the proposed BM-Global at `https://www.github.com/leepei/BM-Global/`.

Keywords: escaping spurious local minima, low-rank models, Burer-Monteiro decomposition, nuclear-norm regularization, nonconvex optimization

1. Introduction

Consider the following regularized convex matrix optimization problem

$$\min_{X\in\mathbb{R}^{m\times n}} \quad F(X) \coloneqq f(X) + \Psi(X), \tag{CVX}$$

where the loss term $f : \mathbb{R}^{m\times n} \to \mathbb{R}$ is lower-bounded, convex, and differentiable with Lipschitz-continuous gradient, and the regularizer $\Psi : \mathbb{R}^{m\times n} \to \mathbb{R} \cup \{+\infty\}$ is convex in the whole space $\mathbb{R}^{m\times n}$ and has the form

$$\Psi(X) \coloneqq \lambda\|X\|_* + \delta_{\mathcal{X}}(X), \quad X \in \mathbb{R}^{m\times n}, \tag{1}$$

where $\lambda \in \mathbb{R} \setminus \{0\}$, $\|\cdot\|_*$ is the nuclear norm, $\mathcal{X}$ is a closed and convex subset of $\mathbb{R}^{m\times n}$, and $\delta_{\mathcal{X}}$ is the indicator function such that

$$\delta_{\mathcal{X}}(X) = \begin{cases} 0 & \text{if } X \in \mathcal{X}, \\ \infty & \text{otherwise.} \end{cases} \tag{2}$$

Clearly, in this case, Ψ is nonsmooth, proper, and closed. When $\lambda \geq 0$, we see directly that Ψ is indeed convex. In scenarios with $\lambda < 0$, the problems of interest are also equipped with a suitable $\mathcal{X}$ that makes Ψ convex.[1] Without loss of generality, we assume that $m \leq n$ throughout, which can be achieved easily by conducting a matrix transpose if necessary. For our case of (1), as long as λ is properly selected, a low-rank optimal solution to (CVX) exists, since the nuclear norm is exactly applying the ℓ_1-norm to the singular values of the given matrix. In practice, singular value decomposition (SVD) for a non-symmetric matrix X is calculated through the eigendecomposition of the symmetric matrix $XX^\top$ (as we assume $m \leq n$), and thus computation of SVDs and of eigendecompositions are nearly identical. We will therefore summarize these two situations simply as requiring eigendecompositions.

We focus on large-scale problems such that mn is extremely large, so forming a (possibly dense) matrix $X \in \mathbb{R}^{m\times n}$ explicitly is spatially and computationally expensive, if not infeasible, and thus a low-rank solution is necessary for practical reasons. The nuclear-norm regularization hence serves to induce a low-rank structure in any solution X^* to (CVX). On the other hand, we assume that $\nabla f(X)$ is either structured or extremely sparse so that $\nabla f(X)v$ for any vector v can be computed efficiently; see Section 5 for some examples of sparse ∇f. This is necessary for the execution of an inexact proximal gradient (PG) step; see details in Section 3.

To deal with the high problem dimensionality in (CVX), a popular approach is the Burer-Monteiro (BM) decomposition (Burer and Monteiro, 2003) that explicitly writes X as a product of two low-rank matrices with a pre-specified rank $k \leq m$. Explicitly, we get

$$\min_{W\in\mathbb{R}^{m\times k}, H\in\mathbb{R}^{n\times k}} \quad F(W,H) \coloneqq f\left(WH^\top\right) + \Psi\left(WH^\top\right). \tag{BM}$$

1. When $m = n$ and $\mathcal{X}$ is a subset of $\mathbb{S}^n_+$, the set of n by n symmetric positive semidefinite matrices, Ψ is convex (in the whole space) even if $\lambda < 0$, as in that case the nuclear norm becomes the trace of X within $\mathcal{X}$, which is an affine function of X, and the feasible region $\mathcal{X}$ is a convex set. Otherwise, we will still need $\lambda \geq 0$ to make Ψ convex.

The spatial cost of $O(mn)$ for storing X in (CVX) is then reduced to $O((m+n)k)$ in (BM). Numerous efficient algorithms for solving (BM) are therefore proposed.

Among applications of (CVX), one of the most prominent example, and also our motivating problem, is the following low-rank matrix completion problem whose target is to recover the whole ground truth matrix $A \in \mathbb{R}^{m\times n}$ from its observed entries enumerated by an index set Ω:

$$\min_{X\in\mathbb{R}^{m\times n}} \quad \frac{1}{2}\|P_\Omega(X-A)\|_F^2 + \lambda\|X\|_*, \tag{MC}$$

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

$$\left(P_\Omega\left(A\right)\right)_{i,j} = \begin{cases} A_{i,j} & \text{if } (i,j)\in\Omega \\ 0 & \text{otherwise} \end{cases}.$$

It is well-known that the factorized form in (BM) of (MC) for a given rank k is

$$\min_{W\in\mathbb{R}^{m\times k}, H\in\mathbb{R}^{n\times k}} \quad \frac{1}{2}\|P_\Omega(WH^\top - A)\|_F^2 + \frac{\lambda}{2}(\|W\|_F^2 + \|H\|_F^2), \tag{MF}$$

which is often called the matrix factorization (MF) problem in the machine learning community. One can easily show that the global optimal objectives of (MC) and (MF) coincide whenever k is large enough such that there is at least one optimal solution X^* of (MC) with $\mathrm{rank}(X^*) \le k$. (See Lemma 1.) Apparently, even the objective evaluation for (MC) requires an eigendecomposition that costs $O(m^3)$, while for (MF), objective evaluation and variable updates all require cost of only $O((m+n)k)$.

Unfortunately, the lower computational and spatial costs of solving (MF) and thus the more general (BM) comes with a price. A disadvantage of the BM method is that the rank k needs to be specified in advance, and a good value for k can be hard to estimate a priori. Another even more severe issue is that the problem (MF) is a nonconvex one, meaning that algorithms for solving it could get stuck at spurious local optima (local but not global optima) or saddle points (*i.e.*, points with zero gradients that are not local extrema). Such points can give terrible performance for predicting missing entries. The simplest example would be that for any $k > 0$, letting W and H be matrices of all zeros in (MF) will directly generate a saddle point, but this clearly is not a solution in general. It is also recently shown by Yalcin et al. (2022); O'Carroll et al. (2022) that there are indeed problems with a sufficiently large k that still possess spurious local minima with a terribly large objective value, and thus how to escape from such saddle points and spurious local minima is a critical issue for the BM approach to produce satisfactory performance.

On the other hand, the convex problem (MC) or (CVX) can be solved directly through PG-type algorithms that are able to find the global optima (Toh and Yun, 2010), but the cost of the eigendecomposition in the proximal operation is extremely expensive, even if only a subset of singular values/vectors is required. State-of-the-art methods for (MC) like those by Hsieh and Olsen (2014); Yao and Kwok (2018) thus resort to approximate eigendecompositions computed through the power method to reduce the time cost of eigendecompositions. The power method also has the benefit that only XV for some thin matrix V is needed at each iteration, so explicit computation of X is not needed. However, we observe that in practice, usually the convex approaches based on PG tend to be rather slow

because their computational cost per iteration is still not comparable to those for (MF) or (BM).

In this work, we propose a highly-efficient algorithm, BM-Global, that combines the advantages of both approaches with theoretical guarantees. Our method fully utilizes the computational and spatial advantages of (BM) to have a running time similar to the state of the art for (BM). Meanwhile, BM-Global also possesses guarantees for convergence to the global optima just like those approaches for solving (CVX). With suitable inexactness conditions for the PG steps, we also obtain a sublinear convergence rate for the general convex case, and further get faster rates when the objective function satisfies the Kurdyka-Łojasiewicz (KL) condition (Kurdyka, 1998; Łojasiewicz, 1963). Our algorithm attains these appealing guarantees through sporadically resorting to convex lifting steps that conduct one iteration of inexact PG on (CVX) to escape from saddle points and spurious local optima at which existing methods for (BM) got stuck at. We emphasize that for alternating between inexact PG and other update steps, our convergence and rate guarantees are novel, as existing analyses for convergence and rates of inexact PG rely on geometrical properties of the PG iterates that will be destroyed when other updates are inserted.

Through the techniques of manifold identification (Hare and Lewis, 2004; Lewis and Zhang, 2013) in our analysis, another major and novel contribution of this work is that *the optimal rank for* (BM) *will provably be found by the proposed method automatically*, so no additional parameter tuning for the optimization side is required for attaining satisfactory results for practical applications of (CVX). Numerical results also show that our method is significantly faster than state-of-the-art solvers for (CVX), and can effectively escape from saddle points and spurious local minima of (BM).

1.1 Related Works

Methods for (CVX). The convex problem (CVX) falls in the category of regularized optimization, and many efficient algorithms in the literature are available. However, most methods for regularized optimization concentrate on the scenario that the proximal operation can be conducted efficiently but obtaining information of the smooth term is the major computational bottleneck, which is apparently not the case for (CVX). Practical methods specifically designed for (CVX) all take into serious account the expensive SVD involved in the nuclear norm (Toh and Yun, 2010; Hsieh and Olsen, 2014; Yao and Kwok, 2018), and they all focus on the most popular setting that f is a quadratic term. In this case, high-order methods like proximal (quasi-)Newton is useless because the subproblem has the same form as the original problem itself. Therefore, these works all consider first-order methods such as the (inexact) PG or accelerated PG (APG) methods (Nesterov, 2013; Beck and Teboulle, 2009a,b). Toh and Yun (2010) used the Lanczos method to conduct approximate SVD, and Hsieh and Olsen (2014) proposed to apply the power method for approximate SVD and to use the rank k decided by their inexact PG method to conduct another convex optimization step with respect to a subproblem of dimension $k \times k$ after each PG step. The power method in Hsieh and Olsen (2014) effectively uses the current iterate as warmstart and is much more efficient than the Lanczos approach in Toh and Yun (2010), but the additional convex optimization step turns out to be time-consuming. To improve the efficiency

of Toh and Yun (2010) and Hsieh and Olsen (2014), Yao and Kwok (2018) combined the two approaches to turn to inexact APG using the power method.

More recently, motivated by PG's ability of manifold identification, Bareilles et al. (2022) proposed to alternate between an exact PG step and a Riemannian (truncated) Newton step on the currently identified manifold for general regularized optimization, and applied this method to a toy problem of (CVX) in their experiments. Different from ours, their usage of manifold identification is for showing that their method could obtain superlinear convergence, but their algorithm is not feasible for large-scale problems considered in this work because they considered exact PG only and required the explicit computation of X_t.

Convergence of Inexact Proximal Gradient The global convergence of our method is achieved through the safeguard of inexact PG steps, but we do not require the inexact PG step to always decrease the objective value. This feature combined with the BM phase makes the analysis difficult. Existing analyses for inexact PG Combettes (2004); Schmidt et al. (2011); Jiang et al. (2012); Hamadouche et al. (2022) utilize telescope sums of inequalities in the form of $\|X_{t+1} - X^*\|^2 \leq$ error term$+\|X_t - X^*\|^2 - \alpha\left(F(X_t) - F(X^*)\right)$ for any $X^* \in \Omega^*$ and some $\alpha > 0$ to prove convergence and rates. Therefore, those approaches cannot allow for alternating between inexact PG and other steps because that will nullify the technique of telescope sums. On the other hand, analyses compatible with other steps like those in Scheinberg and Tang (2016); Bonettini et al. (2016); Lee and Wright (2019) require strict decreasing of the objective in the inexact PG step (either explicitly or implicitly), which imposes an additional burden.

The work of Yang et al. (2022) that applies inexact PG to the semidefinite programming (SDP) relaxation of polynomial optimization problems is probably the closest to our approach in that their method alternates between nonmonotone inexact PG and an alternative step. However, their alternative step is only accepted when it decreases the objective by an absolute amount $\epsilon > 0$, so eventually the alternative steps are always rejected when the objective converges, but we do not have such restrictions. Our analysis also provides more comprehensive convergence guarantees under different conditions as well as identification of the optimal rank using techniques totally different from that of Yang et al. (2022).

Methods for escaping saddle points. There has recently been a thriving interest in studying smooth optimization methods that can escape strict saddle points with at least one negative eigenvalue in the Hessian (Lee et al., 2019; Jin et al., 2017; Royer and Wright, 2018; Carmon et al., 2018; Royer et al., 2020; Agarwal et al., 2017). However, these methods are unable to deal with degenerate saddle points where the smallest eigenvalue of the Hessian is exactly zero, and neither could they deal with spurious local minima that might be arbitrarily far away from the global ones. On the other hand, our method can handle all such difficult cases appearing in (BM) by resorting to convex lifting, and Lemma 1 together with Theorem 3 show that our method indeed will converge to the global optima. Moreover, existing methods for escaping strict saddles are mainly of theoretical interest, and their empirical performance is usually not very impressive, while our method is designed for practical large-scale usage and greatly outperforms state-of-the-art methods for (CVX) and (BM) by a large margin on real-world data, as we shall see in the numerical experiments later.

Absence of spurious stationary points for (MF). To cope with possible spurious local minima and degenerate saddle points of matrix factorization, there is also a growing interest

in analyzing its optimization landscape. Most such works consider the quadratic loss:

$$f(X) = \|P_\Omega(A - X)\|_F^2. \tag{3}$$

Jain et al. (2013); Ge et al. (2016); Sun and Luo (2016); Ge et al. (2017) confined their analyses to minimizing (3) with deliberately selected regularizers, in the factorized form with variables W and H using the alternating minimization method that alternatively minimizes the objective function with respect to H and to W. They proved the absence of spurious local minima for only the ideal case in which each (i, j) belongs to Ω with a fixed probability $p \in (0, 1]$ and the observations are noiseless. They showed that this approach enjoys appealing theoretical guarantees and fast computation under suitable conditions such as variants of the restricted isometry property and its variants. But these assumptions generally fail in practice, and the selected regularizers are not widely adopted. In particular, for applications like recommendation systems, elements of Ω are already biased selections by an existing system and will never obey the independent random assumption. In addition, real-world data always contain noisy observations and measurement corruption. Chen et al. (2019); Ye and Du (2021) focused on (3) with $\Omega = \{1, \ldots, m\} \times \{1, \ldots, n\}$ only, and their techniques could be hard to generalize to other problems. Different from these works, we do not have any assumption on the underlying data, as it has been shown recently in Yalcin et al. (2022); O'Carroll et al. (2022) that some problem classes of matrix factorization and SDP indeed contain numerous spurious local minima. Moreover, we do not confine our algorithm or analysis to the special case of (3) but aim at general f with minimal assumptions. However, our method can still find the global optima with ultra-high practical efficiency in the presence of non-strict saddle points and spurious local optima.

1.2 Organization

This paper is organized as follows. Preliminaries for our algorithmic development are provided in Section 2. In Section 3, we describe our main algorithmic framework. Section 4 provides a comprehensive theoretical analysis of our algorithm, including its global convergence to the global optima, convergence rates, and identification of the right rank for (BM) within a finite number of iterations. We then give more details in Section 5 several realizations of (CVX) in applications. Numerical experiments on real-world problems in such applications are conducted in Section 6, and finally Section 7 concludes this work. Detailed implementations for the applications used in our numerical experiments and additional experimental results are provided in the appendices starting from Page 37.

2. Preliminaries

This section first lays out our notations in this work, and then provides background knowledge that will facilitate further descriptions in our development of algorithm and theory. We use $\mathrm{tr}(\cdot)$ to denote the trace of a square matrix. For any $x \in \mathbb{R}^m$, $\mathrm{diag}(x)$ is the $m \times m$ diagonal matrix whose diagonal entries are those in x. We use $\langle \cdot, \cdot \rangle$ to denote the standard inner product and $\|\cdot\|$ to denote its induced norm. In particular, for vectors $x, y \in \mathbb{R}^n$, this is the standard inner product such that $\langle x, y \rangle \coloneqq \sum_{i=1}^n x_i y_i$, with the norm being the Euclidean norm, and for matrices $A, B \in \mathbb{R}^{m \times n}$, this inner product is defined as $\langle A, B \rangle \coloneqq \mathrm{tr}\left(A^\top B\right)$,

where $A^\top$ is the transpose of A and the corresponding norm is the Frobenius norm. We denote by $\mathbb{R}^m_+$ the nonnegative orthant in the m-dimensional Euclidean space, by $\mathbb{S}^n$ the set of n by n symmetric real matrices, and by $\mathbb{S}^n_+$ the cone of symmetric positive semidefinite matrices in $\mathbb{S}^n$. I_n denotes the identity matrix with dimension $n \times n$, and $e_n \in \mathbb{R}^n$ is the vector of ones. The subscript n is often omitted when the dimensionality is clear. For $x \in \mathbb{R}^m$, we let $[x]_+$ be its Euclidean projection onto $\mathbb{R}^m_+$, and for $X \in \mathbb{S}^n$, $[X]_+$ is its Euclidean projection onto $\mathbb{S}^n_+$. In particular, if X admits an eigendecomposition $X = U\mathrm{diag}(\Sigma)U^\top$, where $U \in \mathbb{R}^{n\times n}$ is orthonormal such that $UU^\top = U^\top U = I_n$ and $\Sigma \in \mathbb{R}^n$, we have that $[X]_+ = U\mathrm{diag}\left([\Sigma]_+\right)U^\top$. We note that SVDs and eigendecompositions are unique up to permutations of the eigenvalues or the singular values, so we will simply say "the" SVD or "the" eigendecomposition to refer to the one such that the singular values or eigenvalues are sorted in descending order. (And this can be an arbitrary one when there are repeated singular values.) Given any set $\mathcal{X}$, we use $\delta_{\mathcal{X}}$ to denote its indicator function described in (2). For any convex function h, we use ∂h to denote its subdifferential.

Throughout this work, we will heavily use the proximal operation. Given a function Ψ, this operation is defined as

$$\mathrm{prox}_{\Psi}(X) \coloneqq \arg\min_{\hat{X}} \ \frac{1}{2}\|X - \hat{X}\|^2 + \Psi(\hat{X}). \tag{4}$$

For Ψ convex, proper, and closed, it is well-known that (4) is well-defined and single-valued everywhere. When $\Psi = \lambda\|\cdot\|_*$ for some $\lambda > 0$, (4) has a closed-form solution (Lewis and Overton, 1996). Given a matrix $X \in \mathbb{R}^{m\times n}$ with rank k and with its SVD written as $X = U\mathrm{diag}(\Sigma)V^\top$ for some $U \in \mathbb{R}^{m\times k}, V \in \mathbb{R}^{n\times k}$ that are orthogonal and $\Sigma \in \mathbb{R}^k_+$, we have

$$\mathrm{prox}_{\beta\|\cdot\|_*}(X) = U\mathrm{diag}\left([\Sigma - \beta e]_+\right)V^\top \tag{5}$$

for any $\beta > 0$. Similarly, if $X \in \mathbb{S}^n$ admits an eigendecomposition $X = U\mathrm{diag}(\Sigma)U^\top$, we have from Lewis and Overton (1996) that for any $\beta > 0$,

$$\mathrm{prox}_{\beta(\lambda\|\cdot\|_* + \delta_{\mathbb{S}^n_+})}(X) = [X]_+ = U\mathrm{diag}\left([\Sigma - \beta\lambda e]_+\right)U^\top. \tag{6}$$

In this paper, we focus on the two scenarios of $\Psi = \lambda\|\cdot\|_*$ and $\Psi = \lambda\|\cdot\|_* + \delta_{\mathbb{S}^n_+\cap\mathcal{S}}(\cdot)$ (for $m = n$) for some closed and convex set $\mathcal{S}$. We will see in Lemma 1 that the former results in the following form of (BM):

$$\min_{U\in\mathbb{R}^{m\times k}, V\in\mathbb{R}^{n\times k}} \quad \tilde{F}(W,H) \coloneqq f\left(WH^\top\right) + \frac{\lambda}{2}\left(\|W\|_F^2 + \|H\|_F^2\right), \tag{BM-nuclear}$$

while the latter leads to

$$\min_{W\in\mathbb{R}^{n\times k}} \quad \tilde{F}(W) \coloneqq f\left(WW^\top\right) + \lambda\|W\|_F^2 + \delta_{\mathcal{S}}\left(WW^\top\right). \tag{BM-PSD}$$

For the latter case, when we deal with (CVX), for easier calculation, we will sometimes consider the smooth term as $\tilde{f}(X) \coloneqq f(X) + \lambda\langle I, X\rangle$ and the regularizer as $\tilde{\Psi}(X) \coloneqq \delta_{\mathbb{S}^n_+\cap\mathcal{S}}(X)$ instead, which is equivalent to the original problem because the nuclear norm on a positive semidefinite matrix is exactly the trace of the same matrix.

The equivalence between the nuclear norm in (CVX) and the Frobenius norm squared in (BM-nuclear) and (BM-PSD) is formally stated in the following lemma.

Lemma 1 *Given any $X \in \mathbb{R}^{m\times n}$, we have*

$$\|X\|_* = \min_{W,H:WH^\top = X} \quad \frac{1}{2}\left(\|W\|_F^2 + \|H\|_F^2\right). \tag{7}$$

Moreover, if the SVD of X is

$$X = USV^\top = \sum_{i=1}^{k} \sigma_i u_i v_i^\top,$$

where $S = \mathrm{diag}(\sigma_1, \ldots, \sigma_k)$ and $\sigma_i > 0$ are the singular values, $k = \mathrm{rank}(X)$, $U = [u_1, \ldots, u_k] \in \mathbb{R}^{m\times k}$ and $V = [v_1, \ldots, v_k] \in \mathbb{R}^{n\times k}$ are both orthonormal, the minima of the right-hand side of (7) are exactly those

$$\hat{W} \coloneqq \left[\sqrt{\sigma_{\tau(1)}}u_{\tau(1)}, \ldots, \sqrt{\sigma_{\tau(k)}}u_{\tau(k)}\right], \hat{H} \coloneqq \left[\sqrt{\sigma_{\tau(1)}}v_{\tau(1)}, \ldots, \sqrt{\sigma_{\tau(k)}}v_{\tau(k)}\right], \tag{8}$$

where τ is any given permutation of $\{1, \ldots, k\}$. Therefore, for any global optimum (W^*, H^*) of (BM-nuclear)*, $X^* \coloneqq W^*(H^*)^\top$ is also a global optimum to* (CVX) *with $\Psi = \lambda\|\cdot\|_*$, provided that there is a global optimum $\hat{X}$ of* (CVX) *with $\mathrm{rank}(\hat{X}) \le k$, and for any optimal solution X^* of* (CVX) *with SVD $X^* = U^*\Sigma^*(V^*)^\top$,*

$$W^* = U^*(\Sigma^*)^{\frac{1}{2}}, \quad H^* = V^*(\Sigma^*)^{\frac{1}{2}}$$

form a global solution of (BM-nuclear) *for any $k \ge \mathrm{rank}(X^*)$. Likewise, for any global optimum W^* of* (BM-PSD)*, $X^* \coloneqq W^*(W^*)^\top$ is also a global optimum to* (CVX) *with $\Psi = \lambda\|\cdot\|_* + \delta_{\mathbb{S}^n_+}(\cdot) + \delta_{\mathcal{S}}(\cdot)$, provided that there is a global optimum $\hat{X}$ of* (CVX) *with $\mathrm{rank}(\hat{X}) \le k$.*

(8) is directly from Rennie and Srebro (2005), and the rest of Lemma 1 are also well-known.

3. Algorithmic framework

In this section, we present a detailed description of the proposed BM-Global that can be split into two phases: the BM phase and the convex lifting phase. The high-level idea of the proposed framework is to fully utilize the efficiency in solving (BM) from the smoothness of the objective function whenever possible. When only limited progress can be further made in the BM phase with the current iterate $(\tilde{W}_t, \tilde{H}_t)$, we turn to the convex lifting phase, which conducts one step of inexact proximal gradient (PG) on (CVX) from the iterate $\tilde{X}_t = \tilde{W}_t\tilde{H}_t^\top$. In the inexact PG step, an approximate eigendecomposition algorithm is employed to obtain the next iterate $X_{t+1} = W_{t+1}H_{t+1}^\top$. The rank of the approximate eigendecomposition is dynamically increased (and the proximal step decreases the rank of its output) to guarantee that the correct rank at the optimum can be found within finite iterations of our algorithm.

(8) provides a convenient way to transform an iterate of (CVX) to that of (BM-nuclear) or (BM-PSD). (Transforming the other way round is straightforward.) Although this transformation requires an eigendecomposition, we will see shortly that our convex lifting step will generate the exact eigendecomposition of its output (which is obtained from conducting

an exact proximal operation on an approximate eigendecomposition of the input matrix), so the transformation can be done almost for free. We emphasize that the matrix $X = WH^\top$ is never explicitly formed when executing BM-Global.

The only requirement we put on the BM phase is a mild and implementable nonincreasing objective condition:

$$\tilde{F}(\tilde{W}_t, \tilde{H}_t) \le \tilde{F}(W_t, H_t) \quad \Leftrightarrow \quad F(\tilde{W}_t\tilde{H}_t^\top) \le F(W_t H_t^\top). \tag{9}$$

With this minimum requirement, many suitable solvers for the BM-phase subproblem can be applied at the user's will, and so is the stopping condition for the selected solver. We can even skip the BM phase from time to time without violating (9) when this skipping is deemed useful.

In the convex lifting phase, we emphasize that since we never explicitly store the dense matrix variable X due to the high spatial cost, exact decomposition for computing the eigendecomposition becomes impractical. This is one of the major reasons to consider approximate eigendecompositions that are computed through an iterative process where each iteration of which only requires the computation of matrix-matrix products of the form XV for some thin matrix V. This product XV can be computed efficiently without explicitly forming X if X can be decomposed into the sum of a low-rank matrix and a highly sparse matrix. This is another reason to consider low-rank problems promoted by the nuclear norm, as the proximal operation of the nuclear norm often leads to a low-rank iterate. See more details in (5) and (6), and Theorem 7 in Section 4.

Although APG is state of the art for (CVX), such as the works of Toh and Yun (2010); Yao and Kwok (2018) that utilized the APG method to obtain theoretical and practical convergence faster than that of the PG method, APG is not applicable in our framework because when we insert other update steps between two APG iterations, existing proofs for convergence guarantees of APG are invalidated. We also note that for the error-bound condition considered in (47), PG could achieve faster convergence rates (see Theorem 5) than APG. Moreover, notwithstanding using a vanilla PG method in our algorithm results in a slower worst-case convergence rate than the APG method, our major workhorse in reducing the objective value efficiently is actually the BM phase, while the PG step mainly serves as a safeguard for global convergence and the mechanism for identifying the correct rank. Therefore, we do not expect the PG step to provide much objective decrease empirically. In the numerical experiments, we will also see that the added BM phase indeed effectively decreases the objective value with a short running time, making the proposed algorithm outperform the APG method of Toh and Yun (2010); Yao and Kwok (2018) significantly.

3.1 Inexact Proximal Gradient Step

Given the iterate $\tilde{X}_t$ and a step size $\alpha_t > 0$ at the tth iteration, the exact PG step $\hat{X}_t^+(\alpha_t)$ is computed by

$$\begin{aligned}\hat{X}_t^+(\alpha_t) &= \operatorname{prox}_{\alpha_t\Psi}(\tilde{X}_t - \alpha_t\nabla f(\tilde{X}_t)) \\ &= \arg\min_Y \left\{ Q_t^{\alpha_t}(Y) \coloneqq \langle \nabla f(\tilde{X}_t), Y - \tilde{X}_t\rangle + \frac{\|Y - \tilde{X}_t\|_F^2}{2\alpha_t} + \Psi(Y) - \Psi(\tilde{X}_t) \right\}.\end{aligned} \tag{10}$$

After finding a suitable α_t ensuring a sufficient decrease in F, the exact PG method then assigns $X_{t+1} = \hat{X}_t^+(\alpha_t)$. For our inexact scheme, we focus on the scheme that the computation of $\nabla f(\tilde{X}_t)$ and the proximal operation in (5) or (6) are exact, and the inexactness in the PG step comes from the approximate eigendecomposition. In particular, the approximate eigendecomposition is the exact eigendecomposition of a matrix approximating the original one, and thus we can easily conduct exact eigenvalue/singular value truncation in (5) or (6) of this approximation matrix.

We can therefore view the calculation of our inexact PG step with such an inexactness quantified by some $\epsilon_t \geq 0$ as

$$X_{t+1} = X_t^+(\alpha_t) = \mathrm{prox}_{\alpha_t\Psi}\left(\tilde{Z}_t\right) = \arg\min_{\hat{Y}} \langle \nabla f(\tilde{X}_t) + \mathcal{E}_t,\, \hat{Y} - \tilde{X}_t\rangle + \frac{1}{2\alpha_t}\left\|\hat{Y} - \tilde{X}_t\right\|_F^2 + \Psi(\hat{Y}), \tag{11}$$

for some $\mathcal{E}_t$ and $\tilde{Z}_t$ that satisfy

$$\|\mathcal{E}_t\|_F \leq \epsilon_t, \quad \tilde{Z}_t = \tilde{X}_t - \alpha_t \nabla f(\tilde{X}_t) - \alpha_t \mathcal{E}_t. \tag{12}$$

Such an inexactness can also be described by the following abstract representation.

$$\min_{G \in \partial Q_t^{\alpha_t}\left(X_t^+(\alpha_t)\right)} \|G\|_F \leq \epsilon_t. \tag{13}$$

For the stepsize choices, the only requirement of our framework is that the stepsize α_t is uniformly bounded and satisfies either of the following criteria for some given $\delta \in (0, 1)$.

$$f(X_{t+1}) \leq f(\tilde{X}_t) + \langle \nabla f(\tilde{X}_t),\, X_{t+1} - \tilde{X}_t\rangle + \frac{\delta}{\alpha_t}\|X_{t+1} - \tilde{X}_t\|_F^2, \quad \text{or} \tag{14}$$

$$F(X_{t+1}) \leq F(\tilde{X}_t) + \delta Q_t^{\alpha_t}(X_{t+1}). \tag{15}$$

We note that (14) does not necessarily imply monotonically decreasing objective values, and it is actually in general independent of how accurately (13) is solved. On the other hand, although the value of $Q_t^{\alpha_t}(X_{t+1})$ is affected by ϵ_t, it is not necessarily negative (although the minimum of $Q_t^{\alpha_t}(\cdot)$ is), and thus the objective value might still be nonmonotone.

To ensure that α_t is bounded, we need to specify an upper bound $\alpha_{\max}$ and a lower bound $\alpha_{\min}$. It is known that if ∇f is L-Lipschitz continuous, then (14) holds for any $\alpha_t < 2\delta/L$, and thus we assign $\alpha_{\min} < 2\delta/L$ to ensure that our algorithm is well-defined. As for (15), note that when $\alpha_t \leq 1/L$, $Q_t^{\alpha_t}(Y)$ is a majorization of $F(Y) - F(\tilde{X}_t)$, so (15) holds. Thus, we can set $\alpha_{\min} < 1/L$.

Since the inexact PG step is not the major tool for reducing the objective value, we simply assign a fixed step size $\alpha_t \equiv \alpha$ in our implementation. We have also experimented with the approach of SpaRSA (Wright et al., 2009) that combines a spectral initialization of Barzilai and Borwein (1988) with backtracking line search, but its empirical performance is worse than the fixed-step variant (see the supplementary materials), likely due to the additional eigendecompositions in backtracking.

We summarize the version of our algorithm for (BM-nuclear) in Algorithm 1, and the version for (BM-PSD) can be obtained by considering only one matrix $W \in \mathbb{R}^{m\times k}$ and replacing SVDs with eigendecompositions. In either case, the approximate SVD (U_t,$\hat{\Sigma}_t$,

V_t) of $\tilde{Z}_t$ is first computed, and we can use it to compute $\epsilon_t^2 = \|\tilde{Z}_t - (\tilde{X}_t - \alpha_t \nabla f(\tilde{X}_t)\|_F^2$ easily by utilizing the fact that both $\tilde{Z}_t$ and $\tilde{X}_t$ are presented in a low-rank factorized form and $\nabla f(\tilde{X}_t)$ is either structured or sparse. We can therefore monitor the progress of the approximate SVD algorithm for reaching a given ϵ_t, and even use it to adjust the rank.

Algorithm 1: BM-Global

input : $\lambda, \delta, \alpha_{\max} \geq \alpha_{\min} > 0$ with $\alpha_{\min} < \delta/(2L)$ or $\alpha_{\min} < 1/L$, initial rank k, a nonnegative sequence $\{\epsilon_t\}$ with $\epsilon_t \to 0$

initialization: $W_0 \in \mathbb{R}^{m\times k}, H_0 \in \mathbb{R}^{n\times k}$ such that $W_0 H_0^\top \in \mathrm{dom}(\Psi)$.

for $t = 0, \ldots$ **do**

  (BM phase) Compute $\tilde{W}_t \in \mathbb{R}^{m\times k}$, $\tilde{H}_t \in \mathbb{R}^{n\times k}$ as an approximate solution to (BM) (starting from (W_t, H_t)) satisfying $\tilde{W}_t \tilde{H}_t^\top \in \mathrm{dom}(\Psi)$ and (9)

  (Convex lifting phase) Decide $\alpha_t \in [\alpha_{\min}, \alpha_{\max}]$ and obtain the SVD U_t, Σ_t, V_t of X_{t+1} (without forming X_{t+1} explicitly) through (5) such that (13) holds and either (14) or (15) is satisfied

  $W_{t+1} \leftarrow U_t \mathrm{diag}(\sqrt{\Sigma_t})$, $H_{t+1} \leftarrow V_t \mathrm{diag}(\sqrt{\Sigma_t})$

  If a partial/inexact eigendecomposition is used, the number of eigenvalues to compute is updated to $k + k_{\mathrm{add}}$ where $k_{\mathrm{add}} \geq 0$

  (Rank update) $k = \mathrm{rank}(\mathrm{diag}(\Sigma_t))$.

Remark 2 *From the description of Algorithm 1, one needs to choose a subproblem solver and a stopping condition for the factorized nonconvex subproblem. We emphasize that the choice of the solver can be arbitrary and highly depends on the application. For example, in our experiments, we applied the polyMF-SS method of (Wang et al., 2017) in the matrix completion problem and Manopt (Boumal et al., 2014) for manifold optimization in the nonlinear semidefinite programming problem. It is also possible to extend the idea of preconditioned gradient-type methods (Tong et al., 2021; Xu et al., 2023) for unregularized ill-conditioned problems to our scenario.*

4. Analysis

This section provides theoretical guarantees for Algorithm 1. In partiular, we first give suitable conditions for ϵ_t to guarantee global convergence, and then further obtain convergence rates by imposing further requirements on ϵ_t. Next, rank identification of Algorithm 1 is proven under a nondegeneracy condition, which shows that for any subsequence $\{X_{t_i}\}_{i=0}^\infty$ of the iterates that converge to a solution X^*, $\mathrm{rank}(X_{t_i}) = \mathrm{rank}(X^*)$ for all i large enough, so the rank k_t in (BM-nuclear) will eventually be automatically adjusted to the optimal value.

4.1 Global Convergence and Worst-case Rates

Our first main theoretical result is the global convergence of our algorithm. In our proofs, we let Ω^* denote the solution set to (CVX), and F^* the optimal objective. For notational simplicity, we denote

$$\mathrm{dist}(X, \Omega^*) := \inf_{X^* \in \Omega^*} \|X - X^*\|_F.$$

Theorem 3 *Consider* (CVX) *with* Ψ *defined in* (1). *Then* Ω^* *is compact and nonempty, and* F *is coercive. If* ∇f *is Lipschitz continuous and*

$$\sum \epsilon_t^2 < \infty \tag{16}$$

in Algorithm 1 *with the condition* (14) *being enforced, then for any initialization* W_0, H_0, *we always have* $\operatorname{dist}(X_t, \Omega^*) \to 0$, *there is at least one limit point of* $\{X_t\}$, *any such limit point is a global solution to* (CVX), *and* $F(X_t) \to F^*$. *Moreover, the same results also apply to* $\{\tilde{X}_t\}$.

Proof The coerciveness of F follows directly from the fact that the nuclear norm is coercive and that f is lower-bounded. This implies that the level sets of F are bounded, and thus so is Ω^*. Moreover, because F is lower semicontinuous, it attains its minimum confined to any compact set, so we see that Ω^* is nonempty.

From (16), we have

$$\infty > c^2 \coloneqq \sum_{t=0}^{\infty} \epsilon_t^2. \tag{17}$$

We first show that $\{X_t\}$ admits at least one limit point. From (13), we have that there exists $G_t \in \mathbb{R}^{m\times n}$ such that

$$G_t \in \nabla f\left(\tilde{X}_t\right) + \frac{1}{\alpha_t}\left(X_{t+1} - \tilde{X}_t\right) + \partial\Psi(X_{t+1}), \quad \|G_t\|_F \le \epsilon_t. \tag{18}$$

From the convexity of $\Psi(\cdot)$, we have

$$\Psi(X_{t+1}) \le \langle \Xi,\, X_{t+1} - X\rangle + \Psi(X), \quad \forall \Xi \in \partial\Psi(X_{t+1}), \quad \forall X. \tag{19}$$

By combining (19) with $X = \tilde{X}_t$, (14) and (18), and defining $\gamma \coloneqq (1-\delta)/\alpha_{\max}$, we get the following inequality:

$$\begin{aligned} F(X_{t+1}) &\le F(\tilde{X}_t) + \langle G_t,\, X_{t+1} - \tilde{X}_t\rangle - \frac{1-\delta}{\alpha_t}\|X_{t+1} - \tilde{X}_t\|_F^2 \\ &\le F(\tilde{X}_t) + \|G_t\|_F\|X_{t+1} - \tilde{X}_t\|_F - \frac{1-\delta}{\alpha_{\max}}\|X_{t+1} - \tilde{X}_t\|_F^2 \\ &\le F(\tilde{X}_t) + \epsilon_t\|X_{t+1} - \tilde{X}_t\|_F - \gamma\|X_{t+1} - \tilde{X}_t\|_F^2 \qquad (20) \\ &\le F(X_t) + \epsilon_t\|X_{t+1} - \tilde{X}_t\|_F - \gamma\|X_{t+1} - \tilde{X}_t\|_F^2, \qquad (21) \end{aligned}$$

where the last inequality is from (9). (21) implies that

$$\gamma\|X_{t+1} - \tilde{X}_t\|_F^2 \le F(X_t) - F(X_{t+1}) + \epsilon_t\|X_{t+1} - \tilde{X}_t\|_F. \tag{22}$$

By summing (22) from $t = 0$ to $t = k$, we have that

$$\gamma \sum_{t=0}^{k} \|X_{t+1} - \tilde{X}_t\|_F^2 \leq F(X_0) - F(X_{k+1}) + \sum_{t=0}^{k} \epsilon_t \|X_{t+1} - \tilde{X}_t\|_F \tag{23}$$

$$\leq F(X_0) - F^* + \sum_{t=0}^{k} \epsilon_t \|X_{t+1} - \tilde{X}_t\|_F$$

$$\leq F(X_0) - F^* + \sqrt{\sum_{t=0}^{k} \epsilon_t^2} \sqrt{\sum_{t=0}^{k} \|X_{t+1} - \tilde{X}_t\|_F^2} \tag{24}$$

$$\leq F(X_0) - F^* + c\sqrt{\sum_{t=0}^{k} \|X_{t+1} - \tilde{X}_t\|_F^2}, \tag{25}$$

where (24) is from the Cauchy-Schwarz inequality. By applying the quadratic formula to (25), we obtain that

$$\sqrt{\sum_{t=0}^{k} \|X_{t+1} - \tilde{X}_t\|_F^2} \leq \frac{c + \sqrt{c^2 + 4\gamma\left(F(X_0) - F^*\right)}}{2\gamma}. \tag{26}$$

This implies that for all $k \geq 0$, we get

$$\sum_{t=0}^{k} \epsilon_t \|X_{t+1} - \tilde{X}_t\|_F \leq \sqrt{\sum_{t=0}^{k} \epsilon_t^2} \sqrt{\sum_{t=0}^{k} \|X_{t+1} - \tilde{X}_t\|_F^2} \leq \frac{c^2 + c\sqrt{c^2 + 4\gamma\left(F(X_0) - F^*\right)}}{2\gamma}. \tag{27}$$

Combining (27) and (23), we have that $\{F(X_t)\}$ is upper-bounded. Then from the coerciveness of F, the sequence $\{X_t\}$ is bounded, and thus it has at least one limit point.

Next, we prove that $\mathrm{dist}(X_t, \Omega^*) \to 0$ by conrtadiction. Suppose this statement is false, then there exists $\sigma > 0$ and a subsequence $\{X_{t_k}\}_k$ such that $\mathrm{dist}(X_{t_k}, \Omega) \geq \sigma > 0$ for any k. Since $\{X_{t_k}\}$ is also a bounded sequence, we have that there exists a subsubsequence $\{X_{\ell_k}\} \subseteq \{X_{t_k}\}$ such that

$$X_{\ell_k} \to \widetilde{X^*} \notin \Omega^*. \tag{28}$$

From (18), we have that

$$G_t + \nabla f(X_{t+1}) - \nabla f(\tilde{X}_t) - \frac{1}{\alpha_t}(X_{t+1} - \tilde{X}_t) \in \partial F(X_{t+1}). \tag{29}$$

From (16), we have $\epsilon_t \to 0$ and hence $G_t \to 0$. From (26), we have that $X_{t+1} - \tilde{X}_t \to 0$. This together with the Lipschitz continuity of ∇f and the boundedness of α_t implies that $\nabla f(X_{t+1}) - \nabla f(\tilde{X}_t) \to 0$ and $\alpha_t^{-1}(X_{t+1} - \tilde{X}_t) \to 0$. These results together imply that

$$G_t + \nabla f(X_{t+1}) - \nabla f(\tilde{X}_t) - \frac{1}{\alpha_t}(X_{t+1} - \tilde{X}_t) \to 0. \tag{30}$$

From (28)–(30) and the outer semi-continuity of ∂F (see (Rockafellar and Wets, 2009, Proposition 8.7) and (Bauschke and Combettes, 2017, Proposition 20.37)) in (29), we have

that $0 \in \partial F(\widetilde{X^*})$. From the convexity of F, we thus get $\widetilde{X^*} \in \Omega^*$, contradicting (28). Therefore, we conclude $\mathrm{dist}(X_t, \Omega^*) \to 0$. This also implies that any limit point of $\{X_t\}$ must lie in Ω^*.

Finally, we prove the convergence of $\{F(X_t)\}$. From the convexity of F and (29), we see that for any $X^* \in \Omega^*$,

$$\begin{aligned} 0 &\le F(X_{t+1}) - F(X^*) \\ &\le \langle G_t + \nabla f(X_{t+1}) - \nabla f(\tilde{X}_t) - \alpha_t^{-1}(X_{t+1} - \tilde{X}_t),\, X_{t+1} - X^* \rangle \\ &\le \left\| G_t + \nabla f(X_{t+1}) - \nabla f(\tilde{X}_t) - \alpha_t^{-1}(X_{t+1} - \tilde{X}_t) \right\|_F \|X_{t+1} - X^*\|_F. \end{aligned} \tag{31}$$

By the boundedness of $\{X_t\}$, $\|X_{t+1} - X^*\|_F$ is upper bounded. Recall that the first norm term in (31) approaches to 0 as shown in (30). Thus, $F(X_t) \to F^*$ by the sandwich lemma.

For the part of $\{\tilde{X}_t\}$, we see that the boundedness of the iterates and the convergence of the objective value follow from (9) and again the coerciveness of F. From this boundedness we then conclude the existence of a limit point, and convergence of $\mathrm{dist}(\tilde{X}_t, \Omega^*)$ follows from the same argument above. ■

Although our global convergence is guaranteed by the inexact PG step, existing analyses for the inexact PG method, like those by Combettes (2004); Schmidt et al. (2011); Jiang et al. (2012); Hamadouche et al. (2022), utilize the geometry of the iterates, and are hence not applicable to our algorithm because our additional BM phase could move the iterates arbitrarily within the level sets and this may make such geometry properties no longer valid. Therefore, another contribution of this work is in developing new proof techniques for obtaining global convergence guarantees for alternating between general nonmonotone inexact PG steps and some other descent optimization steps.

We next provide convergence rate guarantees for Algorithm 1 in the coming two theorems. For such results, we use the definitions below.

$$\triangle F_t := F(X_t) - F^*, \quad \triangle \tilde{F}_t := F(\tilde{X}_t) - F^*.$$

The following theorem and its proof are partially motivated by Scheinberg and Tang (2016).

Theorem 4 *Suppose the conditions in Theorem 3 hold. Then there exists a constant $\beta > 0$ such that the following inequality holds:*

$$\triangle \tilde{F}_{t+1} \le \max\left\{ \xi_t, \frac{\triangle \tilde{F}_t}{1 + \beta^{-1} \triangle \tilde{F}_t} \right\}, \quad \xi_t := \sqrt{\beta \psi_t} + \psi_t, \quad \psi_t := \epsilon_t(\|X_{t+1} - \tilde{X}_t\|_F + \gamma \epsilon_t), \tag{32}$$

where $\gamma := (1-\delta)/\alpha_{\max}$. Moreover, if $\epsilon_t = O(t^{-2})$, then $\triangle F_t = O(t^{-1})$ and $\triangle \tilde{F}_t = O(t^{-1})$.

Proof From the boundedness of $\{F(X_t)\}$ and $\{F(\tilde{X}_t)\}$ obtained in the proof of Theorem 3, we know that there is a value F_0 such that $F(X_t) \le F_0$ and $F(\tilde{X}_t) \le F_0$ for all t, and thus from the coerciveness of F from Theorem 3, there is a nonnegative and finite constant R_0 such that

$$R_0 := \max_{X_1, X_2 \in \{X \mid F(X) \le F_0\}} \|X_1 - X_2\|_F < \infty. \tag{33}$$

Next, from the convexity of f, we have that for any $X \in \mathbb{R}^{m\times n}$,

$$f(\tilde{X}_t) \le f(X) + \langle \nabla f(\tilde{X}_t),\, \tilde{X}_t - X\rangle. \tag{34}$$

Add up (14) and (34), we get

$$f(X_{t+1}) \le f(X) + \langle \nabla f(\tilde{X}_t),\, X_{t+1} - X\rangle + \frac{\delta}{\alpha_t}\|X_{t+1} - \tilde{X}_t\|_F^2. \tag{35}$$

Add up (19) and (35), we get

$$F(X_{t+1}) \le F(X) + \frac{1}{\alpha_t}\langle \tilde{X}_t - X_{t+1},\, X_{t+1} - X\rangle + \frac{\delta}{\alpha_t}\|X_{t+1} - \tilde{X}_t\|_F^2 + \langle G_t,\, X_{t+1} - X\rangle \tag{36}$$

for G_t defined in (18). Choose $X = X^*$ for some $X^* \in \Omega^*$ in (36) and use (33), we get

$$\begin{aligned} F(X_{t+1}) &\le F(X^*) + \frac{R_0}{\alpha_t}\|\tilde{X}_t - X_{t+1}\|_F + \frac{\delta R_0}{\alpha_t}\|\tilde{X}_t - X_{t+1}\|_F + \epsilon_t R_0 \\ &\le F(X^*) + \tilde{c}(\|X_{t+1} - \tilde{X}_t\|_F + \epsilon_t), \end{aligned} \tag{37}$$

where $\tilde{c} \coloneqq R_0(1+\delta+\alpha_{\min})/\alpha_{\min} \in [0,\infty)$ is a constant. Note that F is Lipschitz continuous in any bounded region, so we can further obtain from (37) that

$$F(\tilde{X}_t) \le F(X^*) + \bar{c}(\|X_{t+1} - \tilde{X}_t\|_F + \epsilon_t), \tag{38}$$

where $\bar{c} > 0$ is a constant. From (38), we further get that

$$(F(\tilde{X}_t) - F(X^*))^2 \le 2\bar{c}^2\|X_{t+1} - \tilde{X}_t\|_F^2 + 2\bar{c}^2\epsilon_t^2, \tag{39}$$

Substitute (39) into (20) and use (32), we get

$$F(\tilde{X}_{t+1}) \le F(X_{t+1}) \le F(\tilde{X}_t) + \psi_t - \frac{\gamma}{2\bar{c}^2}(F(\tilde{X}_t) - F(X^*))^2, \tag{40}$$

Let $\beta = 4\bar{c}^2/\gamma$, we then have the following two cases from (40):
Case 1. $\psi_t > \beta^{-1}\triangle\tilde{F}_t^2$.
We have that $\triangle\tilde{F}_t \le \sqrt{\beta\psi_t}$. Combine this and (40), we get

$$\triangle\tilde{F}_{t+1} \le \triangle F_{t+1} \le \sqrt{\beta\psi_t} + \psi_t. \tag{41}$$

Case 2. $\psi_t \le \beta^{-1}\triangle\tilde{F}_t^2$.
Substitute this into (40), we get

$$\triangle\tilde{F}_{t+1} \le \triangle F_{t+1} \le \triangle\tilde{F}_t - \frac{1}{\beta}\triangle\tilde{F}_t^2 \le \triangle\tilde{F}_t. \tag{42}$$

From (42), we have that $\triangle\tilde{F}_{t+1} \le \triangle\tilde{F}_t - \beta^{-1}\triangle\tilde{F}_t\triangle\tilde{F}_{t+1}$, which implies that

$$\triangle\tilde{F}_{t+1} \le \frac{\triangle\tilde{F}_t}{1 + \beta^{-1}\triangle\tilde{F}_t}. \tag{43}$$

Combine (41) and (43), we get (32).

Now, assume that $\epsilon_t = O(t^{-2})$. From (32) and (33), we see that $\xi_t = O(t^{-1})$. Namely, there exists $\kappa \geq 0$ such that $\xi_t \leq \kappa/t$ for all $t \geq 1$. For any $t \in \mathbb{N}$, we first consider the case in which there is some index $\tilde{t}_0 \in \{1, \ldots, t\}$ such that the first term in (32) is larger than the second one and let t_0 be the maximum of such indices. Thus, for any $k \in \{t_0+1, \ldots, t\}$, we have that $\triangle \tilde{F}_{k+1} \leq \triangle \tilde{F}_k/(1+\beta^{-1}\triangle \tilde{F}_k)$, which implies that

$$\frac{1}{\triangle \tilde{F}_k} + \frac{1}{\beta} \leq \frac{1}{\triangle \tilde{F}_{k+1}}.$$

Summing the inequality above from $k = t_0 + 1$ to $k = t$ and telescoping, we have that

$$\frac{1}{\triangle \tilde{F}_{t+1}} \geq \frac{1}{\triangle \tilde{F}_{t_0+1}} + \frac{(t-t_0)}{\beta} \geq \frac{t_0}{\kappa} + \frac{t-t_0}{\beta} \geq \frac{t}{\max\{\kappa, \beta\}}, \tag{44}$$

which implies

$$\triangle \tilde{F}_{t+1} \leq \frac{\max\{\beta, \kappa\}}{t} = O(t^{-1}). \tag{45}$$

Now let us turn to the case in which the second term in (32) is larger than the first one for all $k \in \{1, \ldots, t\}$. Then an analysis analogous to (44) leads to the following inequality:

$$\triangle \tilde{F}_{t+1} \leq \frac{1}{\triangle \tilde{F}_1^{-1} + \beta^{-1} t} = O(t^{-1}). \tag{46}$$

Combine (45) and (46), we obtain that $\triangle \tilde{F}_t = O(t^{-1})$. Finally, viewing from (40) and the optimality of F^*, we get from (45), (46), and $\psi_t = O(t^{-1})$ that $\Delta F_{t+1} \leq \Delta \tilde{F}_t + \psi_t = O(t^{-1})$. ■

In the next result, we show faster convergence rates under a Hölderian error-bound condition

$$\zeta \mathrm{dist}(X, \Omega^*) \leq \left(F(X) - F^*\right)^\theta, \quad \forall X \tag{47}$$

for some $\zeta > 0$ and some $\theta \in [0, 1]$. In particular, when $\theta \geq 1/2$, we obtain linear convergence for the objective. Under convexity of F, it is shown by Bolte et al. (2017) that (47) is equivalent to the Kurdyka-Łojasiewicz (KL) condition (Kurdyka, 1998; Łojasiewicz, 1963).

Theorem 5 *Consider* (CVX) *with* Ψ *defined in* (1)*. Suppose the line-search criterion* (15) *is used with* $\delta \in (0, 1)$ *in Algorithm* 1*. If* F *satisfies* (47)*, then the following convergence results hold.*

(i) When $\theta = 1/2$*: Let*

$$M \coloneqq \min_{\mu \in [0,1]} 1 - \delta\mu + \frac{\delta\mu^2}{2\zeta^2 \alpha_{\min}} < 1. \tag{48}$$

If

$$\sum_{t=1}^{\infty} \frac{\epsilon_t^2}{M^t} < \infty, \tag{49}$$

then $\triangle \tilde{F}_t = O(M^t)$.

(ii) When $0 \le \theta < 1/2$: If

$$\epsilon_t^2 = O\left(t^{-\frac{2-2\theta}{1-2\theta}}\right), \tag{50}$$

then

$$\triangle \tilde{F}_t = O\left(t^{-\frac{1}{1-2\theta}}\right).$$

(iii) When $\theta > 1/2$: If we use (14) instead of (15) and ϵ_t^2 is summable, then there is $T_0 \ge 0$ such that $X_{T_0} \in \Omega^*$.

Proof Let X_t^* and Q_t^* be the unique minimizer and minimum of $Q_t^{\alpha_t}(Y)$ respectively. Because $Q_t^{\alpha_t}(Y)$ is a strongly-convex function with modulus α_t^{-1}, from (13), there exist G_t such that

$$\frac{1}{\alpha_t}\|X_{t+1} - X_t^*\|_F \le \|G_t\|_F \le \epsilon_t. \tag{51}$$

From (15), we have that

$$\begin{aligned} F(X_{t+1}) - F(\tilde{X}_t) &\le \delta Q_t^{\alpha_t}(X_{t+1}) \\ &\le \delta(Q_t^* + \langle G_t,\, X_{t+1} - X_t^*\rangle) \\ &\le \delta(Q_t^* + \|G_t\|_F \|X_{t+1} - X_t^*\|_F) \\ &\le \delta(Q_t^* + \alpha_t \epsilon_t^2), \end{aligned} \tag{52}$$

where the second inequality comes from the convexity of $Q_t^{\alpha_t}(\cdot)$, and the last one comes from (51). From Lemma 5 in Lee and Wright (2019) (also see Equation 70 of Lee, 2023), we have that

$$Q_t^* \le \mu(F^* - F(\tilde{X}_t)) + \frac{\mu^2}{2\alpha_t}\mathrm{dist}(\tilde{X}_t, \Omega^*)^2, \quad \forall \mu \in [0,1]. \tag{53}$$

Substitute (53) into (52) and use (47), we get

$$\begin{aligned} F(\tilde{X}_{t+1}) - F(\tilde{X}_t) &\le F(X_{t+1}) - F(\tilde{X}_t) \\ &\le \delta \min_{\mu\in[0,1]} \left(\mu(F^* - F(\tilde{X}_t)) + \frac{\mu^2}{2\alpha_t}\mathrm{dist}(\tilde{X}_t, \Omega^*)^2 + \alpha_t\epsilon_t^2\right) \\ &\le \delta \min_{\mu\in[0,1]} \left(\mu(F^* - F(\tilde{X}_t)) + \frac{\mu^2}{2\zeta^2\alpha_t}(F(\tilde{X}_t) - F^*)^{2\theta} + \alpha_t\epsilon_t^2\right). \end{aligned} \tag{54}$$

Proof of (i). Let

$$M_t \coloneqq \min_{\mu\in[0,1]} 1 - \delta\mu + \frac{\delta\mu^2}{2\zeta^2\alpha_t}, \quad \forall t \ge 0.$$

Because $0 < \delta < 1$ and $\alpha_t \le \alpha_{\max}$, we have that $0 < M_t < M < 1$ for all $t \ge 0$. From (54) and the assumption that $\theta = 1/2$, we get

$$\triangle \tilde{F}_{t+1}/M^{t+1} \le \triangle \tilde{F}_t/M^t + \delta\alpha_t\epsilon_t^2/M^{t+1}. \tag{55}$$

Because ϵ_t^2/M^t is summable from (49) and $\alpha_t \le \alpha_{\max} < \infty$, (55) clearly shows $\triangle \tilde{F}_t = O(M^t)$.

Proof of (ii). From (54), we have

$$\triangle\tilde{F}_{t+1} \leq \min_{\mu\in[0,1]} \left\{1 - \delta\mu + \frac{\delta\mu^2}{2\zeta^2\alpha_t}\triangle\tilde{F}_t^{2\theta-1}\right\}\triangle\tilde{F}_t + \delta\alpha_t\epsilon_t^2. \tag{56}$$

Clearly, the minimizer of μ in (56) is $\min\{\zeta^2\alpha_t\triangle\tilde{F}_t^{1-2\theta}, 1\}$. Substitute this into (56), we get

$$\triangle\tilde{F}_{t+1} \leq \left(1 - \frac{\delta\zeta^2\alpha_t}{2}\min\left\{\triangle\tilde{F}_t^{1-2\theta}, \frac{1}{\zeta^2\alpha_t}\right\}\right)\triangle\tilde{F}_t + \delta\alpha_t\epsilon_t^2. \tag{57}$$

We first show that $\liminf_{t\to\infty}\triangle\tilde{F}_t = 0$. Assume on the contrary that there exists $\eta > 0$ such that $\triangle\tilde{F}_t \geq \eta$ for all $t \geq 0$. Then, from (57), we have that

$$\triangle\tilde{F}_{t+1} \leq \left(1 - \frac{\delta\zeta^2\alpha_t}{2}\min\left\{\eta^{1-2\theta}, \frac{1}{\zeta^2\alpha_t}\right\} + \frac{\delta\alpha_t\epsilon_t^2}{\eta}\right)\triangle\tilde{F}_t,$$

which implies that $\triangle\tilde{F}_t \to 0$ with a linear rate when t is sufficiently large to make ϵ_t small enough. This contradicts to $\triangle\tilde{F}_t \geq \eta > 0$. Now, since $\liminf_{t\to\infty}\triangle\tilde{F}_t = 0$ and $\sum_{t=1}^{\infty}\alpha_t\epsilon_t^2 < \infty$, there exists $T_0 \geq 0$ such that $\triangle\tilde{F}_{T_0} \leq C_1$ and $\sum_{t=T_0}^{\infty}\delta\alpha_t\epsilon_t^2 < C_1$, where

$$C_1 \coloneqq \frac{1}{2}(\zeta^2\alpha_{\max})^{-\frac{1}{1-2\theta}}.$$

From (57), we thus get

$$\triangle\tilde{F}_t \leq \triangle\tilde{F}_{T_0} + \sum_{k=T_0}^{t-1}\delta\alpha_t\epsilon_t^2 < 2C_1 \leq (\zeta^2\alpha_t)^{-\frac{1}{1-2\theta}}, \quad \forall t \geq T_0. \tag{58}$$

Thus,

$$\min\left\{\triangle\tilde{F}_t^{1-2\theta}, \frac{1}{\zeta^2\alpha_t}\right\} = \triangle\tilde{F}_t^{1-2\theta}, \quad \forall t \geq T_0.$$

Let $M_1 \coloneqq \delta\zeta^2\alpha_{\min}/2$, $M_2 \coloneqq \delta\alpha_{\max}$, then from the equation above and (57), we have that

$$\triangle\tilde{F}_{t+1} \leq \left(1 - M_1\triangle\tilde{F}_t^{1-2\theta}\right)\triangle\tilde{F}_t + M_2\epsilon_t^2, \quad \forall t \geq T_0. \tag{59}$$

From (58) and that $0 < \delta < 1$, we have that $M_1\triangle\tilde{F}_t^{1-2\theta} < 1$ for any $t \geq T_0$. Now we choose $D > 0$ to be a sufficiently large number such that the following three conditions hold.

$$\triangle\tilde{F}_{T_0} \leq DT_0^{\frac{-1}{1-2\theta}}, \tag{60a}$$

$$M_2\epsilon_t^2 + \left(\frac{2M_2}{M_1}\epsilon_t^2\right)^{\frac{1}{2-2\theta}} \leq D(t+1)^{\frac{-1}{1-2\theta}}, \quad \forall t \geq T_0, \tag{60b}$$

$$D^{-(1-2\theta)} \leq \frac{(1-2\theta)M_1}{2}, \tag{60c}$$

where (60b) is guaranteed by (50) and $\theta < 1/2$, and (60c) can be guaranteed by $\theta < 1/2$. Now, we use mathematical induction to prove that $\triangle\tilde{F}_t \leq Dt^{\frac{-1}{1-2\theta}}$ for all $t \geq T_0$. The case

$t = T_0$ directly comes from (60a). Suppose the inequality holds for some $t \geq T_0$, we have the following two cases.

Case 1. $M_2\epsilon_t^2 \leq M_1 \triangle \tilde{F}_t^{2-2\theta}/2$.

From (59), we have that

$$\triangle \tilde{F}_{t+1} \leq \left(1 - \frac{M_1}{2}\triangle \tilde{F}_t^{1-2\theta}\right) \triangle \tilde{F}_t,$$

which leads to

$$\begin{aligned}
\triangle \tilde{F}_{t+1}^{-(1-2\theta)} &\geq \left(1 - \frac{M_1}{2}\triangle \tilde{F}_t^{1-2\theta}\right)^{-(1-2\theta)} \triangle \tilde{F}_t^{-(1-2\theta)} \\
&\geq \left(1 + \frac{(1-2\theta)M_1}{2}\triangle \tilde{F}_t^{1-2\theta}\right) \triangle \tilde{F}_t^{-(1-2\theta)} \\
&= \triangle \tilde{F}_t^{-(1-2\theta)} + \frac{(1-2\theta)M_1}{2} \\
&\geq D^{-(1-2\theta)}t + \frac{(1-2\theta)M_1}{2} \\
&\geq D^{-(1-2\theta)}(t+1), \qquad (61)
\end{aligned}$$

where the second inequality comes from the fact that $(1-x)^{-p} \geq 1 + px$ for any $x < 1$ and $p > 0$, and the last inequality comes from (60c). (61) implies that $\triangle \tilde{F}_{t+1} \leq D(t+1)^{\frac{-1}{1-2\theta}}$.

Case 2. $M_2\epsilon_t^2 > M_1 \triangle \tilde{F}_t^{2-2\theta}/2$.

In this case, we have that

$$\triangle \tilde{F}_t \leq \left(\frac{2M_2}{M_1}\epsilon_t^2\right)^{\frac{1}{2-2\theta}}. \qquad (62)$$

By substituting (62) into (59), we obtain

$$\triangle \tilde{F}_{t+1} \leq \triangle \tilde{F}_t + M_2\epsilon_t^2 \leq \left(\frac{2M_2}{M_1}\epsilon_t^2\right)^{\frac{1}{2-2\theta}} + M_2\epsilon_t^2 \leq D(t+1)^{\frac{-1}{1-2\theta}}, \qquad (63)$$

where the last inequality comes from (60b).

Combining Cases 1 and 2, we get $\triangle \tilde{F}_t \leq Dt^{\frac{-1}{1-2\theta}}$ for any $t \geq T_0$, as desired.

Proof of (iii). From Theorem 3, we get that $\mathrm{dist}(X_t, \Omega^*) \to 0$, and therefore (Yue et al., 2019, Proposition 1) implies $\|X_t - \mathrm{prox}_\Psi(X_t - \nabla f(X_t))\|_F \to 0$. On the other hand, from (Bolte et al., 2017, Theorem 5) and (Mordukhovich et al., 2022, Proposition 2.4), the condition (47) together with the convexity of F implies that there is $\kappa \geq 0$ such that

$$\mathrm{dist}(X, \Omega^*) \leq \kappa \|X - \mathrm{prox}_\Psi(X - \nabla f(X))\|_F^{\frac{\theta}{1-\theta}}.$$

Moreover, as $\theta > 1/2$, we get $\theta/(1-\theta) > 1$. Therefore, we can apply (Lee and Wright, 2022, Theorem 3) to obtain the desired conclusion to complete the proof. ■

By setting $\theta = 0$ in Theorem 5, we recover the same convergence rate in Theorem 4, but instead of $\epsilon_t = O(t^{-2})$ in Theorem 4, Theorem 5 only needs $\epsilon_t = O(t^{-1})$. The difference between the two theorems is that Theorem 4 uses (14) that allows a more aggressive step

size selection but with the price of a higher accuracy in the PG step, while Theorem 5 uses (15) that leads to a more conservative step size to trade for less time spent on computing the approximate SVD. Moreover, for Theorem 5, (47) with $\theta = 0$ directly assumes that the iterates are bounded.

4.2 Rank identification

We proceed on to show that under a nondegeneracy condition, the rank of X_t for any convergent subsequence will eventually become fixed and equivalent to the point of convergence. First, we need the definition below of convex partly smooth functions. This definition involves the usage of $\mathcal{C}^2$-manifold, which means the system of equations defining such a manifold is $\mathcal{C}^2$.

Definition 6 (Partly smooth (Lewis, 2002)) *A convex function Ψ is partly smooth at a point X^* relative to a set $\mathcal{M}$ containing X^* if $\partial\Psi(X^*) \neq \emptyset$ and:*

1. *Around X^*, $\mathcal{M}$ is a $\mathcal{C}^2$-manifold and $\Psi|_{\mathcal{M}}$ is $\mathcal{C}^2$.*
2. *The affine span of $\partial\Psi(X)$ is a translate of the normal space to $\mathcal{M}$ at X^*.*
3. *$\partial\Psi$ is continuous at X^* relative to $\mathcal{M}$.*

Loosely speaking, this means that $\Psi|_{\mathcal{M}}$ is smooth at x^*, but the value of Ψ changes drastically along directions leaving $\mathcal{M}$ around x^*.

It is known (Daniilidis et al., 2014) that at every $X \in \mathbb{R}^{m\times n}$, $\|\cdot\|_*$ is partly smooth with respect to the manifold

$$\mathcal{M}(X) \coloneqq \left\{Y \in \mathbb{R}^{m\times n} \mid \operatorname{rank}(Y) = \operatorname{rank}(X)\right\}. \tag{64}$$

Similarly, if $X \in \mathbb{S}^n$, we also have that $\delta_{\mathbb{S}^n_+}$ is partly smooth everywhere in $\mathbb{S}^n_+$, with respect to the manifold

$$\mathcal{M}_2(X) \coloneqq \left\{Y \in \mathbb{S}^n_+ \mid \operatorname{rank}(Y) = \operatorname{rank}(X)\right\}. \tag{65}$$

Finally, when $\mathcal{S}$ is a polyhedron, it is widely known that $\delta_{\mathcal{S}}$ is also partly smooth everywhere, with respect to the minimal face containing the reference point. As intersections of manifolds are still manifolds, if

$$\Psi(X) = \lambda\|X\|_* + \lambda_2\delta_{\mathbb{S}^n_+}(X) + \lambda_3\delta_{\mathcal{S}}(X) \tag{66}$$

for $\lambda \in \mathbb{R}$, $\lambda_2, \lambda_3 \in \{0, 1\}$ and some polyhedral $\mathcal{S}$, we have that Ψ is partly smooth everywhere, with respect to a submanifold $\bar{\mathcal{M}}(X) \subseteq \mathcal{M}(X)$.

Now we can leverage tools from partial smoothness and manifold identification to show that our algorithm will find the correct rank for (BM) that contains a global optimum.

Theorem 7 *Consider* (CVX) *with Ψ defined in* (1)*. Consider the two sequences of iterates $\{X_t\}$ and $\{\tilde{X}_t\}$ generated by Algorithm* 1 *from some starting point $X_0 = (W_0, H_0)$ with $\epsilon_t \to 0$ in* (13)*. Then the following hold.*

(i) For any subsequence $\{\tilde{X}_{t_i}\}_i$ such that $\tilde{X}_{t_i} \to X^*$ for some $X^* \in \Omega^*$, $X_{t_i+1} \to X^*$ as well.

(ii) For the same subsequence as above, if X^* satisfies the nondegeneracy condition

$$0 \in \operatorname{relint}\left(\partial F\left(X^*\right)\right) \tag{67}$$

and Ψ is as defined in (1) with either $\lambda \geq 0$ or $\lambda < 0$ and Ψ accords with (66), then there is $i_0 \geq 0$ such that $\operatorname{rank}(X_{t_i+1}) = \operatorname{rank}(X^*)$ for all $i \geq i_0$.

Proof

Proof of (i). Let us denote the exact solution of (10) at the t_i-th iteration given $\tilde{X}_{t_i}$ as $X^*_{t_i+1}$, and the real update we use from (13) as X_{t_i+1}. Following the proof of (Yue et al., 2019, Proposition 1), we have from the optimality of X^*, which implies $\operatorname{prox}_{\alpha\Psi}(X^* - \alpha\nabla f(X^*)) = X^*$ for any $\alpha > 0$, that

$$\begin{aligned}
&\left\|X^*_{t_i} - \tilde{X}_{t_i}\right\|_F \\
&= \left\|\operatorname{prox}_{\alpha_{t_i}\Psi}\left(\tilde{X}_{t_i} - \alpha_{t_i}\nabla f(\tilde{X}_{t_i}\right) - \tilde{X}_{t_i} + \left(X^* - \operatorname{prox}_{\alpha_{t_i}\Psi}(X^* - \alpha_{t_i}\nabla f(X^*))\right)\right\|_F \\
&\leq \left\|\tilde{X}_{t_i} - X^*\right\|_F + \left\|\operatorname{prox}_{\alpha_{t_i}\Psi}\left(\tilde{X}_{t_i} - \alpha_{t_i}\nabla f(\tilde{X}_{t_i})\right) - \operatorname{prox}_{\alpha_{t_i}\Psi}\left(X^* - \alpha_{t_i}\nabla f(X^*)\right)\right\|_F \\
&\leq \left\|\tilde{X}_{t_i} - X^*\right\|_F + \left\|\left(\tilde{X}_{t_i} - \alpha_{t_i}\nabla f(\tilde{X}_{t_i})\right) - \left(X^* - \alpha_{t_i}\nabla f(X^*)\right)\right\|_F \qquad (68) \\
&\leq 2\left\|\tilde{X}_{t_i} - X^*\right\|_F + \alpha_{t_i}\left\|\nabla f(\tilde{X}_{t_i}) - \nabla f(X^*)\right\|_F \\
&\leq (2 + L\alpha_{\max})\left\|\tilde{X}_{t_i} - X^*\right\|_F \to 0, \qquad (69)
\end{aligned}$$

where (68) is from the nonexpansiveness of the proximal operation of any convex function, and (69) is from our assumption. (69) then leads to

$$\left\|X^*_{t_i} - \tilde{X}_{t_i}\right\|_F \to 0. \tag{70}$$

On the other hand, (51) shows that

$$0 \leq \left\|X_{t_i+1} - X^*_{t_i}\right\|_F \leq \alpha_{t_i}\epsilon_{t_i} \to 0, \tag{71}$$

where the limit is obtained from that $\epsilon_t \to 0$ and that α_t is upper-bounded. By combining (70) and (71), it is clear that

$$\left\|X_{t_i+1} - \tilde{X}_{t_i}\right\| \to 0 \quad \Rightarrow \quad \|X_{t_i+1} - X^*\| \to 0, \tag{72}$$

proving the desired result.

Proof of (ii). From our arguments preceding the theorem, Ψ is partly smooth at every X relative to a submanifold of either (64) or (65). Therefore, the result of the second item is equivalent to $X_{t_i+1} \in \mathcal{M}(X^*)$ for all i large enough. As $\epsilon_t \to 0$, we see that all conditions of (Lee, 2023, Theorem 1) are satisfied, and therefore $X_{t_i+1} \in \bar{\mathcal{M}}(X^*) \subseteq \mathcal{M}(X^*)$ for all i large enough. The conclusion therefore follows. ■

Due to the flexibility in the BM step, we have less control over the iterates than ordinary PG methods. Therefore, convergence of the whole sequence of iterates cannot be directly guaranteed and we can only get subsequential convergence. However, in our experiments in Section 6, we often observe empirically that the iterates are convergent to a point, and the rank always becomes fixed after a few iterations of BM-Global.

5. Applications

We provide two applications of (CVX). One is our motivating example of matrix completion with $\mathcal{X}$ being the whole space, and the other one is a special class of convex quadratic semidefinite programming problems.

5.1 Matrix Completion

Our first application of (CVX) is the low-rank matrix completion problem (MC). This problem is widely seen in many machine learning tasks like recommendation systems, localization in Internet of Things (IoTs), and image denoising and compression. Interested readers are referred to a recent survey Nguyen et al. (2019) for more details of these applications. A common feature for many of these tasks is that the observed data are extremely sparse in comparison to the unobserved entries that we aim to predict. In other words, $|\Omega| \ll mn$, and thus the resulting gradient of the smooth part ∇f is also sparse, as it can be nonzero only at those entries in Ω. Table 1 provides some examples of the sparsity level of Ω in real-world data used in our numerical experiments.

We observe that the loss term of (MC) has an L-Lipschitz continuous gradient with $L = 1$, and when $\alpha_t = L^{-1} = 1$, $X_t - \alpha_t \nabla f(X_t)$ is the same as replacing the entries of X_t in Ω with $P_\Omega(A)$, hence standard PG with $\alpha_t \equiv L^{-1}$ is also called soft impute for this problem (Mazumder et al., 2010). Often in real applications described above, we can easily have m and n in the scale of millions with $|\Omega|$ rather small, so indeed we are unable to explicitly form X_t and need to rely on low-rank assumptions or to force low-rank approximations for practical reasons. On the other hand, thanks to the extreme popularity and the simple forms of (MC) and (MF), there are many well-developed algorithms for them.

Theoretical analyses for (MC) and (MF) often consider the noiseless case such that the ground truth A is indeed of low rank and we observe entries without any noise, and show that under such cases, one can recover the whole A by solving (MF) with a sufficient rank. However, in practice, the observed entries are often noisy, either due to measurement errors (like in the IoTs case) or randomness in nature (rating in recommendation systems could be affected by factors beyond the users' preference for certain items). We will see in the numerical experiments in Section 6 that it is often the case that solving (MF) alone does not guarantee convergence to a global optimum even if the correct rank is specified, and therefore the convex lifting step in Algorithm 1 is necessary.

For this problem, in the convex lifting step, we adopt a long-step variant of PG by setting α_t close to $2L^{-1}$ to obtain a slightly better empirical performance. For the BM stage, we adopt the state-of-the-art solver polyMF-SS for (MF) developed by Wang et al. (2017) that conducts block coordinate descent with an exact line search, where each block is one column of W and one column of H.

More implementation details of our algorithm tailored for this application are described in the supplementary materials.

We note that this problem satisfies (47) with $\theta = 1/2$ according to Hou et al. (2013), and thus linear convergence is expected according to Theorem 5.

5.2 A class of convex quadratic semidefinite programming problems

Our second application is the following convex quadratic semidefinite programming (QSDP) problem:

$$\min_{X\in\mathbb{S}^n_+} \left(f(X) \coloneqq \frac{1}{2}\|\mathcal{A}(X)-b\|^2 + \langle C,\, X\rangle \right) \quad \text{s.t.} \quad \langle E,\, X\rangle = 0, \tag{QSDP}$$

where $\mathcal{A}:\mathbb{S}^n\to\mathbb{R}^p$ is a linear mapping whose adjoint mapping is denoted by $\mathcal{A}^*$, $b\in\mathbb{R}^p$, $C\in\mathbb{S}^n$, and $E\in\mathbb{S}^n$ denotes the matrix of all ones. The gradient of f is given by

$$\nabla f(X) = \mathcal{A}^*\left(\mathcal{A}(X)-b\right) + C. \tag{73}$$

Thus $\nabla f(\cdot)$ is Lipschitz continuous with modulus $L = \|\mathcal{A}^*\mathcal{A}\|_2$.

The QSDP problem (QSDP) arises in many important applications when one needs to find a low-rank approximation of a given matrix while preserving certain useful structures (via linear constraints). In this part, we introduce the following two data analysis problems.

- The regularized kernel estimation (RKE) problem (Lu et al., 2005): Given a set of n objects and dissimilarity measures d_{ij}^2 for certain object pairs $(i,j)\in\Omega$, the goal of RKE is to find a positive semidefinite matrix X such that the fitted squared distances between the objects induced by X satisfy

  $$X_{ii} + X_{jj} - 2X_{ij} \;\approx\; d_{ij}^2, \quad \forall (i,j)\in\Omega.$$

  To obtain a low-rank solution for X, the following regularized semidefinite least squares problem is often considered:

  $$\min_{X\in\mathbb{S}^n_+} \frac{1}{2}\sum_{(i,j)\in\Omega} w_{ij}(X_{ii}+X_{jj}-2X_{ij}-d_{ij}^2)^2 + \lambda\langle I,\, X\rangle \quad \text{s.t.} \quad \langle E,\, X\rangle = 0, \tag{74}$$

  where $\lambda>0$ is a positive regularization parameter and $w_{ij}>0$ for any $(i,j)\in\Omega$. In the above, the constraint $\langle E,\, X\rangle = 0$ is a normalization to put the center of mass of the realized Euclidean embedding at the origin. As argued in Section 2, $\langle I,\, X\rangle$ is equivalent to the nuclear norm for $X\in\mathbb{S}^n_+$, so (74) induces low-rank solutions.

- The molecular conformation problem (Fang and Toh, 2013): Given a molecule with n atoms and the estimated inter-atomic distances d_{ij} between some pairs $(i,j)\in\Omega$ of atoms, the goal is to determine the positions $x_1,\ldots,x_n\in\mathbb{R}^d$ of all the atoms. Mathematically, the molecular conformation problem can be stated as follows:

  $$\min_{x_i\in\mathbb{R}^d, 1\le i\le n} \frac{1}{2}\sum_{(i,j)\in\Omega} w_{ij}\left(\|x_i-x_j\|^2 - d_{ij}^2\right)^2 - \frac{\rho}{2n}\sum_{i,j=1}^n \|x_i-x_j\|^2 \quad \text{s.t.} \quad \sum_{i=1}^n x_i = 0,$$

  where $w_{ij}>0$ for all $(i,j)\in\Omega$ and the second term involving $\rho>0$ is used to maximize the pairwise separations between atoms. Define the matrix

  $$X \coloneqq \begin{bmatrix} x_1 & \ldots & x_n \end{bmatrix}^\top \begin{bmatrix} x_1 & \ldots & x_n \end{bmatrix} \in \mathbb{S}^n,$$

then, it is easy to check that

$$\|x_i - x_j\|^2 = X_{ii} + X_{jj} - 2X_{ij}, \quad \sum_{i,j=1}^{n} \|x_i - x_j\|^2 = 2n\langle I,\, X\rangle,$$

and the constraint $\sum_{i=1}^{n} x_i = 0$ can be replaced by $\langle E,\, X\rangle = 0$. We therefore get the same QSDP relaxation (74) for the molecular conformation problem with $\lambda := -\rho < 0$.

Although $\lambda < 0$ in this application, the problem is still in the form (CVX) with a regularizer in (1), so the objective function is still partly smooth everywhere with respect to (65). Hence, our algorithm will eventually generate iterates that have the same rank as the global optimum to which the iterates converge. If this optimum is low-rank, then so will the generated iterates be.

In the applications above, we see that for ∇f in (73), C is a sparse and structured matrix (actually the identity matrix) and $\mathcal{A}$ and $\mathcal{A}^*$ are sparse mappings such that only $\nabla_{i,j} f$ with either $(i,j) \in \Omega$ or $i = j$ could be nonzero. We usually have $p \ll n^2$ in applications, and thus the resulting gradient has a sparse part. Driven by the fruitful and important applications of QSDPs in diverse fields, many efficient algorithms for solving them have been developed. We refer the readers to Li et al. (2018) for a comprehensive literature review and a powerful state-of-the-art solver, QSDPNAL, for the problem (QSDP).

To apply Algorithm 1, PG, or APG to (74), we need to perform the projection onto the feasible set

$$\mathcal{X} := \left\{ X \in \mathbb{S}^n_+ \mid \langle E,\, X\rangle = 0 \right\}. \tag{75}$$

The following lemma provides an effective way for performing such a projection.

Lemma 8 *Define $J := I_n - n^{-1}ee^\top \in \mathbb{S}^n$, then for the set $\mathcal{X}$ defined in* (75)*, it holds that*

$$P_{\mathcal{X}}(G) = P_{\mathbb{S}^n_+}(JGJ), \quad \forall G \in \mathbb{S}^n.$$

Proof First, for any $X \in \mathcal{X}$, clearly $Xe = 0$ and $e^\top X = 0$. Therefore, we observe that

$$\begin{aligned}
&\|X - JGJ\|_F^2 \\
&= \left\| X - \left(I_n - \frac{1}{n} ee^\top \right) G \left(I_n - \frac{1}{n} ee^\top \right) \right\|_F^2 \\
&= \left\| X - G + \frac{1}{n} Gee^\top + \frac{1}{n} ee^\top G + \frac{1}{n} ee^\top \right\|_F^2 \\
&= \|X - G\|_F^2 + \frac{2}{n} \langle X - G,\, Gee^\top + ee^\top G + ee^\top \rangle + \frac{1}{n} \left\| Gee^\top + ee^\top G + ee^\top \right\|_F^2 \\
&= \|X - G\|_F^2 - \frac{2}{n} \langle G,\, Gee^\top + ee^\top G + ee^\top \rangle + \frac{1}{n} \left\| Gee^\top + ee^\top G + ee^\top \right\|_F^2.
\end{aligned}$$

As a consequence, it holds that $P_{\mathcal{X}}(G) = P_{\mathcal{X}}(JGJ)$. Moreover, as $JGJe = JG(Je) = JG(0) = 0$, namely, JGJ has an eigenvalue of 0 associated with the eigenvector e, we get that $P_{\mathbb{S}^n_+}(JGJ) \in \mathcal{X}$ because the projection onto $\mathbb{S}^n_+$ only truncates negative eigenvalues to zero in the eigendecomposition. Since $\mathcal{X} \subseteq \mathbb{S}^n_+$, it follows that $P_{\mathcal{X}}(JGJ) = P_{\mathbb{S}^n_+}(JGJ)$. ■

From Lemma 8, we see that the computational bottleneck lies in the eigendecomposition of matrices in $\mathbb{S}^n$, which could be highly expensive or even computationally prohibited in our high dimensional setting. Thus, we need to rely on low-rank approximate eigendecomposition to perform inexact projections.

Similar to (BM), we can also use the BM approach to solve (QSDP). In particular, the factorized problem takes the following form

$$\min_{W\in\mathbb{R}^{n\times k}} g(W) := \frac{1}{2}\left\|\mathcal{A}\left(WW^\top\right)-b\right\|^2 + \langle C, WW^\top\rangle, \quad \text{s.t.} \quad W^\top e = 0. \tag{76}$$

The gradient of the function $g:\mathbb{R}^{n\times k}\to\mathbb{R}$ is then given as

$$\nabla g(W) = 2\mathcal{A}^*\left(\mathcal{A}\left(WW^\top\right)-b\right)W + 2CW,$$

and for any $D\in\mathbb{R}^{n\times k}$, the Hessian operator of g performed on D is given by

$$\nabla^2 g(W)[D] = 2\mathcal{A}^*\left(\mathcal{A}\left(WW^\top\right)-b\right)D + 2\mathcal{A}^*\left(\mathcal{A}\left(WD^\top+DW^\top\right)\right)W + 2CD.$$

Since $\left\{W\in\mathbb{R}^{n\times k} \mid W^\top e = 0\right\}$ defines a Riemannian manifold, by using the above information related to $g(\cdot)$, we can apply many efficient solvers for Riemannian optimization to solve (76). In our experiments in Section 6.2 for this QSDP problem, we use the state-of-the-art solver Manopt (Boumal et al., 2014) in our BM phase.

6. Numerical experiments

We conduct numerical experiments to exemplify the practical efficiency of the proposed algorithmic framework. In particular, we consider the two tasks discussed in Section 5 with large-scale real-world data sets in multicore environments. All algorithms are implemented in MATLAB and C/C++.

6.1 Matrix completion

The first task we consider is the matrix completion problem in the forms of (MC) and (MF). We use one toy example included in the package LIBPMF (`https://www.cs.utexas.edu/~rofuyu/libpmf/`) and four publicly available large-scale recommendation system data sets for this set of experiments.[2] The only preprocessing we did was to tranpose the data matrices when necessary to conform to our blanket assumption of $m\le n$. For all data sets, We use their original training/test split. These data sets are summarized in Table 1. The column $|\Omega_{\text{test}}|$ indicates the number of entries in the test set. For the value of λ on the real-world data, we follow the values provided by Hsieh and Olsen (2014) that were obtained through cross-validation, while the final k is the rank of the final output of our algorithm, obtained by running our algorithm with the given λ till the objective cannot be further improved. The value of λ for the toy example is from some simple tuning to make the final rank not too far away from that of other data sets.

2. movielens100k: `https://www.kaggle.com/prajitdatta/movielens-100k-dataset`. (We used the split from ua). movielens10m: `https://www.kaggle.com/smritisingh1997/movielens-10m-dataset`. (We used the split from ra). Netflix: `https://www.kaggle.com/netflix-inc/netflix-prize-data`. Yahoo-musc: the R2 one at `https://webscope.sandbox.yahoo.com/catalog.php?datatype=r`.

Data set	m	n	$\lvert\Omega\rvert$	$\lvert\Omega_{\text{test}}\rvert$	λ	final k
toy-example	3952	6040	900189	100020	36	62
movielens100k	943	1682	90570	9430	15	68
movielens10m	65133	71567	9301274	698780	100	50
netflix	17770	2649429	99072112	1408395	300	68
yahoo-music	624961	1000990	252800275	4003960	10000	52

Table 1: Data statistics for matrix completion.

Experiments on the first four data sets are conducted on an Amazon AWS EC2 c6i.4xlarge instance with an 8-core Intel Xeon Ice Lake processor and 32GB memory. For the larger yahoo-music data set, an m6i.4xlarge instance that has the same processor but with 64GB memory is used. Our experiments in this subsection utilize all cores available for all algorithms through parallelization by MATLAB and openMP.

For this task, we conduct four sets of experiments. First, we use the first two smaller data sets to see how different numbers of consecutive inexact proximal gradient iterations and consecutive epochs in the BM phase affect the behavior of our algorithm. Next, we empirically examine the result of Theorem 7 by checking how fast BM-Global identifies the active manifold, namely the correct rank. We then compare our whole method with its BM solver subroutine alone to see that our method is as efficient as the BM solver and can escape from stationary points of (MF) that are not global optima. In the last set of experments, we compare our method with the state of the art for (MC). We note that for this problem, ∇f is 1-Lipschitz continuous, and thus a fixed step size of $\alpha = 1.99$ can be used to satisfy (14) without any data-dependent computation. We have also tested a version that follows SpaRSA (Wright et al., 2009) to use the spectral step size initialization strategy of Barzilai and Borwein (1988) together with backtracking linesearch, but it did not result in better performance, and therefore we will use this fixed-step variant throughout. For completeness, we include the experiments with the SpaRSA variant in the supplementary materials.

To compare different methods, we consider two criteria, one from the optimization point of view and the other from the task-oriented angle. In particular, we first run our algorithm till the objective cannot be further improved, and take the obtained output X^* as the numerical global optimal solution. With the knowledge of this X^*, our first criterion is the relative objective

$$\frac{F(X) - F(X^*)}{F(X^*)}. \tag{77}$$

The second measure we use is the relative root mean squared error (RMSE), which is computed as

$$\frac{\text{RMSE}(X) - \text{RMSE}(X^*)}{\text{RMSE}(0) - \text{RMSE}(X^*)}, \quad \text{RMSE}(X) \coloneqq \sqrt{\frac{\|P_{\Omega_{\text{test}}}(X - A)\|_F^2}{|\Omega_{\text{test}}|}}. \tag{78}$$

Although in general the norm of the exact proximal gradient step would also be a better optimization progress measure especially because it does not require the knowledge

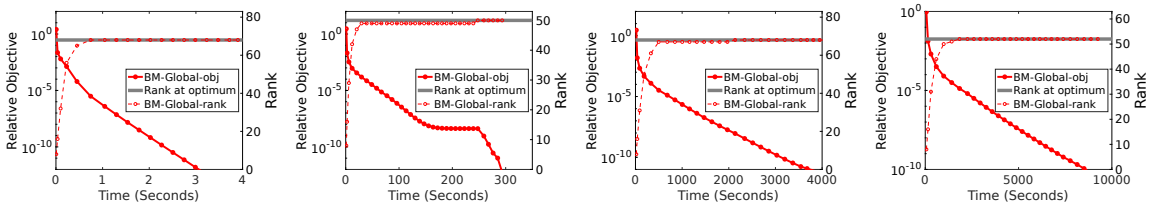
of $F(X^*)$, its computation is impractical in this set of experiments because mn is usually too huge for us to form X_t explicitly and compute its exact SVD that is needed for calculating the exact proximal gradient step.

6.1.1 Parameter tuning for our method

We first use the toy example and movielens100k to finalize details in the parameters setting of our algorithm. In particular, we test the setting of alternating between x consecutive inexact proximal gradient steps and y consecutive iterations of the BM phase solver, with $x \in \{1, \ldots, 5\}$ and $y \in \{1, \ldots, 8\}$, for our fixed step variant with $\alpha \equiv 1.99$. More Details and the results are shown in the supplementary materials. Our result indicates that there is no definite best performer in all cases. But in general, $x = 1$ and $y = 3$ seems to be a rather robust choice. This observation accords with our argument that eigendecompositions are rather expensive and the BM steps should be utilized more often than the proximal gradient steps. We therefore will stick to this setting in all the remaining experiments in this subsection.

6.1.2 Stabilization of the rank

We then show the rank of X_t over iterations of BM-Global In Fig. 1, we use solid lines and dash lines to respectively show the relative objective value and the rank of the iterates of our method. The gray line represents the rank at the optimum X^*. We can see that the rank of X_t increases quickly at first, and eventually stabilizes at the rank of the point of convergence in all cases. Sometimes, the rank remains fixed for a while, then is increased by a small number, and finally stays at the new rank. This is the situation that a safeguard (see the supplementary materials) kicks in to resolve the insufficient rank problem and ensures that the iterates indeed converge to a global optimum. We can also see that when the rank reaches rank(X^*), the relative objective also drops significantly, indicating that finding the right rank is essential in solving (CVX) to a high precision.



(a) movielens100k (b) movielens10m (c) netflix (d) yahoo-music

Figure 1: Rank and relative objective of the iterates of BM-Global over running time.

6.1.3 Comparison between BM-Global and the BM solver alone

We next compare BM-Global with running a BM solver only for (MF). We directly use the solver in our BM phase, namely, the polyMF-SS method of Wang et al. (2017), with their original random start scheme to avoid starting from the origin, which is a known saddle point of (MF). Given any value k, their method starts from a randomized $W \in \mathbb{R}^{m \times k}$ and

$H \in \mathbb{R}^{n\times k}$. We favor their method to directly assign k as the final rank shown in Table 1, but we emphasize that in real-world applications, finding this k will require additional effort in parameter search.

The purpose of this experiment is to show that solving (MF) only can get the iterates stuck at saddle points or spurious local minima, while BM-Global can effectively and efficiently escape from such points. Therefore, we consider the relative objective as the only comparison criterion in this experiment. The results of running time and number of iterations are shown in Fig. 2. For the number of iterations, we count either one inexact proximal gradient step or one epoch of polyMF-SS (one sweep through the whole data) as one iteration.

We observe that in terms of iterations, PolyMF-SS has a small early advantage due to the larger starting rank in (BM). But its convergence quickly slows down, suggesting that likely the iterates are attracted to a saddle point or a spurious local minimum that is strictly worse than the global optima. On the other hand, the story in the running time comparison is very different. We see that the higher rank in PolyMF-SS from the beginning on actually increases the time cost per epoch, and thus the early advantage of PolyMF-SS over BM-Global we observed in terms of iterations is not present in the time comparison. Another observation is that in the numerical experiments, the empirical convergence speed of BM-Global is indeed Q-linear as predicted by Theorem 5.

Overall speaking, BM-Global is as efficient as running a solver for (MF) alone, but it provides multiple advantages including the guarantee of convergence to the global optima. Although in this experiment, the stationary points to which the iterates of PolyMF-SS converge seem to be of good enough quality, we have no guarantee that on other data sets, or even on these data sets but with a different λ, their points of convergence will still be of satisfactory quality.

6.1.4 Comparison with existing methods

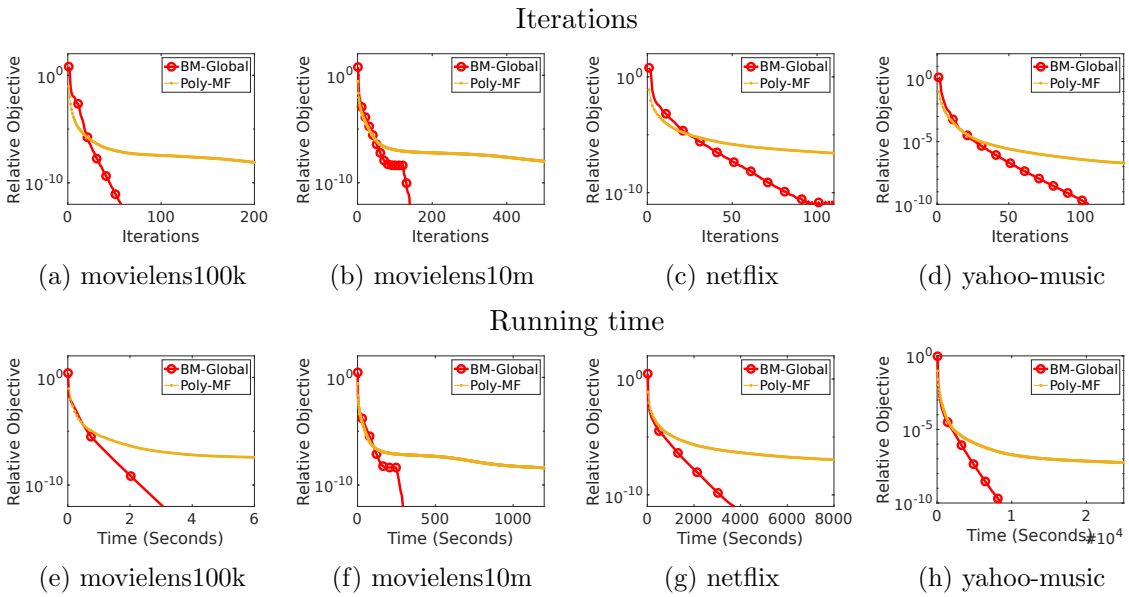
Now that it is clear our method is advantageous over running a solver for (MF) alone, we proceed to compare BM-Global with the state of the art for (MC). In particular, we compare BM-Global with the following:

- Active-ALT (Hsieh and Olsen, 2014): This method alternates between conducting an inexact PG step and solving a lower-dimensional convex subproblem. In the approximate SVD part for inexact PG, Hsieh and Olsen (2014) use the power method with warmstart from the output of the previous iteration plus some random columns as a safeguard.

- AIS-Impute (Yao and Kwok, 2018): An inexact APG method that also uses the power method for approximate SVDs. They use the combination of the outputs of the previous iteration and the iteration preceding it to form the warmstart matrix.

The inexact APG method in Toh and Yun (2010) is not included because the underlying APG part is the same as that of AIS-Impute, but their approximate SVD using Lanczos is shown by Yao and Kwok (2018) to be less efficient.

The results of relative objective and relative RMSE are shown in Fig. 3. Note that the running time for relative RMSE in Fig. 3 is in log scale to make the difference legible.



(a) movielens100k (b) movielens10m (c) netflix (d) yahoo-music

(e) movielens100k (f) movielens10m (g) netflix (h) yahoo-music

Figure 2: Comparison between PolyMF-SS and BM-Global. Top row: iterations v.s. relative objective. Bottom row: running time (seconds) v.s. relative objective.

Clearly, BM-Global outperforms the state of the art for (MC) significantly on both criteria. Particularly, Fig. 3 exemplifies even greater efficiency difference in reaching satisfactory RMSE between BM-Global and existing methods. We can see that the proposed approach is actually magnitudes faster than state of the art in this criterion.

6.2 Convex QSDP

Next, we consider solving the two applications of QSDP described in Section 5.2. As mentioned before, PG and APG can be applied to solve the problem directly. However, based on our empirical experience, both methods require too many iterations and excessive runtime to reach a reasonably good solution, so their numerical results are excluded here. (Interested readers may refer to the supplementary material for the numerical results of the APG methods.) We hence only compare BM-Global with the efficient and robust QSDP solver, QSDPNAL (Li et al., 2018).[3]

Regarding the termination conditions, since QSDPNAL computes both primal and dual iterates, its relative KKT residual is computable (see (Li et al., 2018, Section 5.2) for the definition). Thus, given a specific stopping tolerance `tol`, QSDPNAL is terminated when the maximum relative KKT residual, denoted by η_{kkt}, is less than `tol`. Moreover, when n is large, QSDPNAL may take too much computational time (since it uses full eigendecompositions), so we also cap the running time of QSDPNAL to four hours (initialization overhead excluded) and its maximum number of iterations to 200. For BM-Global, we terminate it

3. Avaliable at `https://blog.nus.edu.sg/mattohkc/softwares/qsdpnal/`.

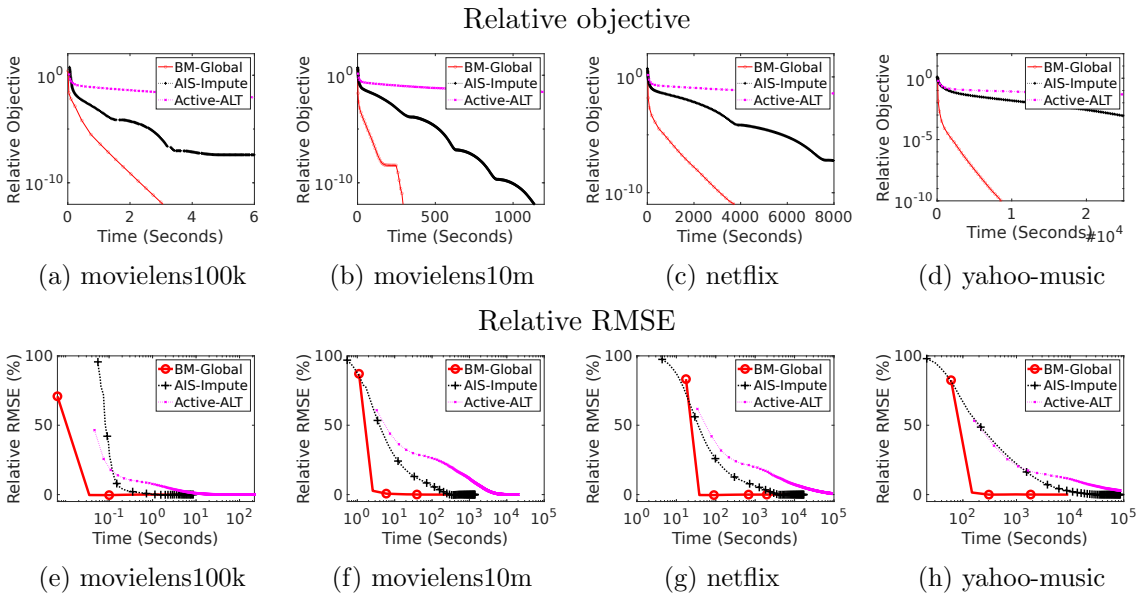


Figure 3: Comparison between BM-Global and existing methods. Top row: running time v.s. relative objective. Bottom row: running time (log scale) v.s. relative RMSE.

when

$$\frac{|f(W_tW_t^\top) - f(W_{t-1}W_{t-1}^\top)|}{(1+|f(W_{t-1}W_{t-1}^\top)|)} < \texttt{tol}.$$

In our experiments, we set $\texttt{tol} = 10^{-6}$ for both methods.

Recall that the first-order optimality condition for problem (QSDP) is given by

$$X - P_{\mathcal{X}}(X - \nabla f(X)) = 0, \quad X \in \mathbb{S}^n.$$

Since we are testing problems with n that can be handled by QSDPNAL that uses full eigendecompositions, we are in fact able to check whether an approximate solution $X \in \mathbb{S}^n$ is optimal numerically, even though this can be time-consuming. Therefore, to compare the quality of the solutions returned by BM-Global and QSDPNAL, we record the relative primal feasibility and the relative optimality, respectively defined as

$$\eta_{\rm prim}(X) := \frac{|\langle E, X\rangle|}{1+\|X\|_F}, \quad \text{and} \quad \eta_{\rm opt}(X) := \frac{\|X - P_{\mathcal{X}}(X - \nabla f(X))\|_F}{1+\|X\|_F + \|\nabla f(X)\|_F}.$$

Experiments for this part are conducted on a Linux PC with an Intel Xeon E5-2650 processor and 96GB memory.

6.2.1 Regularized kernel estimation

We consider problems with dissimilarity measures d_{ij} for $1 \le i, j \le n$ collected in Duin and Pekalska (2009).[4] In our experiments, the data d_{ij} are scaled to the interval $[0, 1]$, and the

4. Data available at `http://prtools.tudelft.nl/Guide/37Pages/distools.html`.

		QSDPNAL					BM-Global			
Name	n	η_{prim}	η_{opt}	η_{kkt}	rnk	Time	η_{prim}	η_{opt}	rnk	Time
BrainMRI	124	6e-11	1e-06	1e-06	5	0.7	2e-16	5e-07	5	0.5
protein	213	4e-13	4e-06	5e-07	24	4.3	5e-11	7e-06	24	1.9
CoilDelftDiff	288	9e-13	4e-04	9e-07	32	5.7	4e-15	5e-08	32	1.6
coildelftsame	288	9e-14	4e-06	5e-07	32	6.7	2e-11	4e-06	32	2.3
CoilYork	288	3e-13	2e-06	4e-07	22	4.8	2e-15	7e-06	22	2.6
Chickenpieces-5-45	446	4e-13	3e-06	7e-07	27	11.5	2e-11	8e-06	28	7.6
newgroups	600	8e-13	3e-05	5e-07	81	39.6	2e-12	2e-04	83	29.6
flowcytodis	612	2e-13	4e-06	9e-07	15	14.8	1e-12	1e-06	15	13.9
DelftPedestrians	689	3e-12	1e-05	2e-07	69	43.7	3e-12	7e-06	69	32.8
WoodyPlants50	791	5e-13	1e-05	7e-07	48	44.3	9e-13	2e-07	48	78.2
delftgestures	1500	1e-14	7e-06	5e-07	76	318.3	3e-13	2e-06	77	390.1
zongker	2000	2e-11	1e-02	8e-07	264	1000.2	6e-12	1e-04	267	665.6
polydish57	4000	2e-12	1e-05	5e-07	100	3508.9	7e-15	3e-05	101	1765.2
polydism57	4000	1e-12	5e-04	8e-07	25	3286.4	1e-16	2e-07	26	305.0

Table 2: Computational results on regularized kernel estimation problems.

elements of the index set Ω are randomly selected such that $|\Omega| \approx n/20$. We set $w_{ij} = 1$ for all $(i,j) \in \Omega$ and $\lambda = \sqrt{n}/10$.

The results are presented in Table 2. Clearly, both methods are able to compute nearly feasible and low-rank solutions. In terms of the optimality measure, BM-Global is able to solve all the problems with $\eta_{\text{opt}} < 10^{-3}$ while QSDPNAL fails to do so for one of the problems. In terms of efficiency, we see that BM-Global is faster in most cases, and BM-Global can even be ten times faster than QSDPNAL in the case of the largest instance.

6.2.2 Molecular conformation

In this experiment, we consider the molecules from the Protein Data Bank (see `https://www.rcsb.org/`) with given noisy and sparse distance data to simulate distances measurable by nuclear magnetic resonance (NMR) experiments. For each molecule, if the distance between two compatible atoms is less than 6Å(6×10^{-8} cm), then the distance can be measured by the NMR experiment; otherwise, we assume that no information is known for this pair. To simulate the sparse set of distances measurable by the NMR experiment, among all the pairwise distances less than 6Å, we select 25% of them to generate our index set Ω. We then add in additional noise to the observed data as follows. Let τ be a given noise level and $\hat{d}_{ij}$ be the exact distance between atom i and atom j for $(i,j) \in \Omega$, we sample two independent random variables $\underline{\epsilon}_{ij}, \overline{\epsilon}_{ij}$ from the normal distribution $\mathcal{N}(0, \pi\tau^2/2)$ and define

$$\underline{d}_{ij} := \max\{1, (1 - |\underline{\epsilon}_{ij}|)\hat{d}_{ij}\}, \quad \overline{d}_{ij} := (1 + |\overline{\epsilon}_{i,j}|)\hat{d}_{ij}.$$

Then, the input distances are set as $d_{ij} := (\underline{d}_{ij} + \overline{d}_{ij})/2$. Given d_{ij}, we set $w_{ij} = 1/d_{ij}^2$. Moreover, we let $\lambda = -10\sqrt{n}/\sum_{(i,j)\in\Omega} d_{ij}^2$. In our tests, we set $\tau = 0.1$. To measure the accuracy of the estimated positions, we record the root mean square deviation (RMSD):

$$\texttt{RMSD} := \frac{1}{\sqrt{n}} \left(\sum_{i=1}^{n} \|x_i - \hat{x}_i\|^2 \right)^2$$

		QSDPNAL						BM-Global				
Name	n	η_{prim}	η_{opt}	η_{kkt}	rnk	RMSD	Time	η_{prim}	η_{opt}	rnk	RMSD	Time
1PBM	126	1e-12	9e-09	7e-07	10	2.7	25.0	1e-13	3e-08	10	2.7	2.0
1AU6	161	3e-12	3e-08	8e-07	12	1.1	34.1	1e-13	2e-08	12	1.1	4.0
1PTQ	402	2e-12	4e-08	8e-07	14	0.7	358.6	2e-16	3e-08	15	0.7	13.9
1CTF	487	3e-12	8e-09	9e-07	13	0.7	570.3	1e-15	3e-08	15	0.8	18.0
1HOE	558	1e-12	2e-08	1e-06	15	0.8	899.6	7e-16	1e-08	16	0.7	32.7
1LFB	641	5e-13	1e-08	3e-06	15	1.2	2739.5	4e-16	3e-08	17	1.5	25.5
1PHT	666	2e-13	8e-08	1e-06	15	1.1	2091.9	8e-16	4e-08	18	1.4	36.3
1F39	767	4e-12	2e-08	3e-06	17	1.3	3230.1	8e-16	2e-08	20	0.8	44.0
1DCH	806	7e-13	2e-08	4e-06	18	1.6	3897.0	3e-16	4e-09	18	1.1	41.8
1HQQ	891	6e-13	2e-08	4e-06	17	2.7	5089.2	7e-16	1e-08	21	1.0	43.5
1POA	914	1e-12	2e-08	4e-06	16	2.1	4734.1	1e-15	4e-09	19	1.1	65.4
1AX8	1003	6e-13	2e-08	4e-06	17	3.0	6397.0	2e-16	4e-09	18	1.7	61.1
1TJO	1394	5e-13	2e-08	5e-06	21	12.3	-	2e-15	9e-10	29	2.3	77.4
1RGS	2015	3e-12	8e-08	6e-06	39	16.1	-	1e-15	4e-09	31	2.3	168.8
1TOA	2138	9e-12	2e-07	8e-06	49	16.8	-	2e-15	2e-09	31	1.0	142.4
1KDH	2846	3e-11	2e-06	2e-05	150	21.9	-	5e-16	2e-09	40	2.1	199.1
1NFG	3501	2e-12	3e-05	1e-04	325	21.2	-	3e-15	3e-09	43	1.0	275.8
1BPM	3672	1e-12	5e-05	2e-04	396	23.8	-	5e-17	8e-10	40	1.4	438.6
1MQQ	5510	1e-14	4e-04	1e-03	911	26.0	-	2e-15	1e-09	61	1.3	947.1

Table 3: Computational results on regularized molecular conformation problems. "-" indicates that the solver is terminated because the maximum running time of four hours is reached.

where x_i is the estimated position and $\hat{x}_i$ is the actual position. Note that a smaller RMSD means a better estimation, and an RMSD of less than 2Å is considered to be good in molecular conformation.

The computational results are presented in Table 3. It is clear that both methods return nearly feasible solutions. However, we can see that BM-Global outperforms QSDPNAL in all other measures. In particular, QSDPNAL often returns solutions with a large suboptimality measure, and those solutions tend to be of a higher rank and give a larger RMSD. On the other hand, the solutions returned by BM-Global are always of low rank with very small RMSD. Moreover, by the presented computational time, we see that BM-Global is much more efficient than QSDPNAL, and its generated solutions are also often much more accurate.

The results in this and the previous subsections also suggest that the numerical performance of QSDPNAL may depend on the sign of λ while BM-Global is robust with respect to parameter selection of λ.

7. Conclusions

In this work, we proposed an efficient algorithm BM-Global for solving the low-rank matrix optimization problem. We utilized both the efficiency from a smooth objective of the Burer-Monteiro decomposition approach and the convexity and partial smoothness of the nuclear-norm-regularized convex form to obtain a highly efficient algorithm with sound theoretical guarantees. Extensive numerical experiments showed that our proposed algorithm outperforms the state of the art for low-rank matrix optimization. Based on

this research, we have released an open-source package of the proposed BM-Global at `https://www.github.com/leepei/BM-Global/`.

Acknowledgement

We thank Po-Wei Wang for providing us the source code of PolyMF-SS, and Ting Kei Pong for pointing us to the work of Hou et al. (2013). This work was partially done when Lee was visiting the Department of Mathematics at the National University of Singapore. Lee's research was supported in part by the JSPS Grant-in-Aid for Research Activity Start-up 23K19981 and Grant-in-Aid for Early-Career Scientists 24K20845. Toh's research is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 3 grant call (MOE-2019-T3-1-010).

References

Naman Agarwal, Zeyuan Allen-Zhu, Brian Bullins, Elad Hazan, and Tengyu Ma. Finding approximate local minima faster than gradient descent. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1195–1199, 2017.

Gilles Bareilles, Franck Iutzeler, and Jérôme Malick. Newton acceleration on manifolds identified by proximal-gradient methods. *Mathematical Programming*, 2022. Online first.

Jonathan Barzilai and Jonathan M Borwein. Two-point step size gradient methods. *IMA journal of numerical analysis*, 8(1):141–148, 1988.

Heinz H. Bauschke and Patrick L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, second edition, 2017.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009a.

Amir Beck and Marc Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE transactions on image processing*, 18(11):2419–2434, 2009b.

Jérôme Bolte, Trong Phong Nguyen, Juan Peypouquet, and Bruce W Suter. From error bounds to the complexity of first-order descent methods for convex functions. *Mathematical Programming*, 165(2):471–507, 2017.

Silvia Bonettini, Ignace Loris, Federica Porta, and Marco Prato. Variable metric inexact line-search-based methods for nonsmooth optimization. *SIAM Journal on Optimization*, 26(2):891–921, 2016.

N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15(42):1455–1459, 2014. URL `https://www.manopt.org`.

Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. Accelerated methods for nonconvex optimization. *SIAM Journal on Optimization*, 28(2):1751–1772, 2018.

Yuxin Chen, Yuejie Chi, Jianqing Fan, and Cong Ma. Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval. *Mathematical Programming*, 176(1):5–37, 2019.

Patrick L Combettes. Solving monotone inclusions via compositions of nonexpansive averaged operators. *Optimization*, 53(5-6):475–504, 2004.

Aris Daniilidis, Dmitriy Drusvyatskiy, and Adrian S. Lewis. Orthogonal invariance and identifiability. *SIAM Journal on Matrix Analysis and Applications*, 35(2):580–598, 2014.

RPW Duin and E Pekalska. Datasets and tools for dissimilarity analysis in pattern recognition (tech. rep. no. 2009 9). simbad (eu, fp7, fet), 2009.

Xingyuan Fang and Kim-Chuan Toh. Using a distributed SDP approach to solve simulated protein molecular conformation problems. In *Distance Geometry*, pages 351–376. Springer, 2013.

Rong Ge, Jason D. Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 2981–2989, 2016.

Rong Ge, Chi Jin, and Yi Zheng. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. In *International Conference on Machine Learning*, pages 1233–1242, 2017.

Anis Hamadouche, Yun Wu, Andrew M. Wallace, and Joao F. C. Mota. Sharper bounds for proximal gradient algorithms with errors, 2022. arXiv:2203.02204.

Warren L. Hare and Adrian S. Lewis. Identifying active constraints via partial smoothness and prox-regularity. *Journal of Convex Analysis*, 11(2):251–266, 2004.

Ke Hou, Zirui Zhou, Anthony Man-Cho So, and Zhi-Quan Luo. On the linear convergence of the proximal gradient method for trace norm regularization. *Advances in Neural Information Processing Systems*, 26, 2013.

Cho-Jui Hsieh and Peder Olsen. Nuclear norm minimization via active subspace selection. In *International Conference on Machine Learning*, pages 575–583, 2014.

Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674, 2013.

Kaifeng Jiang, Defeng Sun, and Kim-Chuan Toh. An inexact accelerated proximal gradient method for large scale linearly constrained convex SDP. *SIAM Journal on Optimization*, 22(3):1042–1064, 2012.

Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M. Kakade, and Michael I. Jordan. How to escape saddle points efficiently. In *International Conference on Machine Learning*, pages 1724–1732, 2017.

Krzysztof Kurdyka. On gradients of functions definable in o-minimal structures. In *Annales de l'institut Fourier*, volume 48, pages 769–783, 1998.

Ching-pei Lee. Accelerating inexact successive quadratic approximation for regularized optimization through manifold identification. *Mathematical Programming*, 2023. In press.

Ching-pei Lee and Stephen J. Wright. Inexact successive quadratic approximation for regularized optimization. *Computational Optimization and Applications*, 72:641–674, 2019.

Ching-pei Lee and Stephen J. Wright. Revisiting superlinear convergence of proximal Newton methods to degenerate solutions, 2022.

Jason D. Lee, Ioannis Panageas, Georgios Piliouras, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. First-order methods almost always avoid strict saddle points. *Mathematical programming*, 176(1):311–337, 2019.

Adrian S. Lewis. Active sets, nonsmoothness, and sensitivity. *SIAM Journal on Optimization*, 13(3):702–725, 2002.

Adrian S. Lewis and Michael L. Overton. Eigenvalue optimization. *Acta numerica*, 5: 149–190, 1996.

Adrian S. Lewis and Shanshan Zhang. Partial smoothness, tilt stability, and generalized Hessians. *SIAM Journal on Optimization*, 23(1):74–94, 2013.

Xudong Li, Defeng Sun, and Kim-Chuan Toh. QSDPNAL: A two-phase augmented Lagrangian method for convex quadratic semidefinite programming. *Mathematical Programming Computation*, 10(4):703–743, 2018.

Stanisław Łojasiewicz. Une propriété topologique des sous-ensembles analytiques réels. In *Les Équations aus Dérivées Partielles*. Éditions du centre National de la Recherche Scientifique, 1963.

Fan Lu, Sündüz Keleş, Stephen J Wright, and Grace Wahba. Framework for kernel regularization with application to protein clustering. *Proceedings of the National Academy of Sciences*, 102(35):12332–12337, 2005.

Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 11:2287–2322, 2010.

Boris S. Mordukhovich, Xiaoming Yuan, Shangzhi Zeng, and Jin Zhang. A globally convergent proximal newton-type method in nonsmooth convex optimization. *Mathematical Programming*, pages 1–38, 2022.

Yurii Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

Luong Trung Nguyen, Junhan Kim, and Byonghyo Shim. Low-rank matrix completion: A contemporary survey. *IEEE Access*, 7:94215–94237, 2019.

Liam O'Carroll, Vaidehi Srinivas, and Aravindan Vijayaraghavan. The Burer-Monteiro SDP method can fail even above the Barvinok-Pataki bound. In *Advances in Neural Information Processing Systems*, 2022.

Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, pages 713–719, 2005.

R. Tyrrell Rockafellar and Roger J.-B. Wets. *Variational analysis*. Springer Science & Business Media, 2009.

Clément W. Royer and Stephen J. Wright. Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. *SIAM Journal on Optimization*, 28(2): 1448–1477, 2018.

Clément W Royer, Michael O'Neill, and Stephen J. Wright. A Newton-CG algorithm with complexity guarantees for smooth unconstrained optimization. *Mathematical Programming*, 180(1):451–488, 2020.

Katya Scheinberg and Xiaocheng Tang. Practical inexact proximal quasi-Newton method with global complexity analysis. *Mathematical Programming*, 160(1-2):495–529, 2016.

Mark Schmidt, Nicolas Roux, and Francis Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in Neural Information Processing Systems*, pages 1458–1466, 2011.

Ruoyu Sun and Zhi-Quan Luo. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, 2016.

Kim-Chuan Toh and Sangwoon Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of optimization*, 6(615-640):15, 2010.

Tian Tong, Cong Ma, and Yuejie Chi. Accelerating ill-conditioned low-rank matrix estimation via scaled gradient descent. *Journal of Machine Learning Research*, 22(150):1–63, 2021.

Po-Wei Wang, Chun-Liang Li, and J. Zico Kolter. Polynomial optimization methods for matrix factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Stephen J. Wright, Robert D. Nowak, and Mário A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

Xingyu Xu, Yandi Shen, Yuejie Chi, and Cong Ma. The power of preconditioning in overparameterized low-rank matrix sensing. In *International Conference on Machine Learning*, pages 38611–38654. PMLR, 2023.

Baturalp Yalcin, Haixiang Zhang, Javad Lavaei, and Somayeh Sojoudi. Factorization approach for low-complexity matrix completion problems: Exponential number of spurious solutions and failure of gradient methods. In *The 25th International Conference on Artificial Intelligence and Statistics*, 2022.

Heng Yang, Ling Liang, Luca Carlone, and Kim-Chuan Toh. An inexact projected gradient method with rounding and lifting by nonlinear programming for solving rank-one semidefinite relaxation of polynomial optimization. *Mathematical Programming*, 2022. Online first.

Quanming Yao and James T. Kwok. Accelerated and inexact soft-impute for large-scale matrix and tensor completion. *IEEE Transactions on Knowledge and Data Engineering*, 41(11):2628–2643, 2018.

Tian Ye and Simon S. Du. Global convergence of gradient descent for asymmetric low-rank matrix factorization. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Man-Chung Yue, Zirui Zhou, and Anthony Man-Cho So. A family of inexact SQA methods for non-smooth convex minimization with provable convergence guarantees based on the Luo–Tseng error bound property. *Mathematical Programming*, 174(1-2):327–358, 2019.

Appendices

Table of Contents

Appendix A. Implementation Details

A.1 Matrix Completion

We now describe our implementation details of BM-Global that are tailored for the matrix completion problem. In particular, we will discuss the mechanism for deciding α_t in (13), the algorithm for obtaining the approximate eigendecomposition using only matrix-vector products, details of the safeguard to ensure $\epsilon_t \downarrow 0$ in (13), initialization strategy for X_0, the solver for (BM-nuclear) and (BM-PSD), and the degree of parallelism of our algorithm.

To avoid redundancy, we focus on the case of (BM-nuclear) in this section, and keep in our mind that it can be easily adapted to the case of (BM-PSD) by straightforward changes from SVDs to eigendecompositions.

A.1.1 Approximate SVD

Let us denote

$$Z_t \coloneqq \tilde{X}_t - \alpha_t \nabla f(\tilde{X}_t). \tag{79}$$

For the proximal operation (5), since $m \leq n$ by our assumption, it is cheaper to compute an approximate eigendecomposition of $Z_t Z_t^\top$ (instead of $Z_t^\top Z_t$) to get

$$Z_t Z_t^\top \approx \hat{U}_t \mathrm{diag}\left(\left(\bar{\Sigma}_t\right)^2\right) \hat{U}_t^\top \tag{80}$$

for some $\bar{\Sigma}_t \in \mathbb{R}^{k_t}$ with $\bar{\Sigma}_t > 0$ and some orthogonal $\hat{U}_t \in \mathbb{R}^{m \times k_t}$ for a given rank k_t. The notation $(\bar{\Sigma}_t)^2$ denotes the element-wise square. Note that here we have removed the eigenvectors corresponding to the eigenvalue 0 as it does not affect the product matrix. We then conduct an exact SVD on $\hat{U}_t^\top Z_t \in \mathbb{R}^{k_t \times n}$ (whose calculation can be done without forming Z_t or $\tilde{X}_t$ explicitly) to obtain

$$\hat{U}_t^\top Z_t = \tilde{U}_t \mathrm{diag}\left(\hat{\Sigma}_t\right) V_t^\top, \tag{81}$$

with cost $O(k_t^3 + nk_t^2)$, which is much cheaper than SVD for Z_t when $k_t \ll m$. The approximate SVD of Z_t is then obtained through

$$Z_t \approx U_t \mathrm{diag}\left(\hat{\Sigma}_t\right) V_t^\top =: \tilde{Z}_t, \quad U_t \coloneqq \hat{U}_t \tilde{U}_t, \tag{82}$$

where $\tilde{Z}_t$ coincides with the one we used in (11) and (12). Clearly, U_t and V_t are both orthonormal, so this is indeed a valid SVD for the matrix

$$\tilde{Z}_t = \hat{U}_t \hat{U}_t^\top Z_t = P_{\mathrm{ran}(\hat{U}_t)}(Z_t).$$

The inexact proximal gradient step is then finished as

$$X_t^+(\alpha_t) = U_t \mathrm{diag}\left(\Sigma_t\right) V_t^\top, \quad \Sigma_t \coloneqq \left[\hat{\Sigma}_t - \lambda \alpha_t e\right]_+. \tag{83}$$

Note that we only store U_t, Σ_t, V_t but do not explicitly form X_{t+1}. Here we slightly abuse the notation to let Σ_t denote only the coordinates of the thresholded vector that have a nonzero value, and let U_t and V_t contain only the columns corresponding to these values to save spatial and computational cost.

A.1.2 An Efficient Algorithm for Approximate Eigendecompositions Using Only Matrix-Matrix Products

As mentioned before, forming X_t and $\tilde{X}_t$ is prohibitively expensive. Therefore, for computing (80), we need to rely on iterative methods that only require evaluating matrix multiplications involving Z_t and $Z_t^\top$, so that we can utilize the decomposition of $\tilde{X}_t = W_t H_t^\top$ as well as the structured assumption of $\nabla f(X)$ made in Section 1. A highly efficient and robust approach is the limited memory block Krylov subspace method (LmSVD) proposed by Liu et al. (2013); see also the references therein for other popular algorithms. LmSVD is an extension of the classic simple subspace iteration (SSI) method for computing the extremal eigenvalues and eigenvectors that extends from the renowned power method (i.e., $k_t = 1$). For any initial guess for right-singular vectors $U^{t,0} \in \mathbb{R}^{m\times k_t}$, the SSI method computes the new iterates $U^{t,i}$ via

$$U^{t,i} \leftarrow \mathrm{orth}\left(Z_t Z_t^\top U^{t,i-1}\right), \quad \forall\, i \geq 1, \tag{84}$$

where $\mathrm{orth}(M)$ extracts an orthonormal basis for the range space of the given matrix M, and i is the iteration counter for SSI. For the ease of description, we abstract the operation $Z_t Z_t^\top(U)$ for an input U as a self-adjoint semidefinite operator $L_t(\cdot)$. Note that in each iteration of SSI, one needs to perform two matrix multiplications (one for $Z_t^\top$ and the other for Z_t) and one orthonormalization that cost $O(mnk_t)$ and $O(mk_t^2)$ flops, respectively. In our case, suppose that $k_t \ll m \leq n$, then the main computational bottleneck is the matrix products. Therefore, LmSVD tries to accelerate the practical convergence via cutting down the total number of iterations to reduce such matrix products without incurring additional heavy computation. To achieve this goal, LmSVD finds the next iterate via replacing $U^{t,i}$ in (84) with an improved candidate $\hat{U}^{t,i}$ via solving the following constrained optimization problem:

$$\hat{U}^{t,i} = \mathrm{argmax}\left\{\langle U,\, Z_t Z_t^\top U\rangle \mid U^T U = I_{k_t},\ \mathrm{ran}(U) \subseteq \mathcal{S}_{t,i}\right\}, \tag{85}$$

where the subspace $\mathcal{S}_{t,i} \subseteq \mathbb{R}^m$ is selected as

$$\mathcal{S}_{t,i} = \mathrm{ran}\left(U^{t,i}, U^{t,i-1}, \ldots, U^{t,i-p_t}\right), \quad U^{t,j} \coloneqq L_t(\hat{U}^{t,j-1}), \quad j = i, i-1, \ldots, i-p_t$$

for some pre-specified $p_t \geq 0$. Let $q_t \coloneqq k_t(p_t+1)$ and

$$P^{t,i} \coloneqq [U^{t,i}, U^{t,i-1}, \ldots, U^{t,i-p_t}] \in \mathbb{R}^{m\times q_t},$$

then $U \in \mathcal{S}_{t,i}$ if and only if there exists $V \in \mathbb{R}^{q_t\times k_t}$ such that

$$U = P^{t,i} V. \tag{86}$$

Direct computation then shows that (85) is equivalent to

$$\max_{V\in\mathbb{R}^{q_t\times k_t}} \langle V,\, ((P^{t,i})^\top L_t(P^{t,i}V)\rangle \quad \text{s.t.} \quad V^\top((P^{t,i})^\top P^{t,i})V = I_{k_t}. \tag{87}$$

However, the matrix $P^{t,i}$ may be rank deficient to cause numerical issues in solving (87). To resolve this issue, LmSVD replaces $P^{t,i}$ with an orthonormal basis of $\mathrm{ran}(P^{t,i})$. To extract such a basis, since $U^{t,i}$ (i.e., the first block in matrix $P^{t,i}$) always has a full column rank, LmSVD first projects the remaining blocks in $P^{t,i}$ to $\ker((U^{t,i})^\top)$ to form

$$P_{t,i} \coloneqq \left(I_m - U^{t,i}(U^{t,i})^\top\right)\left[U^{t,i-1}, \ldots, U^{t,i-p}\right] \in \mathbb{R}^{m\times p_t k_t}, \tag{88}$$

and then consider its orthonormalization. We denote the eigendecomposition of $P_{t,i}^\top P_{t,i} \in \mathbb{R}^{p_t k_t \times p_t k_t}$ by

$$P_{t,i}^\top P_{t,i} = \tilde{U}_{P_{t,i}} \tilde{\Lambda}_{P_{t,i}} \tilde{U}_{P_{t,i}}^T,$$

for matrices $\tilde{U}_{P_{t,i}}, \tilde{\Lambda}_{P_{t,i}} \in \mathbb{R}^{p_t k_t \times p_t k_t}$ with $\tilde{U}_{P_{t,i}}$ orthonormal and $\tilde{\Lambda}_{P_{t,i}}$ diagonal. Clearly, if $\tilde{\Lambda}_{P_{t,i}}$ is nonsingular,

$$\hat{P}^{t,i} \coloneqq \left[U^{t,i}, P_{t,i}\tilde{U}_{P_{t,i}}\tilde{\Lambda}_{P_{t,i}}^{-1/2}\right] \tag{89}$$

is an orthonormal basis of $\mathrm{ran}(P^{t,i})$. In practice, we can drop those columns of $\hat{P}^{t,i}$ that correspond to nearly zero eigenvalues of $P_{t,i}^\top P_{t,i}$. Moreover, there may exist columns of $\hat{P}^{t,i}$ whose norms are nearly zero. To stabilize the numerical computation, one might also want to drop these columns. From here on, we always assume that $\hat{P}^{t,i} \in \mathbb{R}^{m\times q_{t,i}}$, for some $q_{t,i} > 0$, forms an orthonormal basis of $\mathrm{ran}(P^{t,i})$ and it is obtained via performing the above two trimming procedures to the matrix in (89).

After knowing an orthonormal basis $\hat{P}^{t,i}$i of $P^{t,i}$, we then express any $U \in \mathcal{S}_{t,i}$ as

$$U = \hat{P}^{t,i}V$$

for some $V \in \mathbb{R}^{q_{t,i}\times k_t}$. The above expression then yields the following optimization problem to be solved at each iteration of LmSVD.

$$\max_{V\in\mathbb{R}^{q_{t,i}\times k_t}} \langle V, L^{t,i}V\rangle \quad \text{s.t.} \quad V^\top V = I_{k_t}, \tag{90}$$

where $L^{t,i} \coloneqq (\hat{P}^{t,i})^\top L_t(\hat{P}^{t,i}) \in \mathbb{R}^{q_{t,i}\times q_{t,i}}$. The solution $V_*^{t,i}$ for problem (90) is nothing but the k_t leading eigenvectors of the matrix $L^{t,i}$. Therefore, we can compute the full spectral decomposition of $L^{t,i}$ to get $V_*^{t,i}$ and the computational cost is acceptable provided that p_t and k_t are small. The overall algorithm of LmSVD is summarized in Algorithm 2.

We emphasize that LmSVD has an efficient and robust official implementation by Liu et al. (2013).[5] In the present work, we borrow most parts of the implementation of Liu et al. (2013) but impose some minor modifications to adapt for our purposes. First, as we shall see in the following subsection, instead of using a randomly generated initial point $U^{t,0}$, we use a more sophisticated initialization scheme. Second, the implementation of Liu et al. (2013) terminates by following a two-level strategy, but in our implementation, we simply terminate the algorithm as long as the difference between the eigenvalues of $L^{t,i}$ and $L^{t,i-1}$ is small.

5. Available at `https://www.mathworks.com/matlabcentral/fileexchange/46875-lmsvd-m`.

Algorithm 2: LmSVD$(L_t, U^{t,0}, p_t)$

input : A self-adjoint semidefinite operator L_t over $\mathbb{R}^m$, an initial guess $U^{t,0} \in \mathbb{R}^{m\times k_t}$, a threshold $\epsilon > 0$ for ruling out small eigenvalues, and an integer $p_t > 0$

$U^{t,0} \leftarrow \mathrm{orth}(U^{t,0})$

for $i = 0, \ldots,$ **do**

  $p \leftarrow \min\{p_t, i\}$

  $\mathcal{S}_{t,i} \leftarrow \mathrm{ran}\left(U^{t,i}, U^{t,i-1}, \ldots, U^{t,i-p}\right)$ `*Block Krylov subspace selection`

  $\hat{U}^{t,i} \leftarrow \mathrm{argmax}\left\{\langle U, L_t(U)\rangle \mid U^\top U = I_{k_t}, U \in \mathcal{S}_{t,i}\right\}$ `*Block subspace optimization`

  $U^{t,i+1} \leftarrow \mathrm{orth}\left(L_t(\hat{U}^{t,i})\right)$ `*Orthonormalization`

$s \leftarrow \left(\mathrm{diag}\left((U^{t,i+1})^\top L(U^{t,i+1})\right)\right)^{1/2}$

$J_s \leftarrow \mathrm{find}(s > \epsilon)$ `*Rule out small eigenvalues and associated eigenvectors`

$\hat{U}_t \leftarrow (U^{t,i+1})_{:,J_s}$

output: $\hat{U}_t$

A.1.3 Ensuring Sufficient Precision in the Proximal Operation

We notice that (83) suggests that all entries in $\bar{\Sigma}_t$ smaller than the threshold as well as their corresponding columns of V_t and U_t do not contribute to the calculation of $X_t^+(\alpha_t)$, so we just need to compute the entries not truncated by the proximal operation. Therefore, if $\mathrm{rank}(X_{t+1}) = k_{t+1}$, ideally we just need to compute the first k_{t+1} eigenvalues in our approximate eigendecomposition at the tth iteration of proximal gradient. On the other hand, to ensure that we are recovering a global solution X^* of (CVX), it is necessary to check that the ranks of the iterates are large enough so that we do not get stuck at an approximation of X^* with an insufficient rank. To safeguard that our algorithm converges to a global optimum, or more explicitly, to make ϵ_t in (13) decrease to 0 fast enough (see Section 4), we want to ensure that eventually the smallest eigenvalue we obtain will be truncated out, so that we can be certain that all eigenvalues/eigenvectors that contribute to the computation of $X_t^+(\alpha_t)$ have already been obtained. Therefore, we will need a mechanism to adaptively adjust the rank of X_t. As the decrease of the rank is achieved by the truncation in the proximal operation, following our usage of LmSVD described in the previous subsection, what we need is a way to make the initial guess $U^{t,0}$ input to LmSVD have a rank sufficiently higher than that of X_t and $\tilde{X}_t$.

As noted in Lemma 1, we know that the output of approximately solving (BM-nuclear) should be close to the singular vectors of $\tilde{X}_t$ (up to column-wise scaling). Moreover, when X_t and $\tilde{X}_t$ are close to a global optimum X^*, we expect that X_{t+1} will be close to X^* and thus also to X_t and $\tilde{X}_t$, so the SVD of X_{t+1} is also expected to be close to that of X_t and $\tilde{X}_t$. We therefore use U_t from (82) and W_t from the output of the BM phase to form

$$\hat{U}^{t,0} \coloneqq \mathrm{orth}\left([U_t, W_t]\right) \tag{91}$$

as the base of our the warmstart input to the approximate eigendecomposition in obtaining X_{t+1} from Z_t. If the BM phase is not entered or does not produce any change in the iterate, we use

$$\hat{U}^{t,0} \coloneqq \operatorname{orth}\left([U_t, U_{t-1}]\right) \tag{92}$$

instead. To further guarantee that $\epsilon_t \to 0$ in (13), we need to ensure that the rank of $U^{t,0}$ is sufficiently large, and that $\hat{U}^{t,0}$ will approach the singular vectors corresponding to the singular values not truncated out. Ideally, we hope that the output of our approximate eigendecomposition will be exactly all the eigenvalues or singular values that are retained nonzero, plus the largest one that is truncated out in (5). Therefore, we add in one column with randomness to R_t whenever the rank of X_t and X_{t-1} are the same (namely, the rank has stopped increasing) and there is at most one eigenvalue truncated out in the inexact proximal gradient step at the $(t-1)$th iteration. The idea is that the case of truncating only one eigenvalue is the ideal scenario we want and we want the next iteration to still have one eigenvalue to truncate as the safeguard, while if no truncation happened, then we should continue increasing the rank. Utilizing this idea, we retain the singular vector u that corresponds to the largest truncated singular value in the latest iteration where such a truncation took place, and compute its projection u_t to $\ker((\hat{U}^{t,0})^\top)$. (It is possible and acceptable that $u_t = 0$.) The warmstart input is finally formed by

$$U^{t,0} \coloneqq \operatorname{orth}\left(\left[\hat{U}^{t,0}, u_t + \xi_t\right]\right), \quad \xi_t \in \ker\left([\hat{U}^{t,0}, u_t]^\top\right), \quad \|\xi_t\| \le \psi_t, \tag{93}$$

where ξ_t is a random vector and $\{\psi_t\}$ is a sequence such that $\psi_t \downarrow 0$.

When the rank of X_t and $U^{t-1,0}$ are the same, it means no truncation took place in the proximal operation, and we view this as that the maintained u has been added to $U^{t,0}$ as a column in the next iteration when we call LmSVD to compute an approximate eigendecomposition. In this situation, we then seek the eigenvector that corresponds to the next eigenvalue truncated as our new u. When there is no more such vectors available, we simply add in a unit random vector that is orthogonal to the columns of $\hat{U}^{t,0}$.

Here we provide further explanations to our design above. Assume that the eigenvalues in the exact eigendecomposition of $Z_t Z_t^\top$ are $\sigma_1 \ge \sigma_2 \ge \ldots \ge \sigma_m$, $S_1 \coloneqq \{\sigma_1, \ldots, \sigma_k\}$ is the set of those eigenvalues that will not become zero after the truncation in (83), and $S_2 \coloneqq \{\sigma_{k+1}, \ldots, \sigma_m\}$ are those that will become zero after the truncation in (83). To cope with pathological cases in which some eigenvectors corresponding to some $S_3 \subsetneq S_1$ and some corresponding to $S_4 \subset S_2$ are obtained, we inject noise to u so that during the procedure of Algorithm 2, it will approach an eigenvector that corresponds to some value in $S_1 \setminus S_3$ instead of getting stuck at an eigenvector that will be truncated out. (Analysis of the classical SSI suggests that $U^{t,i}$ approaches the leading eigenvectors as long as no column is exactly a multiple of an eigenvector that corresponds to an eigenvalue with a smaller absolute value.) On the other hand, when we are close to an optimal solution X^* of (CVX), and the eigenvectors corresponding to S_1 are all identified or well-approximated, it is natural that we do not want to add in much noise in the initialization of Algorithm 2, as such noise will decelerate the convergence of Algorithm 2. Therefore, in our design, we only add u with noise to $U^{t,0}$ when $\operatorname{rank}(X_t) = \operatorname{rank}(X_{t-1})$ or $\operatorname{rank}(U^{t-1,0}) \le \operatorname{rank}(X_t) + 1$, namely when no truncation happened or when only one vector is truncated. In the latter case, adding u to $U^{t,0}$ is for the purpose of making $U^{t,0}$ contain one eigenvector that is

likely to be truncated, so that we can still have the safeguard for ensuring that we have found the correct rank in the approximate eigendecomposition. We then decrease the level of noise by a certain factor whenever $\operatorname{rank}(X_t) = \operatorname{rank}(R_t) - 1$. That is, when exactly only one eigenvalue is truncated. This corresponds to (93). For the noise level ψ_t, in our implementation, we start with $\psi_0 = 0.1$ so that the noise will not dominate $u_t + \xi_t$, and whenever we need to decrease the significance of the noise, we just let $\psi_t = \psi_{t-1}/2$, and otherwise we assign $\psi_t = \psi_{t-1}$.

A.1.4 Initialization

It can be clearly seen that the point of origin is a saddle point of (BM-nuclear), and this is also the case for many other popular problems that has the form of (BM). Therefore, it is essential to have an effective way to initialize X_0, or equivalently $(\tilde{W}_0, \tilde{H}_0)$, in Algorithm 1. Existing methods for (BM) usually take a random initialization, but such approaches often lead to an unideal initial objective even worse than using $X_0 = 0$. Moreover, it is hard to decide what is an appropriate value for the initial rank – too large the rank takes longer running time, but too small the rank might lead to slow convergence at the early stage. The most straightforward idea would be to conduct one (inexact) proximal gradient step from the origin, but the difficulty is that we will be in lack of a warmstart matrix for the approximate eigendecomposition in Algorithm 2, and we still need to decide the rank of this matrix.

To get a good initialization for X_0, we follow the recent developments in randomized numerical linear algebra by Halko et al. (2011); Martinsson (2019) to combine the HMT method (Halko et al., 2011) with the Nyström method (Nyström, 1930). We imagine that our starting point is actually $X_{-1} = 0$, and then conduct one step of inexact proximal gradient from there with the fixed stepsize $\alpha_{-1} = L^{-1}$ to initialize X_{-1}. Regarding the approximate SVD for

$$Z_{-1} \coloneqq X_{-1} - \alpha_{-1}\nabla f(X_{-1}) = -L^{-1}\nabla f(0),$$

we describe how to obtain the eigendecomposition of $A_{-1} \coloneqq Z_{-1}Z_{-1}^\top$. Given the initial rank k_{-1}, we start with a random matrix $\tilde{U}^{-1,0} \in \mathbb{R}^{m\times k_{-1}}$ whose entries are independently and identically distributed as the standard normal distribution. Then we conduct

$$Q_{-1} = \operatorname{orth}\left(A_{-1}\tilde{U}^{-1,0}\right)$$

as in the HMT method, and use this Q_{-1} as the sketching matrix in the Nyström method to consider the approximation

$$A_{-1} \approx A_{-1}Q_{-1}\left(Q_{-1}^\top A_{-1}Q_{-1}\right)^\dagger \left(A_{-1}Q_{-1}\right)^\top =: \hat{A}_{-1}, \tag{94}$$

where for any matrix B, $B^\dagger$ is its pseudo inverse. The approximation matrix $\hat{A}_{-1}$ here is not really explicitly computed, but only serves as an intermediate variable for our further process. Clearly, $\operatorname{rank}(\hat{A}_{-1}) \le k_{-1}$, and thanks to the randomness from $\tilde{U}_{-1,0}$ and therefore Q_{-1}, with high probability we have $\operatorname{rank}(\hat{A}_{-1}) = k_{-1}$. We can then compute the exact eigendecomposition for $\hat{A}_{-1}$ by separately considering $A_{-1}Q_{-1}$ and $(Q_{-1}^\top A_{-1}Q_{-1})^\dagger$. As

long as k_{-1} is small, the computation of both $A_{-1}Q_{-1}$ and $Q_{-1}^\top A_{-1}Q_{-1} \in \mathbb{R}^{k_{-1}\times k_{-1}}$ is affordable under our assumption of efficient matrix-matrix products involving ∇f, and so is the calculation of the pseudo inverse that costs $O(k_{-1}^3)$. For obtaining the exact eigendecomposition of $\hat{A}_{-1}$, we first compute a QR decomposition of $A_{-1}Q_{-1}$

$$\hat{Q}_{-1}\hat{R}_{-1} = A_{-1}Q_{-1},$$

where $\hat{Q}_{-1}$ is an orthonormal matrix and $\hat{R}_{-1}$ is upper triangular. We then obtain the eigendecomposition of

$$\hat{R}_{-1}(Q_{-1}^\top A_{-1}Q_{-1})^\dagger \hat{R}_{-1}^\top = \tilde{U}_{-1}\mathrm{diag}\left((\hat{\Sigma}_{-1})^2\right)\tilde{U}_{-1}^\top$$

which costs only $O(k_{-1}^3)$ in both forming the matrix on the left-hand side and calculating its eigendecomposition, as the matrices involved are all k_{-1} by k_{-1}. When $k_{-1} \ll m$, this cost is negligible in comparison to other steps. The eigendecomposition of $\hat{A}_{-1}$ is then obtained by setting $\hat{U}_0 = \hat{Q}_{-1}\tilde{U}_{-1}$ to get $\hat{U}_0$ in the right-hand side of (80). The next steps for initializing X_0 then directly follow the procedure of (81)–(83).

The remaining issue is to decide the rank k_{-1} for $\tilde{R}_{-1}$. The most naive idea is to set $k_{-1} = 1$ so that all computation in the initialization is of the lowest possible overhead. However, in this case, we will not be able to fully exploit the multicore advantage of modern computing devices. We therefore set the initial k_{-1} to be the number of cores we can use, so that all the matrix-matrix computation of this step can be fully parallelized without increasing the overhead for initialization.

A.1.5 Solver for (BM-nuclear)

The step of optimizing (BM-nuclear) relies on an off-the-shore solver, and to obtain the best efficiency, the choice should be application-dependent. As we aim for large-scale problems and wish to fully exploit multicore parallelization ubiquitous in modern computers, we will use methods that efficiently utilize multiple computational cores. For (BM-nuclear), as the objective is smooth, many algorithms are available for this choice, ranging from asynchronous, low-order ones to synchronous, high-order approaches. For the low-order ones, we note that since each row of W affects different entries in $WH^\top$, one can update multiple rows simultaneously if $f(X)$ is separable, and the same argument applies to the update of H as well. Therefore, as long as k is no fewer than the number of cores, we should be able to enjoy full parallelism. For problems like (MF), many efficient algorithms such as those by Yu et al. (2014); Zhuang et al. (2013); Wang et al. (2017) are readily available. On the other hand, for (BM-PSD), the property of the constraint set C will limit our choices for the solver. In most applications, however, the constraint $WW^\top \in C$ can be formulated as a smooth manifold $\mathcal{M}$ for W, and one can utilize efficient manifold optimization approaches for such problems; see, for example, Absil et al. (2009); Boumal et al. (2014). The major computation in such manifold optimization approaches is usually the computation of the Riemannian gradient and the Hessian-vector products, which are by nature parallelizable.

A.1.6 Parallelism

A key to cope with large-scale problems in modern computing environments is to utilize multicore parallelization. Although inherently the bottleneck of Algorithm 2, (84), can be

highly parallelized, in practice usually the parallelism is low as this operation is data-heavy but not computation-heavy, so higher parallelism is hindered by the memory bandwidth. On the other hand, solvers for (BM-nuclear) tend to be more computationally intensive, so they can often achieve better parallelism, by utilizing all cores available, than solvers for (CVX). By switching to the BM phase, our algorithm hence exploits better parallelism than state of the art for (CVX). The initial rank k in the previous subsection is also selected to fully exploit the advantage of multiple cores from the beginning on.

A.2 Quadratic SDP

A.2.1 The BM formulations for the RKE problem and the molecular conformation problem.

For the RKE problem and the molecular conformation problem, the corresponding factorized problem is given as

$$\min_{W\in\mathbb{R}^{n\times k}} \left(g\left(W\right) \coloneqq \frac{1}{2}\sum_{(i,j)\in\Omega} w_{ij}\left(\langle E_{ij},\, WW^\top\rangle - d_{ij}^2\right)^2 + \lambda\|W\|_F^2\right), \quad \text{s.t.} \quad W^\top e = 0. \tag{95}$$

The gradient of this function $g:\mathbb{R}^{n\times k}\to\mathbb{R}$ is

$$\nabla g(W) = 2\left(\sum_{(i,j)\in\Omega} w_{ij}\left(\langle E_{ij},\, WW^\top\rangle - d_{ij}^2\right)E_{ij} + \lambda I\right)W.$$

For any given $D\in\mathbb{R}^{n\times k}$, the Hessian operator of g performed on H can be computed as

$$\begin{aligned}\nabla^2 g(W)[D] = {} & 2\sum_{(i,j)\in\Omega} w_{ij}\langle E_{ij},\, WD^\top + DW^\top\rangle E_{ij}W \\ & + 2\sum_{(i,j)\in\Omega} w_{ij}\left(\langle E_{ij},\, WW^\top\rangle - d_{ij}^2\right)E_{ij}D + 2\lambda D.\end{aligned}$$

A.2.2 Implementation Details

Recall that for a given problem size n, Algorithm 1 iterates on a low-rank matrix $W_t\in\mathbb{R}^{n\times k_t}$ with $k_t < n$ dynamically adjusted. In our experiment, we choose the initial k_0 as

$$k_0 = \min\{100, \lfloor 0.15n\rfloor\}.$$

Given k_0, we then run APG for 100 iterations with this rank constraint to generate an initial point W_0. When applying the APG method, we never form the matrix $X \coloneqq WW^\top$ and use only eigendecompositions with rank k_0. In particular, we use MATLAB's built-in `eigs` subroutine to compute the largest k eigenpairs. After the initialization stage, at the t-th iteration of Algorithm 1, we first apply the Manopt solver to solve the factorized problem (95) with the initial point W_t to compute a matrix $\tilde{W}_t$. Using $\tilde{W}_t$, we then perform one inexact PG step with an approximate eigendecomposition with fixed step size $\alpha_t = 1/L$. If all the computed eigenvalues are positive, then it is highly possible that the current rank is too small. In this case, we set $k_{t+1} \leftarrow k_t + \lfloor \frac{n}{20}\rfloor$. Otherwise, we set k_{t+1} as the number of positive eigenvalues returned by `eigs`.

Algorithm 3: The APG method for solving (QSDP)

input : Initial point: $Y_0 = X_0 \in \mathbb{S}^n$, Lipschitz constant $L > 0$ and $\beta_0 = 1$

for $t = 0, \ldots,$ **do**

$X_{t+1} \leftarrow P_{\mathcal{X}}(Y_t - L^{-1}\nabla f(Y_t))$

$\beta_{t+1} \leftarrow \frac{1+\sqrt{1+4\beta_t^2}}{2}$

$Y_{t+1} \leftarrow X_{t+1} + \frac{\beta_t - 1}{\beta_{t+1}}(X_{t+1} - X_t)$

output: X_{t+1}

A.2.3 The accelerated proximal gradient method

Since we utilize the APG method for the initialization of BM-Global for (QSDP), for completeness, we provide a concise description of the APG method in Algorithm 3. To perform the projection $P_{\mathcal{X}}$ for (QSDP) through Lemma 8, we use the `eig` subroutine in MATLAB. (Note that the PG method is equivalent to setting $\beta_t \equiv 1$ in Algorithm 3.)

Appendix B. Additional Experimental Details and More Experiments

B.1 The SpaRSA variant for Matrix Completion

It is known that the convergence speed of standard PG could be slow, thus we also considered following Wright et al. (2009) to use the Barzilai-Borwein (BB) initialization (Barzilai and Borwein, 1988) with linesearch to decide the step size to accelerate the practical convergence of the convex lifting step. Given $\alpha_{\max} \geq \alpha_{\min} > 0$, we first compute

$$\alpha_t^{\mathrm{BB}} := \max\left\{\alpha_{\min}, \min\left\{\alpha_{\max}, \frac{\langle X_t - X_{t-1},\, \nabla f(X_t) - \nabla f(X_{t-1})\rangle}{\|X_t - X_{t-1}\|_F^2}\right\}\right\}. \tag{96}$$

But different from Wright et al. (2009), our backtracking linesearch does not search for an α_t that gives sufficient descent. Instead, given $\beta, \delta \in (0,1)$, we find the smallest nonnegative integer i such that $\alpha_t = \alpha_t^{\mathrm{BB}}\beta^i$ satisfies (14) or (15).

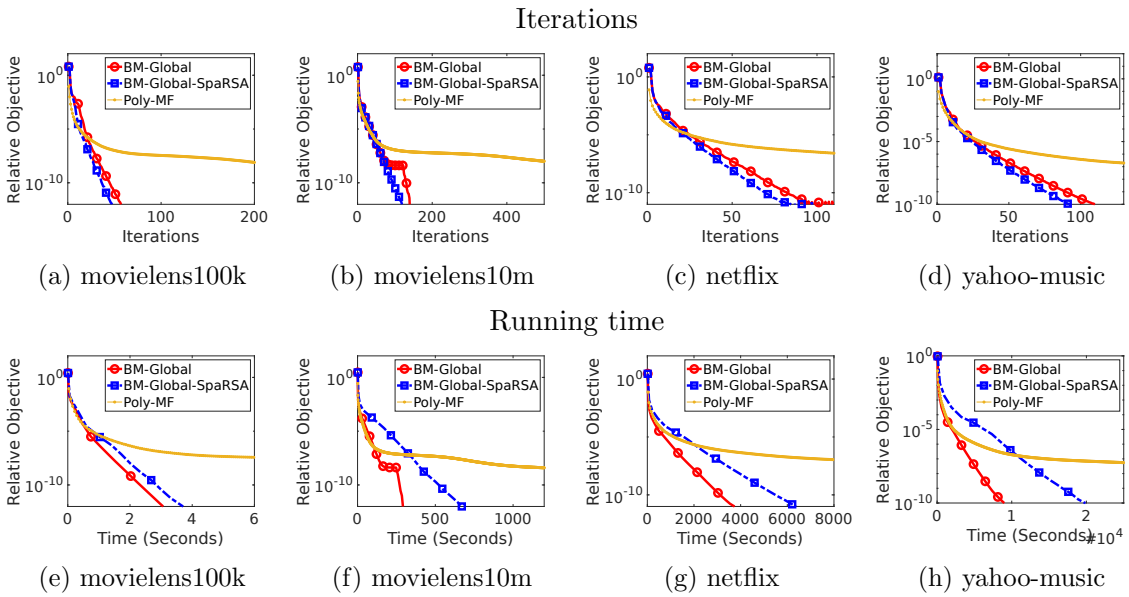
In our scenario, we only have the factorized form $X_t = W_t H_t^\top$ for each t but not the iterate X_t itself, so we use the following formula to calculate the numerator of (96), which is also used as the last term in (14).

$$\begin{aligned}
&\|X_t - X_{t-1}\|_F^2 \\
&= \langle X_t - X_{t-1},\, X_t - X_{t-1}\rangle \\
&= \langle X_t,\, X_t\rangle + \langle X_{t-1},\, X_{t-1}\rangle - 2\langle X_t,\, X_{t-1}\rangle \\
&= \mathrm{tr}\left((W_t^\top W_t)(H_t^\top H_t)\right) + \mathrm{tr}\left((W_{t-1}^\top W_{t-1})(H_{t-1}^\top H_{t-1})\right) + \mathrm{tr}\left((W_{t-1}^\top W_t)(H_{t-1}^\top H_t)\right).
\end{aligned} \tag{97}$$

Similar calculation is also applied to compute the denominator of (96). If $\mathrm{rank}(X_t) = k_t$ and $\mathrm{rank}(X_{t-1}) = k_{t-1}$, the cost of computing (97) is $O((m+n)(k_t^2 + k_{t-1}^2 + k_t k_{t-1})$.

B.2 Experimental Details of Section 6.1.1

We report the required running time for making the relative objective smaller than $\epsilon \in \{10^{-4}, 10^{-8}, 10^{-12}\}$ for both the fixed-step variant and the SpaRSA variant of our algorithm.



(a) movielens100k (b) movielens10m (c) netflix (d) yahoo-music

(e) movielens100k (f) movielens10m (g) netflix (h) yahoo-music

Figure 4: Performance of the SpaRSA variant of BM-Global. Top row: iterations v.s. relative objective. Bottom row: running time (seconds) v.s. relative objective.

The algorithms are terminated either when they have reached $\epsilon = 10^{-12}$ or when they have conducted 100 inexact proximal gradient steps. The results are shown in Tables 4 and 5.

Aside from the selection of $x = 1$ and $y = 3$ made in the main paper for the fixed-step variant, for SpaRSA, we also see that $x = 1$ and $y = 3$ is still robust and we thus use this setting for SpaRSA from now on.

B.3 Comparison with the SpaRSA variant for Matrix Completion

In this subsection, we present experimental results with the SpaRSA variant. The results are shown in Fig. 4. As expected, this variant is faster than the fixed-step variant in terms of iterations, although the difference is minor. On the other hand, it is significantly slower in terms of the real running time. The likely reason is the possible additional approximate SVDs executed in the backtracking procedure, while the inexact PG step in our algorithm is not the major contributor to the objective decrease, so the improvement in the convergence speed of this part cannot counterbalance the corresponding additional cost.

B.4 Parallelism

We also examine the parallelism of our method. In particular, we run our method with $1, 2, 4, 8$ cores and check how much time they respectively take to make (77) no larger than 10^{-6}, and compute the speedup as follows.

$$\text{Speedup}(x) = \frac{\text{Running time of } x \text{ cores}}{\text{Running time of 1 core}}.$$

We can observe from Fig. 5 that except for the small data set movielens100k, PolyMF-SS always achieves the highest speedup because most of its operations are inherently parallel

$\epsilon = 10^{-4}$

	y	1	2	3	4	5	6	7	8
SpaRSA	$x=1$	5.4e+0	1.1e+0	1.2e+0	1.4e+0	1.5e+0	1.6e+0	1.7e+0	1.8e+0
	$x=2$	1.4e+0	1.3e+0	1.8e+0	1.7e+0	1.9e+0	2.1e+0	2.4e+0	2.6e+0
	$x=3$	1.1e+0	1.1e+0	1.1e+0	1.2e+0	1.2e+0	1.2e+0	1.3e+0	1.3e+0
	$x=4$	1.3e+0	1.0e+0	1.1e+0	1.2e+0	1.6e+0	1.7e+0	1.8e+0	1.8e+0
	$x=5$	1.0e+0	1.1e+0	1.3e+0	1.3e+0	1.4e+0	1.5e+0	1.6e+0	1.7e+0
$\alpha_t \equiv 1$	$x=1$	1.2e+0	1.0e+0	1.2e+0	1.3e+0	1.4e+0	1.6e+0	1.7e+0	2.0e+0
	$x=2$	1.2e+0	1.5e+0	1.5e+0	1.7e+0	1.9e+0	2.1e+0	2.3e+0	2.5e+0
	$x=3$	1.5e+0	1.2e+0	1.1e+0	1.2e+0	1.2e+0	1.3e+0	1.3e+0	1.4e+0
	$x=4$	1.2e+0	1.3e+0	1.3e+0	1.4e+0	1.5e+0	1.6e+0	1.7e+0	1.7e+0
	$x=5$	1.3e+0	1.2e+0	1.3e+0	1.4e+0	1.5e+0	1.6e+0	1.8e+0	1.9e+0

$\epsilon = 10^{-8}$

	y	1	2	3	4	5	6	7	8
SpaRSA	$x=1$	1.2e+1	3.7e+0	4.9e+0	5.0e+0	5.5e+0	5.8e+0	7.0e+0	7.1e+0
	$x=2$	3.6e+0	6.3e+0	4.3e+0	4.9e+0	4.7e+0	5.8e+0	5.3e+0	5.6e+0
	$x=3$	3.2e+0	3.6e+0	4.2e+0	4.1e+0	4.5e+0	4.8e+0	5.2e+0	5.7e+0
	$x=4$	3.3e+0	3.2e+0	3.2e+0	3.4e+0	3.6e+0	3.8e+0	4.1e+0	4.3e+0
	$x=5$	3.3e+0	3.7e+0	3.5e+0	4.4e+0	4.5e+0	4.8e+0	5.1e+0	5.4e+0
$\alpha_t \equiv 1$	$x=1$	4.4e+0	3.7e+0	4.9e+0	5.4e+0	6.4e+0	7.1e+0	7.7e+0	8.4e+0
	$x=2$	4.4e+0	3.8e+0	4.2e+0	4.6e+0	5.1e+0	5.2e+0	5.5e+0	5.7e+0
	$x=3$	4.4e+0	3.8e+0	4.1e+0	4.6e+0	4.9e+0	5.2e+0	5.6e+0	6.0e+0
	$x=4$	5.0e+0	3.6e+0	3.6e+0	3.9e+0	4.0e+0	4.3e+0	4.5e+0	4.7e+0
	$x=5$	4.9e+0	4.1e+0	4.2e+0	4.5e+0	4.5e+0	4.8e+0	5.3e+0	5.3e+0

$\epsilon = 10^{-12}$

	y	1	2	3	4	5	6	7	8
SpaRSA	$x=1$	1.6e+1	7.4e+0	8.1e+0	7.9e+0	9.6e+0	9.1e+0	1.0e+1	1.2e+1
	$x=2$	1.1e+1	8.9e+0	7.7e+0	8.2e+0	8.4e+0	9.4e+0	9.2e+0	9.0e+0
	$x=3$	1.0e+1	7.0e+0	7.2e+0	7.2e+0	7.1e+0	7.3e+0	7.8e+0	8.6e+0
	$x=4$	9.0e+0	7.6e+0	7.1e+0	7.0e+0	7.4e+0	7.9e+0	7.6e+0	7.5e+0
	$x=5$	1.1e+1	7.8e+0	6.2e+0	6.8e+0	6.9e+0	7.3e+0	6.9e+0	7.9e+0
$\alpha_t \equiv 1$	$x=1$	1.2e+1	6.8e+0	8.0e+0	7.7e+0	8.8e+0	9.3e+0	1.0e+1	1.0e+1
	$x=2$	1.7e+1	6.9e+0	7.1e+0	7.1e+0	7.8e+0	7.8e+0	8.4e+0	8.7e+0
	$x=3$	1.4e+1	7.2e+0	6.9e+0	7.9e+0	7.8e+0	8.0e+0	8.6e+0	8.5e+0
	$x=4$	1.3e+1	7.7e+0	6.9e+0	7.4e+0	7.7e+0	8.0e+0	8.6e+0	8.8e+0
	$x=5$	1.7e+1	7.6e+0	6.5e+0	7.1e+0	7.4e+0	7.9e+0	8.4e+0	8.3e+0

Table 4: Time (seconds) required for solving (MC) on toy-example using Algorithm 1 to make (77) no larger than ϵ. We alternate between x consecutive inexact proximal gradient steps and y consecutive iterations of the BM phase solver. The fastest one for each case is labeled in red. "-" indicates that the designated ϵ is not reached within 100 inexact proximal gradient steps.

$\epsilon = 10^{-4}$

	y	1	2	3	4	5	6	7	8
SpaRSA	$x=1$	5.7e−1	5.2e−1	4.1e−1	4.6e−1	5.1e−1	5.4e−1	5.7e−1	6.0e−1
	$x=2$	2.0e+0	1.4e+0	7.0e−1	6.8e−1	5.7e−1	5.6e−1	4.9e−1	9.5e−1
	$x=3$	3.8e+0	2.4e+0	1.6e+0	1.6e+0	9.1e−1	9.3e−1	8.0e−1	5.8e−1
	$x=4$	4.1e+0	2.9e+0	2.3e+0	1.7e+0	1.3e+0	1.3e+0	7.7e−1	7.4e−1
	$x=5$	3.9e+0	2.6e+0	2.1e+0	1.5e+0	1.5e+0	1.4e+0	1.1e+0	9.7e−1
$\alpha_t \equiv 1$	$x=1$	7.8e−1	4.5e−1	4.4e−1	4.3e−1	5.3e−1	5.7e−1	6.0e−1	6.3e−1
	$x=2$	1.6e+0	8.0e−1	5.7e−1	5.2e−1	4.5e−1	4.9e−1	5.2e−1	5.5e−1
	$x=3$	2.2e+0	1.4e+0	1.0e+0	1.1e+0	8.1e−1	7.7e−1	5.3e−1	5.6e−1
	$x=4$	2.5e+0	1.6e+0	1.3e+0	1.3e+0	1.0e+0	9.4e−1	6.6e−1	6.3e−1
	$x=5$	2.8e+0	1.9e+0	1.6e+0	1.4e+0	1.1e+0	1.1e+0	8.2e−1	8.1e−1

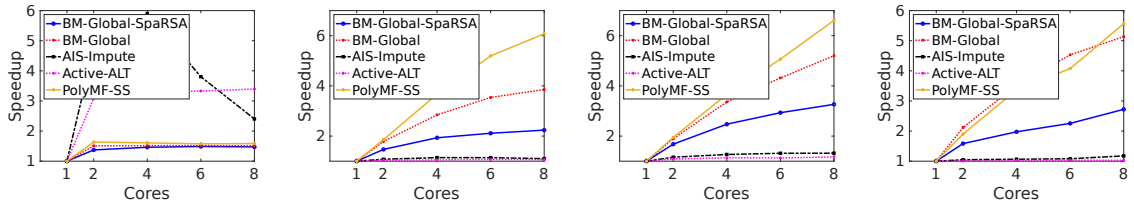
$\epsilon = 10^{-8}$

	y	1	2	3	4	5	6	7	8
SpaRSA	$x=1$	2.4e+0	2.2e+0	1.9e+0	1.7e+0	1.6e+0	1.6e+0	1.7e+0	1.7e+0
	$x=2$	4.6e+0	4.2e+0	3.2e+0	2.4e+0	2.1e+0	1.8e+0	1.7e+0	1.6e+0
	$x=3$	7.0e+0	5.2e+0	4.4e+0	3.5e+0	2.9e+0	2.7e+0	2.7e+0	2.4e+0
	$x=4$	7.6e+0	5.9e+0	4.8e+0	4.0e+0	3.5e+0	3.1e+0	3.0e+0	2.6e+0
	$x=5$	7.2e+0	5.8e+0	4.6e+0	4.2e+0	3.6e+0	3.4e+0	3.3e+0	2.9e+0
$\alpha_t \equiv 1$	$x=1$	2.4e+0	2.0e+0	1.7e+0	1.6e+0	1.6e+0	1.8e+0	2.0e+0	2.2e+0
	$x=2$	3.4e+0	2.7e+0	2.1e+0	1.9e+0	1.8e+0	1.7e+0	1.7e+0	1.7e+0
	$x=3$	4.3e+0	3.4e+0	2.8e+0	2.7e+0	2.3e+0	2.1e+0	2.1e+0	1.9e+0
	$x=4$	4.9e+0	3.8e+0	3.2e+0	3.0e+0	2.6e+0	2.4e+0	2.2e+0	2.1e+0
	$x=5$	-	4.2e+0	3.5e+0	3.2e+0	2.8e+0	2.6e+0	2.5e+0	2.3e+0

$\epsilon = 10^{-12}$

	y	1	2	3	4	5	6	7	8
SpaRSA	$x=1$	4.6e+0	3.8e+0	3.5e+0	3.3e+0	3.4e+0	3.2e+0	2.9e+0	2.9e+0
	$x=2$	7.0e+0	6.6e+0	5.3e+0	4.2e+0	3.8e+0	3.4e+0	3.2e+0	3.0e+0
	$x=3$	9.9e+0	7.5e+0	6.8e+0	5.4e+0	4.6e+0	4.5e+0	4.3e+0	4.0e+0
	$x=4$	-	8.8e+0	7.3e+0	6.2e+0	5.4e+0	5.1e+0	4.7e+0	4.2e+0
	$x=5$	-	8.8e+0	7.3e+0	6.2e+0	5.6e+0	5.5e+0	5.0e+0	4.6e+0
$\alpha_t \equiv 1$	$x=1$	5.5e+0	3.7e+0	3.3e+0	3.1e+0	3.0e+0	3.0e+0	3.1e+0	3.4e+0
	$x=2$	5.7e+0	4.5e+0	3.6e+0	3.4e+0	3.2e+0	3.2e+0	3.1e+0	3.0e+0
	$x=3$	-	5.2e+0	4.3e+0	4.3e+0	3.7e+0	3.6e+0	3.4e+0	3.3e+0
	$x=4$	-	5.8e+0	4.8e+0	4.5e+0	4.1e+0	3.9e+0	3.7e+0	3.5e+0
	$x=5$	-	-	5.3e+0	4.9e+0	4.4e+0	4.2e+0	3.9e+0	3.7e+0

Table 5: Time (seconds) required for solving (MC) on movielens100k using Algorithm 1 to make (77) no larger than ϵ. We alternate between x consecutive inexact proximal gradient steps and y consecutive iterations of the BM phase solver. The fastest one for each case is labeled in red. "-" indicates that the designated ϵ is not reached within 100 inexact proximal gradient steps.



(a) movielens100k (b) movielens10m (c) netflix (d) yahoo-music

Figure 5: Comparison of different algorithms in terms of speedup with respect to different number of cores.

and computationally heavy, and BM-Global comes the next, while AIS-Impute and Active-ALT barely exhibit any parallelism at all.

B.5 Numerical results for the APG method

This subsection presents numerical results of the APG method Algorithm 3 on testing problems considered in Section 6.2.

In our experiments, the APG method is executed with $Y_0 = X_0 = 0$ and a fixed step size $\alpha = 1/L$, where $L := \|\mathcal{A}\mathcal{A}\|_2$ is estimated by the `eigs` subroutine in MATLAB. To perform the projection $P_{\mathcal{X}}$ through Lemma 8, we use the `eig` subroutine in MATLAB. Our stopping condition for the APG method is

$$\frac{|f(X_{t+1}) - f(X_t)|}{1 + |f(X_t)|} \leq \texttt{tol},$$

where `tol` is the tolerance (chosen as 10^{-6} in our experiments). Moreover, we set the maximal number of iterations to be 10000 and the maximal computational time to be four hours. The computational results for the RKE problems and the molecular conformation problems are presented in Tables 6 and 7, respectively. We can observe from the presented results that the APG method usually performs worse than BM-Global and QSDPNAL. Indeed, it returns solutions with a lower accuracy and takes more computational time. Also, the numerical results suggest that the APG method could be sensitive to the sign of the parameter λ, since it performs much better in the cases of $\lambda > 0$ then in the cases of $\lambda < 0$.

References

Pierre-Antoine Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

Jonathan Barzilai and Jonathan M Borwein. Two-point step size gradient methods. *IMA journal of numerical analysis*, 8(1):141–148, 1988.

N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15(42):1455–1459, 2014. URL `https://www.manopt.org`.

Name	n	η_{prim}	η_{opt}	rnk	Time	Iter
BrainMRI	124	2e-16	1e-03	5	0.7	86
protein	213	3e-15	1e-03	27	6.7	285
CoilDelftDiff	288	2e-15	1e-03	32	9.0	234
coildelftsame	288	3e-15	2e-03	32	7.7	189
CoilYork	288	1e-15	1e-03	21	9.2	231
Chickenpieces-5-45	446	4e-16	2e-03	28	32.6	288
newgroups	600	3e-17	2e-03	84	69.7	341
flowcytodis	612	8e-16	1e-03	16	84.6	353
DelftPedestrians	689	2e-15	2e-03	70	105.0	377
WoodyPlants50	791	8e-16	2e-03	49	185.7	478
delftgestures	1500	1e-15	1e-02	106	283.3	191
zongker	2000	1e-15	4e-03	303	1236.5	451
polydish57	4000	2e-15	1e-02	139	8530.7	357
polydism57	4000	3e-15	1e-02	83	8023.2	337

Table 6: Computational results for the APG method on RKE problems.

Name	n	η_{prim}	η_{opt}	rnk	RMSD	Time	Iter
1PBM	126	4e-16	6e-06	14	7.4	30.1	5073
1AU6	161	2e-16	6e-06	15	6.8	55.9	5250
1PTQ	402	2e-15	2e-06	20	10.2	727.8	10000
1CTF	487	5e-16	2e-06	23	11.2	1250.5	10000
1HOE	558	1e-15	2e-06	24	11.6	1831.3	10000
1LFB	641	1e-15	2e-06	28	13.4	2379.5	10000
1PHT	666	5e-16	2e-06	28	12.2	2598.5	10000
1F39	767	1e-15	2e-06	33	13.6	3432.6	10000
1DCH	806	3e-17	2e-06	32	13.4	3776.6	10000
1HQQ	891	1e-15	2e-06	41	15.0	4539.2	10000
1POA	914	9e-16	2e-06	37	14.2	4844.0	10000
1AX8	1003	8e-16	2e-06	40	14.3	5969.8	10000
1TJO	1394	2e-15	2e-06	62	19.0	12512.3	10000
1RGS	2015	6e-16	4e-06	126	20.2	-	4952
1TOA	2138	2e-15	5e-06	136	19.2	-	4303
1KDH	2846	5e-16	1e-05	254	21.9	-	1901
1NFG	3501	3e-15	5e-05	391	21.2	-	903
1BPM	3672	2e-16	6e-05	436	23.8	-	788
1MQQ	5510	7e-16	3e-04	888	26.0	-	247

Table 7: Computational results for the APG method on molecular conformation problems. "-" indicates that the solver is terminated because the maximum running time of four hours is reached.

Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions.

SIAM review, 53(2):217–288, 2011.

Xin Liu, Zaiwen Wen, and Yin Zhang. Limited memory block krylov subspace optimization for computing dominant singular value decompositions. *SIAM Journal on Scientific Computing*, 35(3):A1641–A1668, 2013.

Per-Gunnar Martinsson. Randomized methods for matrix computations. *The Mathematics of Data*, 25(4):187–231, 2019.

Evert J. Nyström. Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben. *Acta Mathematica*, 54:185–204, 1930.

Po-Wei Wang, Chun-Liang Li, and J. Zico Kolter. Polynomial optimization methods for matrix factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Stephen J. Wright, Robert D. Nowak, and Mário A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit S. Dhillon. Parallel matrix factorization for recommender systems. *Knowledge and Information Systems*, 41(3):793–819, 2014.

Yong Zhuang, Wei-Sheng Chin, Yu-Chin Juan, and Chih-Jen Lin. A fast parallel SGD for matrix factorization in shared memory systems. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 249–256, 2013.