M-estimation in Low-rank Matrix Factorization: a General Framework

***Abstract*—Many problems in science and engineering can be reduced to the recovery of an unknown large matrix from a small number of random linear measurements. Matrix factorization arguably is the most popular approach for low-rank matrix recovery. Many methods have been proposed using different loss functions, for example the most widely used L_2 loss, more robust choices such as L_1 and Huber loss, quantile and expectile loss for skewed data. All of them can be unified into the framework of M-estimation. In this paper, we present a general framework of low-rank matrix factorization based on M-estimation in statistics. The framework mainly involves two steps: firstly we apply Nesterov's smoothing technique to obtain an optimal smooth approximation for non-smooth loss function, such as L_1 and quantile loss; secondly we exploit an alternative updating scheme along with Nesterov's momentum method at each step to minimize the smoothed loss function. Strong theoretical convergence guarantee has been developed for the general framework, and extensive numerical experiments has been conducted to illustrate the performance of proposed algorithm.**

***Index Terms*—matrix recovery, M-estimation, matrix factorization, robustness, statistical foundation**

I. Introduction

Motivation. In matrix recovery from linear measurements, we are interested in recovering an unknown matrix $X \in \mathbb{R}^{m\times n}$ from $p < mn$ linear measurements $b_i = \mathrm{Tr}(A_i^\top X)$, where each $A_i \in \mathbb{R}^{m\times n}$ is a measurement matrix, $i = 1,\dots p$. Usually it is expensive or even impossible to fully sample the entire matrix X, and we are left with a highly incomplete set of observations. In general it is not always possible to recovery X under such settings, however, if we impose a low-rank structure on X, it is possible to exploit this structure and efficiently estimate X. The problem of matrix recovery arises in a wide range of applications, such as collaborate filtering [1], image recovery [2], structure from motion, photometric stereo [3], system identification [4], and computer network tomography [5].

Matrix factorization arguably is the most popular and intuitive approach for low-rank matrix recovery. The basic idea is to decompose the low-rank matrix $X \in \mathbb{R}^{m\times n}$ into the product of two matrices:

$$X = U^\top V, \tag{1}$$

where $U \in \mathbb{R}^{r\times m}$ and $V \in \mathbb{R}^{r\times n}$. In many practical problems, the rank of a matrix is known in advance, or can be estimated priori; see, for example, the rigid and nonrigid structures from motion as well as image recovery. The matrix U and V can also be interpreted as latent factors that drive the unknown matrix X.

Matrix factorization is usually based on L_2 loss, i.e. square loss, which is optimal for Gaussian errors, however, its performance may be severely deteriorated when the data is polluted by outliers. For example, in a collaborate filtering system, some popular items have many ratings regardless whether they are useful, while others have few ones. There may even exist shilling attacks, i.e. a user may consistent gives positive feedback for their products or negative feedbacks to their competitors regardless of the items themselves [6]. Recently there are few attempts to address this problem; see, for example, [2], [7]. However, to the best of our knowledge, they focus on either L_1 procedures or quantile related procedures, and are only useful in limited scenarios. For low-rank matrix recovery under M-estimation, He *et al.* [8] studied the use of a few smooth loss function such as Huber and Welsch in this setting. In this paper, we propose a more general framework that is applicable to any M-estimation loss function, smooth or non-smooth, and provide theoretical convergence guarantee on the proposed state-of-the-art algorithm.

This paper introduces a general framework of M-estimations [9]–[11] to matrix factorization. Specifically we consider the loss function, related to an M-estimation, for matrix factorization. The M-estimation is defined in a way similar to the well known terminology "Maximum Likelihood Estimate (MLE)" in statistics, and has many good properties that are similar to those of MLE. In meanwhile it still retains intuitive interpretation. The proposed loss function includes both well-known L_1 and L_2 loss as special cases. In real applications, we choose a suitable M-estimation procedure, according to our knowledge of the data and specific priorities of the problem. For example, the well known Huber M loss enjoys the property of smoothness as L_2 loss and robustness as L_1 loss.

The loss functions of some M-estimation procedures are smooth such as the L_2 loss, while those of the others are nonsmooth such as L_1 and quantile loss. Note that the resulting objective functions both have a bilinear structure due to the decomposition at (1). For nonsmooth cases, we first consider Nesterov's smoothing method to obtain an optimal smooth approximation [12], and the bilinear structure is preserved. The alternating minimization method is hence used to search for the solutions. At each step, we employ Nesterov's momentum method to accelerate the convergence, and it actually is easier to find the global optima. Figure 1 gives the flowchart of our complete algorithm. Theoretical convergence analysis is conducted for both smooth and nonsmooth loss functions.

Contributions. We summarize our contributions below:

1. We propose to do matrix factorization based on the loss

function of an M-estimation. The proposed framework is very general and applicable to any M-estimate loss function, which gives us the flexibility in selecting a suitable loss for the specific problem.

2. We propose to use Nesterov's smoothing technique to obtain an optimal smooth approximation when the loss function is nonsmooth.
3. We consider the Nesterov's momentum method, rather than gradient decedent methods, to perform the optimization at each step of the alternating minimization which accelerate the convergence.
4. We provide theoretical convergence guarantees for the propose algorithm.
5. We illustrate the usefulness of our method by conducting extensive simulation experiments for both synthetic data and real data.

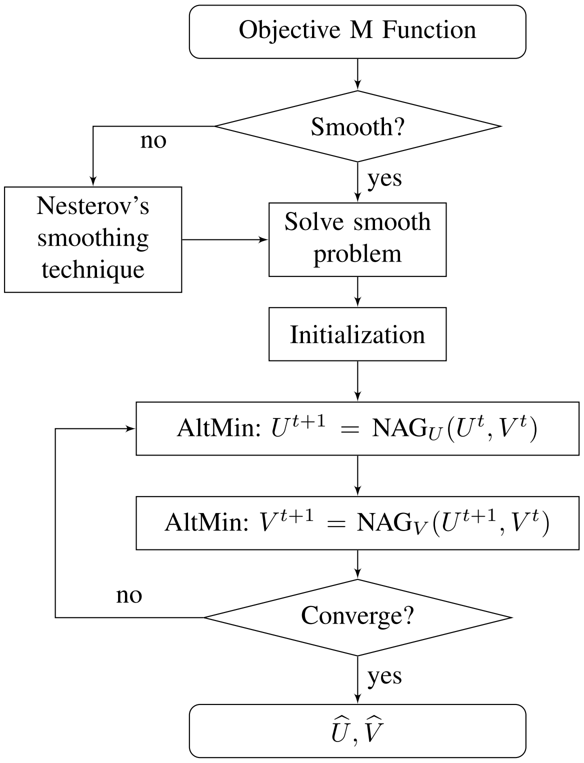
Objective M Function
Smooth?
no
yes
Nesterov's smoothing technique
Solve smooth problem
Initialization
AltMin: $U^{t+1} = \text{NAG}_U(U^t, V^t)$
AltMin: $V^{t+1} = \text{NAG}_V(U^{t+1}, V^t)$
no
Converge?
yes
$\widehat{U}, \widehat{V}$

Figure 1: Flowchart of the whole algorithm

II. Methodology Framework

Let $X^* \in \mathbb{R}^{m\times n}$ be the target low-rank matrix, and $A_i \in \mathbb{R}^{m\times n}$ with $1 \le i \le p$ be given measurement matrices. Here A_i's can be the same or distinct with each other. We assume the observed signals $b = (b_1, ..., b_p)^\top$ have the following structure:

$$b_i = \langle A_i, X^* \rangle + \epsilon_i, \quad i = 1, \dots, p, \tag{2}$$

where $\langle A_i, X\rangle := \text{Tr}(A_i^\top X)$, and ϵ_i is the error term. Suppose that the rank of matrix X^* is no more than r with $r \ll \min(m, n, p)$. We then have the decomposition $X^* = U^{*\top} V^*$, and the matrix can be recovered by solving a non-convex optimization problem,

$$\min_{U\in\mathbb{R}^{r\times m}, V\in\mathbb{R}^{r\times n}} \quad \frac{1}{p}\sum_{i=1}^{p} \mathcal{L}(b_i - \langle A_i, U^\top V\rangle), \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function related to an M-estimation; see, for example, $\mathcal{L}(y) = y^2$ for the L_2 loss, and $|y|$ for the L_1 loss. Here $\mathcal{L}(\cdot)$ usually is convex. Let $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ be an affine transformation with the ith entry of $\mathcal{A}(X)$ being $\langle A_i, X\rangle$, and $\mathcal{M}(x) = p^{-1}\sum_{i=1}^{p} \mathcal{L}(x_i)$ for a vector $x = (x_1, ..., x_p)^\top$. We then rewrite (3) into a compact form

$$\min_{U\in\mathbb{R}^{r\times m}, V\in\mathbb{R}^{r\times n}} \mathcal{M}(b - \mathcal{A}(U^\top V)). \tag{4}$$

A. The case that $\mathcal{M}$ is not smooth

The loss function $\mathcal{L}(\cdot)$, and hence $\mathcal{M}(\cdot)$, may be nonsmooth for some M-estimation; see, for example, the L_1 loss. For this case, we first employ Nesterov's smoothing method to obtain an optimal smooth approximation; see [12] (pp. 129-132).

Specifically, We first assume that the objective function $\mathcal{M}$ has the following structure:

$$\begin{aligned} \mathcal{M}(b - \mathcal{A}(U^\top V)) &= \hat{\mathcal{M}}(b - \mathcal{A}(U^\top V)) \\ &+ \max_u \left\{ \langle B(b - \mathcal{A}(U^\top V)), u\rangle_2 - \hat{\phi}(u) \right\}, \end{aligned} \tag{5}$$

where $\hat{\mathcal{M}}(\cdot)$ is continuous and convex; see equation (2.2) in [12].

Then the objective function $\mathcal{M}$ can be approximated by

$$\begin{aligned} \mathcal{M}_\pi(b - \mathcal{A}(U^\top V)) &= \hat{\mathcal{M}}(b - \mathcal{A}(U^\top V)) \\ &+ \max_u \left\{ \langle B(b - (U^\top V)), u\rangle_2 - \hat{\phi}(u) - \pi d_2(u) \right\}, \end{aligned} \tag{6}$$

where π is the smoothness parameter, which is positive. Note that $\mathcal{M}_\pi(\cdot)$ is smooth, and there are then many available techniques for optimizing it.

B. Alternating Minimization

Due to the bilinear form with respect to U and V in the objective functions (4) and (6), we consider an alternatively minimization scheme. Specifically, in each iteration, we will keep one of U and V fixed, optimize over the other, and then switch in the next iteration until the algorithm converges. The details

Moreover, the direct optimization at (4) may have unstable solutions, and this paper solves this problem by adopting the regularization method via Procrustes flow in [13]. Specifically, rather than (4), we attempt to do the optimization below:

$$\min_{U\in\mathbb{R}^{r\times m}, V\in\mathbb{R}^{r\times n}} \mathcal{M}(b - \mathcal{A}(U^\top V)) + \lambda \|UU^\top - VV^\top\|_F^2, \tag{7}$$

where $\|\cdot\|_F$ stands for the Frobenius norm, and the ad hoc choice of λ is 1/16. Denote by $\mathcal{M}^\lambda(U, V)$ the regularized version at (7), and all algorithms designed below are for this objective function. For the case with nonsmooth $\mathcal{M}$, we can similarly define the regularized objective function, denoted by $\mathcal{M}_\pi^\lambda(U, V)$, and all algorithms for $\mathcal{M}^\lambda(U, V)$ can then be applied.

III. Initialization

When the starting values of U and V are orthogonal (or almost orthogonal) to the true space, the designed algorithm may never be able to converge to true values, and hence an initialization procedure is needed to avoid this situation. We here adopt the singular value projection (SVP), which originates from [14] and was later used by [13], to provide the initial values of U and V for the designed algorithm in the next section. The main difference between our method and the original one is summarized into Line 2 of Algorithm 1, where we use the loss function related to an M-estimation,

$$\nabla_X \mathcal{M}(b - \mathcal{A}(X^t)) = -\frac{1}{p}\sum_{i=1}^{p} \dot{\mathcal{L}}(b - \mathcal{A}(X^t))A_i,$$

and $\dot{\mathcal{L}}(\cdot)$ is the derivative of $\mathcal{L}(\cdot)$.

Algorithm 1: Initialization by SVP algorithm

```
  Input: $\mathcal{A}$, $b$, tolerance $\epsilon_1$, step size $\xi_t$ with
      $t = 0, 1, \cdots$, and $X^0 = 0_{m\times n}$
  Output: $X^{t+1}$
1 Repeat
2 $Y^{t+1} \leftarrow X^t - \xi_t \nabla_X \mathcal{M}(b - \mathcal{A}(X^t))$
3 Compute top $r$ singular vectors of $Y^{t+1}$:$U_r, \Sigma_r, V_r$
4 $X^{t+1} \leftarrow U_r \Sigma_r V_r$
5 $t \leftarrow t + 1$
6 Until $\|X^{t+1} - X^t\|_F \leq \epsilon_1$
```

It is noteworthy to point out that Algorithm 1 can be directly used to recover the matrix X^* if it is iterated for sufficient times. However, the singular value calculation here is time-consuming when the dimension of matrix X^* is large. Furthermore, as an initialization, we do not need a very small tolerance ϵ_1, i.e., a rough output is sufficient. Our simulation experiments show that, after several iterations of the SVP algorithm, the resulting values will be close to the true ones, while it is not the case for the random initialization.

Algorithm 1 actually can be rewritten into a compact form,

$$X^{t+1} \leftarrow \mathcal{P}_r\left(X^t - \xi_t \nabla_X \mathcal{M}(b - \mathcal{A}(X^t))\right),$$

where $\mathcal{P}_r$ denotes the projection onto the space of rank-r matrices. Moreover, the original objective function $\mathcal{M}$, rather than the regularized one at (7), is used in Algorithm 1 and, for the nonsmooth case, we will use $\mathcal{M}_\pi$ at (6).

IV. Algorithm

There are two layers of iterations in our algorithm: the outer layer is the alternating minimization; and the inner layer is to employ Nesterov's momentum algorithm to obtain updated values of U^{t+1} and V^{t+1}.

We first introduce the inner layer, and the Nesterov's momentum method is used to update the values of U^t and V^t to those of U^{t+1} and V^{t+1}. Algorithm 2 gives the details for updating the value of U^t, and can be denoted by NAG$_U$ for simplicity. Here the gradient $\nabla_U \mathcal{M}^\lambda(U, V)$ is defined as the gradient with respect to U, and similarly we can define $\nabla_V \mathcal{M}^\lambda(U, V)$. Moreover, $\boldsymbol{\nu}_{(i)}^t$ stands for the momentum term, γ is the momentum parameter, and η is the learning rate. The value of γ is usually chosen to be around 0.9; see [15] and [16]. Similarly we can give the detailed algorithm for updating the value of V^t, and it can be denoted by NAG$_V$.

Algorithm 2: Nesterov's accelerate gradient (NAG) method

```
  Input: $U^t$, $V^t$, momentum parameter $\gamma$, learning rate
      $\eta$, and tolerance $\epsilon_2$
  Output: $U^{t+1}$
1 Repeat
2 $\boldsymbol{\nu}_{(i)}^t = \gamma \boldsymbol{\nu}_{(i-1)}^t + \eta \nabla_U \mathcal{M}^\lambda(U_{(i-1)}^t - \gamma \boldsymbol{\nu}_{(i-1)}^t, V^t)$
3 $U_{(i)}^t = U_{(i-1)}^t + \boldsymbol{\nu}_{(i)}^t$
4 Until $\|U_{(i)}^t - U_{(i-1)}^t\|_F \leq \epsilon_2$
```

The alternating minimization method is employed for the outer layer of iterations; see Algorithm 3 for details. The final solutions can be denoted by $\widehat{U}$ and $\widehat{V}$, and we then can use $\widehat{U}^\top \widehat{V}$ to approximate the low-rank matrix X^*.

Algorithm 3: Alternating Minimization

```
  Input: $U^0$, $V^0$
  Output: $\widehat{U}$, $\widehat{V}$
1 Repeat
2 1.1.Update $U^t$ with $U^{t+1} = \text{NAG}_U(U^t, V^t)$
3 1.2.Update $V^t$ with $V^{t+1} = \text{NAG}_V(U^{t+1}, V^t)$
4 Until converge
```

V. Convergence results

Suppose that U and V are a pair of solutions, i.e. $X = U^\top V$. It then holds that, for an orthonormal matrix R satisfying $R^\top R = I_r$, $U^\dagger = RU$ and $V^\dagger = RV$ are another pair of solutions. To evaluate the performance of the proposed algorithm, we first define a distance between two matrices,

$$\text{dist}(U, U^\dagger) = \min_{R\in\mathbb{R}^{r\times r}:R^\top R = I_r} \|U - RU^\dagger\|_F,$$

where $U, U^\dagger \in \mathbb{R}^{r\times m}$ with $m \geq r$; see [13].

Theorem 1. *Let $X \in \mathbb{R}^{m\times n}$ be a rank r matrix, with singular values $\sigma_1(X) \geq \sigma_2(X) \geq \cdots \geq \sigma_r(X) > 0$ and condition number $\kappa = \sigma_1(X)/\sigma_r(X)$. Denote by $X = A^\top \Sigma B$ the corresponding SVD decomposition. Let $U = A^\top \Sigma^{1/2} \in \mathbb{R}^{m\times r}$ and $V = B^\top \Sigma^{1/2} \in \mathbb{R}^{n\times r}$. Assume that $\mathcal{A}$ satisfies a rank-6r RIP condition with RIP constant $\sigma_{6r} < \frac{1}{25}$, $\xi_t = \frac{1}{p}$. Then using $T_0 \geq 3\log(\sqrt{r}\kappa) + 5$ iterations in Algorithm 1 yields a solution U_0, V_0 obeying*

$$\text{dist}\left(\begin{bmatrix} U_0 \\ V_0 \end{bmatrix}, \begin{bmatrix} U \\ V \end{bmatrix}\right) \leq \frac{1}{4}\sigma_r(U). \quad (8)$$

Furthermore, starting from any initial solution obeying (8), the t-th iterate of Algorithm 3 satisfies that

$$\text{dist}\left(\begin{bmatrix} U_t \\ V_t \end{bmatrix}, \begin{bmatrix} U \\ V \end{bmatrix}\right) \leq \frac{1}{4}(1-\tilde{\tau}_1)^t \frac{\tilde{\mu}}{\tilde{\xi}} \frac{1+\delta_r}{1-\delta_r} \sigma_r(U) \qquad (9)$$

Equation (8) in the above theorem guarantees that, under the RIP assumption on linear measurements $\mathcal{A}$, our algorithm can achieve a good initialization. Equation (9) tells us that, starting from a sufficiently accurate initialization, the algorithm exhibits linear convergence rate. Moreover, the specific convergence rates of both initialization at (8) and alternating minimization at (9) also depend on the RIP constant δ_{6r}.

Jain *et al.* [14] and Tu *et al.* [13] built the convergence result for alternating minimization under least square matrix factorization, however, their proving methods cannot be directly adopted to derive the results at (8). This paper extends [14]'s SVP method to more general objective function based on M-estimation, and a detailed proof for the convergence of SVP initialization is also provided. For the linear convergence at (9), Tu *et al.* [13] gave a similar result, while the gradient descent method was used for the alternating minimization. This paper adopts the proving technique in [17] to establish the linear converge of the alternating minimization based on Nesterov's momentum method.

When $\mathcal{M}$ is not smooth, the regularized objective function $\mathcal{M}^\lambda$, defined as in (7), is also not smooth, while the function $\mathcal{M}^\lambda_\pi$ is smooth. Denote $\{U^\ddagger, V^\ddagger\} = \min_{U,V} \mathcal{M}^\lambda(U,V)$ and $\{U^{\pi\ddagger}, V^{\pi\ddagger}\} = \min_{U,V} \mathcal{M}^\lambda_\pi(U,V)$.

Theorem 2. *(Convergence of optimal solution of smoothed objective function) As $\pi \to 0^+$, we have $U^{\pi\ddagger\top} V^{\pi\ddagger} \to U^{\ddagger\top} V^\ddagger$.*

The above theorem guarantees that the optimal solution of the smoothed object function converges to that of the nonsmooth one as the smoothness parameter π tends to zero.

Theorem 2 actually is one of our important contributions. In the literature, most of state-of-art smoothing methods was given without theoretical justifications; see, for example, [18]. Theorem 2 implies that the optimal solution for smooth approximation will indeed converge to the solution for nonsmooth objective function, i.e. the smooth and nonsmooth objective function will lead to same results under some regularity conditions. Under matrix factorization settings, Yang *et al.* [19] considered Nesterov's smoothing method to obtain a smooth approximation, while it handled the nonnegative matrix only, which actually is a much simpler problem compared with the one in this paper. Moreover, no strong theoretical convergence guarantee was provided there.

VI. Simulation settings

As a robust alternative to L_2 loss, L_1 loss brings computation difficulties since it is not smooth. Using the Nesterov's smoothing method, we obtained the well-known Huber loss:

$$\mathcal{L}_\mu(a) = \begin{cases} \frac{1}{2\mu}|a|^2 & \text{for} \quad |a| \leq \mu \\ |a| - \frac{\mu}{2} & \text{otherwise} \end{cases},$$

where μ is the smoothness parameter, and $\mathcal{L}_\mu(a)$ will approach the L_1 loss as μ decreases. For the simulation section, we compare the performance of Huber and L_2 loss.

A. Synthetic data

We first generate a matrix X of size $m \times n$ by sampling each entry from the Gaussian distribution $\mathcal{N}(0,1)$, and the true matrix X^* is obtained via truncated singular value decomposition (SVD) by keeping the first r largest singular values. The sampling matrices $A_i,\ i = 1, \ldots, p$ are independently produced by sampling each entry from the Gaussian distribution $\mathcal{N}(0,1)$. To evaluate the robustness of proposed method, we consider eight distributions for the error term ϵ_i: (a) No error; (b) $\mathcal{N}(0,2)$; (c) $\mathcal{N}(0,10)$; (d) log$\mathcal{N}(0,1)$; (e) Cauchy; (f) $t(3)$; (g) Pareto$(1,1)$; (h) $0.9\mathcal{N}(0,1) + 0.1\mathcal{N}(100,1)$. Note that (e) and (g) are heavy-tailed with the first order moment being infinite. We consider two loss functions for the recovery: Huber and L_2 loss, and there are 100 replications for each error distribution with $m = n = 100$, $r = 10$ and $p = 5000$.

We consider the following three metrics for evaluation: 1) relative error (RE): $||X^* - \widehat{U}^\top \widehat{V}||_F / ||X^*||_F$; 2) recovery rate (RR): fraction of element-wise relative error smaller than 5%, where the element-wise relative error is defined as $|(X^*_{ij} - (\widehat{U}^\top \widehat{V})_{ij})/X^*_{ij}|$; 3) test error (MSE): a test set $\{\mathcal{A}^*, b^*\}$ with $p^* = 100$ was generated using the same pipeline, and the test error is defined as $p^{*-1}\|\mathcal{A}^*(\widehat{U}^\top \widehat{V}) - b^*\|^2$. For both RE and MSE, a smaller value is desired, while a value closer to one indicates better performance for RR. Table I presents the results based on the average of 100 replications, and we highlight preferable figures by boldface.

When the data is very heavy-tailed (error (e) and (g)), the algorithm will diverge for some cases. Table II shows the percentages of convergence. For the other six distributions, the algorithm converges for all replications. Moreover, we observe that scaling down the learning rate can make the algorithm converge, but this may severely slow down the computation. Actually, when starting with a larger learning rate, the algorithm with L_2 loss tends to diverge, while that with the Huber loss will converge faster.

When there is no error, Huber and L_2 loss are comparable, both reaching a recovery rate over 96%. In terms of the metrics presented, L_2 loss is better than Huber loss when the error is normally distributed. However, Huber loss is more robust when error introduces bias, skewness or outliers. Huber loss has substantial advantage over L_2 loss especially when the error is skewed (error d) or heavy-tailed (error f). When the linear measurements are biased (error h) or contaminated by heavy-tailed outliers (errors e & g), matrix recovery using L_2 recovers only less than 7% of entries and has very large MSE, while in contrast, employing Huber loss allows the procedure to recover at least 40% of the entries and the recovery error is less than 7% in general.

Figure 2 gives box plots of recovery rates for different error distributions and loss functions. It can be seen that recovery rates vary within a small scale, and the observations are consistent with those in Table I. Overall, the algorithm

Error	Loss	RE	RR	MSE
(a)	Huber	0.003	0.960	0.030
	L_2	**0.003**	**0.969**	**0.022**
(b)	Huber	0.030	0.650	2.885
	L_2	**0.028**	**0.671**	**2.487**
(c)	Huber	0.163	0.185	84.628
	L_2	**0.141**	**0.211**	**63.040**
(d)	Huber	**0.028**	**0.667**	**2.617**
	L_2	0.038	0.580	4.571
(e)	Huber	**0.039**	**0.570**	**4.832**
	L_2	1.083	0.037	4742
(f)	Huber	**0.020**	**0.757**	**1.238**
	L_2	0.024	0.711	1.862
(g)	Huber	**0.068**	**0.399**	**14.68**
	L_2	1.825	0.021	12451
(h)	Huber	**0.025**	**0.704**	**1.920**
	L_2	0.494	0.063	791.1

Table I: Simulation results for synthetic data with eight error distributions: (a) No error; (b) $\mathcal{N}(0,2)$; (c) $\mathcal{N}(0,10)$; (d) $\log\mathcal{N}(0,1)$; (e) Cauchy; (f) $t(3)$; (g) Pareto$(1,1)$; (h) $0.9\mathcal{N}(0,1)+0.1\mathcal{N}(100,1)$.

Error	Huber	L_2
(e)	99.4%	80.0%
(g)	100.0%	62.8%

Table II: Percentage of the replications with the algorithm converging and two error distributions: (e) Cauchy; (g) Pareto$(1,1)$.

with Huber loss is comparable to that with L_2 loss in special cases (error a-c) and significantly better in general cases (error d-h).

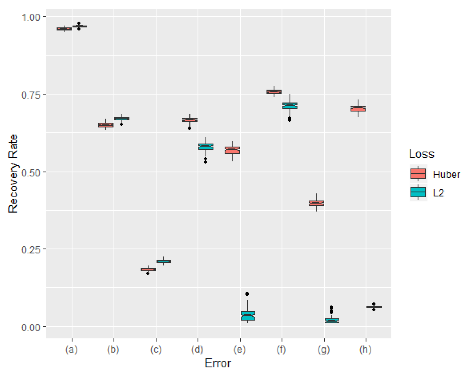

Figure 2: Box plots of recovery rates under Huber (left) and L_2 (right) loss functions with eight error distributions: (a) No error; (b) $\mathcal{N}(0,2)$; (c) $\mathcal{N}(0,10)$; (d) $\log\mathcal{N}(0,1)$; (e) Cauchy; (f) $t(3)$; (g) Pareto$(1,1)$; (h) $0.9\mathcal{N}(0,1)+0.1\mathcal{N}(100,1)$.

B. Real data

We further demonstrate the efficiency and robustness of our low-rank matrix recovery algorithm by applying it to two real examples.

The first is a chlorine concentration dataset, available in [20]. The dataset contains a matrix of chlorine concentration levels collected in a water distribution system. The observations at each row are collected at a certain location, and the columns correspond to observations at consecutive time points.

The second is compressed sensing of the MIT logo [14], [21], which is a 38×72 gray-scale image, see Figure 3.

Error	Loss	RE	RR
(a)	Huber	**0.095**	**0.130**
	L_2	0.102	0.121
(c)	Huber	0.448	0.032
	L_2	**0.432**	**0.032**
(i)	Huber	**0.097**	**0.132**
	L_2	12.587	0.002
(j)	Huber	0.114	0.109
	L_2	N.A.	

Table III: Performance of Chlorine concentration recovery with four error distributions: (a) No error; (c) $\mathcal{N}(0,10)$; (i) 1% outliers with outliers sampled from $\mathcal{N}(0,10\|X^*\|_F)$; (j) 5% outliers with outliers sampled from $\mathcal{N}(0,10\|X^*\|_F)$.

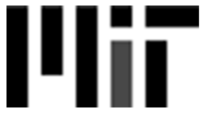

Figure 3: MIT logo

1) Chlorine concentration recovery: We use a sub-matrix of size 120×180 as our ground truth matrix X^* and generated $p=4200$ sensing matrices and linear measurements according to (2). We set $r=6$ since the rank-6 truncated SVD of X^* achieves a relative low error of 0.069. In this experiment, as in [22], we use outliers to replace measurements b_is, and the outliers are sampled from $\mathcal{N}(0,10\|X^*\|_F)$. The replacement happens with probability 1% or 5%, which is denoted by error distributions (i) and (j), respectively. As a comparison, we also consider the cases with no error and the error distribution of $\mathcal{N}(0,10)$, which correspond to error distributions (a) and (c) in synthetic data, respectively.

Table III gives the sensing performance under different levels of outliers. The recovery rates are all low, and this possibly is due to the fact that X^* has many entries close to zero. As a result, the relative error might be an more informative metric here. The Huber and L_2 loss have comparable performance when there is no error (see also Figure 4) or normal error of a moderate scale (see also Figure 5). Replacing 1% or 5% of the measurements with $\mathcal{N}(0,10\|X^*\|_F)$ outliers hardly affects the recovery if Huber loss is used. However, for L_2 loss, the result is severely affected by outliers. When there is 5% outliers, the algorithm is either too slow or diverges, thus the result is not available (N.A.).

Figure 4-7 visualize the element-wise comparison between the true matrix and recovery results by plotting the second row of the matrices in a plot. Figure 4 shows that the recovery is good when there is no error. Figure 5 shows that both procedures are affected by the $\mathcal{N}(0,10)$ noise, but the reconstructed values still share the same pattern with the true values. When there are 1% outliers, using the L_2 loss results in large fluctuations and could not recover the matrix (see Figure

6). In contrast, using Huber loss allows us to recover the true X^* even when 5% of the observations are outliers, see Figure 7.

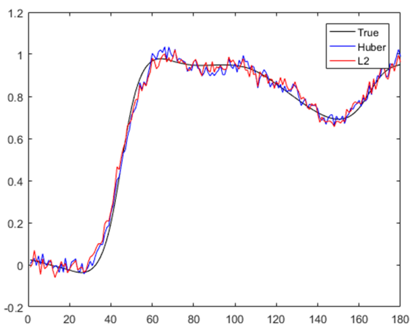

Figure 4: Chlorine concentration recovery: no error

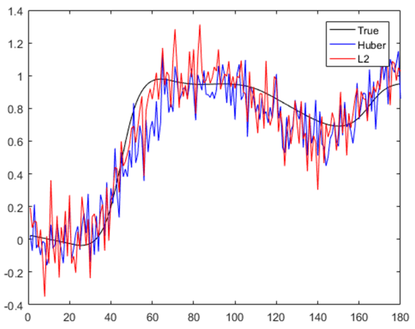

Figure 5: Chlorine concentration recovery: $\mathcal{N}(0, 10)$ error

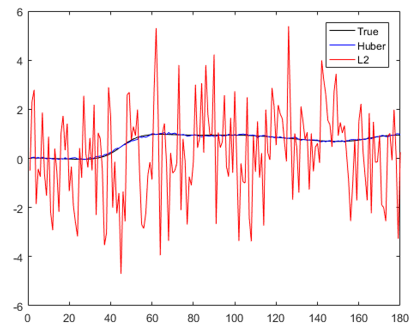

Figure 6: Chlorine concentration recovery: 1% outlier

2) Compressed sensing of MIT logo: We use the gray-scale logo as the ground truth matrix, which can be well-approximated by a rank-4 matrix (RE = 0.0194 by truncated SVD). We set $m = 38, n = 72, r = 4$, and take $p = 1200$ measurements using (2). Figure 8 visually compares the sensing results under L_2 loss and Huber loss.

If Huber loss is used, the MIT logo can be recovered in all 4 scenarios examined. Recall that the $\mathcal{N}(0, 10)$ noise is the most adverse case for Huber loss among the eight error distributions tested using synthetic dataset. In this MIT logo experiment, the reconstructed image is still recognizable when

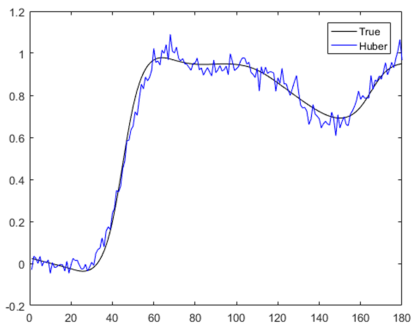

Figure 7: Chlorine concentration recovery: 5% outlier

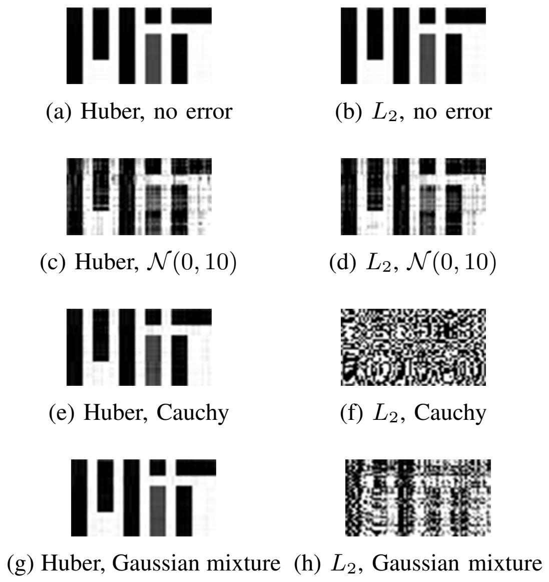
(a) Huber, no error
(b) L_2, no error
(c) Huber, $\mathcal{N}(0, 10)$
(d) L_2, $\mathcal{N}(0, 10)$
(e) Huber, Cauchy
(f) L_2, Cauchy
(g) Huber, Gaussian mixture
(h) L_2, Gaussian mixture

Figure 8: Compressed sensing of MIT logo. (a) No error; (b) $\mathcal{N}(0, 2)$; (c) $\mathcal{N}(0, 10)$; (d) $\log\mathcal{N}(0, 1)$; (e) Cauchy; (f) $t(3)$; (g) Pareto$(1, 1)$; (h) $0.9\mathcal{N}(0, 1) + 0.1\mathcal{N}(100, 1)$.

error follows $\mathcal{N}(0, 10)$. For the other 3 types of errors, the recovery error is hardly noticeable.

On the other hand, using L_2 loss allows us to recover the logo when there is no or Normal error. Under Normal error, L_2 loss has a slight advantage over Huber loss, as the outcome is a bit clearer. The reconstructed image is very blur when Gaussian mixture error presents, and can not be recognized under Cauchy error. Numerical metrics are supplied in Table IV.

VII. Conclusion

This paper proposes to use the loss function of an M-estimation in the matrix factorization for low-rank matrix recovery problem. The M-estimation is very general in statistics, and it includes the famous L_1, L_2 and quantile loss functions as special cases. Especially the Huber M-estimation enjoys the robustness to outliers, while the loss function is smooth. The alternating minimization method is hence applied and, at each step, Nesterov's momentum method is employed to accelerate

Error	Loss	RE	RR
(a)	Huber	**0.024**	**0.547**
	L_2	0.026	0.538
(c)	Huber	0.275	0.105
	L_2	**0.235**	**0.113**
(e)	Huber	**0.063**	**0.380**
	L_2	9.795	0.002
(h)	Huber	**0.043**	**0.487**
	L_2	0.951	0.035

Table IV: Recovery of MIT logo with four error distributions: (a) No error; (c) $\mathcal{N}(0,10)$; (e) Cauchy; (h) $0.9\mathcal{N}(0,1)+0.1\mathcal{N}(100,1)$.

the algorithm. The simulation experiments on both synthetic data and real data demonstrate that the Huber M loss function will result in a more robust method for skewed and/or heavy-tailed data.

Appendix

Appendix A: Proof Sketch of Theorem 1

The proof idea for the convergence is the same nature as the combined analysis of [13], [23] and [17], thus we only provide the proof road map.

Below only provide theorems when $\mathcal{M}$ is smooth, for nonsmooth $\mathcal{M}$, will provide proof later. Assumptions for function $\mathcal{M}(\cdot)$:

1 This assumption is ξ-strongly convex assumption [24]: $\mathcal{M}(\mathcal{A}(Y)-b)-\mathcal{M}(\mathcal{A}(X)-b)\geq\langle\nabla\mathcal{M}(\mathcal{A}(X)-b),Y-X\rangle+\xi\|\mathcal{A}(X)-\mathcal{A}(Y)\|_2^2$.

2 This assumption basically means that $\mathcal{M}(\mathcal{A}(Y)-b)-\mathcal{M}(\mathcal{A}(X)-b)\leq\langle\nabla\mathcal{M}(\mathcal{A}(X)-b),Y-X\rangle+\eta\|\mathcal{A}(X)-\mathcal{A}(Y)\|_2^2$.

3 Without loss of generality, assume X^* is the optimal matrix, then $\mathcal{M}(\mathcal{A}(X^*)-b)=0$, and $\mathcal{M}(X)\geq 0$.

4 Assume that $\mathcal{M}$ also defines a matrix norm when act on a matrix X, and $c_1\|X\|_2^2\leq\mathcal{M}(X)\leq c_2\|X\|_2^2$.

Remark: Assumption 1-2 defines the condition number of function $\mathcal{M}(\cdot)$, similar assumptions also appears in [25]. Usually $\kappa=\eta/\xi$ is called the condition number of function f [26].

Lemma A1. *(Restricted Isometry Property (RIP)) A linear map $\mathcal{A}$ satisfies the r-RIP with constant δ_r, if*

$$(1-\delta_r)\|X\|_F^2\leq\|\mathcal{A}(X)\|_2^2\leq(1+\delta_r)\|X\|_F^2$$

is satisfied for all matrices $X\in\mathbb{R}^{m\times n}$ of rank at most r.

Lemma A2. *[27] Let $\mathcal{A}$ satisfy 2r-RIP with constant δ_{2r}. Then for all matrices X,Y of rank at most r, we have*

$$|\langle\mathcal{A}(X),\mathcal{A}(Y)\rangle-\langle X,Y\rangle|\leq\delta_{2r}\|X\|_F\|Y\|_F.$$

The next lemma characterizes the convergence rate of initialization procedure:

Lemma A3. *[28] Let $X\in\mathbb{R}^{m\times n}$ be an arbitrary matrix of rank r. Also let $b=\mathcal{A}(X)\in\mathbb{R}^p$ be p linear measurements. Consider the iterative updates*

$$Y^{t+1}\leftarrow\mathcal{P}_r\left(Y^t-\xi_t\nabla_X\mathcal{M}(\mathcal{A}(Y^t)-b)\right).$$

where Y^i are $m\times n$ matrices. Then

$$\|Y^t-X\|_F\leq\psi(\mathcal{A})^t\|Y^0-X\|_F.$$

holds. Here $\psi(\mathcal{A})$ is defined as

$$\psi(\mathcal{A})=2sup_{\|X\|_F=\|Y\|_F=1,rank(X)\leq 2r,rank(Y)\leq 2r}$$
$$|\langle\mathcal{A}(X),\mathcal{A}(Y)\rangle-\langle X,Y\rangle|.$$

We can prove this lemma by using the results of Theorem A1.

First we will prove that the initialization procedure is indeed converge to the true value of X.

A. proof of equation (8) in Theorem 1

Lemma A4 (Lemma for initialization)**.** *Assume that $\xi\frac{1+\delta_{2k}}{1-\delta_{2k}}-\eta>0$. Denote $\tilde{\mathcal{M}}(X)=\mathcal{M}(\mathcal{A}(X)-b)$. Let X^* be an optimal solution and let X^t be the iterate obtained by algorithm 1 at t-th iteration. Then*

$$\tilde{\mathcal{M}}(X^{t+1})\leq\tilde{\mathcal{M}}(X^t)+\left(\xi\frac{1+\delta_{2k}}{1-\delta_{2k}}-\eta\right)\|\mathcal{A}(X^*-X^t)\|_2^2.$$

Proof. From assumption, we have

$$\begin{aligned}&\tilde{\mathcal{M}}(X^{t+1})-\tilde{\mathcal{M}}(X^t)\\&\leq\langle\nabla\tilde{\mathcal{M}}(X^t),X^{t+1}-X^t\rangle+\xi\|\mathcal{A}(X^{t+1})-\mathcal{A}(X^t)\|_2^2\\&\leq\langle\nabla\tilde{\mathcal{M}}(X^t),X^{t+1}-X^t\rangle+\xi(1+\delta_{2k})\|X^{t+1}-X^t\|_F^2\end{aligned}$$

where the last inequality comes from RIP. Let $Y^{t+1}=X^t-\frac{1}{2\xi(1+\delta_{2k})}\nabla\tilde{\mathcal{M}}(X^t)$, and

$$f_t(X)=\langle\nabla\tilde{\mathcal{M}}(X^t),X-X^t\rangle+\xi(1+\delta_{2k})\|X-X^t\|_F^2.$$

Then

$$\begin{aligned}&f_t(X)=\xi(1+\delta_{2k})\\&\left[\|X-Y^{t+1}\|_F^2-\frac{1}{4\xi^2(1+\delta_{2k})^2}\|\nabla\tilde{\mathcal{M}}(X^t)\|_F^2\right]\end{aligned}$$

By definition, $\mathcal{P}_k(Y^{t+1})=X^{t+1}$, then $f_t(X^{t+1})\leq f_t(X^*)$. Thus

$$\begin{aligned}&\tilde{\mathcal{M}}(X^{t+1})-\tilde{\mathcal{M}}(X^t)\leq f_t(X^{t+1})\leq f_t(X^*)\\=\;&\langle\nabla\tilde{\mathcal{M}}(X^t),X^*-X^t\rangle+\xi(1+\delta_{2k})\|X^*-X^t\|_F^2\\\leq\;&\nabla\tilde{\mathcal{M}}(X^t),X^*-X^t\rangle+\xi\frac{1+\delta_{2k}}{1-\delta_{2k}}\|\mathcal{A}(X^*)-\mathcal{A}(X^t)\|_F^2\\\leq\;&\nabla\tilde{\mathcal{M}}(X^t),X^*-X^t\rangle+\eta\|\mathcal{A}(X^*)-\mathcal{A}(X^t)\|_F^2\\&+\left(\xi\frac{1+\delta_{2k}}{1-\delta_{2k}}-\eta\right)\|\mathcal{A}(X^*)-\mathcal{A}(X^t)\|_F^2\\\leq\;&\tilde{\mathcal{M}}(X^*)-\tilde{\mathcal{M}}(X^t)\\&+\left(\xi\frac{1+\delta_{2k}}{1-\delta_{2k}}-\eta\right)\|\mathcal{A}(X^*)-\mathcal{A}(X^t)\|_F^2\end{aligned}$$

□

Theorem A1. *Let $b=\mathcal{A}(X^*)+e$ for rank k matrix X^* and an error vector $e\in\mathbb{R}^p$, $D=\frac{1}{C^2}+\left(\xi\frac{1+\delta_{2k}}{1-\delta_{2k}}-\eta\right)\left(\frac{2}{C^2}+\sqrt{\frac{2}{c_1}}\frac{1}{C}+\frac{1}{c_1}\right)$. Then, under the assumption that $D<1$, algorithm 1 with step size $\eta_t=\frac{1}{2\xi(1+\delta_{2k})}$*

outputs a matrix X of rank at most k such that $\mathcal{M}(\mathcal{A}(X)-b) \le (C^2+\varepsilon)\frac{\|e\|^2}{2}$, $\varepsilon \ge 0$, in at most $\left\lceil \frac{1}{\log D}\log\frac{(C^2+\varepsilon)\|e\|^2}{2c_2\|b\|_2^2}\right\rceil$ iterations.

Proof. Let the current solution X^t satisfy $\mathcal{M}(X^t) \ge \frac{C^2\|e\|^2}{2}$, by lemma A4 and $b - \mathcal{A}(X^*) = e$, we have

$$\begin{aligned}
\mathcal{M}(X^{t+1}) &\le \frac{\|e\|^2}{2} + \left(\xi\frac{1+\delta_{2k}}{1-\delta_{2k}} - \eta\right)\|b-\mathcal{A}(X^t)-e\|^2 \\
&\le \frac{\|e\|^2}{2} + \left(\xi\frac{1+\delta_{2k}}{1-\delta_{2k}} - \eta\right)\big(\|b-\mathcal{A}(X^t)\|^2 \\
&\quad -2e^\top(b-\mathcal{A}(X^t)) + \|e\|^2\big) \\
&\le \frac{\mathcal{M}(X^t)}{C^2} + \left(\xi\frac{1+\delta_{2k}}{1-\delta_{2k}} - \eta\right)\Bigg(\frac{2\mathcal{M}(X^t)}{C^2} \\
&\quad +\frac{\mathcal{M}(X^t)}{c_1} + \frac{\sqrt{2}\mathcal{M}(X^t)}{C\sqrt{c_1}} \\
&= D\mathcal{M}(X^t).
\end{aligned}$$

Since $D < 1$, combine the fact that $\mathcal{M}(X^0) \le c_2\|b\|^2$, by taking $t = \left\lceil \frac{1}{\log D}\log\frac{(C^2+\varepsilon)\|e\|^2}{2c_2\|b\|_2^2}\right\rceil$, we complete the proof. □

The following lemma was adapted from [13]:

Lemma A5. *Let $X_1, X_2 \in \mathbb{R}^{m\times n}$ be two rank r matrices with SVD decomposition $X_1 = U_1^\top\Sigma_1V_1$, $X_2 = U_2^\top\Sigma_2V_2$, for $l = 1,2$, define $X_l = U_l^\top\Sigma_l^{1/2} \in \mathbb{R}^{m\times r}$, $Y_l = V_l^\top\Sigma_l^{1/2} \in \mathbb{R}^{n\times r}$. Assume X_1, X_2 obey $\|X_2 - X_1\| \le \frac{1}{2}\sigma_r(X_1)$. Then:*

$$dist^2\left(\begin{bmatrix} X_2 \\ Y_2 \end{bmatrix}, \begin{bmatrix} X_1 \\ Y_1 \end{bmatrix}\right) \le \frac{2}{\sqrt{2}-1}\frac{\|X_2-X_1\|_F^2}{\sigma_r(X_1)}.$$

Combine lemma A1, A2 and A5, follow the similar route of [13], we can prove that using more than $3\log(\sqrt{r}\kappa)+5$ iterations of initialization algorithm 1, we can obtain

$$\text{dist}\left(\begin{bmatrix} U_0 \\ V_0 \end{bmatrix}, \begin{bmatrix} U \\ V \end{bmatrix}\right) \le \frac{1}{4}\sigma_r(U).$$

Thus we finished the proof of convergence in the initialization procedure, next we will study the linear decay rate for each iteration for penalized object function.

Before study the theoretical properties, we first rearrange object function (7) by using the uplifting technique so that it's easy to simultaneously consider $\mathcal{M}$ and the regularization term, to see this, consider rank r matrix $X \in \mathbb{R}^{m\times n}$ with SVD decomposition $X = U^\top\Sigma V$, define Sym : $\mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ as

$$\text{Sym}(X) = \begin{bmatrix} 0_{m\times m} & X \\ X^\top & 0_{n\times n} \end{bmatrix}.$$

Given the block matrix $\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$ with $\mathbf{A}_{11} \in \mathbb{R}^{m\times m}$, $\mathbf{A}_{12} \in \mathbb{R}^{m\times n}$, $\mathbf{A}_{21} \in \mathbb{R}^{n\times m}$, $\mathbf{A}_{22} \in \mathbb{R}^{n\times n}$. Define $\mathcal{P}_{\text{diag}}(\mathbf{A}) = \begin{bmatrix} \mathbf{A}_{11} & 0_{m\times n} \\ 0_{n\times m} & \mathbf{A}_{22} \end{bmatrix}$, $\mathcal{P}_{\text{off}}(\mathbf{A}) = \begin{bmatrix} 0_{m\times m} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & 0_{n\times n} \end{bmatrix}$, define $\mathcal{B} : \mathbb{R}^{(m+n)\times(m+n)} \to \mathbb{R}^{(m+n)\times(m+n)}$ as the uplift version of $\mathcal{A}$ operator:

$$\mathcal{B}(X)_k = \langle B_k, X\rangle, \text{ where } B_k = \text{Sym}(X).$$

Define $W = [U^\top; V^\top]$, Then as a result, we can rewrite object function (7) as:

$$\begin{aligned}
g(W) &:= g(U,V) \\
&= \mathcal{M}(b - \mathcal{A}(U^\top V)) + \lambda\|UU^\top - VV^\top\|_F^2 \\
&= \frac{1}{2}\mathcal{M}\left(\mathcal{B}(\text{Sym}(U^\top V)) - \text{Sym}(X)\right) \qquad (10) \\
&\quad + \frac{1}{2}\lambda\|\text{Sym}(U^\top V) - \text{Sym}(X)\|_F^2.
\end{aligned}$$

From equation (10) we can see that actually the non-penalize part and penalize part have similar structure.

As a result, although we made the assumption 1,2,4 on function $\mathcal{M}$, we can see from equation (10) that after adding the penalization term, the penalized object function still retains similar property, as for assumption 3, we can also use some location transform techniques to make the penalized object function satisfies this assumption, as a result, it does not make so much difference whether we deal with penalized object function or un-penalized object function.

Then similar to [13], the alternating minimization incorporate with Nesterov's momentum algorithm with respect to U and V (sub vector of W), respectively, are actually can be written as the NAG algorithm applied to $g(W)$ with respect to W.

For the convergence analysis of Nesterov's momentum algorithm, we employ follow lemma, which is a theorem in [17]:

Lemma A6. *For minimization problem $\min_{x\in\mathcal{X}} f(x)$, where x is a vector, using the Lyapunov function*

$$\tilde{V}_k = f(y_k) + \xi\|z_k - x^*\|^2$$

it can be shown

$$\tilde{V}_{k+1} - \tilde{V}_k = -\tau_k\tilde{V}_k + \varepsilon_{k+1} \qquad (11)$$

where the error is expressed as

$$\varepsilon_{k+1} = \left(\frac{\tau_k^2}{4\xi}\right)\|\nabla f(x_k)\|^2 + \left(\tau_k\eta - \frac{\xi}{\tau_k}\right)\|x_k - y_k\|^2,$$

τ_k is the step size in Nesterov's momentum algorithm, usually equals $1/\sqrt{\kappa}$, $y_{k+1} = x_k - \frac{1}{2\eta}\nabla f(x_k)$, $x_{k+1} = \frac{1}{1+\tau_k}y_k + \frac{\tau_k}{1+\tau_k}z_k$, $z_{k+1} = z_k + \tau_k\left(x_{k+1} - z_k - \frac{1}{2\xi}\nabla f(x+k+1)\right)$.

Assume that $\tau_0 = 0$, $\tau_1 = \tau_2 = \cdots = \tilde{\tau}$, and $\varepsilon_1, \cdots, \varepsilon_{k+1}$ has a common upper bound $\tilde{\varepsilon}$, then (11) implies:

$$\begin{aligned}
|\tilde{V}_{k+1}| &= |(1-\tilde{\tau})^{k+1}\tilde{V}_0 + \sum_{i=1}^{k+1}(1-\tilde{\tau})^{i-1}\varepsilon_{k+2-i}| \\
&\le (1-\tilde{\tau})^{k+1}|\tilde{V}_0| + \frac{\tilde{\varepsilon} - \tilde{\varepsilon}(1-\tilde{\tau})^k}{\tilde{\tau}}
\end{aligned}$$

Substitute x_k with $W_{k+1} = [U^{k+1}, V^{k+1}]^\top$, f with g, if we want to deal with the convergence analysis with respect to $W_t - W^*$ and $W_0 - W^*$, we need to handle two parts, the first part is the error part with respect to $\tilde{\varepsilon}$, this can be solved by choosing initial estimate close to true value, as a result $\|\nabla f(x_1)\|$ can be arbitrary close to 0. For the sake of notation simplicity, assume that $\frac{\tilde{\varepsilon}-\tilde{\varepsilon}(1-\tilde{\tau})^k}{\tilde{\tau}} \leq \varepsilon^\dagger$. Since $\tilde{V}_k$ still satisfies assumption 1 and 2, without loss of generality, assume the corresponding parameter are $\tilde{\xi}$ and $\tilde{\eta}$.

In the next, we want to seek the relation of $W_t - W^*$ with $\tilde{V}_t$. This will involve the assumption 1 and 2 as well as lemma A1. With a rough handle of the gradient part in assumption 1 and 2, we can obtain

$$\tilde{\xi}\|\mathcal{A}(W_t) - \mathcal{A}(W^*)\|_2^2 \leq (1-\tilde{\tau})^t \tilde{\mu}\|\mathcal{A}(W_0) - \mathcal{A}(W^*)\|_2^2 + \tilde{\varepsilon}$$

Notice that $\tilde{\varepsilon}$ can be made arbitrary small so that

$$\tilde{\xi}\|\mathcal{A}(W_t) - \mathcal{A}(W^*)\|_2^2 \leq (1-\tilde{\tau}_1)^t \tilde{\mu}\|\mathcal{A}(W_0) - \mathcal{A}(W^*)\|_2^2$$

and $1 - \tilde{\tau}_1$ still larger than 0 smaller than 1. Employ the Restricted Isometry Property,

$$\tilde{\xi}(1-\delta_r)\|W_t - W^*\|_2^2 \leq (1-\tilde{\tau}_1)^t \tilde{\mu}(1+\delta_r)\|W_0 - W^*\|_2^2$$

Thus

$$\begin{aligned} &\text{dist}\left(\begin{bmatrix} U_t \\ V_t \end{bmatrix}, \begin{bmatrix} U \\ V \end{bmatrix}\right) \\ &\leq (1-\tilde{\tau}_1)^t \frac{\tilde{\mu}}{\tilde{\xi}} \frac{1+\delta_r}{1-\delta_r} \text{dist}\left(\begin{bmatrix} U_0 \\ V_0 \end{bmatrix}, \begin{bmatrix} U \\ V \end{bmatrix}\right) \\ &\leq \frac{1}{4}(1-\tilde{\tau}_1)^t \frac{\tilde{\mu}}{\tilde{\xi}} \frac{1+\delta_r}{1-\delta_r} \sigma_r(U) \end{aligned}$$

Theorem 1 is proved.

Remark on equation (10): From (10), we provide a guideline with respect to the selection of λ compared with [13], by combine (10) and assumption 4.

Appendix B: Proof Sketch of Theorem 2

Proof. Employ Theorem 1 in [29] and Lemma 3 in [30] we know that

$$\mathcal{M}_\pi^\lambda(b - \mathcal{A}(U^{\pi*\top} V^{\pi*})) \to \mathcal{M}^\lambda(b - \mathcal{A}(U^{*\top} V^*))$$

Giving the fact that $\mathcal{M}_\pi^\lambda$ and $\mathcal{M}^\lambda$ are convex, as well as the restricted isometry property, we finish the proof of Theorem 2. □

References

[1] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, no. 8, pp. 30–37, 2009.

[2] Z. Lin, C. Xu, and H. Zha, "Robust matrix factorization by majorization minimization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 1, pp. 208–220, 2018.

[3] R. Basri, D. Jacobs, and I. Kemelmacher, "Photometric stereo with general, unknown lighting," *International Journal of Computer Vision*, vol. 72, no. 3, pp. 239–257, 2007.

[4] M. A. Davenport and J. Romberg, "An overview of low-rank matrix recovery from incomplete observations," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 608–622, 2016.

[5] D. Bindel, "Matrix factorizations for computer network tomography," in *Householder Symposium XVIII on Numerical Linear Algebra*, p. 27, 2011.

[6] S. C. Ponz, *Machine Learning based Models for Matrix Factorization*. PhD thesis, 2017.

[7] R. Zhu, D. Niu, L. Kong, and Z. Li, "Expectile matrix factorization for skewed data analysis," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[8] R. He, T. Tan, and L. Wang, "Robust recovery of corrupted low-rankmatrix by implicit regularizers," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 4, pp. 770–783, 2014.

[9] J. Tukey, "A survey of sampling from contaminated distributions," in *Contributions to probability and statistics* (I. Olkin, ed.), pp. 448–485, Stanford University Press, 1960.

[10] F. Hampel, *Contribution to the theory of robust estimation*. Phd thesis, University of California, Berkeley, 1968.

[11] P. J. Huber and E. M. Ronchetti, *Robust Statistics*. Wiley, 2009.

[12] Y. Nesterov, "Smooth minimization of non-smooth functions," *Mathematical programming*, vol. 103, no. 1, pp. 127–152, 2005.

[13] S. Tu, R. Boczar, M. Simchowitz, M. Soltanolkotabi, and B. Recht, "Low-rank solutions of linear matrix equations via procrustes flow," in *International Conference on Machine Learning*, pp. 964–973, 2016.

[14] P. Jain, R. Meka, and I. S. Dhillon, "Guaranteed rank minimization via singular value projection," in *Advances in Neural Information Processing Systems*, pp. 937–945, 2010.

[15] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*, pp. 1139–1147, 2013.

[16] J. Zhang and I. Mitliagkas, "Yellowfin and the art of momentum tuning," *arXiv preprint arXiv:1706.03471*, 2017.

[17] R. Brooks, "Convergence analysis of deterministic and stochastic methods for convex optimization," Master's thesis, University of Waterloo, 2017.

[18] A. Y. Aravkin, A. Kambadur, A. C. Lozano, and R. Luss, "Sparse quantile huber regression for efficient and robust estimation," *arXiv preprint arXiv:1402.4624*, 2014.

[19] Z. Yang, Y. Zhang, W. Yan, Y. Xiang, and S. Xie, "A fast non-smooth nonnegative matrix factorization for learning sparse representation," *IEEE access*, vol. 4, pp. 5161–5168, 2016.

[20] S. Papadimitriou, J. Sun, and C. Faloutsos, "Streaming pattern discovery in multiple time-series," in *Proceedings of the 31st international conference on Very large data bases*, pp. 697–708, VLDB Endowment, 2005.

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

[22] Y. Li, Y. Chi, H. Zhang, and Y. Liang, "Nonconvex low-rank matrix recovery with arbitrary outliers via median-truncated gradient descent," *arXiv preprint arXiv:1709.08114*, 2017.

[23] E. J. Candes, X. Li, and M. Soltanolkotabi, "Phase retrieval via wirtinger flow: Theory and algorithms," *IEEE Transactions on Information Theory*, vol. 61, no. 4, pp. 1985–2007, 2015.

[24] D. Park, A. Kyrillidis, C. Caramanis, and S. Sanghavi, "Finding low-rank solutions to matrix problems, efficiently and provably," *arXiv preprint arXiv:1606.03168*, 2016.

[25] W. Su, S. Boyd, and E. Candes, "A differential equation for modeling nesterov's accelerated gradient method: Theory and insights," in *Advances in Neural Information Processing Systems*, pp. 2510–2518, 2014.

[26] S. Bhojanapalli, A. Kyrillidis, and S. Sanghavi, "Dropping convexity for faster semi-definite optimization," in *Conference on Learning Theory*, pp. 530–582, 2016.

[27] E. J. Candes, "The restricted isometry property and its implications for compressed sensing," *Comptes rendus mathematique*, vol. 346, no. 9-10, pp. 589–592, 2008.

[28] S. Oymak, B. Recht, and M. Soltanolkotabi, "Sharp time–data tradeoffs for linear inverse problems," *IEEE Transactions on Information Theory*, vol. 64, no. 6, pp. 4129–4158, 2018.

[29] N. Guan, D. Tao, Z. Luo, and J. Shawe-Taylor, "Mahnmf: Manhattan non-negative matrix factorization," *arXiv preprint arXiv:1207.3438*, 2012.

[30] O. Fercoq and P. Richtárik, "Smooth minimization of nonsmooth functions with parallel coordinate descent methods," *arXiv preprint arXiv:1309.5885*, 2013.