A Unified Framework for Structured Low-rank Matrix Learning

Pratik Jawanpuria [1] **Bamdev Mishra** [1]

Abstract

We consider the problem of learning a low-rank matrix, constrained to lie in a linear subspace, and introduce a novel factorization for modeling such matrices. A salient feature of the proposed factorization scheme is it decouples the low-rank and the structural constraints onto separate factors. We formulate the optimization problem on the Riemannian spectrahedron manifold, where the Riemannian framework allows to develop computationally efficient conjugate gradient and trust-region algorithms. Experiments on problems such as standard/robust/non-negative matrix completion, Hankel matrix learning and multi-task learning demonstrate the efficacy of our approach.

1. Introduction

Our focus in this paper is on learning structured low-rank matrices and we consider the following problem:

$$\begin{array}{ll} \min\limits_{\mathbf{W}\in\mathbb{R}^{d\times T}} & CL(\mathbf{Y},\mathbf{W}) + \|\mathbf{W}\|_*^2, \\ \text{subject to} & \mathbf{W}\in\mathcal{D}, \end{array} \tag{1}$$

where $\mathbf{Y}\in\mathbb{R}^{d\times T}$ is a given matrix, $L:\mathbb{R}^{d\times T}\times\mathbb{R}^{d\times T}\to\mathbb{R}$ is a convex loss function, $\|\cdot\|_*$ denotes the nuclear norm regularizer, $C>0$ is the cost parameter, and $\mathcal{D}$ is the *linear* subspace corresponding to structural constraints. It is well known the nuclear norm regularization promotes low rank solutions since $\|\mathbf{W}\|_*$ is equal to the ℓ_1-norm on the singular values of $\mathbf{W}$ (Fazel et al., 2001). The linear subspace $\mathcal{D}$ in problem (1) is represented as $\mathcal{D} := \{\mathbf{W} : \mathcal{A}(\mathbf{W})\diamondsuit\mathbf{0}\}$, where $\mathcal{A}:\mathbb{R}^{d\times T}\to\mathbb{R}^n$ is a linear map and $\diamondsuit$ represents equality ($=$) or greater than equal to ($\geq$) constraint.

Low-rank matrices are commonly learned in several machine learning applications such as matrix completion (Abernethy et al., 2009; Boumal and Absil, 2011), multi-task learning (Amit et al., 2007; Argyriou et al., 2008; Zhang and Yeung, 2010; Jawanpuria and Nath, 2012), multivariate regression (Yuan et al., 2007; Journée et al., 2010), to name a few. In addition to the low-rank constraint, other structural constraints may exist, e.g., entry-wise *non-negative/bounded* constraints (Kannan et al., 2012; Marecek et al., 2017; Fang et al., 2017). Several linear dynamical system models require learning a low-rank *Hankel* matrix (Fazel et al., 2013; Markovsky and Usevich, 2013). A Hankel matrix has the structural constraint that all its anti-diagonal entries are the same. In robust matrix completion and robust PCA problems (Wright et al., 2009), the matrix is learned as a superimposition of a low-rank matrix and a sparse matrix. This sparse structure is modeled effectively by choosing a robust loss function (Cambier and Absil, 2016), such as the ℓ_1-loss.

We propose a generic framework to the structured low-rank matrix learning problem (1) that is well suited for handling a range of smooth/non-smooth loss functions L, structural constraints $\mathbf{W}\in\mathcal{D}$, and is scalable for large-scale problem instances. Using the duality theory, we introduce a novel modeling of structured low-rank matrix $\mathbf{W}$ of rank r as $\mathbf{W} = \mathbf{U}\mathbf{U}^\top(\mathbf{Z}+\mathbf{A})$, where $\mathbf{U}\in\mathbb{R}^{d\times r}$ and $\mathbf{Z},\mathbf{A}\in\mathbb{R}^{d\times T}$. It can be observed that our factorization naturally decouples the low-rank and structural constraints on $\mathbf{W}$. The low-rank of $\mathbf{W}$ is enforced by $\mathbf{U}$, the structural constraint is modeled by $\mathbf{A}$, and the loss function specific structure is modeled by $\mathbf{Z}$. The decoupling of low-rank and structural constraints onto separate factors makes the resulting optimization conceptually simpler. To the best of our knowledge, such a decoupling of constraints has not been studied in the existing structured low-rank matrix learning literature (Fazel et al., 2013; Markovsky and Usevich, 2013; Yu et al., 2014; Cambier and Absil, 2016; Fang et al., 2017).

Our approach leads to an optimization problem on the *Riemannian spectrahedron* manifold. We exploit the Riemannian framework to develop computationally efficient conjugate gradient and trust-region algorithms. The proposed algorithms perform well in several applications such as standard/robust/non-negative matrix completion, Hankel matrix learning application, and multi-task learning. Our algorithms readily scale to the Netflix data set, even with the non-smooth ℓ_1-loss and ϵ-SVR (ϵ-insensitive support vector regression) loss functions.

[1]Microsoft, India. Correspondence to: Pratik Jawanpuria <pratik.jawanpuria@microsoft.com>, Bamdev Mishra <bamdevm@microsoft.com>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

The main contributions of this work are:

- we propose a novel factorization $\mathbf{W} = \mathbf{U}\mathbf{U}^\top(\mathbf{Z} + \mathbf{A})$ for modeling structured low-rank matrices.
- we present a unified framework to learn structured low-rank matrix for several applications with different constraints and loss functions.
- we develop efficient Riemannian conjugate gradient and trust-region algorithms, which obtain state-of-the-art generalization performance across applications.

The proofs of all the theorems, additional details and experiments are provided in the longer version of the paper (Jawanpuria and Mishra, 2017). Our codes are available at `https://pratikjawanpuria.com/`. We begin by discussing the related works in the next section.

2. Related Work

Matrix completion: Existing low-rank matrix learning literature has been primarily focused on problem (1) with the square loss and in the absence of the structural constraint $\mathbf{W} \in \mathcal{D}$. Singular value thresholding (Cai et al., 2010), proximal gradient descent (Toh and Yun, 2010), active subspace selection (Hsieh and Olsen, 2014) are some of the algorithms proposed to solve (1) without the structural constraint $\mathbf{W} \in \mathcal{D}$. Alternatively, several works (Wen et al., 2012; Mishra and Sepulchre, 2014; Boumal and Absil, 2015) propose to learn a low-rank matrix by fixing the rank explicitly, i.e. $\mathbf{W} = \mathbf{U}\mathbf{V}^\top$, where $\mathbf{U} \in \mathbb{R}^{d\times r}$, $\mathbf{V} \in \mathbb{R}^{T\times r}$ and the rank r is fixed *a priori*.

Robust matrix completion: A matrix completion problem where few of the observed entries are perturbed/outliers (Candès et al., 2011). Recent works (He et al., 2012; Cambier and Absil, 2016) model it as a low-rank matrix completion problem with robust loss functions such as the ℓ_1-loss or the pseudo-Huber loss. In particular, Cambier and Absil (2016) develop a large-scale Riemannian conjugate gradient algorithm for this application.

Non-negative matrix completion: Certain recommender system and image completion based applications desire matrix completion with non-negative entries (Kannan et al., 2012; Sun and Mazumder, 2013; Tsagkatakis et al., 2016; Fang et al., 2017), i.e $\mathbf{W} \geq \mathbf{0}$. Kannan et al. (2012) present a block coordinate descent algorithm that learns $\mathbf{W}$ as $\mathbf{W} = \mathbf{U}\mathbf{V}^\top$ for a given rank r. Recently, Fang et al. (2017) propose a large-scale alternating direction method of multipliers (ADMM) algorithm for solving (1) in the fixed-rank setting.

Hankel matrix learning: The Hankel constraint is a linear equality constraint. Fazel et al. (2013) propose the ADMM approaches to solve (1) with the above constraint. On the other hand, Yu et al. (2014) learn a low-rank Hankel matrix by relaxing the structural constraint with a penalty term in the objective function. Markovsky and Usevich (2013) model the low-rank Hankel matrix learning problem as a non-linear least square problem in the fixed rank setting and propose a second-order algorithm.

Multi-task feature learning: The goal in multi-task feature learning (Argyriou et al., 2008; Jawanpuria and Nath, 2011) is to jointly learn a low-dimensional latent feature representation common across several classification/regression problems (tasks). Problems such as multi-class classification (Amit et al., 2007), multi-variate regression (Yuan et al., 2007), matrix completion with side information (Xu et al., 2013), among others, may be viewed as special cases of multi-task feature learning.

In the following section, we present a unified framework for solving the above problems.

3. Structured Low-rank Matrix Learning

For notational simplicity, we consider only equality structural constraint $\mathcal{A}(\mathbf{W}) = \mathbf{0}$ to present the main results. However, our framework also admits inequality constraints $\mathcal{A}(\mathbf{W}) \geq 0$. We use the notation $\mathcal{P}^d$ to denote the set of $d \times d$ positive semi-definite matrices with *unit* trace.

Problem (1) is a convex problem with linear constraint. In addition to the trace-norm regularizer, the loss function L may also be non-smooth (as in the case of ℓ_1-loss or ϵ-SVR loss). Dealing with (1) directly or characterizing the nature of its optimal solution in the general setting is non trivial. To this end, we propose an equivalent partial dual of (1) in the following section. The use of dual framework often leads to a better understanding of the primal problem (Jawanpuria et al., 2015). In our case, the duality theory helps in discovering a novel factorization of its optimal solution, which is not evident directly from the primal problem (1). This subsequently helps in the development of computationally efficient algorithms.

3.1. Decoupling of Constraints and Duality

The following theorem presents a dual problem equivalent to the primal problem (1).

Theorem 1 *Let L^* be the Fenchel conjugate function of the loss: $L : \mathbb{R}^{d\times T} \to \mathbb{R}$, $v \mapsto L(\mathbf{Y}, v)$. An equivalent partial dual problem of (1) with $\mathcal{A}(\mathbf{W}) = \mathbf{0}$ constraint is*

$$\min_{\Theta\in\mathcal{P}^d} \quad \max_{\mathbf{Z}\in\mathbb{R}^{d\times T},\, s\in\mathbb{R}^n} \quad f(\Theta, \mathbf{Z}, s), \tag{2}$$

where the function f is defined as

$$f(\Theta, \mathbf{Z}, s) := -CL^*(\frac{-\mathbf{Z}}{C}) - \frac{1}{2}\left\langle \Theta(\mathbf{Z} + \mathcal{A}^*(s)), \mathbf{Z} + \mathcal{A}^*(s)\right\rangle$$

and $\mathcal{A}^* : \mathbb{R}^n \to \mathbb{R}^{d\times T}$ is the adjoint of $\mathcal{A}$.

Proof sketch: The proof employs a variational characterization of the trace-norm regularizer (Argyriou et al., 2006) and the KKT conditions of (1).

Our next result gives the expression of an optimal solution of the primal problem (1).

Theorem 2 *(Representer theorem) Let $\{\bar{\Theta}, \bar{\mathbf{Z}}, \bar{s}\}$ be an optimal solution of (2). Then, a corresponding optimal solution $\bar{\mathbf{W}}$ of (1) is*

$$\bar{\mathbf{W}} = \bar{\Theta}(\bar{\mathbf{Z}} + \mathcal{A}^*(\bar{s})).$$

Remark 1: Θ in (2) is a positive semi-definite with unit trace. Hence, an optimal $\bar{\Theta}$ in (2) is a low-rank matrix.

Remark 2: Theorem 1 gives an expression for $\bar{\mathbf{W}}$ of (1) as a function of $\bar{\Theta}$ and $\bar{\mathbf{Z}} + \mathcal{A}^*(\bar{s})$. The low-rank constraint is enforced through $\bar{\Theta}$, the loss-specific structure (encoded in L^*) is enforced through $\bar{\mathbf{Z}}$, and the structural constraint is enforced through $\mathcal{A}^*(\bar{s})$. Overall, such a decoupling of constraints onto separate variables facilitates the use of simpler optimization techniques as compared to the case where all the constraints are enforced on a single variable.

As discussed earlier, an optimal $\bar{\Theta}$ of (2) is a low-rank positive semi-definite matrix. However, an algorithm for (2) need not produce intermediate iterates that are low rank. For large-scale optimization, this observation as well as other computational efficiency concerns motivate a fixed-rank parameterization of Θ as discussed in the following.

3.2. A Novel Fixed-rank Factorization of W

We model $\Theta \in \mathcal{P}^d$ as a rank r matrix in the following way: $\Theta = \mathbf{U}\mathbf{U}^\top$, where $\mathbf{U} \in \mathbb{R}^{d\times r}$ and $\|\mathbf{U}\|_F = 1$. The proposed modeling has several benefits in large-scale low-rank matrix learning problems, where $r \ll \min\{d, T\}$ is a common setting. *First*, the parameterization ensures that $\Theta \in \mathcal{P}^d$ constraint is always satisfied. This saves the costly projection operations to ensure $\Theta \in \mathcal{P}^d$. Enforcing $\|\mathbf{U}\|_F = 1$ constraint costs $O(rd)$. *Second*, the dimension of the search space of problem (2) with $\Theta = \mathbf{U}\mathbf{U}^\top$ is $rd - 1 - r(r-1)/2$, which is much lower than the dimension $(d(d+1)/2 - 1)$ of $\Theta \in \mathcal{P}^d$. By restricting the search space for Θ, we gain computational efficiency. *Third*, increasing the parameter C in (1) and (2) promotes low training error but high rank of the solution, and vice-versa. The proposed fixed-rank parameterization decouples this trade-off.

Remark 3: With the proposed fixed-rank parameterization of Θ, the expression for $\mathbf{W}$ becomes $\mathbf{U}\mathbf{U}^\top(\mathbf{Z} + \mathcal{A}^*(s))$.

Instead of solving a minimax objective as in (2), we solve a minimization problem after incorporating the $\Theta = \mathbf{U}\mathbf{U}^\top$ parameterization as follows:

$$\min_{\mathbf{U}\in\mathbb{R}^{d\times r}, \|\mathbf{U}\|_F=1} \; g(\mathbf{U}), \tag{3}$$

where the function g is defined as

$$g(\mathbf{U}) \coloneqq \max_{\mathbf{Z}\in\mathbb{R}^{d\times T}, s\in\mathbb{R}^n} \; -CL^*(-\mathbf{Z}/C) - \frac{1}{2}\left\|\mathbf{U}^\top(\mathbf{Z} + \mathcal{A}^*(s))\right\|_F^2. \tag{4}$$

It should be noted that (3) is the proposed *generic* structured low-rank matrix learning problem. The application-specific details are modeled within the sub-problem (4). In Section 5, we present specialized expressions of (4), tailored for various applications. We propose a unified optimization framework for solving (3) in Section 4.

The fixed-rank parameterization, $\Theta = \mathbf{U}\mathbf{U}^\top$, results in non-convexity of the overall optimization problem (3), though sub-problem (4) is a convex optimization problem. We end this section by stating sufficient conditions of obtaining a globally optimal solution of (2) from a solution of (3).

Theorem 3 *Let $\hat{\mathbf{U}}$ be a feasible solution of (3) and $\{\hat{\mathbf{Z}}, \hat{s}\}$ be an optimal solution of the convex problem in (4) at $\mathbf{U} = \hat{\mathbf{U}}$. Let σ_1 be the maximum singular of the matrix $\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s})$. A candidate solution for (2) is $\{\hat{\Theta}, \hat{\mathbf{Z}}, \hat{s}\}$, where $\hat{\Theta} = \hat{\mathbf{U}}\hat{\mathbf{U}}^\top$. The duality gap ($\Delta$) associated with $\{\hat{\Theta}, \hat{\mathbf{Z}}, \hat{s}\}$ is given by*

$$\Delta = \frac{1}{2}\Big(\sigma_1^2 - \Big\|\hat{\mathbf{U}}^\top(\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s}))\Big\|_F^2\Big).$$

Furthermore, if $\hat{\mathbf{U}}$ is a rank deficient local minimum of (3), then $\{\hat{\Theta}, \hat{\mathbf{Z}}, \hat{s}\}$ is a global minimum of (2), i.e., $\Delta = 0$.

The value of the duality gap Δ can be used as to verify whether a candidate solution $\{\hat{\Theta}, \hat{\mathbf{Z}}, \hat{s}\}$ is a global optimum of (2). The cost of computing σ_1 is computationally cheap as it requires only a few *power iteration* updates.

4. Optimization on Spectrahedron Manifold

The matrix $\mathbf{U}$ lies in, what is popularly known as, the *spectrahedron* manifold $\mathcal{S}_r^d \coloneqq \{\mathbf{U} \in \mathbb{R}^{d\times r} : \|\mathbf{U}\|_F = 1\}$. Specifically, the spectrahedron manifold has the structure of a compact Riemannian quotient manifold (Journée et al., 2010). The quotient structure takes the rotational invariance of the constraint $\|\mathbf{U}\|_F = 1$ into account. The Riemannian optimization framework embeds the constraint $\mathbf{U} \in \mathcal{S}_r^d$ into the search space, conceptually translating the constrained optimization problem (3) into an *unconstrained* optimization over the spectrahedron manifold. The Riemannian optimization framework generalizes various classical first- and second-order Euclidean algorithms (e.g., the conjugate gradient and trust region algorithms) to manifolds and provide concrete convergence guarantees (Edelman et al., 1998; Absil et al., 2008; Journée et al., 2010; Sato and Iwai, 2013; Sato et al., 2017). In particular, Absil et al. (2008) provide a systematic way of implementing Riemannian conjugate gradient (CG) and trust region (TR) algorithms.

Algorithm 1 Proposed first- and second-order algorithms for (3)

Input: matrix $\mathbf{Y}$, rank r, regularization parameter C.
Initialize $\mathbf{U} \in \mathcal{S}_r^d$.
repeat
1: Solve for $\{\mathbf{Z}, s\}$ by computing $g(\mathbf{U})$ in (4). Section 5 discusses solvers for specific applications.
2: Compute $\nabla g(\mathbf{U})$ as given in Lemma 1.
3: Riemannian CG step: compute a conjugate direction $\mathbf{V}$ and step size α using Armijo line search. It makes use of $\nabla g(\mathbf{U})$.
3: Riemannian TR step: compute a search direction $\mathbf{V}$ which minimizes the trust region sub-problem. It makes use of $\nabla g(\mathbf{U})$ and its directional derivative. Step size $\alpha = 1$.
4: Update $\mathbf{U} = (\mathbf{U} + \alpha\mathbf{V}) / \|\mathbf{U} + \alpha\mathbf{V}\|_F$ (retraction step)
until convergence
Output: $\{\mathbf{U}, \mathbf{Z}, s\}$ and $\mathbf{W} = \mathbf{U}\mathbf{U}^\top(\mathbf{Z} + \mathcal{A}^*(s))$.

We implement the Riemannian conjugate gradient (CG) and trust-region (TR) algorithms for (3). These require the notions of the *Riemannian gradient* (first-order derivative of the objective function on the manifold), *Riemannian Hessian* along a search direction (the *covariant* derivative of the Riemannian gradient along a tangential direction on the manifold), and the *retraction* operator (that ensures that we always stay on the manifold). The Riemannian gradient and Hessian notions require computations of the Euclidean gradient and the directional derivative of this gradient along a given search direction, which are expressed in the following lemma.

Lemma 1 *Let $\{\hat{\mathbf{Z}}, \hat{s}\}$ be an optimal solution of the convex problem (4) at $\mathbf{U}$. Then, the gradient of $g(\mathbf{U})$ at $\mathbf{U}$ is given by the following expression:*

$$\nabla g(\mathbf{U}) = -(\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s}))(\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s}))^\top \mathbf{U}.$$

Let $\mathrm{D}\nabla g(\mathbf{U})[\mathbf{V}]$ denote the directional derivative of the gradient $\nabla g(\mathbf{U})$ along $\mathbf{V} \in \mathbb{R}^{d\times r}$. Let $\{\dot{\mathbf{Z}}, \dot{s}\}$ denote the directional derivative of $\{\mathbf{Z}, s\}$ along $\mathbf{V}$ at $\{\hat{\mathbf{Z}}, \hat{s}\}$. Then,

$$\begin{aligned}\mathrm{D}\nabla g(\mathbf{U})[\mathbf{V}] &= (\dot{\mathbf{Z}} + \mathcal{A}^*(\dot{s}))(\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s}))^\top \mathbf{U} \\ &+ (\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s}))\big((\dot{\mathbf{Z}} + \mathcal{A}^*(\dot{s}))^\top \mathbf{U} - (\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s}))^\top \mathbf{V}\big).\end{aligned}$$

The terms $\{\dot{\mathbf{Z}}, \dot{s}\}$ are computed from the first-order KKT conditions of the convex problem (4) at $\{\hat{\mathbf{Z}}, \hat{s}\}$. In particular, when $L^*(\cdot)$ is differentiable (e.g., the square loss), $\{\dot{\mathbf{Z}}, \dot{s}\}$ are obtained by solving the following linear system:

$$\mathrm{D}\nabla L^*(-\mathbf{Z}/C)[\dot{\mathbf{Z}}] - \mathbf{Q} = \mathbf{0} \quad \text{and} \quad \mathcal{A}\big(\mathbf{Q}\big) = \mathbf{0}$$

where $\mathbf{Q} = (\mathbf{U}\mathbf{V}^\top + \mathbf{V}\mathbf{U}^\top)(\hat{\mathbf{Z}} + \mathcal{A}^*(\hat{s})) + \mathbf{U}\mathbf{U}^\top(\dot{\mathbf{Z}} + \mathcal{A}^*(\dot{s}))$. In various applications such as matrix completion or multi-task learning, both the gradient $\nabla g(\mathbf{U})$ and its directional derivation $\mathrm{D}\nabla g(\mathbf{U})[\mathbf{V}]$ can be computed by closed-form expressions.

Riemannian CG algorithm: It computes the Riemannian *conjugate* gradient direction by employing the first-order information $\nabla g(\mathbf{U})$ (Lemma 1). We perform *Armijo* line search on $\mathcal{S}_r^d$ to compute a step-size that sufficiently decreases $g(\mathbf{U})$ on the manifold. We update along the conjugate direction with the step-size by *retraction*.

Riemannian TR algorithm: It solves a Riemannian trust-region *sub-problem* (in a neighborhood) at every iteration. Solving the trust-region sub-problem leads to a search direction that minimizes a *quadratic* approximation of $g(\mathbf{U})$ on the manifold. Solving this sub-problem does not require inverting the full Hessian of the objective function. It makes use of $\nabla g(\mathbf{U})$ and its directional derivative $\mathrm{D}\nabla g(\mathbf{U})[\mathbf{V}]$.

Overall algorithm: Algorithm 1 summarizes the proposed first- and second-order algorithms for solving (3).

Computational complexity: The computational complexity of Algorithm 1 is the sum of the cost of manifold related operations and the cost of application specific ingredients. The spectrahedron manifold operations cost $O(dr+r^3)$. The following section discusses the application specific computational costs.

Although we have focused on batch algorithms, our framework can be extended to stochastic settings, *e.g.*, when the columns are streamed one by one (Bonnabel, 2013). In this case, when a new column is received, we perform a (stochastic) gradient update on $\mathcal{S}_r^d$.

5. Specialized Formulations for Applications

The expression of $g(\mathbf{U})$ in (4) depends on the functions $L(\cdot)$ and $\mathcal{A}(\cdot)$ employed in the application at hand. Below, we discuss $g(\mathbf{U})$ for popular applications.

5.1. Matrix Completion

Given a partially observed matrix $\mathbf{Y}$ at indices Ω, the aim here is to learn the full matrix $\mathbf{W}$ (Toh and Yun, 2010; Cai et al., 2010). Let Ω_t be the set of indices that are observed in y_t, the t^{th} column of $\mathbf{Y}$. Let $y_{t_{\Omega_t}}$ and $\mathbf{U}_{\Omega_t}$ represents the rows of y_t and $\mathbf{U}$, respectively, that correspond to the indices in Ω_t. Then, the function $g(\mathbf{U})$ in (4) for low-rank matrix completion problem with square loss is (5) in Ta-

Table 1. Specialized expression for (4) across different structured low-rank matrix learning applications.

Application	$g(\mathbf{U}\mathbf{U}^\top)$ as defined in (4)	
Matrix completion	$\sum_{t=1}^{T} \max_{z_t \in \mathbb{R}^{\lvert\Omega\rvert_t}} \langle y_{t_{\Omega_t}}, z_t \rangle - \frac{1}{4C}\|z_t\|^2 - \frac{1}{2}\|\mathbf{U}_{\Omega_t}^\top z_t\|^2$	(5)
Robust matrix completion	$\sum_{t=1}^{T} \max_{z_t \in [-C,C]^{\lvert\Omega\rvert_t}} \langle y_{t_{\Omega_t}}, z_t \rangle - \frac{1}{2}\|\mathbf{U}_{\Omega_t}^\top z_t\|^2$	(6)
Non-negative matrix completion	$\sum_{t=1}^{T} \max_{s_t \in [0,\infty)^{\lvert\Omega\rvert_t}} \max_{z_t \in \mathbb{R}^{\lvert\Omega\rvert_t}} \langle y_{t_{\Omega_t}}, z_t \rangle - \frac{1}{4C}\|z_t\|^2 - \frac{1}{2}\|\mathbf{U}_{\Omega_t}^\top z_t + \mathbf{U}^\top s_t)\|^2$	(7)
Hankel matrix learning	$\max_{s_t \in \mathbb{R}^d \forall t, z \in \mathbb{R}^{d+T-1}} \langle y, z \rangle - \frac{1}{4C}\|z\|^2 - \frac{1}{2}\sum_{t=1}^{T}\|\mathbf{U}^\top s_t\|^2$ subject to : $\sum_{(i,t):i+t=k} s_{ti} = z_k \ \forall k \in \{2,3,\ldots,d+T\}, i \in \{1,2,\ldots,d\}, t \in \{1,2,\ldots,T\}$	(8)
Multi-task feature learning	$\sum_{t=1}^{T} \max_{z_t \in \mathbb{R}^{n_t}} \langle y_t, z_t \rangle - \frac{1}{4C}\|z_t\|^2 - \frac{1}{2}\|\mathbf{U}^\top \mathbf{X}_t^\top z_t\|^2$	(9)

ble 1. Problem (5) is a least-squares problem for each z_t and is solved efficiently in closed-form with $O(|\Omega|r^2)$ time complexity. By taking into account the manifold operation costs (discussed in Section 4), the overall per-iteration computational cost of Algorithm 1 is $O(|\Omega|r^2 + dr + r^3)$.

5.2. Robust Matrix Completion

We solve for this application by employing the ℓ_1-loss function in low-rank matrix completion problem. The expression for $g(\mathbf{U})$ is (6) in Table 1. Coordinate descent (CD) algorithm is employed to efficiently solve (6). The cost of computing $g(\mathbf{U})$ is $O(|\Omega|kr^2)$, where k is the number of iterations of the CD algorithm. Overall, the per-iteration computational cost of Algorithm 1 is $O(|\Omega|kr^2 + dr + r^3)$. We also experiment with the ϵ-SVR loss in this setting.

5.3. Non-negative Matrix Completion

In this application, problem $g(\mathbf{U})$ with the square loss gets specialized to (7) in Table 1. Since the dual variables s_t have non-negative constraints (due to the constraint $\mathbf{W} \geq 0$ in (1)), we model (7) as a non-negative least squares (NNLS) problem. We employ the NNLS algorithm of Kim et al. (2013) to solve for the variable s_t in (7). In each iteration of NNLS, z_t is solved in closed form. If k is the number of iterations of NNLS, then the cost of computing $g(\mathbf{U})$ is $O(dTkr + |\Omega|kr^2)$. Overall, the per-iteration computational cost of Algorithm 1 is $O(dTkr + |\Omega|kr^2 + dr + r^3)$.

It should be noted that (7) is computationally challenging as it has dT entry-wise non-negativity constraints. In our initial experiments, we observed that the solution $[s_1, \ldots, s_T]$ is highly sparse. For large-scale problems, we exploit this observation for an efficient implementation.

5.4. Hankel Matrix Learning

Hankel matrices have the structural constraint that its anti-diagonal entries are the same. A Hankel matrix corresponding to a vector $y = [y_1, y_2, \ldots, y_7]$ is as follows:

$$\begin{bmatrix} y_1 & y_2 & y_3 & y_4 & y_5 \\ y_2 & y_3 & y_4 & y_5 & y_6 \\ y_3 & y_4 & y_5 & y_6 & y_7 \end{bmatrix}.$$

Hankel matrices play an important role in determining the order or complexity of linear time-invariant systems (Fazel et al., 2013; Markovsky and Usevich, 2013). The aim in such settings is to find the minimum order of the system that explains the observed data (i.e. low error). A low-order system is usually desired as it translates into cost benefit and ease of analysis.

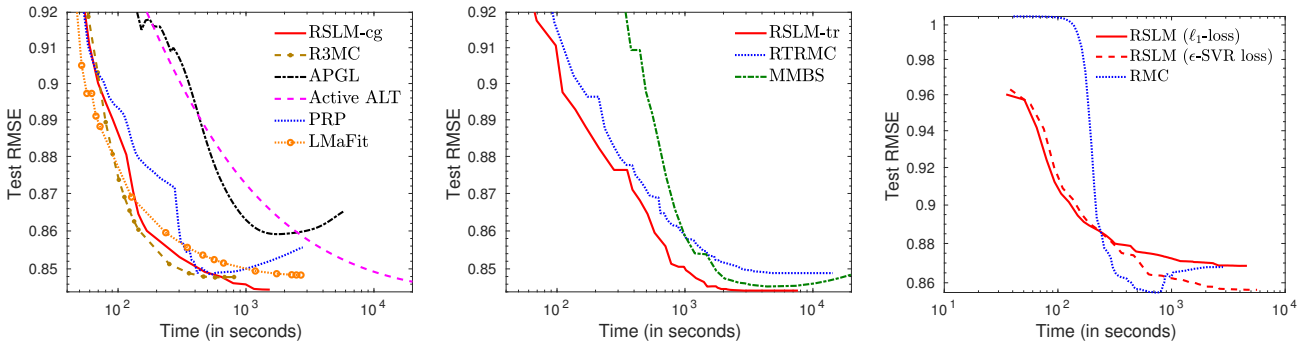
The function $g(\mathbf{U})$ gets specialized to (8) in Table 1. We solve (8) via the conjugate gradient algorithm. The equality constraints are handled efficiently by using an affine projection operator. The computational cost of computing $g(\mathbf{U})$ is $O(dTkr)$, where k is the number of iterations of the inner conjugate gradient algorithm. Overall, the per-iteration complexity of Algorithm 1 is $O(dTkr + dr + r^3)$.

5.5. Multi-task Feature Learning

In this application, every task t has input/output instances $\{\mathbf{X}_t, y_t\}$, where $\mathbf{X}_t \in \mathbb{R}^{n_t \times d}$ and $y_t \in \mathbb{R}^{n_t}$. The aim is to learn the model parameter w_t for each task t such that they share a low-dimensional latent space. The prediction function for task t is $h_t(x) = \langle x, w_t \rangle$. Argyriou et al. (2008) solve the convex primal (1) to learn the task parameters in this setting. The function $g(\mathbf{U})$ for multi-task feature learning is (9) in Table 1, which can be further specialized for problems such as matrix completion with side information, inductive matrix completion, and multi-variate regression.

6. Experiments

In this section, we evaluate the generalization performance as well as computational efficiency of our approach against state-of-the-art in different applications. It should be emphasized that state-of-the-art in each application are different



(a) First-order MC algorithms (b) Second-order MC algorithms (c) Robust MC algorithms

Figure 1. Evolution of test RMSE on the Netflix data set for first- and second-order matrix completion (MC) algorithms as well as robust MC algorithms. Our algorithms converge to the best generalization performance in all the experiments.

Table 2. Data set statistics

Data set	d	T	$\lvert\Omega\rvert$
ML1m	3 706	6 040	1 000 209
ML10m	10 677	71 567	10 000 054
ML20m	26 744	138 493	20 000 263
Netflix	17 770	480 189	100 198 805

and to the best of our knowledge there does not exist a unified framework for solving such applications. All our algorithms are implemented using the Manopt toolbox (Boumal et al., 2014). We term our algorithm as **R**iemannian **S**tructured **L**ow-rank **M**atrix learning (**RSLM**).

6.1. Matrix Completion

Baseline techniques: Our first- and second-order matrix completion algorithms are denoted by RSLM-cg and RSLM-tr, respectively. We compare against state-of-the-art fixed-rank and nuclear norm minimization based matrix completion solvers: APGL: accelerated proximal gradient algorithm for nuclear norm minimization (Toh and Yun, 2010), Active ALT: first-order nuclear norm solver based on active subspace selection (Hsieh and Olsen, 2014), R3MC: fixed-rank Riemannian preconditioned non-linear conjugate gradient algorithm (Mishra and Sepulchre, 2014), LMaFit: nonlinear successive over-relaxation algorithm based on alternate least squares (Wen et al., 2012), MMBS: fixed-rank second-order nuclear norm minimization algorithm (Mishra et al., 2013), RTRMC: fixed-rank second-order Riemannian preconditioned algorithm on the Grassmann manifold (Boumal and Absil, 2011; 2015), and PRP: a recent proximal Riemannian pursuit algorithm (Tan et al., 2016).

Data sets and experimental setup: We compare the performance of the above algorithms on movie recommendation data sets: Netflix (Recht and Ré, 2013), MovieLens10m (ML10m), and MovieLens20m (ML20m) (Harper and Konstan, 2015). The data set statistics is provided in Table 2. For every data set, we create five random $80/20$ train/test splits. For every split, the regularization parameters for respective algorithms are cross-validated to obtain their best performance. All the fixed algorithms (R3MC, LMaFit, MMBS, RTRMC, RSLM) are provided the rank $r = 10$. In other approaches, the maximum rank parameter is set to 10.

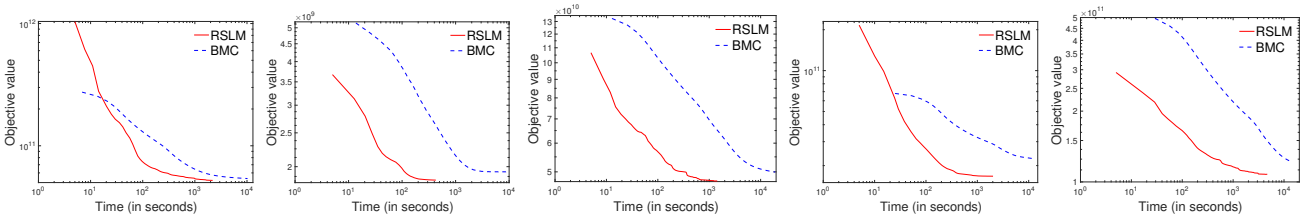
Table 3. Mean test RMSE on matrix completion problems. The proposed first-order (RSLM-cg) and second-order (RSLM-tr) algorithms obtain the best generalization performance.

	Netflix	ML10m	ML20m
RSLM-tr	**0.8443**	**0.8026**	**0.7962**
RSLM-cg	**0.8449**	**0.8026**	**0.7963**
R3MC	0.8478	0.8070	0.7982
RTRMC	0.8489	0.8161	0.8044
APGL	0.8587	0.8283	0.8160
Active ALT	0.8463	0.8116	0.8033
MMBS	0.8454	0.8226	0.8053
LMaFit	0.8484	0.8082	0.7996
PRP	0.8488	0.8068	0.7987

Results: Figures 1(a)&(b) display the evolution of test RMSE against the training time on the Netflix data set (Recht and Ré, 2013) for first- and second-order algorithms, respectively. RSLM-cg is among the most efficient first-order method and RSLM-tr is the best second-order method. Table 3 reports the test RMSE of all the algorithms on the three data sets. We observe that both our algorithms obtain the best generalization performance.

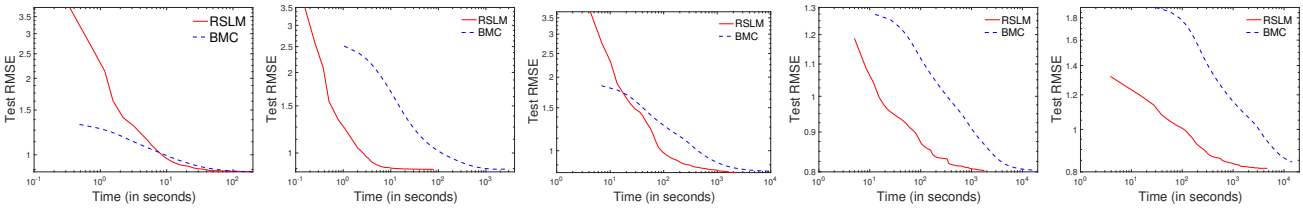
6.2. Robust Matrix Completion

Baseline techniques: We develop CG implementation of RSLM with two different robust loss functions: ℓ_1-loss and ϵ-SVR loss. The non-smooth nature of ℓ_1-loss and ϵ-SVR loss makes them challenging to optimize in large-scale low-rank setting. For evaluation, we compare RSLM against



(a) ML10m, $(r, C) : (5, 1e4)$ (b) ML10m, $(r, C) : (10, 1e2)$ (c) ML10m, $(r, C) : (10, 1e4)$ (d) ML20m, $(r, C) : (5, 1e3)$ (e) ML20m, $(r, C) : (5, 1e4)$

Figure 2. Convergence behavior of RSLM and BMC for different values of regularization parameter and rank on non-negative matrix completion problems. RSLM achieves better objective value in lesser training time than BMC. Both the axes are in $\log_{10}$ scale.



(a) ML1m, $(r, C) : (10, 1e3)$ (b) ML1m, $(r, C) : (20, 1e3)$ (c) ML10m, $(r, C) : (5, 1e4)$ (d) ML10m, $(r, C) : (10, 1e4)$ (e) ML20m, $(r, C) : (5, 1e4)$

Figure 3. Evolution of test RMSE of RSLM and BMC for different values of regularization parameter and rank on non-negative matrix completion problems. RSLM achieves better generalization performance than BMC. Both the axes are in $\log_{10}$ scale.

Table 4. Mean test RMSE on non-negative matrix completion problems. The proposed algorithm RSLM obtains the best generalization performance.

Data set	r	RSLM	BMC	BMA
ML1m	5	**0.8651**	0.8672	0.9476
	10	0.8574	**0.8529**	0.9505
	20	**0.8678**	0.8691	0.9520
ML10m	5	**0.8135**	0.8237	0.8989
	10	**0.8031**	0.8038	0.8832
	20	**0.8148**	0.8842	0.8904
ML20m	5	**0.8142**	0.8454	–
	10	**0.8014**	0.8477	–
	20	**0.8065**	0.9130	–

state-of-the-art RMC algorithm (Cambier and Absil, 2016). RMC is a scalable Riemannian CG algorithm, employing the smooth pseudo-Huber loss function.

Data sets and experimental setup: We compare the performance of all the three algorithms on the Netflix data set. We follow the same experimental setup as described for the case of matrix completion. The rank r for both RSLM and RMC is fixed at $r = 10$.

Results: Figure 1(c) shows the results on the Netflix data set. We observe that RSLM scales effortlessly on the Netflix data set even with non-smooth loss functions and obtains the best generalization performance with the ϵ-SVR loss. The test RMSE obtained at convergence are: **0.857** (RSLM ϵ-SVR loss), 0.869 (RSLM ℓ_1-loss), and 0.868 (RMC).

6.3. Non-negative Matrix Completion

Baseline techniques: We include the following methods in our comparisons: BMC (Fang et al., 2017) and BMA (Kannan et al., 2012). BMC is a recently proposed ADMM based algorithm with carefully designed update rules to ensure an efficient computational and space complexity. BMA is based on co-ordinate descent algorithm.

Data sets and experimental setup: We compare the performance of the above algorithms on three data sets (Table 2): MovieLens1m (ML1m), MovieLens10m (ML10m), and MovieLens20m (ML20m). The experimental setup is the same as described for the case of matrix completion. The performance of all the algorithms are evaluated at three ranks: $r = 5, 10, 20$.

Results: We observe in Table 4 that RSLM outperforms both BMC and BMA. The improvement obtained by RSLM over BMC is more pronounced with larger data sets. BMA is not able to run on MovieLens20m due to high memory and time complexity.

Figures 2(a)-(e) compare the convergence behavior of RSLM and BMC for different values of parameters C and r on all three data sets. Both algorithms aim to minimize the same primal objective (1) for a given rank r. Though the proposed approach solves the proposed fixed-rank dual formulation (3), we compute the corresponding primal objective value of every iterate for the plots. We observe that our algorithm RSLM is significantly faster than BMC in converging to a lower objective value.

Table 5. RMSE on problems that involve learning a low-rank Hankel matrix. The proposed algorithm, RSLM, obtains the best generalization performance.

Data set	r	RSLM	SLRA	DADM	GCG
D_1:	5	**0.0156**	0.0159	0.0628	0.1068
$d = 100$,	10	**0.0171**	0.0190	0.0291	0.0846
$T = 100$	20	**0.0177**	0.0295	0.0196	0.0667
D_2:	5	**0.0059**	0.0164	0.0331	0.0531
$d = 100$,	10	**0.0060**	0.0077	0.0250	0.0386
$T = 1000$	20	**0.0071**	0.0108	0.0159	0.0345
D_3:	5	**0.0149**	**0.0149**	0.0458	0.0458
$d = 1000$,	10	**0.0043**	0.0049	0.0314	0.0340
$T = 10000$	20	**0.0039**	0.0053	0.0288	0.0330

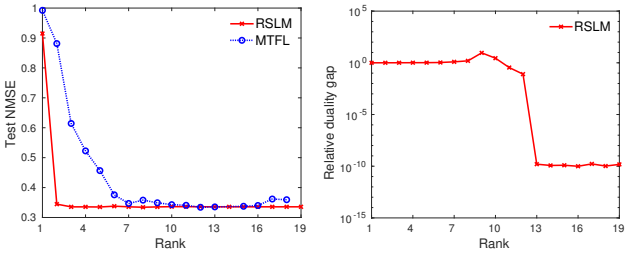


Figure 4. Multi-task feature learning application: (a) Generalization performance vs rank. Our algorithm, RSLM, obtains better generalization performance than MTFL (Argyriou et al., 2008); (b) Relative duality gap vs rank for our algorithm.

Figures 3(a)-(e) plot the evolution of test RMSE against training time for algorithms RSLM and BMC on different data sets with different ranks (and the corresponding best C parameter). We observe that the RSLM outperforms BMC in converging to a lower test RMSE at a much faster rate.

6.4. Hankel Matrix Learning

Baseline techniques: We compare our algorithm RSLM with three methods (discussed in Section 2): GCG (Yu et al., 2014), SLRA (Markovsky, 2014; Markovsky and Usevich, 2014), and DADM (Fazel et al., 2013). Since GCG and DADM employ the nuclear norm regularizer, we tune the regularization parameter to vary the rank of their solution.

Data sets and experimental setup: Given a vector $y \in \mathbb{R}^{d+T-1}$, we obtain a $d \times T$ Hankel matrix as discussed in Section 5.4. The true parameter y is generated as the impulse response (skipping the first sample) of a discrete-time random linear time-invariant system of order r_0 (Markovsky, 2011; Fazel et al., 2013). The noisy estimate of these parameters ($\bar{y}$) are generated as $\bar{y} = y + \sigma\epsilon$, where $\sigma\epsilon$ is the measurement noise. We set $\sigma = 0.05$ and generate ϵ from the standard Gaussian distribution $N(0, 1)$. It should be noted that $\bar{y}$ is the *train* data and y is the *true* data.

We generate three different data set D_1, D_2, D_3 of varying size and order. D_1 has $(r_0, d, T) = (5, 100, 100)$, where r_0 is the true order of the underlying system. The other two data sets, D_2 and D_3, has the configurations $(10, 100, 1000)$ and $(20, 1000, 10000)$, respectively. We evaluate the algorithms on all the three data sets and report their RMSE with respect to the true data (Liu and Vandenberghe, 2009) at different ranks ($r = 5, 10, 20$) in Table 5.

Results: We observe from Table 5 that the proposed algorithm RSLM obtains the best result in all the three data sets and across different ranks. In addition, RSLM also usually obtains the lowest true RMSE for a data set at the rank r equal to r_0 of the data set. This implies that RSLM is able to identify the minimal order of the systems corresponding to the data sets.

6.5. Multi-task Feature Learning

Experimental setup: We compare the generalization performance of our algorithm RSLM with the convex multi-task feature learning algorithm MTFL (Argyriou et al., 2008). Optimal solution for MTFL at different ranks is obtained by tracing the solution path with respect to the regularization parameter, whose value is varied as $\{2^{-8}, 2^{-7}, \ldots, 2^{24}\}$. For RSLM, we fix the value of the regularization parameter C, and vary the rank r to obtain different ranked solutions. The experiments are performed on two benchmark multi-task regression data sets: a) Parkinsons: we need to predict the Parkinson's disease symptom score of 42 patients (Frank and Asuncion, 2010); b) School: we need to predict performance of all students in 139 schools (Argyriou et al., 2008). We report the normalized mean square error over the test set (test NMSE).

Results: Figure 4(a) present the results on the Parkinsons data set. We observe from the figure that our method achieves the better generalization performance at low ranks compared to MTFL. Figure 4(b) plots the variation of duality gap with rank for our algorithm. We observe that as the rank is increased, we converge to the globally optimal solution of (2) and obtain the duality gap close to zero. Similar results are obtained on the School data set.

7. Conclusion

We have proposed a novel factorization for structured low-rank matrix learning problems, which stems from the application of duality theory and rank-constrained parameterization of positive semi-definite matrices. This allows to develop a conceptually simpler and unified optimization framework for various applications. State-of-the-art performance of our algorithms on several applications shows the effectiveness of our approach.

Acknowledgement

We thank Léopold Cambier, Ivan Markovsky, and Konstantin Usevich for useful discussions. We also thank Rodolphe Jenatton, Syama Rangapuram, and Matthias Hein for giving feedback on an earlier version of this work. We are grateful to Adams Wei Yu and Mingkui Tan for making their codes available. Most of this work was done when PJ and BM were at Amazon.com, India.

References

J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10(Mar):803–826, 2009.

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

Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *International Conference on Machine Learning (ICML)*, pages 17–24, 2007.

A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Neural Information Processing Systems conference (NIPS)*, 2006.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73:243–272, 2008.

S. Bonnabel. Stochastic gradient descent on Riemannian manifolds. *IEEE Transactions on Automatic Control*, 58 (9):2217–2229, 2013.

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

N. Boumal and P.-A. Absil. Low-rank matrix completion via preconditioned optimization on the Grassmann manifold. *Linear Algebra and its Applications*, 475:200–239, 2015.

N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt: a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15(Apr): 1455–1459, 2014.

J. F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

L. Cambier and P. A. Absil. Robust low-rank matrix completion by Riemannian optimization. *SIAM J. Sci. Comput.*, 38(5):S440—S460, 2016.

E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3): 11:1–11:37, 2011.

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

H. Fang, Z. Zhen, Y. Shao, and C.-J. Hsieh. Improved bounded matrix completion for large-scale recommender systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1654–1660, 2017.

M. Fazel, H. Hindi, and S. P. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *American Control Conference*, pages 4734–4739, 2001.

M. Fazel, P. T. Kei, D. Sun, and P. Tseng. Hankel matrix rank minimization with applications to system identification and realization. *SIAM Journal on Matrix Analysis and Applications*, 34(3):946–977, 2013.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

F. M. Harper and J. A. Konstan. The MovieLens datasets: history and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, 2015.

J. He, L. Balzano, and A. Szlam. Incremental gradient on the Grassmannian for online foreground and background separation in subsampled video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1568–1575, 2012.

C.-J. Hsieh and P. A. Olsen. Nuclear norm minimization via active subspace selection. In *International Conference on Machine learning (ICML)*, pages 575–583, 2014.

P. Jawanpuria and B. Mishra. Structured low-rank matrix learning: algorithms and applications. Technical report, arXiv preprint arXiv:1704.07352, 2017.

P. Jawanpuria and J. S. Nath. Multi-task multiple kernel learning. In *SIAM International Conference on Data Mining (SDM)*, pages 828–83, 2011.

P. Jawanpuria and J. S. Nath. A convex feature learning formulation for latent task structure discovery. In *International Conference on Machine learning (ICML)*, 2012.

P. Jawanpuria, M. Lapin, M. Hein, and B. Schiele. Efficient output kernel learning for multiple tasks. In *Neural Information Processing Systems conference (NIPS)*, pages 1189–1197, 2015.

M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

R. Kannan, M. Ishteva, and H. Park. Bounded matrix low rank approximation. In *IEEE International Conference on Data Mining (ICDM)*, pages 319–328, 2012.

D. Kim, S. Sra, and I. S. Dhillon. A non-monotonic method for large-scale non-negative least squares. *Optimization Methods and Software*, 28(5):1012–1039, 2013.

Z. Liu and L. Vandenberghe. Semidefinite programming methods for system realization and identification. In *IEEE CDC*, pages 4676–4681, 2009.

J. Marecek, P. Richtarik, and M. Takac. Matrix completion under interval uncertainty. *European Journal of Operational Research*, 256(1):35–43, 2017.

I. Markovsky. *Low Rank Approximation: Algorithms, Implementation, Applications*. Springer Publishing Company, Incorporated, 2011.

I. Markovsky. Recent progress on variable projection methods for structured low-rank approximation. *Signal Processing*, 96(Part B):406–419, 2014.

I. Markovsky and K. Usevich. Structured low-rank approximation with missing data. *SIAM Journal on Matrix Analysis and Applications*, 34(2):814–830, 2013.

I. Markovsky and K. Usevich. Software for weighted structured low-rank approximation. *J. Comput. Appl. Math.*, 256:278–292, 2014.

B. Mishra and R. Sepulchre. R3MC: A Riemannian three-factor algorithm for low-rank matrix completion. In *Proceedings of the 53rd IEEE Conference on Decision and Control (CDC)*, pages 1137–1142, 2014.

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

B. Recht and C. Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. *Mathematical Programming Computation*, 5(2):201–226, 2013.

H. Sato and T. Iwai. A new, globally convergent Riemannian conjugate gradient method. *Optimization: A Journal of Mathematical Programming and Operations Research*, 64(4):1011–1031, 2013.

H. Sato, H. Kasai, and B. Mishra. Riemannian stochastic variance reduced gradient. Technical report, arXiv preprint arXiv:1702.05594, 2017.

D. L. Sun and R. Mazumder. Non-negative matrix completion for bandwidth extension: A convex optimization approach. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2013.

M. Tan, S. Xiao, J. Gao, D. Xu, A. van den Hengel, and Q. Shi. Proximal riemannian pursuit for large-scale trace-norm minimization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5877–5886, 2016.

K. C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *Pacific Journal of Optimization*, 6(3):615–640, 2010.

G. Tsagkatakis, B. Geelen, M. Jayapala, and P. Tsakalides. Non-negative matrix completion for the enhancement of snapshot mosaic multispectral imagery. In *IS&T International Symposium on Electronic Imaging, Image Sensors and Imaging Systems*, 2016.

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

J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Neural Information Processing Systems conference (NIPS)*, pages 2080–2088, 2009.

M. Xu, R. Jin, and Z. Zhou. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2301–2309, 2013.

A. W. Yu, W. Ma, Y. Yu, J. G. Carbonell, and S. Sra. Efficient structured matrix rank minimization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1350–1358, 2014.

M. Yuan, A. Ekici, Z. Lu, and R. Monteiro. Dimension reduction and coefficient estimation in multivariate linear regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(3):329–346, 2007.

Y. Zhang and D. Y. Yeung. A convex formulation for learning task relationships in multi-task learning. In *Uncertainty in Artificial Intelligence*, 2010.