Fast and Low Memory Cost Matrix Factorization: Algorithm, Analysis, and Case Study

Yan Yan, Mingkui Tan, Ivor W. Tsang, Yi Yang, Qinfeng Shi, and Chengqi Zhang

Abstract—Matrix factorization has been widely applied to various applications. With the fast development of storage and internet technologies, we have been witnessing a rapid increase of data. In this paper, we propose new algorithms for matrix factorization with the emphasis on efficiency. In addition, most existing methods of matrix factorization only consider a general smooth least square loss. Differently, many real-world applications have distinctive characteristics. As a result, different losses should be used accordingly. Therefore, it is beneficial to design new matrix factorization algorithms that are able to deal with both smooth and non-smooth losses. To this end, one needs to analyze the characteristics of target data and use the most appropriate loss based on the analysis. We particularly study two representative cases of low-rank matrix recovery, i.e., collaborative filtering for recommendation and high dynamic range imaging. To solve these two problems, we respectively propose a stage-wise matrix factorization algorithm by exploiting manifold optimization techniques. From our theoretical analysis, they are both are provably guaranteed to converge to a stationary point. Extensive experiments on recommender systems and high dynamic range imaging demonstrate the satisfactory performance and efficiency of our proposed method on large-scale real data.

Index Terms—Matrix factorization, matrix recovery, efficient optimization, convergence guarantees, recommender systems, HDR imaging, batch image alignment

1 Introduction

With many advances of Internet technology and the prevalence of electronic devices such as smart phones, it is increasingly easy for users to produce large-scale data. For example, Nokia 808 PureView can capture images with more than 40 mega-pixels. When being processed, an individual image requires around 320 MB memory space to store each RGB channel. This large data size imposes two efficiency challenges for real-world computer vision tasks, i.e., storage and time cost. Let us take batch image alignment, which requires the alignment of a number (e.g., 16) of images of target objects, as an example. Stacking all of these high-resolution images will generate a matrix with more than 640 million elements in each channel, which costs 5.12 GB memory. Some commonly used algorithms require many computation and storage resources. Singular value decompositions (SVDs), for instance, require cubic computation complexity and quadratic storage complexity, which may be unaffordable on large matrices. Consequently, there is a great demand for designing efficient and scalable algorithms for these real-world applications [1], [2], [3], [4], [5].

- *Y. Yan, I.W. Tsang, Y. Yang, and C. Zhang are with the Centre for Artificial Intelligence (CAI), University of Technology Sydney, Sydney, NSW 2007, Australia. E-mail: yan.yan-3@student.uts.edu.au, {ivor.tsang, yi.yang, chengqi.zhang}@uts.edu.au.*
- *M. Tan is with the South China University of Technology, Guangzhou Shi, Guangdong Sheng 510630, China. E-mail: mingkuitan@scut.edu.cn.*
- *Q. Shi is with the Australian Centre for Visual Technologies, University of Adelaide, Adelaide, SA 5000, Australia. E-mail: qinfeng.shi@ieee.org.*

Manuscript received 20 Sept. 2017; revised 23 Oct. 2018; accepted 1 Nov. 2018. Date of publication 19 Nov. 2018; date of current version 8 Jan. 2020. (Corresponding author: Yan Yan.)
Recommended for acceptance by D. Cai.
Digital Object Identifier no. 10.1109/TKDE.2018.2882197

Low-rank matrix recovery (MR) has been exploited to deal with a variety of real-world applications. The main aim of low-rank MR is to recover the underlying low-rank structure of data from the the noisy observations or missing elements. Some real-world problems can be formulated as low-rank MR problems, e.g., collaborative filtering for recommender systems [6], [7], [8], high dynamic range (HDR) imaging [9], [10], [11], batch image alignment [12], etc. Besides these computer vision applications, low-rank MR is also gaining increasing popularity in many other tasks, e.g., recommender systems, where users' preferences on products are modeled as low-rank structures. As aforementioned, the increasing scale of real-world data requires better efficiency of low-rank MR algorithms.

A general low-rank MR problem can be cast as the following optimization problem:

$$\min_{\mathbf{X}} f(\mathbf{X}), \qquad \text{s.t. } \operatorname{rank}(\mathbf{X}) \leq r, \tag{1}$$

where $\mathbf{X} \in R^{m \times n}$ is the recovered low-rank matrix, $f(\mathbf{X})$ is a loss function that measures the fidelity between $\mathbf{X}$ and an observation matrix $\mathbf{Y} \in R^{m \times n}$, and $\operatorname{rank}(\mathbf{X})$ retrieves the rank of $\mathbf{X}$. Here r denotes the maximum rank of $\mathbf{X}$, which is often a small number, leading to a low-rank constraint on $\mathbf{X}$. Note that $f(\mathbf{X})$ is determined by the specific requirements of a certain application, e.g., HDR imaging, multi-label image classification.

Problem (1) is NP-hard due to the rank constraint, i.e., $\operatorname{rank}(\mathbf{X}) \leq r$. Researchers thus have proposed many approaches to solve it. Existing methods can be grouped into two main categories, i.e., *nuclear norm* based approaches and

matrix factorization approaches. Nuclear norm-based methods leverage the nuclear norm as a convex surrogate of the low-rank constraint [13], [14], [15]. However, nuclear norm minimization usually introduces expensive SVDs, e.g., singular value thresholding [16] and proximal gradient methods [17]. This becomes computationally unaffordable when the size of $\mathbf{X}$ is large, particularly on real-world data.

Matrix factorization is another critical approach for low-rank MR. It explicitly factorizes $\mathbf{X}$ into smaller factor matrices, e.g., $\mathbf{X} = \mathbf{U}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{m\times r}$ and $\mathbf{V} \in \mathbb{R}^{n\times r}$ [6], [18], [19]. This results in the problem: $\min_{\mathbf{U},\mathbf{V}} f(\mathbf{U}\mathbf{V}^\top) + \Phi(\mathbf{U},\mathbf{V})$, where $\Phi(\mathbf{U},\mathbf{V})$ is a regularization term of $\mathbf{U}$ and $\mathbf{V}$. Matrix factorization methods are often more efficient for large scale problems than nuclear norm based methods, but they still suffer from some limitations. First, it is unclear how to estimate the rank of $\mathbf{X}$, i.e., r for some real-world applications. Model selection can be applied, but it is often computationally unaffordable for large scale problems. Second, most existing matrix factorization works only focus on a general scenario in which the loss function $f(\mathbf{X})$ is differentiable, e.g., least square loss [20], [21], [22], [23], [24], [25], where $f(\mathbf{X}) = ||\mathbf{Y} - \mathbf{X}||_F^2$. This general scenario simplifies the optimization process but ignores the particular requirements of certain real-world applications. Moreover, some algorithms may be stuck in an unsatisfactory local minimum [18].

In this paper we consider two typical applications of low-rank MR, namely recommender systems for ordinal rating data and HDR imaging, which introduce specific requirements to the problem formulation respectively. In real-world recommender systems, rating data are provided by users on a variety of products, e.g., movies or music tracks. These rating data are often in discrete binary (e.g., like/dislike), or ordinal values (e.g., $1, 2, \ldots, 5$). Existing works have demonstrated the efficacy of maximum margin matrix factorization (M^{3}F) on these data [6], [7], [18], [26], which introduces a large margin for each pair of two consecutive ordinal values (we present the definition in Section 4.1). However, their optimization algorithms may suffer from poor efficiency on large scale data. For example, semi-definite programming (SDP) is applied [6], but SDP usually scales poorly.

In HDR imaging, outlier detection is required to eliminate ghosting artifacts, which are introduced when multiple low dynamic range (LDR) images are combined. Recent contributions to the literature show that sparse large outliers can be detected by robust ℓ_1 loss [9], [10], [11]. This thus leads to the robust matrix factorization problem, in which $f(\mathbf{X}) = ||\mathbf{Y} - \mathbf{X}||_1$, where $\mathrm{rank}(\mathbf{X}) \leq r$. Nevertheless, robust ℓ_1 loss cannot be directly optimized by most existing matrix factorization algorithms due to its non-smoothness.

To cope with the specific requirements of real applications of low-rank MR, we propose an algorithm by exploiting the Riemannian manifold geometry for each of the mentioned problems. Particularly, to deal with the discrete data in recommender systems, we propose an active Riemannian subspace search for M^{3}F (ARSS-M^{3}F). This algorithm leverages an efficient block-wise nonlinear Riemannian conjugate gradient (BNRCG) algorithm to optimize the M^{3}F problem, and exploits a simple and effective method to estimate the rank of the underlying rating matrix. In addition, we propose an efficient augmented Lagrangian multiplier (ALM) framework to handle robust matrix factorization where non-smooth ℓ_1 loss is employed. The proposed framework divides the original problem to simpler subproblems.

The main contributions of this paper are summarized as follows:

- We propose ARSS-M^{3}F algorithm to deal with the discrete ordinal data for recommender systems.
- We propose a robust matrix factorization framework to enable matrix factorization algorithms to handle non-smooth ℓ_1 loss based on ALM.
- We give the theoretical convergence analysis for both of the above two algorithms.
- Extensive experiments on collaborative filtering, HDR imaging and batch image alignment demonstrate the superior performance and efficiency of our proposed method.

The remainder of the paper is organized as follows. We first review related work in Section 2. We then present preliminary knowledge of the Riemannian geometry of fixed-rank matrices in Section 3. In Section 4, we present the M^{3}F for discrete ordinal rating data. We present the framework of matrix factorization on robust ℓ_1 loss in Section 5. In Section 6, we detail our experiments on discrete ordinal rating data and their application to HDR imaging and batch image alignment. Lastly we conclude this paper in Section 7.

2 Related Work

In this section, we review the recent literature on matrix factorization. Most existing works on matrix factorization focus on the general cases in which convex and smooth least square loss is required [20], [21], [22], [23], [24], [25]. Based on the specific application requirements, some works also consider exploiting the maximum margin loss to cope with discrete ordinal rating data for recommender systems, while other works focus on the outlier detection task required by some real-world problems. Given that the outlier detection task is very crucial in many computer vision applications, we also give a brief review of robust principal analysis (RPCA), which has been widely used in computer vision.

2.1 Matrix Factorization

Most matrix factorization algorithms are designed to deal with general cases such as convex and smooth least square loss. Many proposed works based on manifold optimization have recently been proposed. According to these approaches [22], [27], the fixed-rank matrices belong to a smooth matrix manifold. A low rank geometric conjugate gradient (LRGeomCG) method is proposed in [22]. First- and second-order Riemannian trust-region methods are applied to solve low rank matrix completion by exploiting the low rank constraint in [28]. In [29], the authors propose a linear regression algorithm whose parameter is a fixed-rank matrix based on Riemannian manifold geometry. In [30], a quotient geometric matrix completion method is proposed.

An online algorithm for tracking subspaces, Grassmannian rank-one update subspace estimation (GROUSE) is proposed in [31]. In [32], the authors propose to solve matrix factorization problems by alternating minimization, and give the global optimality. To obtain the desired accuracy, this algorithm requires a fresh set of measurements in each

iteration [32], [33]. The authors in [24] propose gradient methods based on scaled metric on the Grassmann manifold. A low-rank matrix fitting algorithm (LMAFIT) is proposed to solve large scale matrix completion problems by exploiting nonlinear successive over-relaxation [23]. However, all these matrix factorization methods consider only the general least square loss, rather than a specific loss in real applications.

The importance of automatic latent factor detection (i.e., the model selection problem) has been recognized by many researchers [7], [19], [26]. For example, a probabilistic $\mathrm{M^3F}$ model is proposed in [7], [26], in which the number of latent factors can be inferred from data. However, these methods are usually very expensive because the probabilistic model requires a large amount of computation, which is avoided in our method.

Maximum margin matrix factorization ($\mathrm{M^3F}$) is proposed to deal with discrete ordinal rating data in collaborative filtering [6]. It can be formulated as an SDP problem, thus it can be solved using standard SDP solvers [6]. However, the SDP solver scales very poorly. To improve efficiency, a fast $\mathrm{M^3F}$ method is proposed to solve the matrix factorization problem by investigating the gradient-based optimization method [18]. Nevertheless, these methods are still not capable of tackling large scale $\mathrm{M^3F}$ problems, and in the mean time, their optimization could be stuck in an unsatisfactory local minimum.

A number of $\mathrm{M^3F}$ extensions have been introduced in the last decade [34], [35], [36]. The authors in [35] presented a method using $\mathrm{M^3F}$ to optimize ranking rather than ratings. Other researcher further improved the performance of $\mathrm{M^3F}$ by casting it within ensemble approaches [37], [38].

There are a few works on matrix factorization in the robust loss setting [39], [40]. These works apply a similar approach, i.e., augmented Lagrangian multipliers, to decompose the ℓ_1 regularized matrix factorization problem, which often results in an augmented Lagrangian function with independent variables. An alternating direction method is applied to update these variables.

Recent papers studied the optimality of low-rank problems [41], [42]. Specifically, the algorithm proposed in [41] deals with the convex nuclear norm minimization problem, which decomposes the original problem into two alternating sub-problems, i.e., a fixed-rank non-convex optimization on the quotient manifold and rank-one increment update. For the fixed-rank optimization, [1] presents a second-order trust region algorithm which has been previously proved to yield quadratic rate of convergence to the local minimum. The algorithm escapes the local minimum by the rank-one increment update until it reaches the global minimum, which is implemented by monitoring the convergence behavior via the convergence criterion and duality gap.

However, we would like to emphasize that the global optimality analyzed in this paper is the one with respect to the convex nuclear norm minimization problem (i.e., Problem (1) in [41]), rather than the fixed-rank optimization problem. As claimed in Proposition 2.2 in [41], the local minimum of the fixed-rank problem is also the global optimum of the nuclear norm minimization problem under a certain condition. Then the authors make use of this result to design the algorithm with the convergence guarantee to the global solution to the original nuclear norm minimization problem (Problem (1) in [41]). In contrast, our theoretical analysis reveals the convergence to the stationary point where the Riemannian gradient of the objective function vanishes.

In [42], the authors propose a framework that includes several low-rank matrix problems, i.e., matrix sensing, matrix completion and robust PCA, as special cases. In the unified framework, the theoretical analysis shows that all local minima are also global optima and there is no high-order saddle point. Specifically, their analysis replies on the objective function of the framework, which is quadratic over the observed matrix (denoted by M in [42]). This property provides the hessian of the objective function, playing a crucial role in the theoretical analysis. However, our paper considers the squared hinge loss, which does not have second-order continuous derivatives. Their results may not be applied in our situation.

2.2 Robust Principal Component Analysis (RPCA)

Robust Principal Component Analysis is proposed to recover the low-rank matrix contaminated by sparse large outliers by exploiting the robust ℓ_1 loss, which is known to introduce sparsity [2], [4], [43], [44]. RPCA has recently been intensively studied and successfully applied in a number of computer vision tasks, including background modeling from video [45], [46], shadow and specularity removal from face images [45], robust video denoising [47], small target detection [48], batch image alignment [12], robust photometric stereo [49] and HDR [11]. The low-rank structure in RPCA is achieved by minimizing the nuclear norm as a convex relaxation of the low rank constraint [15], [50], which can be computed by summing all the singular values. In [51], the authors propose to solve the RPCA problem using the alternating direction method of multipliers (ADMM) by leveraging the techniques of augmented Lagrangian multipliers. Two variations of the ADMM algorithm are proposed in [51], namely exact ALM (EALM) and inexact ALM (IALM). EALM has been proved to have a Q-linear convergence speed. While IALM has the same convergence speed, it requires fewer SVDs in the update [51].

A number of variations of nuclear norm-based methods exploit the underlying low-rank structure to deal with RPCA problems. The authors in [10] propose to minimize the partial sum of singular values (PSSV), which minimizes the smallest $l-p$ singular values of $\mathbf{X}$, where $l=\min(m,n)$ and p is the prior knowledge of the rank. The aim of PSSV is to preserve the information of the largest p singular values and make the remaining values as small as possible.

3 Preliminaries: Riemannian Geometry of Fixed-Rank Matrices

This section provides preliminary knowledge about Riemannian geometry and Riemannian optimization.

Providing that r is a known scalar, the manifold of fixed-rank-r matrices can be denoted as follows [22]:

$$\mathcal{M}_r = \{\mathbf{X} \in \mathbb{R}^{m\times n} : \mathrm{rank}(\mathbf{X}) = r\}. \tag{2}$$

Let $\mathrm{St}_r^m = \{\mathbf{U} \in \mathbb{R}^{m\times r} : \mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}\}$ denote the Stiefel manifold of $m \times r$ real and orthonormal matrices. The manifold $\mathcal{M}_r$ can then be rewritten as the following SVD-style representation:

$$\mathcal{M}_r = \{\mathbf{U}\mathrm{diag}(\boldsymbol{\sigma})\mathbf{V}^\mathsf{T} : \mathbf{U} \in \mathrm{St}_r^m, \mathbf{V} \in \mathrm{St}_r^n, ||\boldsymbol{\sigma}||_0 = r\}, \tag{3}$$

where $\boldsymbol{\sigma} \in \mathbb{R}^{\min(m,n)}$ denote the singular values, and $\mathrm{diag}(\boldsymbol{\sigma})$ denotes the diagonal matrix with $\boldsymbol{\sigma}$ on the diagonal.

The tangent space $T_{\mathbf{X}}\mathcal{M}_r$ of $\mathcal{M}_r$ at $\mathbf{X} = \mathbf{U}\mathrm{diag}(\boldsymbol{\sigma})\mathbf{V}^{\mathsf{T}} \in \mathbb{R}^{m\times n}$ is represented as follows:

$$\begin{aligned} T_{\mathbf{X}}\mathcal{M}_r = \{&\mathbf{U}\mathbf{M}\mathbf{V}^{\mathsf{T}} + \mathbf{U}_p\mathbf{V}^{\mathsf{T}} + \mathbf{U}\mathbf{V}_p^{\mathsf{T}} : \mathbf{M} \in \mathbb{R}^{r\times r}, \\ &\mathbf{U}_p \in \mathbb{R}^{m\times r}, \mathbf{U}_p^{\mathsf{T}}\mathbf{U} = \mathbf{0}, \mathbf{V}_p \in \mathbb{R}^{n\times r}, \mathbf{V}_p^{\mathsf{T}}\mathbf{V} = \mathbf{0}\}. \end{aligned} \tag{4}$$

We denote the tangent bundle by the disjoint union of all tangent spaces

$$\begin{aligned} T\mathcal{M}_r &= \bigcup_{\mathbf{X}\in\mathcal{M}_r} \{\mathbf{X}\} \times T_{\mathbf{X}}\mathcal{M}_r \\ &= \{(\mathbf{X}, \mathbf{H}) \in \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n} : \mathbf{X} \in \mathcal{M}_r, \mathbf{H} \in T_{\mathbf{X}}\mathcal{M}_r\}. \end{aligned} \tag{5}$$

If the Euclidean inner product on $\mathbb{R}^{m\times n}$ is restricted to the tangent bundle, in which the inner product is given by $\langle \mathbf{A}, \mathbf{B}\rangle = \mathrm{tr}(\mathbf{A}^{\mathsf{T}}\mathbf{B})$ with $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$, $\mathcal{M}_r$ is turned to a Riemannian manifold with Riemannian metric $g_{\mathbf{X}}(\zeta, \xi) = \langle \zeta, \xi\rangle$ on $\mathcal{M}_r$ where $\zeta, \xi \in T_{\mathbf{X}}\mathcal{M}_r$.

Based on the specified metric of Riemannian manifold on the tangent space, the Riemannian gradient, which is used to update matrices on the manifold, can be computed. Suppose that $f(\mathbf{X})$ is a smooth function of $\mathbf{X}$. To obtain $\mathbf{grad}\, f(\mathbf{X})$, it is necessary to first compute the gradient of $f(\mathbf{X})$ in Euclidean space. Let $\mathbf{G}$ denote the gradient of $f(\mathbf{X})$ in Euclidean space at $\mathbf{X} = \mathbf{U}\mathrm{diag}(\boldsymbol{\sigma})\mathbf{V}^{\mathsf{T}}$. The Riemannian gradient of $f(\mathbf{X})$ on $\mathcal{M}_r$, denoted by $\mathbf{grad}\, f(\mathbf{X})$, can be computed as the orthogonal projection of $\mathbf{G}$ onto $T_{\mathbf{X}}\mathcal{M}_r$, which is represented as follows:

$$\mathbf{grad}\, f(\mathbf{X}) = \mathrm{P}_{T_{\mathbf{X}}\mathcal{M}_r}(\mathbf{G}), \tag{6}$$

where

$$\mathrm{P}_{T_{\mathbf{X}}\mathcal{M}_r}(\mathbf{Z}) : \mathbf{Z} \mapsto P_U\mathbf{Z}P_V + P_U^{\perp}\mathbf{Z}P_V + P_U\mathbf{Z}P_V^{\perp}, \tag{7}$$

is the orthogonal projection of any $\mathbf{Z} \in \mathbb{R}^{m\times n}$ onto the tangent space at $\mathbf{X} = \mathbf{U}\mathrm{diag}(\boldsymbol{\sigma})\mathbf{V}^{\mathsf{T}}$, where $P_U = \mathbf{U}\mathbf{U}^{\mathsf{T}}$ and $P_U^{\perp} = \mathbf{I} - \mathbf{U}\mathbf{U}^{\mathsf{T}}$ for any $\mathbf{U} \in \mathrm{St}_r^m$.

In addition, there are two crucial operations in the optimization on the manifold, namely *Retraction* and *Vector Transport*. The purpose of retraction is to map a tangent vector in the tangent space back to the manifold after updating at each iteration, while vector transport maps a tangent vector from a tangent space to another tangent space. These two operations are both necessary when updating the matrix $\mathbf{X}$. According to [22], the retraction can be performed by the following closed form:

$$R_{\mathbf{X}}(\mathbf{H}) = \mathrm{P}_{\mathcal{M}_r}(\mathbf{X} + \mathbf{H}) = \sum_{i=1}^{r} \sigma_i \mathbf{p}_i \mathbf{q}_i^{\mathsf{T}}, \tag{8}$$

where $\sum_{i=1}^{r} \sigma_i \mathbf{p}_i \mathbf{q}_i^{\mathsf{T}}$ is the best rank-r approximation of $\mathbf{X} + \mathbf{H}$. The computation of the vector transport can be represented as follows:

$$\mathcal{T}_{\mathbf{X}\to\mathbf{Y}} : T_{\mathbf{X}}\mathcal{M}_r \to T_{\mathbf{Y}}\mathcal{M}_r, \boldsymbol{\eta} \mapsto \mathrm{P}_{T_{\mathbf{Y}}\mathcal{M}_r}(\boldsymbol{\eta}), \tag{9}$$

where $\mathrm{P}_{T_{\mathbf{Y}}\mathcal{M}_r}(\boldsymbol{\eta})$ is the orthogonal projection of $\boldsymbol{\eta}$ onto the tangent space $T_{\mathbf{Y}}\mathcal{M}_r$ at $\mathbf{X} = \mathbf{U}\mathrm{diag}(\boldsymbol{\sigma})\mathbf{V}^{\mathsf{T}}$, whose computation can be found in its definition (7). For more details of Riemannian geometry, see [22], [27].

4 Maximum Margin Matrix Factorization for Collaborative Filtering

This section investigates matrix factorization as it is applied in the recommender system setting. Our proposed method handles the maximum margin loss function efficiently by avoiding expensive SDP solvers used in M^{3}F [6], and additionally estimates the rank of the recovered matrix in a stage-wise manner (as shown in Algorithm 3), which enables general matrix factorization approaches to cope with ordinal rating data in recommender systems.

4.1 Problem Formulation

In recommender systems, users usually provide discrete ratings on items such as music and movies. Typically, ratings are represented as $+1/-1$ (like and dislike), or 1 to 5 stars, or similar. Maximum margin matrix factorization (M^{3}F) is proposed to improve performance in such a scenario [6]. Let Ω be a subset containing the indices of the observed entries. Given an observation matrix $\mathbf{Y} \in \mathbb{R}^{m\times n}$, assume that for any $ij \in \Omega$, we have $\mathbf{Y}_{ij} \in \{1, 2, \ldots, L\}$, where L is the maximum rating value. By using $L+1$ thresholds, i.e., $\theta_0 \le \theta_2 \le \cdots \le \theta_L$, the real valued $\mathbf{X}_{ij}$ can be related to the discrete values in the following hard margin approach:

$$\theta_{\mathbf{Y}_{ij}-1} + 1 \le \mathbf{X}_{ij} \le \theta_{\mathbf{Y}_{ij}} - 1.$$

We set $\theta_0 = -\infty$ and $\theta_L = +\infty$ by default, while the remaining $L-1$ thresholds can be determined based on the data. We denote these $L-1$ thresholds as $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_{L-1}]^{\mathsf{T}} \in \mathbb{R}^{L-1}$. In a soft-margin setting, we introduce a slack variable for each entry of $\mathbf{X}$ in the following manner:

$$\boldsymbol{\xi}_{ij} = \sum_{z=1}^{L-1} (h(T_{ij}^z \cdot (\theta_z - \mathbf{X}_{ij})))^2, \forall ij \in \Omega, \tag{10}$$

where $T_{ij}^z = \begin{cases} +1 & \text{for } z \ge Y_{ij} \\ -1 & \text{for } z < Y_{ij} \end{cases}$ and $h(z) = \max(0, 1-z)$.

As mentioned in the related works, M^{3}F is optimized by general SDP solvers [6]. It can be very computationally unaffordable if the scale of rating data is large. Among general MF optimization algorithms, most existing methods assume the smoothness of the loss functions [20], [21], [22], [23], [24], [25], which is not capable of dealing with the non-smooth hinge loss.

To handle the discrete ordinal rating data in this paper, we apply the squared hinge loss

$$\ell(\mathbf{X}, \boldsymbol{\theta}) = \frac{1}{2} \sum_{ij\in\Omega} \boldsymbol{\xi}_{ij}. \tag{11}$$

In addition, we add a regularization term $\Upsilon(\mathbf{X}) = \frac{1}{2}(||\mathbf{X}||_F^2 + \nu||\mathbf{X}^{\dagger}||_F^2)$ to prevent over-fitting, where $\mathbf{X}^{\dagger}$ is the pseudo-inverse, $\nu > 0$ is a trade-off parameter, and $||\mathbf{X}^{\dagger}||_F^2$ is a barrier to avoid the decrease of the rank of $\mathbf{X}$ [22]. We set ν as a small value by default (e.g., $\nu = 0.0001$) in experiments. Thus, our objective function can be formulated as below:

$$\min_{\mathbf{X},\boldsymbol{\theta}} \; f(\mathbf{X}, \boldsymbol{\theta}), \;\; \text{s.t.} \;\; \mathrm{rank}(\mathbf{X}) = r, \tag{12}$$

where $f(\mathbf{X}, \boldsymbol{\theta}) = \lambda\Upsilon(\mathbf{X}) + \ell(\mathbf{X}, \boldsymbol{\theta})$ and $0 < \lambda < 1$ denotes the regularization parameter. Note that the regularizer $\Upsilon(\mathbf{X})$ is

important for preventing from the issue of over-fitting in the context of $\mathrm{M^3F}$. An experiment demonstrating the effectiveness of the regularizer is given in Section 6.1.2.

Algorithm 1. Compute Riemannian Gradient $\mathbf{grad}f(\mathbf{X})$

1: Let $\Xi = \lambda \mathrm{diag}(\boldsymbol{\sigma} - \nu/\boldsymbol{\sigma}^3)$, and compute $\widehat{\mathbf{G}}$ via (14).
2: Compute $\mathbf{G}_u = \widehat{\mathbf{G}}^\mathsf{T}\mathbf{U}$, and $\mathbf{G}_v = \widehat{\mathbf{G}}\mathbf{V}$.
3: Compute $\widehat{\mathbf{M}} = \mathbf{U}^\mathsf{T}\mathbf{G}_v$.
4: Compute $\mathbf{U}_p = \mathbf{G}_v - \mathbf{U}\widehat{\mathbf{M}}$, and $\mathbf{V}_p = \mathbf{G}_u - \mathbf{V}\widehat{\mathbf{M}}^\mathsf{T}$.
5: Update $\mathbf{M} = \widehat{\mathbf{M}} + \Xi$.
6: Output $\mathbf{U}_p$, $\mathbf{V}_p$, and $\mathbf{M}$, and $\mathbf{grad}f(\mathbf{X},\theta) = \mathbf{U}\mathbf{M}\mathbf{V}^\mathsf{T} + \mathbf{U}_p\mathbf{V}^\mathsf{T} + \mathbf{U}\mathbf{V}_p^\mathsf{T}$.

After obtaining the final real valued matrix $\mathbf{X}$, the projection from real values to discrete values is easily achieved by the following approach:

$$\mathbf{Y}^*_{ij} = \max\{z+1 | \mathbf{X}_{ij} \geq \theta_z, z = 0, \ldots, L-1\}. \tag{13}$$

However, Problem (12) is non-convex due to the rank constraint $\mathrm{rank}(\mathbf{X}) = r$, and optimization is thus difficult. We note that $\mathbf{X}$ is restricted on fixed-rank matrices and accordingly propose to solve Problem (12) based on the Riemannian geometries on fixed-rank matrices.

Following [22], a tangent vector $\boldsymbol{\eta} \in \mathcal{T}_\mathbf{X}\mathcal{M}_r$ is represented as $\boldsymbol{\eta} = \mathbf{U}\mathbf{M}\mathbf{V}^\mathsf{T} + \mathbf{U}_p\mathbf{V}^\mathsf{T} + \mathbf{U}\mathbf{V}_p^\mathsf{T}$ (see Eq. (4) for details). Let $\widehat{\mathbf{G}}$ and $\mathbf{grad}f(\mathbf{X},\theta)$ denote the gradient of $\ell(\mathbf{X},\theta)$ in Euclidean space and the Riemannian gradient of $f(\mathbf{X},\theta)$ w.r.t. $\mathbf{X}$, respectively. Based on Eqs. (10) and (11), we obtain the following:

$$\widehat{\mathbf{G}}_{ij} = \frac{\partial \ell(\mathbf{X},\theta)}{\partial \mathbf{X}_{ij}} = \sum_{l=1}^{L-1} T^l_{ij} \cdot h(T^l_{ij} \cdot (\theta_l - \mathbf{X}_{ij})). \tag{14}$$

Let Ξ denote the gradient of $\Upsilon(\mathbf{X})$ w.r.t. $\mathbf{X}$, then it can be computed by $\Xi = \lambda \mathrm{diag}(\boldsymbol{\sigma} - \nu/\boldsymbol{\sigma}^3)$. According to the definition of the Riemannian gradient, at $\mathbf{X} = \mathbf{U}\mathrm{diag}(\boldsymbol{\sigma})\mathbf{V}^\mathsf{T}$, we have

$$\mathbf{grad}f(\mathbf{X},\theta) = \mathrm{P}_{T_\mathbf{X}\mathcal{M}_r}(\widehat{\mathbf{G}}),$$

where $P_{T_\mathbf{X}\mathcal{M}_r}(\widehat{\mathbf{G}})$ is the projection of $\widehat{\mathbf{G}}$ onto the tangent space $\mathcal{T}_\mathbf{X}\mathcal{M}_r$, defined in (7). The detailed computation of $\mathbf{grad}f(\mathbf{X},\theta)$ is summarized in Algorithm 1.

Lemma 1 ([52]). *Suppose* $\mathbf{U}_p$, $\mathbf{V}_p$, *and* $\mathbf{M}$ *are obtained from Algorithm 1, then* $\mathbf{grad}f(\mathbf{X},\theta) = \mathbf{U}\mathbf{M}\mathbf{V}^\mathsf{T} + \mathbf{U}_p\mathbf{V}^\mathsf{T} + \mathbf{U}\mathbf{V}_p^\mathsf{T}$.

Proof. Proof is in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2018.2882197. □

For the optimization of the thresholds θ, it is easy to compute the gradient of $f(\mathbf{X},\theta)$ w.r.t. θ, which we denote as $\mathbf{g} \in \mathbb{R}^{L-1}$. For each element of $\mathbf{g}$, say g_l for $1 \leq l \leq L-1$, we have the following equation:

$$g_l = \frac{\partial f(\mathbf{X},\theta)}{\partial \theta_z} = \sum_{ij \in \Omega} -T^z_{ij} \cdot h(T^z_{ij} \cdot (\theta_z - \mathbf{X}_{ij})). \tag{15}$$

4.2 Block-Wise Nonlinear Riemannian Conjugate Gradient Descent for $\mathrm{M^3F}$

Due to the presence of the fixed-rank constraint, Problem (12) is NP-hard. Manifold optimization usually shows advantages in terms both of efficiency and convergence [22]. To enable MF to handle $\mathrm{M^3F}$ problems, we therefore design a new algorithm which adopts manifold optimization techniques employed in [22] and meanwhile alternatingly updates the recovered matrix $\mathbf{X}$ and thresholds θ.

There are two variables to be optimized in Problem (12): the low rank rating matrix $\mathbf{X} \in \mathcal{M}_r$ and the thresholds $\theta \in \mathbb{R}^{L-1}$. We propose a block-wise algorithm to update these two variables alternately, which we call Block-wise Nonlinear Riemnnian Conjugate Gradient, summarized in Algorithm 2. At each iteration of BNRCG, we first update $\mathbf{X}$ with fixed θ by minimizing $f(\mathbf{X},\theta)$ via Nonlinear Riemannian Conjugate Gradient (Steps 3, 4, 8 and 9) and then update θ with fixed $\mathbf{X}$ by minimizing $f(\mathbf{X},\theta)$ (Steps 10 and 11). Steps 3-9 are described below in details.

Due to the presence of the rank constraint, $\mathbf{X}$ is required to stay on the manifold of the fixed-rank matrices. Update $\mathbf{X}$ on manifold is thus quite different from the conventional gradient methods in Euclidean space. It is necessary to always update $\mathbf{X}$ following a search direction on the manifold as long as the rank of $\mathbf{X}$ does not change. Suppose that $\mathbf{X}_t$ is the variable in the tth iteration in BNRCG in Euclidean space, and $\boldsymbol{\eta}_t$ denotes the search direction of the current iteration. Then we have the following equation:

$$\boldsymbol{\eta}_t = -\mathbf{grad}f(\mathbf{X}_t) + \beta_t \boldsymbol{\eta}_{t-1}, \tag{16}$$

where β_t can be calculated by a Polak-Ribière (PR+) rule [22]

$$\beta_t = \frac{\mathbf{grad}f(\mathbf{X}_t)^\mathsf{T}(\mathbf{grad}f(\mathbf{X}_t) - \mathbf{grad}f(\mathbf{X}_{t-1}))}{\langle \mathbf{grad}f(\mathbf{X}_{t-1}), \mathbf{grad}f(\mathbf{X}_{t-1})\rangle}. \tag{17}$$

Nevertheless, we cannot simply perform the computation represented in Eqs. (16) and (17), since $\mathbf{grad}f(\mathbf{X}_t) \in \mathcal{T}_{\mathbf{X}_t}\mathcal{M}_r$ and $\mathbf{grad}f(\mathbf{X}_{t-1}), \boldsymbol{\eta}_{t-1} \in \mathcal{T}_{\mathbf{X}_{t-1}}\mathcal{M}_r$, which are in different tangent spaces. To make (17) valid, we apply a previously mentioned geometric operation, vector transport, which is defined in Eq. (9).

Apart from the validity of (17), one has to always make $\boldsymbol{\eta}_t$ gradient-related, i.e., $\langle \boldsymbol{\eta}_t, \mathbf{grad}f(\mathbf{X}_{t-1}), \boldsymbol{\eta}_{t-1}\rangle < 0$ (See Definition 4.2.1 of [27]). This can be ensured by Steps 5 to 7, which is crucial for the convergence analysis later on.

The step sizes in Steps 8 and 11 in Algorithm 2 are computed by the line search method. In Step 8, if the descent direction is given by $\boldsymbol{\eta}_t \in T_{\mathbf{X}_t}\mathcal{M}_r$, the step size α_t is determined by the following Armijo rule:

$$f(R_{\mathbf{X}_t}(\alpha_t\boldsymbol{\eta}_t), \boldsymbol{\theta}_t) \leq f(\mathbf{X}_t, \boldsymbol{\theta}_t) + c_1\alpha_t\langle \mathbf{grad}f(\mathbf{X}_t, \boldsymbol{\theta}_t), \boldsymbol{\eta}_t\rangle, \tag{18}$$

where $0 < c_1 < 1$. When updating $\boldsymbol{\theta}_{t+1}$ by the standard gradient descent method, the step size γ_t can be similarly computed by the line search on the following Armijo rule:

$$f(\mathbf{X}_{t+1}, \boldsymbol{\theta}_{t+1}) \leq f(\mathbf{X}_{t+1}, \boldsymbol{\theta}_t) + d_1\gamma_t\langle \mathbf{g}_t, -\mathbf{g}_t\rangle, \tag{19}$$

where d_1 is the parameter and $0 < d_1 < 1$.

By applying the following Armijo line search strategy, one is able to find a step size that fulfills the Armijo rule (e.g., (18) and (19))

For a smooth function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$,
find the smallest integer $i = 0, 1, 2, \ldots$
such that $f(\mathbf{x} + \delta^i \mathbf{d}) \leq f(\mathbf{x}) + c_0 \delta^i \langle \nabla f(\mathbf{x}), \mathbf{d} \rangle$,
where d is a descent direction, $c_0, \delta \in (0, 1)$.

In [53], finite reduction of the step size to make Armijo rule hold is generally analyzed. For more details of the finite termination property of Armijo line search for our specific problem, please refer to discussion in Appendix A, available in the online supplemental material.

Algorithm 2. BNRCG for Fixed-Rank $\mathrm{M}^3\mathrm{F}$

1: Given $\mathrm{rank}(\mathbf{X}) = r$. Initialize $\mathbf{X}_1$, $\boldsymbol{\eta}_0$, and $\boldsymbol{\theta}_1$. Let $t = 1$.
2: **while** stopping conditions not achieve **do**
3: Compute $\mathbf{H}_t = -\mathbf{grad} f(\mathbf{X}_t, \theta_t)$ according to (6).
4: Compute the conjugate direction with PR+ rule:
$\boldsymbol{\eta}_t = \mathbf{H}_t + \beta_t \mathcal{T}_{\mathbf{X}_{t-1} \to \mathbf{X}_t}(\boldsymbol{\eta}_{t-1}) \in T\mathcal{M}_r$.
5: **if** $\langle \mathbf{grad} f(\mathbf{X}_t, \theta_t), \boldsymbol{\eta}_t \rangle \geq 0$ **then**
6: Set $\boldsymbol{\eta}_t = \mathbf{H}_t$.
7: **end if**
8: Choose a step size α_t by Armijo line search (18).
9: Set $\mathbf{X}_{t+1} = R_{\mathbf{X}_t}(\alpha_t \boldsymbol{\eta}_t)$.
10: Compute $\mathbf{g}_t$ according to (15).
11: Choose a step size γ_t by Armijo line search (19) and set $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \gamma_t \mathbf{g}_t$.
12: Let $t = t + 1$.
13: **end while**

Next, we show that Algorithm 2 is guaranteed to converge to a stationary point of $f(\mathbf{X}, \boldsymbol{\theta})$ by the following Theorem.

Theorem 1. *The BNRCG algorithm is guaranteed to converge to a stationary point* $(\mathbf{X}^*, \boldsymbol{\theta}^*)$ *of* $f(\mathbf{X}, \boldsymbol{\theta})$ *where* $\mathbf{grad} f(\mathbf{X}^*, \boldsymbol{\theta}^*) = \mathbf{0}$ *and* $\nabla_{\boldsymbol{\theta}} f(\mathbf{X}^*, \boldsymbol{\theta}^*) = \mathbf{0}$.

Proof. Proof is in Appendix A, available in the online supplemental material. □

4.3 Automatic Latent Factor Detection by Active Subspace Search

Although all the fixed-rank matrices lie on the same manifold, there is still a challenge: how to estimate the rank of the recovered matrix? In this section, we present an approach to estimate the rank in a stage-wise manner.

We have introduced BNRCG to alternately update $\mathbf{X}$ and $\boldsymbol{\theta}$ with the rank r of $\mathbf{X}$ fixed. However, r is often unknown in practice. In this section, we present an active subspace search method to tackle this issue by estimating the rank of $\mathbf{X}$ automatically. Our approach is built upon BNRCG and treats BNRCG as a subproblem. We name this method ARSS-M^3F and summarize it in Algorithm 3.

As shown in Algorithm 3, after initializing $\mathbf{X}^0 = \mathbf{0}$ where $\xi^0 = \mathbf{b}$, ARSS-M^3F performs the following two main stages at each iteration:

- It identifies the most active subspace by the worst-case analysis (Steps 3-5).
- It finds the solution of the fixed rank $\mathrm{M}^3\mathrm{F}$ problem by BNRCG (Steps 6-9).

In the first main stage, we compute the gradient $f(\mathbf{X}, \theta)$ w.r.t. $\mathbf{X}$, denoted by $\mathbf{G}$, and find the most active subspace by conducting a truncated SVD on $\mathbf{G}$ with the dimensionality of ρ. In the second main stage, we initialize $\mathbf{X}^k = R_{\mathbf{X}^{k-1}}(-t_{\min} \bar{\mathbf{X}}_\rho)$ where the step size $t_{\min}$ is determined by the line search method on the following condition:

$$f(R_{\mathbf{X}^{k-1}}(-t_{\min} \mathbf{G}^{k-1})) \leq f(\mathbf{X}^{k-1}) - \frac{t_{\min}}{2} \langle \mathbf{G}^{k-1}, \mathbf{G}^{k-1} \rangle. \tag{20}$$

Then we use the initialized $\mathbf{X}^k$ as the input of BNRCG (Algorithm 2) where $\mathbf{X}^k$ and θ^k are updated alternately. We increase the rank by $r = r + \rho$ in Step 9, and then set r as the estimated rank of Algorithm 2. Considering the Inequality (20), the objective value $f(\mathbf{X}^k)$ monotonically decreases w.r.t. the iteration index k. As a result, we have the following stopping condition for Algorithm 3:

$$(f(\mathbf{X}^{k-1}) - f(\mathbf{X}^k))/(\rho f(\mathbf{X}^{k-1})) \leq \epsilon, \tag{21}$$

where ϵ is a stopping tolerance. As the rank of $\mathbf{X}$ increases gradually stage by stage, we can ultimately estimate the rank of the underlying low rank matrix.

Algorithm 3. ARSS-M^3F

1: Initialize $\mathbf{X}^0 = \mathbf{0}$, $r = 0$, $\xi^0 = \mathbf{b}$ and θ. Let $k = 1$.
2: **while** stopping condition not achieved **do**
3: Find active subspaces as follows:
4: (a): Compute $\mathbf{G} = \frac{\partial f(\mathbf{X}^k, \theta)}{\partial \mathbf{X}^k}$;
5: (b): Do thin SVD on $\mathbf{G}$: $[\mathbf{P}_\rho, \Sigma_\rho, \mathbf{Q}_\rho] = \mathrm{SVD}(\mathbf{G}, \rho)$.
6: Let $\bar{\mathbf{X}}_\rho = \mathbf{P}_\rho \Sigma_\rho \mathbf{Q}_\rho^{\mathsf{T}}$ and perform master problem optimization:
7: (a): Find an appropriate step size $t_{\min}$ by (20).
8: (b): initialize $\mathbf{X}^k = R_{\mathbf{X}^{k-1}}(-t_{\min} \bar{\mathbf{X}}_\rho)$ (Warm Start).
9: (c): Let $r = r + \rho$ and update $\mathbf{X}^k$ and θ^k by BNRCG (Algorithm 2).
10: Set $k = k + 1$.
11: **end while**

Now we discuss a little bit on the computational expense on SVDs. First, our algorithm does not require full SVDs. Take a look at BNRCG in Algorithm 2, which restricts the solution at each of the tth iteration X_t on a manifold $\mathcal{M}_r$. Recall in Eq. (3), all matrices on $\mathcal{M}_r$ have r non-zero singular values. As long as the solution is restricted on the manifold $\mathcal{M}_r$, we can only consider the largest r singular values.

In the view of ARSS-M3F in Algorithm 3, the estimated rank r is initialized by $r = 0$ and updated by $r = r + \rho$ (Line 9). As can be seen, the estimated rank r is usually very small, so the SVDs in BNRCG is partial SVDs, rather than full SVDs. In the experiment, specifically in Table 2, we find that the average rank estimated by ARSS-M3F on real-world data is usually very small, e.g., 8, 12.

Second, part form the unnecessary full SVDs, our method maintains the active basis by first performing partial SVDs in the beginning of the algorithm, where the rank r is usually set to a very small value, e.g., 1 or 2. In the subsequent iterations, therefore, the algorithm takes advantage of the maintained active basis to avoid repeated partial SVDs.

5 Robust Matrix Factorization for Outlier Detection

In this section, we investigate robust matrix factorization where sparse outliers occur. Many literatures cope with

sparse outliers via RPCA, while there are few MF approaches proposed based on manifold optimization. Similarly, armed with manifold optimization techniques, we propose a stage-wise algorithm to deal with outlier detection, which is also shown to inherit the efficiency to handle large-scale data.

Outlier removal is a requirement for matrix recovery algorithms in many real applications where sparse large errors occur. ℓ_1 loss is known to be insensitive to outliers, and achieves great success in outlier removal. However, the ℓ_1 loss is non-smooth, and most existing matrix factorization methods focus on smooth functions rather than non-smooth loss. For example, Algorithm 2 requires a smooth loss function $f(\mathbf{X}, \boldsymbol{\theta})$. This section thus enables matrix factorization algorithms to deal with non-smooth loss based on ALM. We take ℓ_1 loss as an example for convenience.

Algorithm 4. Matrix Factorization for Non-Smooth ℓ_1-Norm Loss by ALM

1: Initialize $\mathbf{E}^0 = \mathbf{0}, \mathbf{Z}^0 = \mathbf{0}, \mathbf{X}^0 = \mathbf{0}, \mu^0 > 0$ and $\rho > 1$. Set $k = 0$.
2: **while** not converge **do**
3: **while** not converge **do**
4: $\mathbf{X}^{k+1} = \mathrm{LRGeomCG}(\mathbf{X}^k)$ (Algorithm 5).
5: $\mathbf{E}^{k+1} = \mathcal{S}_{\frac{1}{\mu^k}}(\mathcal{P}_\Omega(\mathbf{Y} - \mathbf{X}^{k+1} - \frac{\mathbf{Z}^k}{\mu^k}))$ (Eq. (27)).
6: **end while**
7: $\mathbf{Z}^{k+1} = \mathbf{Z}^k + \mu^k(\mathcal{P}_\Omega(\mathbf{E}^{k+1} + \mathbf{X}^{k+1} - \mathbf{Y}))$.
8: $\mu^{k+1} = \rho\mu^k$.
9: Set $k = k + 1$.
10: **end while**

First, we specify the loss function of a matrix factorization problem as follows:

$$\min_{\mathbf{X}} ||\mathcal{P}_\Omega(\mathbf{Y}) - \mathcal{P}_\Omega(\mathbf{X})||_1, \text{ s.t. } \mathrm{rank}(\mathbf{X}) = r, \tag{22}$$

where $\mathbf{X} \in \mathbb{R}^{m\times n}$ is the low-rank matrix to be recovered, and $\mathbf{Y} \in \mathbb{R}^{m\times n}$ is the observation matrix. $\mathcal{P}_\Omega$ denotes the orthogonal projection onto the linear space of matrices support on Ω: $\mathcal{P}_\Omega(\mathbf{X}) = \mathbf{X}_{ij}$ if $(ij) \in \Omega$; $\mathcal{P}_\Omega(\mathbf{X}) = 0$ otherwise. By introducing an auxiliary variable $\mathcal{P}_\Omega(\mathbf{E} + \mathbf{X}) = \mathcal{P}_\Omega(\mathbf{Y})$, Problem (22) becomes:

$$\min_{\mathbf{X},\mathbf{E}} \quad ||\mathbf{E}||_1, \text{ s.t. } \mathcal{P}_\Omega(\mathbf{E} + \mathbf{X}) = \mathcal{P}_\Omega(\mathbf{Y}), \ \mathrm{rank}(\mathbf{X}) = r. \tag{23}$$

To deal with the equality constraint, we construct the augmented Lagrangian function as follow:

$$\begin{aligned}\mathcal{L}(\mathbf{X}, \mathbf{E}, \mathbf{Z}, \mu) = \ & ||\mathbf{E}||_1 + \langle \mathbf{Z}, (\mathcal{P}_\Omega(\mathbf{E} + \mathbf{X} - \mathbf{Y}))\rangle \\ & + \frac{\mu}{2}||\mathcal{P}_\Omega(\mathbf{E} + \mathbf{X} - \mathbf{Y})||_F^2,\end{aligned} \tag{24}$$

where $\mathbf{X} \in \mathcal{M}_r$, $\mathbf{Z} \in \mathbb{R}^{m\times n}$ are the Lagrangian multipliers, and $\mu > 0$ is a penalty parameter.

The optimization can then be performed by alternately updating the four variables $\mathbf{X}$, $\mathbf{E}$, $\mathbf{Z}$ and μ. The outline of exact ALM is summarized in Algorithm 4, but we apply the inexact ALM framework for the sake of efficiency in our experiments. Inexact ALM, widely used in many works [51], performs the innder loop of Algorithm 4 starting from Line 3 only once in each outer loop. In the following two sections, we present the updating details of $\mathbf{X}$ and $\mathbf{E}$, respectively.

5.1 Optimization of X

When updating $\mathbf{X}$, we fix all the remaining variables in the augmented Lagrangian function (24). Let k denotes the index of iteration. We have the following relation:

$$\begin{aligned}\mathbf{X}^{k+1} &= \arg\min_{\mathbf{X}\in\mathcal{M}_r} \langle \mathbf{Z}^k, \mathcal{P}_\Omega(\mathbf{E}^k + \mathbf{X} - \mathbf{Y})\rangle \\ &\quad + \frac{\mu^k}{2}||\mathcal{P}_\Omega(\mathbf{E}^k + \mathbf{X} - \mathbf{Y})||_F^2 \\ &= \arg\min_{\mathbf{X}\in\mathcal{M}_r} \frac{\mu^k}{2}\Big(\frac{||\mathbf{Z}^k||_F^2}{(\mu^k)^2} + ||\mathcal{P}_\Omega(\mathbf{E}^k + \mathbf{X} - \mathbf{Y})||_F^2 \\ &\quad + \frac{2}{\mu^k}\langle \mathbf{Z}^k, \mathcal{P}_\Omega(\mathbf{E}^k + \mathbf{X} - \mathbf{Y})\rangle\Big) - \frac{\mu^k}{2}\frac{||\mathbf{Z}^k||_F^2}{(\mu^k)^2} \\ &= \arg\min_{\mathbf{X}\in\mathcal{M}_r} ||\mathcal{P}_\Omega(\mathbf{M}^k) - \mathcal{P}_\Omega(\mathbf{X})||_F^2,\end{aligned} \tag{25}$$

where $\mathbf{M}^k = \mathbf{Y} - \mathbf{E}^k + \frac{\mathbf{Z}^k}{\mu^k}$. Problem (25) can be minimized by LRGeomCG proposed in [22], which shares a number of geometric concepts and operations with BNRCG in Algorithm 2, e.g., retraction, vector transport and Armijo line search. We thus provide a brief summary of this algorithm in Algorithm 5. Moreover, Algorithm 5 is guaranteed to converge to a stationary point [22].

Algorithm 5. LRGeomCG($\mathbf{X}^{\text{initial}}$) [22]

1: Initialize $\mathbf{X}_1 = \mathbf{X}^{\text{initial}}$, $\boldsymbol{\eta}_0 = \mathbf{0}$, and $t = 1$.
2: **while** stopping conditions not achieved **do**
3: Compute $\mathbf{H}_t = -\mathbf{grad} f(\mathbf{X}_t)$ according to (6).
4: Compute the conjugate direction with PR+ rule:
$\boldsymbol{\eta}_t = \mathbf{H}_t + \beta_t \mathcal{T}_{\mathbf{X}_{t-1}\to\mathbf{X}_t}(\boldsymbol{\eta}_{t-1}) \in T\mathcal{M}_r$.
5: Choose a step size α_t and set $\mathbf{X}_{t+1} = R_{\mathbf{X}_t}(\alpha_t\boldsymbol{\eta}_t)$.
6: Let $t = t + 1$.
7: **end while**

5.2 Optimization of E

When updating $\mathbf{E}$, we fix all the remaining variables in the augmented Lagrangian function (24). Let k denotes the index of iteration, and then, similarly to the update of $\mathbf{X}$, we have the following relation:

$$\begin{aligned}\mathbf{E}^{k+1} &= \arg\min_{\mathbf{E}} ||\mathbf{E}||_1 + \langle \mathbf{Z}^k, \mathcal{P}_\Omega(\mathbf{E} + \mathbf{X}^{k+1} - \mathbf{Y})\rangle \\ &\quad + \frac{\mu^k}{2}||\mathcal{P}_\Omega(\mathbf{E} + \mathbf{X}^{k+1} - \mathbf{Y})||_F^2 \\ &= \arg\min_{\mathbf{E}} \frac{1}{\mu^k}||\mathbf{E}||_1 + \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{E} - \mathbf{N}^k)||_F^2,\end{aligned} \tag{26}$$

where $\mathbf{N} = \mathbf{Y} - \mathbf{X}^{k+1} - \frac{\mathbf{Z}^k}{\mu^k}$. The above problem is known as a soft-thresholding problem, and we are able to solve it easily by the following soft-thresholding operator:

$$\mathcal{S}_{\frac{1}{\mu}}(\mathbf{A}_{ij}) = \mathrm{sign}(\mathbf{A}_{ij})\max\Big(|\mathbf{A}_{ij}| - \frac{1}{\mu^k}, 0\Big), \tag{27}$$

where $\mathrm{sign}(\mathbf{A}_{ij})$ returns the sign of $\mathbf{A}_{ij}$. In our problem, to obtain $\mathbf{E}^{k+1}$, $\mathbf{A} = \mathcal{P}_\Omega(\mathbf{E}^k - (\mathbf{Y} - \mathbf{X}^{k+1} - \frac{\mathbf{Z}^k}{\mu^k}))$.

5.3 Convergence Analysis

Problem (23) is non-convex due to the presence of the rank constraint. We provide the theoretical analysis to show that Algorithm 4 is guaranteed to converge to a stationary point.

Lemma 2. *Assume that for any* $\mathbf{v} \in \frac{\partial \|\mathbf{E}^k\|_1}{\partial \mathbf{E}^k}$ *we have* $\|\mathbf{v}\|_1 \leq B$ *where* $k \geq 0$ *and* $B \geq 0$. *Given the non-decreasing sequence* $\{\mu^k\}$ *with* $\mu^0 > 0$, *and* $\mathbf{Z}^0 = \mathbf{0}$, *then the sequence* $\{\mathbf{Z}^k\}$ *in Algorithm 4 is bounded, i.e.,* $\|\mathbf{Z}^k\|_F \leq B$ *for every* k.

Proof. Proof is in Appendix A, available in the online supplemental material. □

Remark 1. We emphasize in proof of Lemma 2 that the assumption (i.e., $\|\mathbf{v}\|_F \leq B$ for any $\mathbf{v} \in \frac{\partial \|\mathbf{E}^{k+1}\|_1}{\partial \mathbf{E}^{k+1}}$) in Lemma 2 always holds in our case. Consider the computation of $\mathbf{E}^{k+1}$, i.e., $\mathbf{E}^{k+1} = \mathcal{S}_{\frac{1}{\mu^k}}(\mathcal{P}_\Omega(\mathbf{Y} - \mathbf{X}^{k+1} - \frac{\mathbf{Z}^k}{\mu^k}))$. In this soft-thresholding operator, $\mathbf{Y}$ is the given observation matrix and thus all elements in $\mathbf{Y}$ are bounded. Then $\mathbf{E}^{k+1}$ is bounded if $\mathbf{X}^{k+1}$ and $\frac{\mathbf{Z}^k}{\mu^k}$ are bounded. $\mathbf{X}^{k+1}$ is derived by solving Problem (25), whose convergence is guaranteed in [22]. Thus, all elements in $\mathbf{X}^{k+1}$ are bounded as long as $\mathbf{E}^k$ and $\frac{\mathbf{Z}^k}{\mu^k}$ are bounded. Given that we initialize $\mathbf{E}^0 = \mathbf{0}$ and $\mathbf{Z}^0 = \mathbf{0}$ and the non-decreasing sequence $\{\mu^k\}$ with $\mu^0 > 0$, for any $k \geq 0$, all elements in $\mathbf{X}^k$ is bounded. This shows that $\|\mathbf{v}\|_F \leq B$ always hold in our case.

Theorem 2. *Suppose that the sequences* $\{\mathbf{X}^k\}$, $\{\mathbf{E}^k\}$, $\{\mathbf{Z}^k\}$ *and* $\{\mu^k\}$ *are generated by Algorithm 4. Then any accumulation point* $(\mathbf{X}^*, \mathbf{E}^*)$ *of Algorithm 4 is a stationary point, where the gradient of Problem (24) w.r.t.* $\mathbf{X}$ *vanish.*

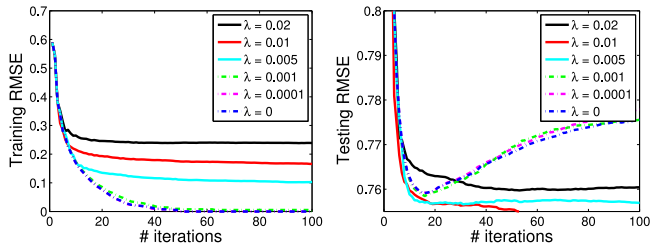
Proof. Proof is in Appendix A, available in the online supplemental material. □

6 Experiments

In this section, we investigate the performance of the proposed algorithms on three experiments, namely, collaborative filtering, HDR imaging and batch image alignment. We emphasize that HDR imaging is very similar to batch image alignment except that HDR imaging usually considers RGB images, while batch image alignment often consider grey scale images. In the collaborative filtering experiment, we apply the proposed ARSS-M^{3}F (Algorithm 3) to exploit the discrete ordinal data in recommender systems. In the HDR imaging and batch image alignment experiments, we apply the proposed robust matrix factorization algorithm (Algorithm 4) to detect sparse outliers.

6.1 Collaborative Filtering

We demonstrate the performance of the proposed methods, namely BNRCG-M^{3}F with fixed-rank problems and ARSS-M^{3}F, by comparing them with several related state-of-the-art methods: FM^{3}F [18], GROUSE [31], LMAFIT [23], ScGrassMC [24], LRGeomCG [22], RTRMC [28] and libpmf [54], on both synthetic and real-world CF tasks. Seven datasets are used in the experiments, comprising three synthetic datasets and four real-world datasets, Movielens 1M, Movielens 10M [55], Netflix [56] and Yahoo! Music Track 1 [57]. The two small scale synthetic datasets are respectively used to investigate the sensitivity of the regularization parameter and compare the convergence behavior of the M^{3}F algorithms. The large scale ordinal synthetic dataset, along with the four real-world recommender datasets, is used to demonstrate the performance and efficiency of our methods. The size of Netflix is 480,189 by 17,770 with 100,480,507 entries, and Yahoo! Music is 1,000,990 by 624,961 with 262,810,175 entries. They are both of large size and very sparse (98.82 and 99.96 percent zeros respectively). By comparing the results achieved on them, we aim to show the satisfactory efficiency of our proposed algorithms in real applications.



(a) RMSE on training data (b) RMSE on testing data

Fig. 1. RMSE of BNRCG-M^{3}F on binary rating data.

The root-mean-square error (RMSE) on both the training set and the testing set will be used as the comparison metric: $\text{RMSE} = \sqrt{\sum_{ij\in\Pi}(\mathbf{Y}^*_{ij} - \mathbf{Y}_{ij})^2/|\Pi|}$, where $\mathbf{Y}^*$ denotes the reconstructed ratings according to (13), and $|\Pi|$ denotes number of elements in the set Π. All the experiments are conducted in Matlab on a work station with an Intel(R) Xeon(R) CPU and 64 GB memory.

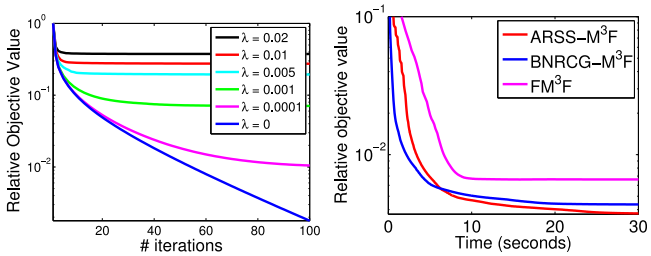
6.1.1 Synthetic Experiments

In the synthetic experiments where we know the ground-truth, we will demonstrate four points: 1) The sensitivity of the regularization of the proposed M^{3}F methods; 2) The efficiency of BNRCG-M^{3}F and ARSS-M^{3}F over other methods; 3) The importance of the squared hinge loss measure over other measures for rating data, e.g., the least square error; 4) The effectiveness of latent factor detection by ARSS-M^{3}F. To demonstrate the above points, we study three synthetic problems on two scales.

Motivated by [20], [24], we first generate a ground-truth low-rank matrix for each of the three synthetic problems by $\widehat{\mathbf{X}} = \widehat{\mathbf{U}}\text{diag}(\widehat{\boldsymbol{\delta}})\widehat{\mathbf{V}}^\mathsf{T}$, where $\boldsymbol{\delta}$ is an r-sparse vector with each non-zero entry sampled from *Gaussian* distribution $\mathcal{N}(0, 1000)$, $\widehat{\mathbf{U}} \in \text{St}^m_r$ and $\widehat{\mathbf{V}} \in \text{St}^n_r$. In both of the small-scale problems, $\widehat{\mathbf{X}}$ is of size $1{,}000 \times 1{,}000$ with $r = 20$, while the large-scale problem $\widehat{\mathbf{X}}$ is of size $20{,}000 \times 20{,}000$ with $r = 50$. After sampling the original entries, we respectively produce the binary ratings by $\widehat{\mathbf{Y}}_{ij} = \text{sgn}(\widehat{\mathbf{X}}_{ij})$, and the ordinal ratings $\{1, 2, 3, 4, 5\}$ by projecting the entries of $\widehat{\mathbf{X}}$ into five bins according to their values, which results in a rating matrix $\widehat{\mathbf{Y}}$. Once $\widehat{\mathbf{Y}}$ has been generated, we sample $l = r(m + n - r) \times \zeta_{os}$ entries from $\widehat{\mathbf{Y}}$ uniformly to form the observed ratings $\mathbf{Y}$, where ζ_{os} is the oversampling factor [51]. In the experiments we set $\zeta_{os} = 3.5$.

6.1.2 Sensitivity of Regularization Parameter

In this section, we perform experiments on the small-scale binary matrix to demonstrate the sensitivity of regularization. To illustrate the impact of regularization in the proposed methods, we test BNRCG-M^{3}F with various regularization parameters λ. Fig. 1 reports the training RMSE and testing RMSE. The convergence is shown in Fig. 2a. As can be seen, regularization is crucial to prevent overfitting.



(a) Binary rating data (b) Ordinal rating data

Fig. 2. Relative objective values of various methods.

6.1.3 Convergence of M³F on Ordinal Rating Data

In this section, we perform experiments on the small-scale ordinal matrix. We compare the proposed algorithms with the six baseline methods and collect the convergence behavior of the three M³F methods. The ground-truth rank is used as the estimated rank for all methods excluding ARSS-M³F.

The convergence behavior of our methods and FM³F is illustrated in Fig. 2b, which shows that our methods can converge faster to a better stationary point. Table 2 reports the resultant RMSE on the testing set and the computational time of each method on the small-scale synthetic ordinal rating dataset.

6.1.4 Efficiency of M³F on Ordinal Rating Data

In this section, we perform experiments on the large-scale ordinal matrix. We compare our methods with the five baseline algorithms. We use the ground-truth rank as the estimated rank for all methods except ARSS-M³F. The average estimated rank of ARSS-M³F is 42, which is close to the groundtruth rank of 50. According to the estimated rank in the two synthetic datasets, the latent factor detection of ARSS-M³ is effective. The RMSE on the testing set and computational time of each algorithm are listed in Table 2.

6.1.5 Real-World Experiments

To demonstrate the significance of the hinge loss to the rating data and effectiveness of latent factor estimation of our method in real-world data experiments, we study four large scale datasets, namely Movielens 1M, Movielens 10M, Netflix and Yahoo! Music Track 1. The baseline methods comprise FM³F, GROUSE, LMAFIT, ScGrassMC, LRGeomCG and RTRMC.

TABLE 1
Statistics of the Real-World Recommender Datasets

DataSets	# users	# items	# ratings
Movielens 1M	6,040	3,952	1,000,209
Movielens 10M	71,567	10,681	10,000,054
Netflix	480,189	17,770	100,480,507
Yahoo! Music Track 1	1,000,990	624,961	262,810,175

Table 1 lists the size statistics of the four datasets. The vast majority (99.71 percent) of ratings in Yahoo! Music Track 1 are multiples of ten. For convenience, we only consider these ratings. For Movielen 10 M and Yahoo! Music Track 1, we map the ratings to ordinal integer values prior to the experiment. For each dataset, we sample 80 percent of data into the training set and the rest into the testing set.

Table 2 reports the computational time of all comparison methods and testing RMSE on the four datasets. The resultant RMSE demonstrates that our method can recover the matrix with lower error than least square loss. Note that in all experiments on both synthetic and real-world data, no model selection cost is included for any comparison method. If model selections are considered, the time cost of the comparison methods will be much higher. Some results for GROUSE and M³F are not available due to their high computation cost. Table 2 shows that ARSS-M³F and BNRCG-M³F recover the rating matrix efficiently and outperform other comparison methods in terms of RMSE on the four real-world datasets. It is worth mentioning that although LRGeomCG is faster on Yahoo dataset, it achieves much worse RMSE than the M³F-based methods.

6.2 HDR Imaging

In this section, we perform HDR imaging experiments to investigate the performance of our proposed matrix factorization algorithm for non-smooth ℓ_1-norm loss. We compare our proposed method with four baselines, including PSSV [10], LMAFIT [40] and RegL1 [39]. Experiments are performed on six datasets, namely `Waterfall`, `Forest`, `Desk`, `USYD`, `Desk-full` and `USYD-full`. These datasets are summarized in Table 3. All the experiments are

TABLE 2
Experimental Results on Synthetic and Real-World Datasets

Methods	Small Synthetic*		Large Synthetic*		Movielens 1M†		Movielens 10M†		Netflix†		Yahoo Music†	
	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time
FM³F [18]	0.3811	11.99	0.3899	2186	0.9344	212.2051	0.9143	13001	1.0971	65662	-	-
GROUSE [31]	0.4718	27.84	0.512	11214	0.9225	39.4184	0.8653	3853	-	-	-	-
LMAFIT [23]	0.4701	6.08	0.4973	827	0.9373	19.9465	0.8424	832	0.9221	4374	24.222	24349
ScGrassMC [24]	0.4638	10.19	0.4714	2149	0.9372	21.3109	0.8427	917	0.9192	5787	24.7982	37705
LRGeomCG [22]	0.4679	6.01	0.4904	814	0.9321	10.2484	0.849	312	0.9015	3151	25.2279	8666
RTRMC [28]	0.4676	8.68	0.4715	884	0.9311	14.1038	0.846	673	0.9102	6465	24.5971	32592
libpmf [54]	0.4900	0.19	0.5115	5.42	**0.9082**	0.27	**0.8021**	63.40	1.6008	727.57	**23.2410**	3917
BNRCG-M³F	0.3698	5.34	0.3915	635	0.9285	13.4437	0.8437	714	0.9022	4118	23.8573	24631
ARSS-M³F	**0.3693**	5.33	**0.3684**	542	0.9222	9.5482	0.8411	650	**0.9001**	3583	23.7902	22065

Computational time is recorded in seconds.
**No model selection cost is included for any fix-rank method as the ground-truth rank is available.*
†The rank detected by ARSS-M³F is used as the estimated rank for other methods. Thus, no model selection is considered. The average rank estimated by ARSS-M³F on Movielens 1M, Movielens 10M, Netflix, and Yahoo Music is 8, 14, 16, and 28, respectively.

TABLE 3
Summary of LDR Datasets

Datasets	# LDR images	Resolution
Forest	4	1,024 × 683
Waterfall [9]	8	767 × 505
desk	6	1,536 × 1,152
desk-full	6	4,608 × 3,456
USYD	6	726 × 1,088
USYD-full	6	3,264 × 4,896

conducted in Matlab on a work station with an Intel(R) Xeon(R) CPU and 32 GB memory.

The aim of HDR imaging is to remove the outliers from a set of low dynamic range images, and generate a ghost-free HDR image. The outliers in the LDR images can be moving objects and areas of under-saturation or over-saturation. Many contemporary devices, e.g., digital cameras and smart phones, can generate images with very high resolution. For example, an iPhone can capture 12-megapixel images. To investigate the efficiency of HDR algorithms, we collect two 16-megapixel datasets by a mirrorless camera, i.e., Desk-full and USYD-full, whose size imposes a challenge on HDR algorithms. As a result, we produce two smaller dataset from Desk-full and USYD-full by resizing LDR images to $1/3$ of the original size, and name them Desk and USYD.

Of these baselines, PSSV is an RPCA-like algorithm, while LMAFIT and RegL1 are built upon matrix factorization. We introduce them briefly as follows:

- *PSSV* [10] Unlike the conventional RPCA which minimizes the nuclear norm of the matrix to be recovered for a low-rank property, PSSV minimizes the partial sum of singular values, whose formulation can be represented as below:

$$\min_{\mathbf{X},\mathbf{E}} \quad ||\mathbf{X}||_r + \lambda||\mathbf{E}||_1, \quad \text{s.t. } \mathbf{Y} = \mathbf{X} + \mathbf{E}, \tag{28}$$

  where $||\mathbf{X}||_r = \sum_{i=r}^{\min(m,n)} \sigma_i$ and σ_i represents the ith singular value of $\mathbf{X}$ by descending order. According to [10], this method outperforms traditional RPCA algorithms.
- *LMAFIT* [40] The formulation of LMAFIT is matrix factorization and is similar to our model

$$\min_{\mathbf{X},\mathbf{U},\mathbf{V}} ||\mathbf{Y} - \mathbf{X}||_1, \quad \text{s.t. } \mathbf{X} = \mathbf{U}\mathbf{V}^{\mathsf{T}}. \tag{29}$$

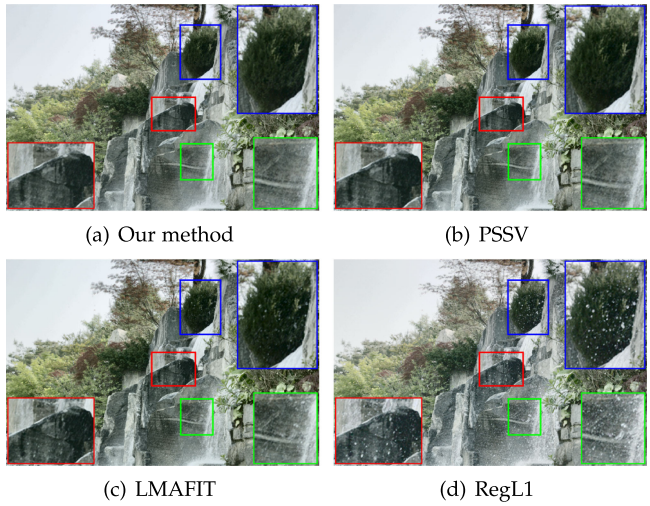
- *RegL1* [39] The matrix factorization formulation of RegL1 is as below:

$$\min_{\mathbf{U},\mathbf{V}} ||\mathbf{Y} - \mathbf{U}\mathbf{V}^{\mathsf{T}}||_1 + \lambda||\mathbf{V}||_*, \quad \text{s.t. } \mathbf{U}^{\top}\mathbf{U} = \mathbf{I}. \tag{30}$$

  This model requires the factor matrix $\mathbf{U}$ to be an orthogonal matrix and $\mathbf{V}$ to be low-rank.

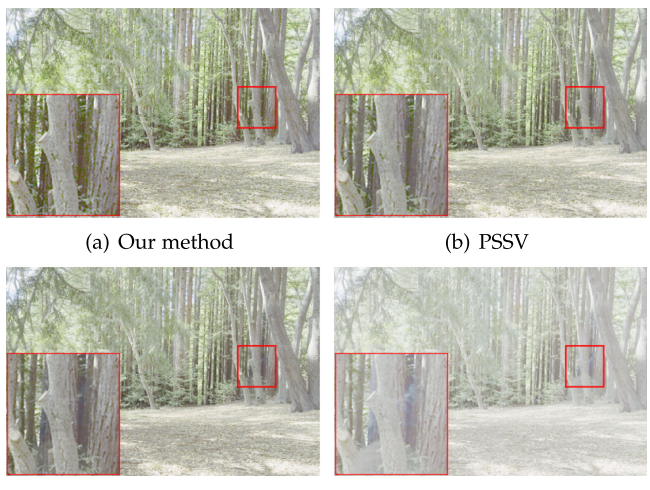
By comparing these baselines, we aim to demonstrate that our proposed matrix factorization algorithm is capable of dealing with outlier removal problems in a matrix factorization manner, and even to perform better than existing matrix factorization algorithms for ℓ_1 loss.

Following [10], we assume that the captured camera image is presented as $I = \kappa R \Delta t$ in each RGB channel, where



(a) Our method (b) PSSV (c) LMAFIT (d) RegL1

Fig. 3. HDR imaging results on `Waterfall`. *Water drops* around the target waterfall are the target outliers to be removed. We show the detail of some regions by zooming in via the boxes outlined in red, green and blue. As shown, our method achieves similar outlier detection performance to PSSV. LMAFIT removes most of these water drops, but not all. RegL1 leaves many drops in the HDR image. This figure is best viewed in color.

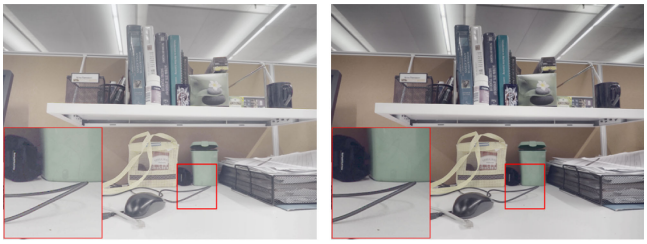


(a) Our method (b) PSSV (c) LMAFIT (d) RegL1

Fig. 4. HDR imaging results on `Forest`. The *walking person* is the target outlier to be removed. We show the detail of some regions by zooming in via the outlined box in red. As shown, our method and PSSV achieve closely similar outlier detection performance. LMAFIT cannot remove all the pixels of the walking person. RegL1 achieves the worst result. This figure is best viewed in color.

κ is a positive scalar, R is the sensor irradiance and Δt is the exposure time. The observed matrix $\mathbf{Y} \in \mathbb{R}^{m \times n}$ is the intensity matrix, which is generated by vectorizing all of the LDR images and stacking them together, i.e., $\mathbf{Y} = [\text{vec}(I_1), \text{vec}(I_2), \ldots \text{vec}(I_n)]$, where m is the number of pixels in each LDR image, and n is the number of LDR images. Ideally, the intensity of each LDR image is linear correlated, so the rank of the matrix to be recovered $\mathbf{X}$ should be 1.

After obtaining the underlying low-rank matrices for each channel, we apply the method proposed in [58] to construct the HDR image. We list the HDR results of our proposed method and other baselines in Figs. 3, 4, 5, 6, 7, and 8 for WaterFall, Forest, Desk, Desk-full, USYD and USYD-full respectively. Due to the excessive memory requirement, we



(a) Our method (b) PSSV

(c) LMAFIT (d) RegL1

Fig. 5. HDR imaging results on `Desk`. The *toy* is the target outlier to be removed. We show the detail of some regions by zooming in via the box outlined in red. As shown, our method and PSSV achieves close outlier detection performance. LMAFIT cannot remove all the pixels of the toy. RegL1 achieves the worst result. This figure is best viewed in color.

(a) Our method (b) LMAFIT (c) RegL1

Fig. 6. HDR imaging results on `Desk-full`. The *toy* is the target outlier to be removed. PSSV requires excessive memory, so we dit not obtain PSSV results. We show the detail of some regions by zooming in via the box outlined in red. As shown, our method outperforms the other two baselines. LMAFIT cannot remove all the pixels of the toy. RegL1 achieves the worst result. This figure is best viewed in color.

(a) Our method (b) PSSV (c) LMAFIT (d) RegL1

Fig. 7. HDR imaging results on `USYD`. The *walking person* is the target outlier to be removed. We show the detail of some regions by zooming in via the box outlined in red. As shown, our method and LMAFIT achieve similar outlier detection performance. PSSV and RegL1 cannot remove all the pixels of the walking person. We observe there are some outlier pixels in the bottom of the PSSV red outline box (legs), and the top right of the RegL1 red outline box (the arm). This figure is best viewed in color.

(a) Our method (b) LMAFIT (c) RegL1

Fig. 8. HDR imaging results on `USYD-full`. The *walking person* is the target outlier to be removed. PSSV requires excessive memory, so we did not obtain these results. We show the detail of some regions by zooming in via the outlined box in red. As shown, our method and LMAFIT achieve similar outlier detection performance. LMAFIT and RegL1 cannot remove all the pixels of the walking person. We observe that there are outlier pixels in the top of the red outline box of both PSSV and RegL1 (the arm). This figure is best viewed in color.

do not obtain the HDR results of PSSV on Desk-full and USYD-full.

As can be observed from these HDR results in Figs. 3, 4, 5, 6, 7, and 8, our proposed algorithm generally achieves the best outlier detection performance on all of the six datasets. Among comparison methods, PSSV obtains very close outlier detection performance on Waterfall, Forest and Desk, compared to our method, but perform poorly on USYD. LMAFIT achieves competitive results on USYD, but cannot remove outliers well for other datasets. RegL1 generally achieves the worst performance on these six datasets. This may result from the Gauss-Seidel-like optimization, which could ultimately achieve an unsatisfactory local minimum.

We list the training time of the proposed method and all baselines in Table 4. LMAFIT shows the best training speed but yields unsatisfactory outlier detection results. RegL1 has a slower training speed in the most situations than our algorithm. PSSV achieves slower training speed than our method in most cases, and there is an additional issue for PSSV in which it requires excessive memory for SVDs on large matrices. We demonstrate by the results in this table that the proposed method is more stable in outlier removal and more efficient for large matrix recovery.

6.3 Batch Image Alignment

Batch image alignment is similar to HDR imaging to some extent, and requires to solve a series of robust low-rank matrix recovery problems. It aims to align a number of images of a single target object to a fixed canonical template. In all these images, misalignment can occur in the image plane, as well as outlier objects. Similarly to HDR imaging, the key challenge of batch image alignment is to detect outliers. In [12], the authors proposed to solve a number of low-rank matrix decomposition problems based on nuclear norm and achieves some success. The formulation of our matrix factorization algorithm on the image alignment problem can be written as follows:

$$\begin{aligned} &\min_{\mathbf{X},\mathbf{E},\tau} \ ||\mathbf{E}||_1 \\ &\text{s.t.} \ \ \mathbf{Y} \circ \tau = \mathbf{X} + \mathbf{E}, \ \text{rank}(\mathbf{X}) = 1, \end{aligned} \tag{31}$$

where $\mathbf{Y} \circ \tau = [\text{vec}(I_1 \circ \tau_1), \text{vec}(I_2 \circ \tau_2), \ldots, \text{vec}(I_n \circ \tau_n)]$, I_i is the ith image and τ_i is the corresponding transformation of the ith image.

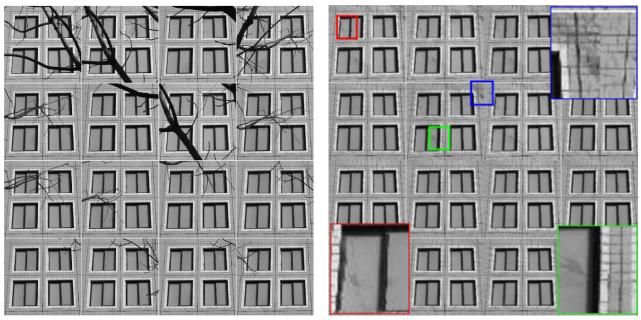
We perform our proposed algorithm on the *Windows* dataset [12], which consists of 16 misaligned images. All the experiments are conducted in Matlab on a work station with

TABLE 4
Training Time (in Seconds) of HDR Imaging Experiments

Datasets	Waterfall	Forest	Desk	USYD	Desk-full	USYD-full
LMAFIT [40]	42.0	4.4	37.3	5.2	340.5	338.1
RegL1 [39]	50.7	118.5	1193.5	397.3	2531.1	6294.2
PSSV [10]	154.7	69.5	3405.5	108.1	–	–
Ours	113.3	76.0	373.4	73.7	4756.0	3426.2

** Of the above four methods, LMAFIT, RegL1, and our method are all designed in a matrix factorization way, while PSSV is based on a nuclear-norm-like approach. We observe in our experiments that LMAFIT is always the most efficient algorithm, but it generally cannot achieve better outlier detection performance than PSSV or our method (Figs. 3, 4, 5, 6, 7, and 8). PSSV and our method are the two best performers, achieving close outlier detection performance except on USYD, where our method outperforms PSSV. In terms of memory consumption, PSSV requires more memory than other competing methods, so we have not obtained the results on the two large datasets, i.e., Desk-full and USYD-full. On the other three datasets, i.e., Waterfall, Forest, and USYD, PSSV has a similar running time to our method. However, on Desk, we observe that the running time of PSSV is approximately nine times higher than our method.*
– The training time of PSSV on Desk-full and USYD-full is not available since it requires excessive memory to run. We therefore use "–" to denote the results of PSSV on these two datasets.

an Intel(R) Xeon(R) CPU and 32 GB memory. In this experiment, we set the resolution of each image, i.e., I_i, to 1000×1000, leading to recovered $\mathbf{X} \in R^{1000000\times 16}$. RASL [12] is chosen as the competing algorithm, which is based on nuclear norm minimization and also optimized in an ALM approach. The results are shown in Fig. 9. From Fig. 9, we observe that our algorithm removes the outliers (e.g., the tree branches on some of the windows) more effectively than RASL. The main reason our algorithm obtains better outlier removal results could be the hard low-rank constraint introduced in our algorithm, i.e., $\mathrm{rank}(\mathbf{X}) = 1$. In contrast to the hard constraint in matrix factorization, RASL approximates the low-rank structure on $\mathbf{X}$ by minimizing the nuclear norm. This is a relatively soft constraint on the low-rank matrix $\mathbf{X}$, which make it possible to preserve a few outliers on $\mathbf{X}$.



(a) Aligned results by RASL (b) Recovered low-rank results by RASL

(c) Aligned results by our method (d) Recovered low-rank results by our method

Fig. 9. Batch image alignment results. We selectively zoom in on three outlier regions on the results recovered by the low-rank component of our method and RASL (presented in boxes outlined in red, green and blue). It can be easily observed that our method removes more outliers, e.g., tree branches. RASL detects most of the outliers but cannot completely remove them and leaves some noise. In contrast to RASL, our method recovers the target windows completely from the noise. This result may be achieved by the hard constraint on the low-rank $\mathbf{X}$ in our method. This figure is best viewed in color.

TABLE 5
Training Time (in Seconds) of the Image Alignment Experiment

Datasets	Windows
RASL [40]	5259.3
Ours	2988.2

We additionally illustrate the training time for both algorithms of the image alignment experiment in Table 5 to show the efficiency. From the results in Table 5, our method can achieve better efficiency even when our method removes more outliers than RASL.

7 Conclusion

Matrix factorization achieves state-of-the-art performance on low-rank matrix recovery. Different applications require various loss functions; For example, the rating data in recommender systems are usually in discrete ordinal values, and maximum margin loss tends to perform better than least squared loss. In some computer vision applications, non-smooth ℓ_1-norm loss is applied. We propose an effective optimization algorithm to enable the original method to handle the non-smooth case based on ALM. Extensive experiments demonstrate that our method is competitive in both smooth and non-smooth cases in terms of effectiveness and efficiency.

References

[1] M. Wang, W. Fu, S. Hao, D. Tao, and X. Wu, "Scalable semi-supervised learning by efficient anchor graph regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1864–1877, Jul. 2016.
[2] M. Wang, X. Liu, and X. Wu, "Visual classification by ℓ_1-hypergraph modeling," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 9, pp. 2564–2574, Sep. 2015.
[3] L. Nie, M. Wang, L. Zhang, S. Yan, B. Zhang, and T.-S. Chua, "Disease inference from health-related questions via sparse deep learning," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 8, pp. 2107–2119, Aug. 2015.
[4] J. Wang, M. Wang, P. Li, L. Liu, Z. Zhao, X. Hu, and X. Wu, "Online feature selection with group structure analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 3029–3041, Nov. 2015.
[5] Y. Yan, F. Nie, W. Li, C. Gao, Y. Yang, and D. Xu, "Image classification by cross-media active learning with privileged information," *IEEE Trans. Multimedia*, vol. 18, no. 12, pp. 2494–2502, Dec. 2016.

[6] N. Srebro, J. D. M. Rennie, and T. S. Jaakola, "Maximum-margin matrix factorization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2005, pp. 1329–1336.

[7] M. Xu, J. Zhu, and B. Zhang, "Fast max-margin matrix factorization with data augmentation," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. III-978–III-986.

[8] M. Tan, I. W. Tsang, and L. Wang, "Is matching pursuit solving convex problems?" Nanyang Technological University, *CoRR*, vol. abs/1302.5010, 2013.

[9] T.-H. Oh, J.-Y. Lee, Y.-W. Tai, and I. S. Kweon, "Robust high dynamic range imaging by rank minimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 6, pp. 1219–1232, Jun. 2015.

[10] T. H. Oh, Y. W. Tai, J. C. Bazin, H. Kim, and I. S. Kweon, "Partial sum minimization of singular values in robust PCA: Algorithm and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 744–758, Apr. 2016.

[11] C. Lee, Y. Li, and V. Monga, "Ghost-free high dynamic range imaging via rank minimization," *IEEE Signal Process. Lett.*, vol. 21, no. 9, pp. 1045–1049, Sep. 2014.

[12] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2233–2246, Nov. 2012.

[13] M. Fazel, "Matrix rank minimization with applications," PhD thesis, Dept. Elect. Eng., Stanford Univ., Stanford, CA, USA, 2002.

[14] E. J. Candés and T. Tao, "The power of convex relaxation: Near-optimal matrix completion," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2053–2080, May 2010.

[15] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Rev.*, vol. 52, no. 3, pp. 471–501, 2010.

[16] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM J. Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[17] K.-C. Toh and S. Yun, "An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems," *Pacific J. Optimization*, vol. 6, pp. 615–640, 2010.

[18] J. D. M. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proc. Int. Conf. Mach. Learn.*, 2005, pp. 713–719.

[19] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1257–1264.

[20] M. Tan, I. W. Tsang, L. Wang, B. Vandereycken, and S. J. Pan, "Riemannian pursuit for big matrix recovery," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. II-1539–II-1547.

[21] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proc. 45th Annu. ACM Symp. Theory Comput.*, 2013, pp. 665–674.

[22] B. Vandereycken, "Low-rank matrix completion by Riemannian optimization," *SIAM J. Optimization*, vol. 23, no. 2, pp. 1214–1236, 2013.

[23] Z. Wen, W. Yin, and Y. Zhang, "Solving a low-rank factorization model for matrix completion by a non-linear successive over-relaxation algorithm," *Math. Program. Comput.*, vol. 4, no. 4, pp. 333–361, 2012.

[24] T. Ngo and Y. Saad, "Scaled gradients on Grassmann manifolds for matrix completion," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1412–1420.

[25] N. Boumal and P.-A. Absil, "RTRMC: A Riemannian trust-region method for low-rank matrix completion," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 406–414.

[26] M. Xu, J. Zhu, and B. Zhang, "Nonparametric max-margin matrix factorization for collaborative prediction," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 64–72.

[27] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ, USA: Princeton Univ. Press, 2008.

[28] N. Boumal and P.-A. Absil, "RTRMC: A Riemannian trust-region method for low-rank matrix completion," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 406–414.

[29] G. Meyer, S. Bonnabel, and R. Sepulchre, "Linear regression under fixed-rank constraints: A Riemannian approach," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 542–552.

[30] B. Mishra, K. A. Apuroop, and R. Sepulchre, "A riemannian geometry for low-rank matrix completion," *CoRR*, vol. abs/1211.1550, 2012.

[31] L. Balzano, R. Nowak, and B. Recht, "Online identification and tracking of subspaces from highly incomplete information," in *Proc. Annu. Allerton Conf. Commun. Control Comput.*, 2010, pp. 704–711.

[32] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proc. Forty-fifth Annu. ACM Symp. Theory Comput.*, Palo Alto, California, USA, 2013, pp. 665–674, doi: 10.1145/2488608.2488693.

[33] K. Wei, J.-F. Cai, T. F. Chan, and S. Leung, "Guarantees of Riemannian optimization for low rank matrix recovery," *SIAM J. Matrix Anal. Appl.*, vol. 37, no. 3, pp. 1198–1222, 2016.

[34] M. Weimer, A. Karatzoglou, and A. Smola, "Improving maximum margin matrix factorization," *Mach. Learn.*, vol. 72, no. 3, pp. 263–276, 2008.

[35] M. Weimer, A. Karatzoglou, Q. V. Le, and A. Smola, "Maximum margin matrix factorization for collaborative ranking," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1593–1600.

[36] A. Karatzoglou, M. Weimer, and A. J. Smola, "Collaborative filtering on a budget," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 389–396.

[37] D. DeCoste, "Collaborative prediction using ensembles of maximum margin matrix factorizations," in *Proc. Int. Conf. Mach. Learn.*, 2006, pp. 249–256.

[38] M. Wu, "Collaborative filtering via ensembles of matrix factorizations," in *Proc. KDD Cup Workshop*, 2007, pp. 43–47.

[39] Y. Zheng, G. Liu, S. Sugimoto, S. Yan, and M. Okutomi, "Practical low-rank matrix approximation under robust L1-norm," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1410–1417.

[40] Y. Shen, Z. Wen, and Y. Zhang, "Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization," *Optimization Methods Softw.*, vol. 29, no. 2, pp. 239–263, 2014.

[41] B. Mishra, G. Meyer, F. Bach, and R. Sepulchre, "Low-rank optimization with trace norm penalty," *SIAM J. Optimization*, vol. 23, no. 4, pp. 2124–2149, 2013.

[42] R. Ge, C. Jin, and Y. Zheng, "No spurious local minima in nonconvex low rank problems: A unified geometric analysis," in *Proc. 34th Int. Conf. Mach. Learn.*, International Convention Centre, Sydney, Australia, vol. 70, pp. 1233–1242, Aug. 2017.

[43] S. Wang, X. Chang, X. Li, G. Long, L. Yao, and Q. Z. Sheng, "Diagnosis code assignment using sparsity-based disease correlation embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3191–3202, Dec. 2016.

[44] X. Chang, F. Nie, Y. Yang, C. Zhang, and H. Huang, "Convex sparse PCA for unsupervised feature learning," *ACM Trans. Knowl. Discovery Data*, vol. 11, no. 1, 2016, Art. no. 3.

[45] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, 2011, Art. no. 11.

[46] Z. Ma, X. Chang, Y. Yang, N. Sebe, and A. G. Hauptmann, "The many shades of negativity," *IEEE Trans. Multimedia*, vol. 19, no. 7, pp. 1558–1568, Jul. 2017.

[47] H. Ji, C. Liu, Z. Shen, and Y. Xu, "Robust video denoising using low rank matrix completion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1791–1798.

[48] C. Gao, D. Meng, Y. Yang, Y. Wang, X. Zhou, and A. G. Hauptmann, "Infrared patch-image model for small target detection in a single image," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 4996–5009, Dec. 2013.

[49] L. Wu, A. Ganesh, B. Shi, Y. Matsushita, Y. Wang, and Y. Ma, "Robust photometric stereo via low-rank matrix completion and recovery," in *Proc. Asian Conf. Comput. Vis.*, 2010, pp. 703–717.

[50] E. J. Candès and T. Tao, "The power of convex relaxation: Near-optimal matrix completion," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2053–2080, May 2010.

[51] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," UIUC, *arXiv preprint arXiv:1009.5055*, 2010.

[52] Y. Yan, M. Tan, I. Tsang, Y. Yang, C. Zhang, and Q. Shi, "Scalable maximum margin matrix factorization by active Riemannian subspace search," in *Proc. Int. Conf. Artif. Intell.*, 2015, pp. 3988–3994.

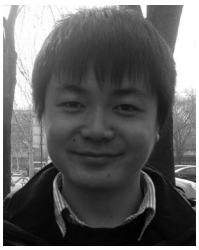
[53] M. Hintermüller and R. John-von Neumann Haus, "Nonlinear optimization," *Vorlesungsskript Universität Berlin*, 2010.

[54] W.-S. Chin, Y. Zhuang, Y.-C. Juan, and C.-J. Lin, "A fast parallel stochastic gradient method for matrix factorization in shared memory systems," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 1, 2015, Art. no. 2.

[55] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 1999, pp. 230–237.

[56] J. Bennett and S. Lanning, "The netflix prize," in *Proc. KDD Cup Workshop*, 2007, pp. 3–6.

[57] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer, "The Yahoo! music dataset and KDD-Cup'11," in *Proc. JMLR Workshop Conf. Proc.*, 2012, pp. 3–18.
[58] P. E. Debevec and J. Malik, "Recovering high dynamic range radiance maps from photographs," in *Proc. ACM Annu. Conf. Comput. Graph. Interactive Techn.*, 2008, Art. no. 31.

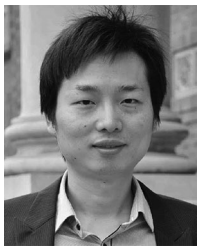


Yan Yan received the BE degree in computer science from Tianjin University, Tianjin, China, in 2013. He is currently working toward the PhD degree in the Research Centre for Artificial Intelligence (CAI), University of Technology Sydney, Australia. His current research interests include machine learning and computer vision.

Mingkui Tan received the bachelor's degree in environmental science and engineering from Hunan University, Changsha, China, in 2006, the master's degree in control science and engineering from Hunan University, Changsha, China, in 2009, and the PhD degree in computer science from Nanyang Technological University, Singapore, in 2014. He is currently working as a professor with the School of Software Engineering, South China University of Technology. His research interests include compressive sensing, big data learning, and large-scale optimization.

Ivor W. Tsang received the PhD degree in computer science from the Hong Kong University of Science and Technology, in 2007. He is an ARC future fellow and professor of artificial intelligence with the University of Technology Sydney (UTS). He is also the research director of the UTS Priority Research Centre for Artificial Intelligence (CAI). He has more than 140 research papers published in refereed international journals and conference proceedings, including the *Journal of Machine Learning Research*, the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, the *IEEE Transactions on Neural Networks and Learning Systems*, NIPS, ICML, etc. In 2009, he was conferred the 2008 Natural Science Award (Class II) by the Ministry of Education, China. In 2013, he received his prestigious Australian Research Council Future Fellowship for his research regarding machine learning on big data. Besides these, he received the prestigious IEEE Transactions on Neural Networks Outstanding 2004 Paper Award in 2006, the 2014 IEEE Transactions on Multimedia Prized Paper Award, and a number of best paper awards and honors from reputable international conferences, including the Best Student Paper Award at CVPR 2010, etc. He was also awarded the ECCV 2012 Outstanding Reviewer Award.

Yi Yang received the PhD degree in computer science from Zhejiang University, Hangzhou, China, in 2010. He is currently a professor with the University of Technology Sydney, NSW, Australia. He was a post-doctoral research with the School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania. His current research interests include machine learning and its applications to multimedia content analysis and computer vision, such as multimedia indexing and retrieval, surveillance video analysis, and video content understanding.



Qinfeng Shi received the bachelor's and master's degrees in computer science and technology from Northwestern Polytechnical University (NPU), in 2003 and 2006, respectively, and the PhD degree in computer science from the Australian National University (ANU), in 2011. He is a senior lecturer with the School of Computer Science, University of Adelaide. He is leading a machine learning team focusing on graphical models, deep learning, and computer vision. He received an ARC Discovery Early Career Researcher Award (DECRA) (2012-2014).

Chengqi Zhang received the bachelor's degree in computer science from Fudan University, in March 1982, the master's degree in computer science from Jilin University, in March 1985, and the PhD degree in computer science from the University of Queensland, in October 1991, followed by a doctor of science degree from Deakin University, in October 2002. He has been appointed as a distinguished professor with the University of Technology Sydney from 27 February 2017 to 26/February 2022, an executive director UTS Data Science from 3 January 2017 to 2 January 2021, an honorary professor with the University of Queensland from 1 January 2015 to 31 December 2017, an adjunct professor with the University of New South Wales from 20 March 2017 to 20 March 2020, and a research professor of information technology with the UTS from 14 December 2001. In addition, he has been selected as the chairman of the Australian Computer Society National Committee for Artificial Intelligence since November 2005, and the chairman of the IEEE Computer Society Technical Committee of Intelligent Informatics (TCII) since June 2014.