Fast Hybrid Algorithm for Big Matrix Recovery

Tengfei Zhou, Hui Qian*, Zebang Shen, Congfu Xu
College of Computer Science and Technology, Zhejiang University, China
{zhoutengfei,qianhui,shenzebang,xucongfu}@zju.edu.cn

Abstract

Large-scale Nuclear Norm penalized Least Square problem (NNLS) is frequently encountered in estimation of low rank structures. In this paper we accelerate the solution procedure by combining non-smooth convex optimization with smooth Riemannian method. Our methods comprise of two phases. In the first phase, we use Alternating Direction Method of Multipliers (ADMM) both to identify the fix rank manifold where an optimum resides and to provide an initializer for the subsequent refinement. In the second phase, two superlinearly convergent Riemannian methods: Riemannian NewTon (NT) and Riemannian Conjugate Gradient descent (CG) are adopted to improve the approximation over a fix rank manifold. We prove that our Hybrid method of ADMM and NT (HADMNT) converges to an optimum of NNLS at least quadratically. The experiments on large-scale collaborative filtering datasets demonstrate very competitive performance of these fast hybrid methods compared to the state-of-the-arts.

Introduction

Low Rank Matrix Recovery (LRMR) aims to estimate a low rank structure by its noisy observations. There are many important applications in which the problem under study can naturally be modeled as a LRMR, such as collaborative filtering (Jaggi and Sulovsk 2010), multitask learning (Pong et al. 2010), multivariate regression (Mishra 2014), and image inpainting (Lu et al. 2015). Commonly, LRMR can be cast as Nuclear Norm penalized Least Square problem (NNLS):

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{A}(\mathbf{X}) - b\|^2 + \lambda\|\mathbf{X}\|_* \triangleq F(\mathbf{X}) \qquad (1)$$

where $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ is a linear operator, $b \in \mathbb{R}^p$ stores noisy linear measurements of an unknown low rank matrix $\mathbf{X}_G$, $\lambda > 0$ is regularizer parameter, and $\|\mathbf{X}\|_* = \sum_{i=1}^{\min\{m,n\}} \sigma_i(\mathbf{X})$ is the nuclear norm (in which $\sigma_i(\mathbf{X})$ is the i-th singular value of matrix $\mathbf{X}$).

Theoretically $\mathbf{X}_G$ can be recovered by solving NNLS under mild conditions (Negahban and Wainwright 2012). Many algorithms, such as (Mazumder, Hastie, and Tibshirani 2010; Toh and Yun 2010; Lin et al. 2009; Yang and Yuan 2013; Jaggi and Sulovsk 2010; Avron et al. 2012) , have been devised to solve NNLS. However, scalability issue always exists (especially when we want to solve large-scale NNLS with high accuracy), since these algorithms have to perform top-ρ SVD of an $m \times n$ matrix in each iteration, which is computationally prohibitive when m and n are large. In addition, most of them suffer sublinear convergence rates.

Besides the convex formulation (1), LRMR can also be cast as nonconvex fix rank optimization problem (when rank is known):

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathcal{A}(\mathbf{X}) - b\|^2 \quad \text{subject to} \quad \mathbf{X} \in \mathcal{M}_r \qquad (2)$$

where the feasible set $\mathcal{M}_r = \{\mathbf{X}|\mathrm{rank}(\mathbf{X}) = r\}$ is the fix rank manifold. Since fix rank manifold is smooth, (2) is a smooth optimization problem and can be solved by Riemannian optimization methods such as Riemannian conjugate gradient descent (Vandereycken 2013; Mishra and Sepulchre 2014a), Riemannian trust region (Mishra et al. 2014), and Riemannian Newton (Absil, Amodei, and Meyer 2014). Theoretically, for NNLS these fix rank solvers are more scalable than convex solvers, because large-scale top-ρ SVD is avoided in each iteration. According to (Absil, Mahony, and Sepulchre 2009), they converge superlinearly in general.

However, in most LRMR applications, the rank of the data under study is unknown, which often precludes the use and potential advantage of the fix rank manifold based methods. For such a regime, we propose two hybrid methods which combine convex optimization framework with scalable Riemannian optimization techniques to solve large-scale NNLS with high accuracy. Our hybrid methods comprise of two phases. In the first phase we use ADMM both to identify the *active fix rank manifold* $\mathcal{M}_{r^*}$ where an optimum $\mathbf{X}^*$ resides and to provide an initializer for the subsequent refinement. In the second phase, we optimize $F(\mathbf{X})$ over $\mathcal{M}_{r^*}$ by Riemannian optimization techniques. We prove that ADMM will identify the active fix rank manifold $\mathcal{M}_{r^*}$ in finite steps, which means the amount of computation in first phase is small. In the second phase two superlinearly convergent Riemannian optimization approaches, Riemannian NewTon (NT) and Riemannian Conjugate Gradient decent (CG), are adopted to minimize $F(\mathbf{X})$ over $\mathcal{M}_{r^*}$. Furthermore, we prove that our hybrid method of ADMM and NT

*Corresponding author

converges to one of the global optima at quadratic convergence rate. Our methods are more scalable and faster than state-of-the-art convex solvers because ours only need to compute finite number large-scale top-ρ SVDs in the first phase (less than 50 times in our experiments), and in the second phase no large SVD is performed.

Preliminaries

Fix Rank Manifold

Let $\mathcal{M}_r = \{\mathbf{X}|\mathrm{rank}(\mathbf{X}) = r\}$ be the fix rank manifold. In practice, storing $\mathbf{X} \in \mathcal{M}_r$ as an $m \times n$ matrix has $O(mn)$ memory complexity, which is much more memory demanding than saving it by its polar factorization (Mishra et al. 2014). That is, we can represent $\mathbf{X}$ as $(\mathbf{U}, \mathbf{B}, \mathbf{V}) \in \overline{\mathcal{M}}_r$ such that $\mathbf{X} = \mathbf{U}\mathbf{B}\mathbf{V}^T$, where

$$\overline{\mathcal{M}}_r := \mathsf{St}(r, m) \times S_{++}(r) \times \mathsf{St}(r, n). \tag{3}$$

Above, $\mathsf{St}(r, m)$ is the Stiefel manifold of $m \times r$ matrices with orthogonal columns and $S_{++}(r)$ is the cone of $r \times r$ positive definite matrices. We term such representation of low rank matrix as *polar representation*. And we say two polar representations are equivalent if they represent the same matrix. Define mapping

$$\pi : \overline{\mathcal{M}}_r \to \mathcal{M}_r : (\mathbf{U}, \mathbf{B}, \mathbf{V}) \mapsto \mathbf{U}\mathbf{B}\mathbf{V}^T. \tag{4}$$

Then the equivalence class of polar representation is

$$\pi^{-1}(\pi(\mathbf{U}, \mathbf{B}, \mathbf{V})) = \{(\mathbf{U}\mathbf{O}, \mathbf{O}^T\mathbf{B}\mathbf{O}, \mathbf{V}\mathbf{O})|\mathbf{O} \in O(r)\} \tag{5}$$

where $O(r)$ is the set of $r \times r$ orthogonal matrices. Since mapping π is submersion, $\mathcal{M}_r$ can be viewed as quotient manifold of $\overline{\mathcal{M}}_r$ over equivalence class (5):

$$\mathcal{M}_r \backsimeq \overline{\mathcal{M}}_r / O(r). \tag{6}$$

In following sections we also call $\overline{\mathcal{M}}_r$ *total space*.

For brevity, we denote any polar representation of matrix $\mathbf{X}$ by $\overline{\mathbf{X}}$ where $\overline{\mathbf{X}} = (\mathbf{U}, \mathbf{B}, \mathbf{V}) \in \overline{\mathcal{M}}_r$. To lighten the notation we also denote a point of $\overline{\mathcal{M}}_r$ by $\overline{\mathbf{X}}$. The distinction is clear from context. The tangent vector of $\mathcal{M}_r$ at point $\mathbf{X}$ is denoted like this: $\zeta_{\mathbf{X}}, \eta_{\mathbf{X}}, \xi_{\mathbf{X}}$. Similarly, we denote the tangent vector of $\overline{\mathcal{M}}_r$ at point $\overline{\mathbf{X}}$ like this: $\bar{\zeta}_{\overline{\mathbf{X}}}, \bar{\eta}_{\overline{\mathbf{X}}}, \bar{\xi}_{\overline{\mathbf{X}}}$. As $\overline{\mathcal{M}}_r$ is a product manifold, its tangent space is the product of tangent space of its components:

$$T_{\overline{\mathbf{X}}}\overline{\mathcal{M}} = T_{\mathbf{U}}\mathsf{St}(r, m) \times T_{\mathbf{B}}S_{++}(r) \times T_{\mathbf{V}}\mathsf{St}(r, n) \tag{7}$$

where $T_{\mathbf{U}}\mathsf{St}(r, m) = \{\bar{\zeta}_{\mathbf{U}} \in \mathbb{R}^{m\times r} | \mathbf{U}^T\bar{\zeta}_{\mathbf{U}} + \bar{\zeta}_{\mathbf{U}}^T\mathbf{U} = \mathbf{0}\}$ and $T_{\mathbf{B}}S_{++}(p) = \{\bar{\zeta}_{\mathbf{B}} \in \mathbb{R}^{r\times r} | \bar{\zeta}_{\mathbf{B}} \text{ is a symmetric matrix}\}$.

Nuclear Norm is Smooth on Fix Rank Manifold

Suppose $\mathbf{X} \in \mathcal{M}_r$ has polar factorization $\mathbf{X} = \mathbf{U}\mathbf{B}\mathbf{V}^T$. Then

$$\|\mathbf{X}\|_* = \|\mathbf{B}\|_* = \mathrm{tr}(\mathbf{B}) \tag{8}$$

where $\mathrm{tr}(\cdot)$ means the trace of a matrix. Therefore, nuclear norm can be expressed as a smooth function $\mathrm{tr}(\mathbf{B})$ on total space $\overline{\mathcal{M}}_r$. By Proposition 3.4.5 of (Absil, Mahony, and Sepulchre 2009), we can infer that nuclear norm is smooth on fix rank manifold. Thus, the objective function $F(\mathbf{X})$ is smooth on such manifold, and minimizing $F(\mathbf{X})$ over the fix rank manifold $\mathcal{M}_r$ can be solved by Riemannian optimization methods (Absil, Mahony, and Sepulchre 2009). Therefore, if the rank r^* of some solution to problem (1) is known, it can be reduced to the following smooth optimization problem:

$$\min_{\mathbf{X}\in\mathcal{M}_{r^*}} \frac{1}{2}\|\mathcal{A}(\mathbf{X}) - b\|^2 + \lambda\|\mathbf{X}\|_* \tag{9}$$

which can be solved by Riemannian optimization.

Assumptions

Suppose $\mathbf{X}^*$ is an optimum of problem (1) and let $r^* = \mathrm{rank}(\mathbf{X}^*)$. To design efficient solver, we make the following assumptions.

A.1 $\mathbf{X}^*$ is low rank: $r^* \ll \min\{m, n\}$.

A.2 $\mathbf{0}$ belongs to the relative interior of the subdifferential $\partial F(\mathbf{X}^*)$.

A.3 There exists $\gamma > 0$ such that for any matrix $\mathbf{X} \in \mathcal{M}_{r^*}$ we have $\|\mathcal{A}(\mathbf{X} - \mathbf{X}^*)\|^2 \geq \gamma\|\mathbf{X} - \mathbf{X}^*\|_F^2$.

A.1 is natural, since the goal of using nuclear norm penalty is to obtain a low rank solution. For A.2, it is actually quite mild, since $\mathbf{X}^*$ is an optimum, we have $\mathbf{0} \in \partial F(\mathbf{X}^*)$. Note that A.2 is commonly used in literatures such as (Hare and Lewis 2004; Liang, Fadili, and Peyré 2014; Lee and Wright 2012; Liang et al. 2015).

The next two lemmas say that A.3 holds with high probability for matrix sensing (Recht, Fazel, and Parrilo 2010) and matrix completion, if at least $O(r^*(m+n)\log(m+n))$ linear measurements are made.

Lemma 1 *Suppose $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ is formed by Gaussian ensemble (Recht, Fazel, and Parrilo 2010). If the sample size $p > cr^*(m+n)\log(m+n)$ then A.3 holds with probability at least $1 - 2\exp(-p/32)$ where $c > 0$ is a global constant.*

Lemma 2 *Suppose $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ samples p elements uniformly from a matrix. Denote the spikiness (Negahban and Wainwright 2012) of a matrix $\mathbf{X}$ by $\alpha_{sp}(\mathbf{X})$. For any matrix $\mathbf{X}$ satisfying $\alpha_{sp}(\mathbf{X}-\mathbf{X}^*) \leq \alpha$, as long as $p > c\alpha^2 r^*(m+n)\log(m+n)$, A.3 holds with probability greater than $1 - c_1\exp(-c_2(m+n)\log(m+n))$ where $c, c_1, c_2 > 0$ are global constants.*

ADMM for NNLS

One of the classical methods for solving NNLS is ADMM. Both its technical and implementation details have been discussed in (Lin et al. 2009; Yang and Yuan 2013). Here we briefly review the sketch of the solver. The novel result for solving NNLS by ADMM is that we prove that ADMM identifies the active manifold $\mathcal{M}_{r^*}$ in finite steps.

NNLS can be rewritten as the following constrained optimization problem:

$$\min_{\mathbf{X},\mathbf{Y}} \frac{1}{2}\|\mathcal{A}(\mathbf{Y}) - b\|^2 + \lambda\|\mathbf{X}\|_* \quad \text{subject to} \quad \mathbf{Y} - \mathbf{X} = \mathbf{0}. \tag{10}$$

One can compose the augmented Lagrangian function as follow:

$$\mathcal{L}(\mathbf{X},\mathbf{Y},\mathbf{\Lambda}) = \frac{1}{2}\|\mathcal{A}(\mathbf{Y})-b\|^2 + \lambda\|\mathbf{X}\|_* - \mathrm{tr}(\mathbf{\Lambda}^T(\mathbf{Y}-\mathbf{X})) + \frac{\beta}{2}\|\mathbf{Y}-\mathbf{X}\|_F^2$$

where $\mathbf{\Lambda}$ is the Lagrangian multiplier matrix and $\beta > 0$. So iterations of ADMM can be generated by minimizing $\mathbf{X}$ (and $\mathbf{Y}$ alternately) and updating the multiplier $\mathbf{\Lambda}$:

$$\begin{aligned}\mathbf{X}^{(k+1)} &= \arg\min_{\mathbf{X}} \mathcal{L}(\mathbf{X},\mathbf{Y}^{(k)},\mathbf{\Lambda}^{(k)}),\\ \mathbf{Y}^{(k+1)} &= \arg\min_{\mathbf{Y}} \mathcal{L}(\mathbf{X}^{(k+1)},\mathbf{Y},\mathbf{\Lambda}^{(k)}),\\ \mathbf{\Lambda}^{(k+1)} &= \mathbf{\Lambda}^{(k)} - \beta(\mathbf{Y}^{(k+1)}-\mathbf{X}^{k+1}).\end{aligned} \tag{11}$$

Theorem 3 *Suppose $\{\mathbf{X}^{(k)}\}$ is the sequences generated by ADMM iterations (11). Then $\{\mathbf{X}^{(k)}\}$ converges to $\mathbf{X}^*$ which is an optimum of problem (1). Moreover if A.1 and A.2 hold, then there exists $K > 0$ such that $\mathrm{rank}(\mathbf{X}^{(k)}) = \mathrm{rank}(\mathbf{X}^*)$, $\forall k > K$.*

The above theorem shows that ADMM identifies the active manifold $\mathcal{M}_{r^*}$ in finite steps. Note that Theorem 3 differs from the finite identification property in (Liang et al. 2015) in that they are under different assumptions.

Hybrid Methods

According to Theorem 3, ADMM iterations for NNLS can be naturally divided into two phases. The first phase consists of the first K iterations. In this phase ADMM identifies the fix rank manifold $\mathcal{M}_{r^*}$ where the optimum $\mathbf{X}^*$ resides. Specifically it implies that for any $k > K$ the rank of $\mathbf{X}^{(k)}$ is equal to r^*. The second phase consists of the remaining iterations of ADMM. In this phase, ADMM generates sequence $\{\mathbf{X}^{(k)}\}_{k>K}$ which converges to $\mathbf{X}^*$. As $\mathbf{X}^{(k)}$ resides in $\mathcal{M}_{r^*} \forall k > K$, ADMM here actually minimizes the fix rank constrained optimization problem (9) in the second phase.

Since empirical result given in Figure 1 (a) shows that ADMM identifies the active manifold $\mathcal{M}_{r^*}$ in few iterations, it has therefore been conjectured that the sublinear convergence of ADMM (He and Yuan 2012) is mainly caused by slow convergent speed of the second phase. Thus, if we use superlinearly convergent methods instead in this phase, more efficient algorithms can be devised. Based on this intuition we design two hybrid methods for NNLS. The outline of our hybrid method is as follows. It carries out a two-phase procedure: identifying the active fix rank manifold $\mathcal{M}_{r^*}$ by ADMM, and using Riemannian optimization methods to refine the approximation over $\mathcal{M}_{r^*}$.

Riemannian Optimization Phase

Among various Riemannian optimization methods, Riemaninan NT and CG have been proved to converge superlinearly. In this section we derive both NT and CG algorithms to solve the smooth optimization problem (9). Before describing NT and CG, we must give the fix rank manifold $\mathcal{M}$ a structure of Riemannian quotient manifold (we remove the subscript of fix rank manifold $\mathcal{M}_{r^*}$ and total space $\overline{\mathcal{M}}_{r^*}$ to simplify notations in the following section). To accomplish this purpose, we choose the second order derivative of function $g(\mathbf{U},\mathbf{B},\mathbf{V}) = \|\mathbf{U}\mathbf{B}\mathbf{V}^T - \mathbf{I}\|_F^2$, namely $D^2 g(\overline{\mathbf{X}})[\bar{\zeta},\bar{\eta}]$, as the Riemannian metric for total space $\overline{\mathcal{M}}$, because such metric has preconditioning effect which can handle the ill-condition issue for real-word datasets (Mishra and Sepulchre 2014b; 2014a). The Riemannian quotient manifold structure of $\mathcal{M}$ is listed in Table 1, and the mathematical derivations are given in the supplement.

Algorithm 1 NT: Riemannian Newton method for problem (9)

Input: Rank r, the polar representation $\overline{\mathbf{X}^{(0)}}$ of matrix $\mathbf{X}^0$.
Output: Polar representation of local optimum.

1: $k = 0$
2: **repeat**
3: Solve the following linear equations of $\overline{\zeta^{(k)}}$ by tCG

$$\begin{cases}\Pi_{\overline{\mathbf{X}^{(k)}}}\left(\overline{\nabla}_{\overline{\zeta^{(k)}}}\overline{\mathrm{grad}F(\mathbf{X}^{(k)})}\right) = -\overline{\mathrm{grad}F(\mathbf{X}^{(k)})}\\ \overline{\zeta^{(k)}} \in \mathcal{H}_{\overline{\mathbf{X}^{(k)}}}\end{cases} \tag{12}$$

4: Set $\overline{\mathbf{X}^{(k+1)}} = R_{\overline{\mathbf{X}^{(k)}}}(\overline{\zeta^{(k)}})$
5: k=k+1
6: **until** convergence
7: **return** $\overline{\mathbf{X}^{(k)}}$

NT Method In Riemannian NT, the search direction $\zeta_{\mathbf{X}} \in T_{\mathbf{X}}\mathcal{M}$ is decided by Riemannian Newton equation

$$\mathrm{Hess}(\mathbf{X})(\zeta_{\mathbf{X}}) = -\mathrm{grad}F(\mathbf{X}) \tag{13}$$

where $\mathrm{Hess}(\cdot)$ is the Riemannian Hessian of F and $\mathrm{grad}F(\cdot)$ is the Riemannian gradient of F. In Riemannian optimization, the Hessian is a linear operator defined by Riemannian connection ∇ (Absil, Mahony, and Sepulchre 2009):

$$\mathrm{Hess}(\mathbf{X})(\zeta_{\mathbf{X}}) = \nabla_{\zeta_{\mathbf{X}}}\mathrm{grad}F(\mathbf{X}). \tag{14}$$

For Riemannian quotient manifold $\mathcal{M}$, the Newton equation (13) is horizontally lifted to horizontal space $\mathcal{H}_{\overline{\mathbf{X}}}$ (Absil, Mahony, and Sepulchre 2009):

$$\overline{\mathrm{Hess}(\mathbf{X})(\zeta_{\mathbf{X}})} = -\overline{\mathrm{grad}F(\mathbf{X})}. \tag{15}$$

Plug (14) into (15) and refer Table 1 for the expression of horizontal lift of Riemannian gradient and Riemannian connection, the Newton equation (15) can be rewritten as:

$$\Pi_{\overline{\mathbf{X}}}(\overline{\nabla}_{\bar{\zeta}_{\overline{\mathbf{X}}}}\overline{\mathrm{grad}F(\mathbf{X})}) = -\overline{\mathrm{grad}F(\mathbf{X})} \tag{16}$$

where $\bar{\zeta}_{\overline{\mathbf{X}}}$ is the horizontal lift of the search direction $\zeta_{\mathbf{X}}$, and expression of $\overline{\nabla}_{\bar{\zeta}_{\overline{\mathbf{X}}}}\overline{\mathrm{grad}F(\mathbf{X})}$ is listed in Table 1. The above linear system of $\bar{\zeta}_{\overline{\mathbf{X}}}$ can be solved by truncated CG method. After $\bar{\zeta}_{\overline{\mathbf{X}}}$ is computed, NT performs updating by retraction along $\bar{\zeta}_{\overline{\mathbf{X}}}$, the expression of retraction is given in Table 1. We summarize NT in Algorithm 1.

CG Method In the k-th iteration, Riemannian CG composes search direction $\zeta^{(k)} \in T_{\mathbf{X}^{(k)}}\mathcal{M}$, then updates by retraction: $\mathbf{X}^{(k+1)} = R_{\mathbf{X}^{(k)}}(\alpha^{(k)}\zeta^{(k)})$ where $\alpha^{(k)} > 0$ is a

Item	Expression
Riemannian metric $\langle\bar{\eta}_{\overline{\mathbf{X}}}, \bar{\zeta}_{\overline{\mathbf{X}}}\rangle_{\overline{\mathbf{X}}}$	$\mathrm{tr}(\mathbf{B}^2\bar{\eta}_{\mathbf{U}}^T\bar{\zeta}_{\mathbf{U}}) + \mathrm{tr}(\bar{\eta}_{\mathbf{B}}^T\bar{\zeta}_{\mathbf{B}}) + \mathrm{tr}(\mathbf{B}^2\bar{\eta}_{\mathbf{V}}^T\bar{\zeta}_{\mathbf{V}})$
Horizontal space $\mathcal{H}_{\overline{\mathbf{X}}}$	$\bar{\zeta}_{\overline{\mathbf{X}}} \in T_{\overline{\mathbf{X}}}\overline{\mathcal{M}} : \mathbf{U}^T\bar{\zeta}_{\mathbf{U}}\mathbf{B}^2 + \mathbf{B}\bar{\zeta}_{\mathbf{B}} - \bar{\zeta}_{\mathbf{B}}\mathbf{B} + \mathbf{V}^T\bar{\zeta}_{\mathbf{V}}\mathbf{B}^2 \in S_{sym}(r)$
Projection of a vector in ambient space onto tangent space $\Psi_{\overline{\mathbf{X}}}(\mathbf{Z}_{\mathbf{U}}, \mathbf{Z}_{\mathbf{B}}, \mathbf{Z}_{\mathbf{V}})$	$(\mathbf{Z}_{\mathbf{U}} - \mathbf{U}\mathbf{B}_{\mathbf{U}}\mathbf{B}^{-2}, Sym(\mathbf{Z}_{\mathbf{B}}), \mathbf{Z}_{\mathbf{V}} - \mathbf{V}\mathbf{B}_{\mathbf{V}}\mathbf{B}^{-2})$ where $\mathbf{B}_{\mathbf{U}}, \mathbf{B}_{\mathbf{V}}$ are solutions of equations: $\mathbf{B}^2\mathbf{B}_{\mathbf{U}} + \mathbf{B}_{\mathbf{U}}\mathbf{B}^2 = 2\mathbf{B}^2(Sym(\mathbf{Z}_{\mathbf{U}}^T\mathbf{U}))\mathbf{B}^2$ $\mathbf{B}^2\mathbf{B}_{\mathbf{V}} + \mathbf{B}_{\mathbf{V}}\mathbf{B}^2 = 2\mathbf{B}^2(Sym(\mathbf{Z}_{\mathbf{V}}^T\mathbf{V}))\mathbf{B}^2$
Projection of a vector of tangent space to Horizontal space $\Pi_{\overline{\mathbf{X}}}(\bar{\zeta}_{\overline{\mathbf{X}}})$	$(\bar{\zeta}_{\mathbf{U}} - \mathbf{U}\mathbf{\Omega}, \bar{\zeta}_{\mathbf{B}} + \mathbf{\Omega}\mathbf{B} - \mathbf{B}\mathbf{\Omega}, \bar{\zeta}_{\mathbf{V}} - \mathbf{V}\mathbf{\Omega})$, where $\mathbf{\Omega}$ is the solution to equation: $\mathbf{\Omega}\mathbf{B}^2 + \mathbf{B}^2\mathbf{\Omega} - \mathbf{B}\mathbf{\Omega}\mathbf{B} = \frac{1}{2}Skw(\mathbf{U}^T\bar{\zeta}_{\mathbf{U}}\mathbf{B}^2) + Skw(\mathbf{B}\bar{\zeta}_{\mathbf{B}}) + \frac{1}{2}Skw(\mathbf{V}^T\bar{\zeta}_{\mathbf{V}}\mathbf{B}^2)$
Retraction on total space $R_{\overline{\mathbf{X}}}(\bar{\zeta}_{\overline{\mathbf{X}}})$	$(R_{\mathbf{U}}(\bar{\zeta}_{\mathbf{U}}), R_{\mathbf{B}}(\bar{\zeta}_{\mathbf{B}}), R_{\mathbf{V}}(\bar{\zeta}_{\mathbf{V}}))$ where $R_{\mathbf{U}}(\bar{\zeta}_{\mathbf{U}}) = uf(\mathbf{U} + \bar{\zeta}_{\mathbf{U}})$ $R_{\mathbf{B}}(\bar{\zeta}_{\mathbf{B}}) = \mathbf{B}^{\frac{1}{2}}\exp(\mathbf{B}^{-\frac{1}{2}}\bar{\zeta}_{\mathbf{B}}\mathbf{B}^{-\frac{1}{2}})\mathbf{B}^{\frac{1}{2}}$ $R_{\mathbf{V}}\bar{\zeta}_{\mathbf{V}} = uf(\mathbf{V} + \bar{\zeta}_{\mathbf{V}})$
Horizontal lift of vector transport defined on fix rank manifold $\overline{\mathcal{T}_{\xi_{\mathbf{X}}}(\zeta_{\mathbf{X}})}$	$\Pi_{R_{\overline{\mathbf{X}}}(\bar{\xi}_{\overline{\mathbf{X}}})}(\Psi_{R_{\overline{\mathbf{X}}}(\bar{\xi}_{\overline{\mathbf{X}}})}(\bar{\zeta}_{\overline{\mathbf{X}}}))$
Riemannian connection on total space $\overline{\nabla}_{\bar{\xi}_{\overline{\mathbf{X}}}}\bar{\eta}_{\overline{\mathbf{X}}}$	$\Psi_{\overline{\mathbf{X}}}(D\bar{\zeta}_{\mathbf{U}}[\bar{\xi}_{\overline{\mathbf{X}}}] + \mathbf{A}_{\mathbf{U}}, D\bar{\zeta}_{\mathbf{B}}[\bar{\xi}_{\overline{\mathbf{X}}}] + \mathbf{A}_{\mathbf{B}}, D\bar{\zeta}_{\mathbf{V}}[\bar{\xi}_{\overline{\mathbf{X}}}] + \mathbf{A}_{\mathbf{V}})$ where $\mathbf{A}_{\mathbf{U}} = \bar{\zeta}_{\mathbf{U}}Sym(\bar{\xi}_{\mathbf{B}}\mathbf{B})\mathbf{B}^{-2} + \bar{\xi}_{\mathbf{U}}Sym(\bar{\zeta}_{\mathbf{B}}\mathbf{B})\mathbf{B}^{-2}$ $\mathbf{A}_{\mathbf{B}} = -\frac{1}{2}Sym(\mathbf{B}\bar{\xi}_{\mathbf{U}}^T\bar{\zeta}_{\mathbf{U}} + \bar{\xi}_{\mathbf{U}}^T\bar{\zeta}_{\mathbf{U}}\mathbf{B}) - \frac{1}{2}Sym(\mathbf{B}\bar{\xi}_{\mathbf{V}}^T\bar{\zeta}_{\mathbf{V}} + \bar{\xi}_{\mathbf{V}}^T\bar{\zeta}_{\mathbf{V}}\mathbf{B})$ $\mathbf{A}_{\mathbf{V}} = \bar{\zeta}_{\mathbf{V}}Sym(\bar{\xi}_{\mathbf{B}}\mathbf{B})\mathbf{B}^{-2} + \bar{\xi}_{\mathbf{V}}Sym(\bar{\zeta}_{\mathbf{B}}\mathbf{B})\mathbf{B}^{-2}$
Horizontal lift of Riemannian connection on fix rank manifold $\overline{\nabla_{\xi_{\mathbf{X}}}\zeta_{\mathbf{X}}}$	$\Pi_{\overline{\mathbf{X}}}(\overline{\nabla}_{\bar{\xi}_{\overline{\mathbf{X}}}}\bar{\eta}_{\overline{\mathbf{X}}})$
Horizontal lift of Riemannian gradient $\overline{\mathrm{grad}F(\mathbf{X})}$	$\Psi_{\overline{\mathbf{X}}}(\mathbf{S}\mathbf{V}\mathbf{B}^{-1}, \mathbf{U}^T\mathbf{S}\mathbf{V} + \lambda\mathbf{I}, \mathbf{S}^T\mathbf{U}\mathbf{B}^{-1})$ where $\mathbf{S} = \mathcal{A}^*(\mathcal{A}(\mathbf{U}\mathbf{B}\mathbf{V}^T) - b)$

Table 1: Differential structures for Riemannian optimization. Here $S_{sym}(r)$ means the set of $r \times r$ symmetric matrices. The ambient space means vector space $\mathbb{R}^{m\times r} \times \mathbb{R}^{r\times r} \times \mathbb{R}^{n\times r}$. $Sym(\mathbf{X}) = 1/2(\mathbf{X} + \mathbf{X}^T)$, $Skw(\mathbf{X}) = 1/2(\mathbf{X} - \mathbf{X}^T)$. $uf(\cdot)$ extracts the orthogonal factor of a full rank matrix. $D\bar{\zeta}_{\overline{\mathbf{X}}}[\bar{\xi}_{\overline{\mathbf{X}}}] = \lim_{t\downarrow 0}(\bar{\zeta}_{\overline{\mathbf{X}}+t\bar{\xi}_{\overline{\mathbf{X}}}} - \bar{\zeta}_{\overline{\mathbf{X}}})/t$. The definition of horizontal space, retraction, vector transport, Riemannian connection and horizontal lift can be found in (Absil, Mahony, and Sepulchre 2009).

suitable stepsize. The search direction $\zeta^{(k)}$ is composed by the following recurrence

$$\begin{cases}\zeta^{(0)} = -\mathrm{grad}F(\mathbf{X}^{(0)}),\\ \zeta^{(k)} = -\mathrm{grad}F(\mathbf{X}^{(k)}) + \beta^k\mathcal{T}_{\alpha^{k-1}\zeta^{(k-1)}}\zeta^{(k-1)}, \quad k>1,\end{cases} \tag{17}$$

where β^k is computed by Polak-Ribière formula (Absil, Mahony, and Sepulchre 2009), and vector transport $\mathcal{T}_{\alpha^{k-1}\zeta^{(k-1)}}(\cdot)$ maps previous search direction $\zeta^{(k-1)}$ onto the current tangent space $T_{\mathbf{X}^{(k)}}\mathcal{M}$. The vector transport is required since $\zeta^{(k-1)} \in T_{\mathbf{X}^{(k-1)}}\mathcal{M}$ and $\mathrm{grad}F(\mathbf{X}^{(k)}) \in T_{\mathbf{X}^{(k)}}\mathcal{M}$ belong to different linear spaces and we cannot linearly combine them directly. Like NT, the recurrence (17) is horizontally lifted to the horizontal space:

$$\begin{cases}\overline{\zeta^{(0)}} = -\overline{\mathrm{grad}F(\mathbf{X}^{(0)})},\\ \overline{\zeta^{(k)}} = -\overline{\mathrm{grad}F(\mathbf{X}^{(k)})} + \beta^k\overline{\mathcal{T}_{\alpha^{k-1}\zeta^{(k-1)}}\zeta^{(k-1)}}, \quad k>1,\end{cases} \tag{18}$$

where $\overline{\mathcal{T}_{\alpha^{k-1}\zeta^{(k-1)}}\zeta^{(k-1)}}$ and $\overline{\mathrm{grad}F(\mathbf{X}^{(k)})}$ are horizontal lift of the vector transport and the Riemannian gradient respectively (see Table 1). We summarize CG in Algorithm 2.

Algorithm 2 CG: Riemannian conjugate gradient descent for problem (9)

Input: Rank r,polar representation $\overline{\mathbf{X}^{(0)}}$.
Output: Polar representation of local optimum.

```
k = 0
repeat
    Compute the gradient grad F(X^(k)) by Table 1.
    Compute search direction ζ^(k) by recurrence (18).
    if ⟨ζ^(k), −grad F(X^(k))⟩_{X^(k)} < 0 then
        ζ^(k) = −grad F(X^(k)).
    end if
    Set X^(k+1) = R_{X^(k)}(α^(k) ζ^(k)) where α^(k) > 0 is a
    suitable stepsize.
    k = k + 1
until convergence
return X^(k)
```

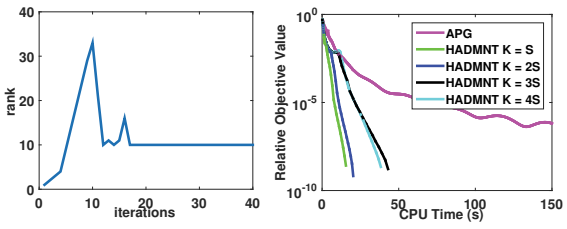
The Hybrid Algorithms

We provide two hybrid methods: the Hybrid of ADMM and NT (HADMNT), and the Hybrid of ADMM and CG (HADMCG) in Algorithm 3.

The next theorem says that HADMNT converges at least quadratically to the solution of NNLS. We leave the convergence analysis of HADMCG up for future work.

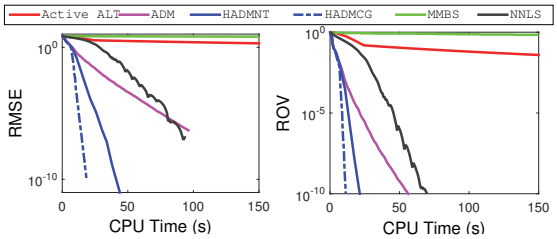
Theorem 4 *Suppose $\{\mathbf{X}^{(k)}\}$ is a sequence generated by ADMM which converges to $\mathbf{X}^*$. Suppose A.1-A.3 are satisfied. Then there exists integral K such that (1) $\mathrm{rank}(\mathbf{X}^{(k)}) = \mathrm{rank}(\mathbf{X}^*), \forall k \geq K$; (2) when initialized by polar representation of $\mathbf{X}^{(K)}$ NT generates sequence $(\mathbf{U}^{(l)}, \mathbf{B}^{(l)}, \mathbf{V}^{(l)}) \in \overline{\mathcal{M}_{\mathrm{rank}(\mathbf{X}^*)}}$ such that $\mathbf{U}^{(l)}\mathbf{B}^{(l)}\mathbf{V}^{(l)T}$ converges to $\mathbf{X}^*$ at least quadratically.*

Remark. Practically, instead of setting K in advance, we stop the first phase of our hybrid methods when $\mathrm{rank}(\mathbf{X}^{(k)})$ satisfies some convergent conditions, that is, $\mathrm{rank}(\mathbf{X}^{(k)}) =$



(a) Rank during ADMM iterations (b) ROV v.s. CPU time

Figure 1: Experimental results on ML1M dataset.



(a) RMSE v.s. CPU time (b) ROV v.s. CPU time

Figure 2: Performance of NNLS solvers for the noiseless case.

$\text{rank}(\mathbf{X}^{(k-1)}) = \ldots = \text{rank}(\mathbf{X}^{(k-c)})$ where c is a constant. So K is not a parameter in our implementation code. Suppose S is the step number after which the convergent condition of the first phase is satisfied. Figure 1 (b) shows that there is no need to compute more ADMM iterations to ensure the superlinear convergence of HADMNT.

Algorithm 3 HADMNT (or HADMCG) for problem (1)

Input: $\mathcal{A}$,λ,b,c,K
Output: optimum of NNLS

Phase 1: Finite Identification Phase:
Initialize $\mathbf{X}^{(0)} = \mathbf{Y}^{(0)} = \mathbf{\Lambda}^{(0)} = \mathbf{0}$
for $k = 1, 2, 3, \ldots, K$ **do**
    Generate $\mathbf{X}^{(k)}, \mathbf{Y}^{(k)}, \mathbf{\Lambda}^{(k)}$ by iteration given in (11)
end for
Phase 2: Riemannian Optimization Phase:
$\overline{\mathbf{X}^{(K)}} = \text{SVD}\left(\mathbf{X}^{(K)}\right)$
$\overline{\mathbf{X}^{*}} = \text{NT}\left(\text{rank}(\mathbf{X}^{(K)}), \overline{\mathbf{X}^{(K)}}\right)$ {CG can be used as an alternative to NT.}
return $\pi\left(\overline{\mathbf{X}^{*}}\right)$

Experiments

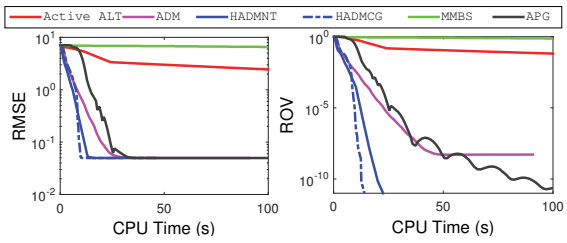
We validate the performance of our hybrid methods by conducting empirical study on synthetic and real matrix recovery tasks. The baselines include four state-of-the-art *NNLS solvers*: ADMM (Yang and Yuan 2013), Active ALT (Hsieh and Olsen 2014), APG (Toh and Yun 2010), and MMBS (Mishra et al. 2013) and four recently published *non-convex solvers*: LMaFit (Wen, Yin, and Zhang 2012),LRGeomCG (Vandereycken 2013), R3MC (Mishra and Sepulchre 2014a), and RP (Tan et al. 2014). Note that some related works such as Lifted CD (Dudik, Harchaoui, and Malick 2012) and SSGD (Avron et al. 2012) are not compared in this paper, since our baselines have been shown to be the state-of-the-art (Hsieh and Olsen 2014). The codes of baselines except for ADMM are download from the homepages of their authors. All experiments are conducted in the same machine with Intel Xeon E5-2690 3.0GHz CPU and 128GB RAM.

Simulations

We use synthetic data to exhibit the convergence rates of the six different NNLS solvers. We do not compare with non-convex solvers in simulation since their objective functions are not the same as that of NNLS. Three different cases are considered in simulations: the noiseless case, the noisy case, and the ill-conditioned case. Two metrics are used to compare the convergence rate and the statistical accuracy of compared methods. Specifically, we use the Relative Objective Value (ROV) to indicate convergence rate: $ROV^{(t)} = (F(\mathbf{X}^{(t)}) - F^*)/F(\mathbf{X}^{(0)})$ where F^* is the minimal value of objective function obtained by iterating APG for 3000 times; and we use the Root Mean Square Error: $RMSE^{(t)} = \|\mathbf{X}^{(t)} - \mathbf{X}_G\|_F/\sqrt{mn}$ to indicate the statistical accuracy where $\mathbf{X}_G$ is the ground truth matrix. For fairness, all the six solvers are initialized by $\mathbf{X}^{(0)} = \mathbf{0}$. And their regularizer parameters λ are set to identical values.

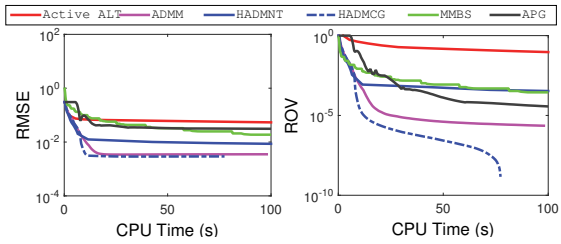
Data Generating Following (Mazumder, Hastie, and Tibshirani 2010) and (Mishra et al. 2013), we generated the ground truth matrix $\mathbf{X}_G$ by two scenarios: (1) $\mathbf{X}_G = \mathbf{L}\mathbf{R}^T$ for the noiseless and the noisy case, where $\mathbf{L} \in \mathbb{R}^{m\times r}$ and $\mathbf{R} \in \mathbb{R}^{n\times r}$ are random matrices with i.i.d normal distributed entries, and (2) $\mathbf{X}_G = \mathbf{U}_G\text{diag}(\sigma)\mathbf{V}_G^T$ for the ill-conditioned case, where $\mathbf{U}_G \in \text{St}(r,m), \mathbf{V}_G \in \text{St}(r,n)$, and singular values are imposed with exponential decay. The measurement vector b is generated by sampling $c\,r(m+n-r)$ elements uniformly from $\mathbf{X}_G$ and then adding a noise vector e. In the noiseless case, $e = 0$. In the noisy or the ill-conditioned case, e is a Gaussian noise with predefined Signal to Noise Ratio (SNR). In our simulation, we set both m and n to 5000, and rank r to 50. The oversampling ratio c is fixed as 3.

Noiseless Case In this scenario, the optimization problem (1) is expected to recover the ground true exactly, if an extremely small regularizer λ is used. As a result, we set $\lambda = 10^{-10}$ for all solvers. And we report ROV and RMSE w.r.t CPU time in Figure 2. In the noiseless case, both curves can indicate the rate of convergence. From Figure 2(a) one can see that HADMCG and HADMNT converge superlinearly to the optimum and are significantly faster than other baselines. In Figure 2(b) ROV shows a similar phenomenon. It is rational to conjecture that using random SVD and solving sub-problem approximately may make Active ALT underperform others. The unsatisfactory performance of MMBS may be directly due to the Riemannian trust region method called in each iteration.



(a) RMSE v.s. CPU time (b) ROV v.s. CPU time

Figure 3: Performance of NNLS solvers for the noisy case.



(a) RMSE v.s. CPU time (b) ROV v.s. CPU time

Figure 4: Performance of NNLS solvers for the ill-condition case.

Noisy Case We set the SNR to 0.01. The regularization parameter λ is set to $0.04\sqrt{(m \log m)/p}$, where p is the number of sampling elements. Such choice of λ is suggested by Negahban and Wainwright(2012). We report the performance of the compared methods in Figure 3. From Figure 3 (a) one can see that our hybrid methods, ADMM, and APG achieve the same RMSE when converging. This is because they solve the same convex optimization problem. HADMNT and HADMCG converge to the optimum much faster than other baselines. Figure 3 (b) also implies that HADMNT and HADMCG converge superlinearly. MMBS and Active ALT still perform unsatisfactorily on this task.

Ill-conditioned Case We impose exponential decay in singular values (the singular values is generated with specified Condition Number (CN) by MatLab command $1000 * \text{logspace}(-\log(\text{CN}), 0, 50)$ where CN is set to 10^6). Moreover we perturb the observed entries by a Gaussian noise with SNR = 0.01. The regularizer parameter is set to the same value as in the noisy case. We report ROV and RMSE w.r.t CPU time in Figure 4. From Figure 4(b) we can see that large CN makes the optimization of $F(\mathbf{X})$ more challenging. It also shows that HADMCG finally enters superlinearly convergence phase, while, for the other five methods, their convergence rates become almost vanished after 40 seconds. Figure 4(a) illustrates the same phenomenon. It exhibits that HADMCG outperforms other methods both in speed and in accuracy.

Experiments on Recommendation

In recommendation task, the ratings given by users to items are partially observed. We need to predict the unobserved ratings based on observed ones. Three largest public available recommendation datasets are used in our comparison: Movielens 10M (ML10M) (Herlocker et al. 1999), NetFlix (KDDCup 2007), and Yahoo music (Dror et al. 2012). Their statistics are showed in Table 2. We randomly partition the datasets into two groups: 80% ratings for training and the remaining 20% ratings for testing. We repeat the experiments 5 times and report the average testing RMSE and CPU time.

Dataset	users	Items	Ratings
ML10M	69,878	10,677	10,000,054
NetFlix	2,649,429	17,770	100,480,507
Yahoo	1,000,990	624,961	252,800,275

Table 2: Statistics of datasets.

Dataset	ML10M		NetFlix		Yahoo	
	RMSE	Time(s)	RMSE	Time	RMSE	Time
Active ALT	0.8106	1609	0.8472	24330	-	-
ADMM	0.8101	77.95	0.8666	1715	22.25	15080
APG	0.8066	188.7	0.8402	2259	23.10	15920
MMBS	0.8067	28760	-	-	-	-
LMaFit	0.8100	140.6	0.8507	1828	23.14	15020
LRGeomCG	0.8246	146.6	0.8633	1133	22.49	17540
R3MC	0.8206	94.57	0.8532	1000	22.38	17270
RP	0.8119	283.3	0.8473	736.4	23.28	12380
HADMNT	0.8062	69.45	0.8498	821.8	23.83	14740
HADMCG	**0.8039**	**53.10**	**0.8398**	**708.2**	**22.00**	**7027**

Table 3: Performance comparison on recommendation tasks.

In our experiments, the regularizer parameters λ of the six NNLS solvers (including our hybrid solvers) are set to identical values. We set $\lambda = 20$ for both ML10M and NetFlix, and 200 for Yahoo Music. The six NNLS solvers and the non-convex method RP are initialized by $\mathbf{0}$. The rank parameters r of fix rank methods, (namely LMafit, LRGeomCG and R3MC) are set to the rank estimated by our hybrid methods. Since $\mathbf{0}$ is not a valid initializer for the fix rank methods, LMafit,LRGeomCG and R3MC are initialized by top-r SVD of the training matrix. We terminate these methods once a pre-specified training RMSE is achieved or they iterate more than 500 times.

The comparison results are given in Table 3. From it one can see that both HADMNT and HADMCG outperform other NNLS solvers in speed. That is because HADMNT and HADMCG are superlinearly convergent methods. Especially, our HADMCG method outperforms other methods both in speed and accuracy. One can also find that fix rank methods (LMaFit, LRGeomCG and R3MC) do not show superior advantage over others even though they do not need to estimate rank. The probable reason is that ill-condition of the recommendation dataset may slow down their convergence, and also they may be trapped in local optimum when minimizing a non-convex objective function.

Conclusion

In this paper we propose two hybrid methods, HADMNT and HADMCG, to solve large-scale NNLS. In theory, we prove HADMNT converges to an optimum of NNLS at least

quadratically. Pratically both HADMNT and HADMCG are faster than state-of-the-art NNLS solvers.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China (Grant No: 61472347 and Grant No: 61272303).

References

Absil, P.-A.; Amodei, L.; and Meyer, G. 2014. Two newton methods on the manifold of fixed-rank matrices endowed with riemannian quotient geometries. *Computational Statistics* 29(3-4):569–590.

Absil, P.-A.; Mahony, R.; and Sepulchre, R. 2009. *Optimization algorithms on matrix manifolds*. Princeton University Press.

Avron, H.; Kale, S.; Sindhwani, V.; and Kasiviswanathan, S. P. 2012. Efficient and practical stochastic subgradient descent for nuclear norm regularization. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 1231–1238.

Dror, G.; Koenigstein, N.; Koren, Y.; and Weimer, M. 2012. The yahoo! music dataset and kdd-cup'11. In *KDD Cup*, 8–18.

Dudik, M.; Harchaoui, Z.; and Malick, J. 2012. Lifted coordinate descent for learning with trace-norm regularization. In *AISTATS*.

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

He, B., and Yuan, X. 2012. On the o(1/n) convergence rate of the douglas-rachford alternating direction method. *SIAM Journal on Numerical Analysis* 50(2):700–709.

Herlocker, J. L.; Konstan, J. A.; Borchers, A.; and Riedl, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 230–237.

Hsieh, C.-J., and Olsen, P. 2014. Nuclear norm minimization via active subspace selection. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 575–583.

Jaggi, M., and Sulovsk, M. 2010. A simple algorithm for nuclear norm regularized problems. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 471–478.

KDDCup. 2007. Acm sigkdd and netflix. In *Proceedings of KDD Cup and Workshop*.

Lee, S., and Wright, S. J. 2012. Manifold identification in dual averaging for regularized stochastic online learning. *The Journal of Machine Learning Research* 13:1705–1744.

Liang, J.; Fadili, J.; Peyré, G.; and Luke, R. 2015. Activity identification and local linear convergence of douglas–rachford/admm under partial smoothness. In *Scale Space and Variational Methods in Computer Vision*. Springer. 642–653.

Liang, J.; Fadili, J.; and Peyré, G. 2014. Local linear convergence of forward–backward under partial smoothness. In *Advances in Neural Information Processing Systems*, 1970–1978.

Lin, Z.; Ganesh, A.; Wright, J.; Wu, L.; Chen, M.; and Ma, Y. 2009. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)* 61.

Lu, C.; Zhu, C.; Xu, C.; Yan, S.; and Lin, Z. 2015. Generalized singular value thresholding. *AAAI*.

Mazumder, R.; Hastie, T.; and Tibshirani, R. 2010. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research* 11:2287–2322.

Mishra, B., and Sepulchre, R. 2014a. R3mc: A riemannian three-factor algorithm for low-rank matrix completion. In *IEEE 53rd Annual Conference on Decision and Control*, 1137–1142.

Mishra, B., and Sepulchre, R. 2014b. Riemannian preconditioning. *arXiv preprint arXiv:1405.6055*.

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

Mishra, B.; Meyer, G.; Bonnabel, S.; and Sepulchre, R. 2014. Fixed-rank matrix factorizations and riemannian low-rank optimization. *Computational Statistics* 29(3-4):591–621.

Mishra, B. 2014. *A Riemannian approach to large-scale constrained least-squares with symmetries*. Ph.D. Dissertation, Université de Namur.

Negahban, S., and Wainwright, M. J. 2012. Restricted strong convexity and weighted matrix completion: Optimal bounds with noise. *The Journal of Machine Learning Research* 13(1):1665–1697.

Pong, T. K.; Tseng, P.; Ji, S.; and Ye, J. 2010. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization* 20(6):3465–3489.

Recht, B.; Fazel, M.; and Parrilo, P. A. 2010. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review* 52(3):471–501.

Tan, M.; Tsang, I. W.; Wang, L.; Vandereycken, B.; and Pan, S. J. 2014. Riemannian pursuit for big matrix recovery. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 1539–1547.

Toh, K.-C., and Yun, S. 2010. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization* 6(615-640):15.

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

Wen, Z.; Yin, W.; and Zhang, Y. 2012. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation* 4(4):333–361.

Yang, J., and Yuan, X. 2013. Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization. *Mathematics of Computation* 82(281):301–329.