A SMOOTHING MAJORIZATION METHOD FOR l_2-l_p MATRIX MINIMIZATION[*]

Yue Lu[†] Liwei Zhang[‡] and Jia Wu [§]

January 20, 2013

Abstract. We discuss the l_2-l_p (with $p \in (0,1)$) matrix minimization for recovering low rank matrix. A smoothing approach is developed for solving this non-smooth, non-Lipschitz and non-convex optimization problem, in which the smoothing parameter is used as a variable and a majorization method is adopted to solve the smoothing problem. The convergence theorem shows that any accumulation point of the sequence generated by the smoothing approach satisfies the necessary optimality condition for the l_2-l_p problem. As an application, we use the proposed smoothing majorization method to solve matrix completion problems. Numerical experiments indicate that our method is very efficient for obtaining the high quality recovery solution for matrix completion problems.

Key Words. Low rank problem, l_2-l_p minimization, majorization method, lower bound analysis, smoothing method.

1. Introduction

Recently, *low rank minimization problem* has attracted wide attention in many areas of application science and engineering, such as computer vision [1], Euclidean space embedding [2], machine learning [3, 4, 5] and so on. There are numerous models used to describe the low rank minimization problem, the simplest one of them is the *rank minimization problem*

$$\begin{array}{ll} \text{minimize} & \operatorname{rank}(X) \\ \text{subject to} & X \in \mathcal{C}, \end{array} \tag{1.1}$$

where $\mathcal{C}$ is a nonempty closed convex subset of $\mathcal{M}_{m\times n}$. The above problem (1.1) was considered by Fazel [6], in which the computational complexity of (1.1) is analyzed and it is proved to be an NP-hard problem. For solving Problem (1.1), Fazel et al. [6, 7] suggested to approximate the rank function by the nuclear norm and proposed the following convex optimization problem

$$\begin{array}{ll} \text{minimize} & \|X\|_* \\ \text{subject to} & X \in \mathcal{C}, \end{array} \tag{1.2}$$

[*] Supported by the National Natural Science Foundation of China under projects No. 11071029 and No. 91130007.

[†] Institute of Operations Research and Control Theory, School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, PR China. E-mail: jinjin403@sina.com

[‡] Institute of Operations Research and Control Theory, School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, PR China. E-mail: lwzhang@dlut.edu.cn

[§] Institute of Operations Research and Control Theory, School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, PR China. E-mail:wujia@dlut.edu.cn

where $\|X\|_*$ denotes the nuclear norm of X, which is defined as the sum of all the singular values of X. Many important problems can be formulated as (1.2), the most popular is *matrix completion problem*

$$\begin{array}{ll} \text{minimize} & \|X\|_* \\ \text{subject to} & X_{ij} = M_{ij},\ (i,j) \in \Omega, \end{array} \tag{1.3}$$

where Ω is an index set of the entries of M. Various algorithms have been designed to solve the matrix completion problem, such as singular value thresholding algorithm (SVT) [8], fixed-point continuation algotithm (FPCA) [9], accelerated proximal gradient algorithm [10] and alternating-direction-type algorithm [11].

If we define $\mathcal{P}_\Omega$ to denote the orthogonal projector onto the span of matrices vanishing outside of Ω so that the (i,j) entry of $\mathcal{P}_\Omega(X)$ is equal to X_{ij} if $(i,j) \in \Omega$ and zero otherwise, Problem (1.3) is written as

$$\begin{array}{ll} \text{minimize} & \|X\|_* \\ \text{subject to} & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M). \end{array} \tag{1.4}$$

Furthermore, if we use the general linear operator $\mathcal{A}$ and vector b to replace the orthogonal projector and $\mathcal{P}_\Omega(M)$ respectively, Problem (1.4) is generalized to the following form

$$\begin{array}{ll} \text{minimize} & \|X\|_* \\ \text{subject to} & \mathcal{A}(X) = b, \end{array}$$

which can be viewed as a convex relaxation of *affine rank minimization problem*

$$\begin{array}{ll} \text{minimize} & \operatorname{rank}(X) \\ \text{subject to} & \mathcal{A}(X) = b. \end{array}$$

Candès and Tao [12] and Donoho [13] show that the affine rank minimization problem under certain condition may be solved through the following convex problem

$$\begin{array}{ll} \text{minimize} & \|X\|_* \\ \text{subject to} & \|b - \mathcal{A}(X)\|_2 \leq \delta, \end{array} \tag{1.5}$$

where $\delta > 0$ evaluates the uncertainty about the observation b with noise. Instead of Problem (1.5), people often consider the problem as follows

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2 + \tau\|X\|_* \\ \text{subject to} & X \in \mathcal{M}_{m\times n}. \end{array} \tag{1.6}$$

The above problem (1.6) is called the *nuclear norm regularized linear least square problem*, which is regarded as the convex approximation to the problem

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2 + \tau \cdot \operatorname{rank}(X) \\ \text{subject to} & X \in \mathcal{M}_{m\times n}. \end{array} \tag{1.7}$$

In this paper, we will consider another approximation to Problem (1.7), which is the following l_2-l_p model

$$(\mathbf{P}) \qquad \left\{ \begin{array}{ll} \text{minimize} & F(X) := \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2 + \frac{\tau}{p}\|X\|_p^p \\ \text{subject to} & X \in \mathcal{M}_{m\times n}, \end{array} \right. \tag{1.8}$$

where $\mathcal{A}$ is a linear operator from $\mathcal{M}_{m\times n}$ to $\mathcal{R}^q$, which is given by

$$\mathcal{A}(X) := \left(\langle A_1, X\rangle, \langle A_2, X\rangle, \cdots, \langle A_q, X\rangle\right)^T,$$

where $A_i \in \mathcal{M}_{m\times n}(i = 1, 2, \cdots, q)$ and $b \in \mathcal{R}^q$. The function $\|\cdot\|_p^p$ $(0 < p < 1)$ is not a norm and it is defined as

$$\|X\|_p^p := \sum_{i=1}^{m} \sigma_i^p(X), \tag{1.9}$$

where $\sigma_i(X)(i = 1, 2, \cdots, m)$ are the singular values of X. It is easy to show that the function $\|X\|_p^p$ defined in (1.9) is an intermediate value between the rank function

$$\text{rank}(X) := \sum_{i=1}^{m} \sigma_i^0(X) = \text{the number of nonzero singular values.}$$

and the nuclear norm

$$\|X\|_* := \sum_{i=1}^{m} \sigma_i(X) = \text{the sum of all singular values.}$$

If let $m = n$ and X is a diagonal matrix, Problem (1.8) is reduce to

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|Ax - b\|_2^2 + \frac{\tau}{p}\|x\|_p^p \\ \text{subject to} & x \in \mathcal{R}^n, \end{array} \tag{1.10}$$

where x denotes the n dimensional vector whose components are the entries of diagonal matrix X and A is a $q \times n$ matrix. The term $\|x\|_p^p$ in (1.10), in some sense, characterizes the sparsity of the solutions of $Ax = b$. Problem (1.10) is extensively studied in recent years, for instance Chen el al. [14] gives the lower bound estimates of nonzero entries in solutions of (1.10) and [15] introduces the smoothing technique to tackle the non-convex, non-Lipschize regularization term $\|x\|_p^p$ and gives a SQP-type algorithm. Desides these, many numerical algorithms have been designed for recovering sparse vectors, see for example [16, 17, 18].

Back to Problem (1.8), a few authors have already made contributions. Mohan el al. [19] and Lai el al. [20] use the skill of iterative reweighted least squares (IRLS-p) and respectively combine with the assumption of $\tau-$Null space property ($\tau-$NSP) [19] and the certain restricted isometry properties (RIPs) [21] for the operator $\mathcal{A}$ to analyze the convergence of Problem (1.8). However, the lower bound analysis for the singular values at the optimal solution of Problem (1.8) is still unknown, which is an important motivation that stimulates us to study it.

This paper is organized as follows. In Section 2, we present the lower bound for the singular values at the local optimal solution of Problem (**P**) defined in (1.8). Next, the properties of the smoothing function for the objective function will be analyzed and the approximation of the smoothing model to Problem (**P**) is established in Section 3. In Section 4, we will design an algorithm for the smoothing model using the idea of majorization method. As an important application, we use the smoothing majorization method to solve a large number of matrix completion problems and report numerical results in Section 5. Finally, we presents some concluding remarks.

2. Lower Bound Analysis

Let $\mathcal{M}_{m\times n}$ be the space of all $m \times n$ matrices equipped with the standard trace inner product $\langle X, Y\rangle := \textbf{trace}(X^T Y)$ and its induced Frobenius norm $\|\cdot\|_F$. The operator norm of a given matrix is denoted by $\|\cdot\|$. Without loss of generality, we assume $m \leq n$ throughout this paper. Let $\mathcal{O}^m, \mathcal{O}^n$ denote the set of all m dimensional orthogonal matrices and the set of all n dimensional orthogonal matrices, respectively. Let X admit the following singular value decomposition (SVD):

$$X = U(X)[\textbf{Diag}\sigma(X)\ 0]V(X)^T. \tag{2.1}$$

where $U(X) \in \mathcal{O}^m$, $V(X) \in \mathcal{O}^n$. $\sigma(X) := (\sigma_1(X), \sigma_2(X), \cdots, \sigma_m(X))^T$ and $\sigma_1(X) \geq \sigma_2(X) \geq \cdots \geq \sigma_m(X) \geq 0$ to denote the singular values of X (counting multiplicity) being arranged in non-increasing order. And $\textbf{Diag}\sigma(X)$ is defined as

$$\textbf{Diag}\sigma(X) := \begin{pmatrix} \sigma_1(X) & & & \\ & \sigma_2(X) & & \\ & & \ddots & \\ & & & \sigma_m(X) \end{pmatrix}.$$

For simplicity, we denote $\mathbf{Diag}\sigma(X)$ by $\Sigma(X)$. Hence, (2.1) can be written as

$$X = U(X)[\Sigma(X)\ 0]V(X)^T. \tag{2.2}$$

The set of such matrices $(U(X), V(X))$ in (2.2) is denoted by $\mathcal{O}^{m,n}(X)$, i.e.

$$\mathcal{O}^{m,n}(X) := \left\{(U,V) \in \mathcal{O}^m \times \mathcal{O}^n : X = U[\Sigma(X)\ 0]V^T\right\}.$$

If X^* is a local minimizer of Problem (**P**) and $\text{rank}(X^*) = r$, then

$$\sigma_1(X^*) \geq \sigma_2(X^*) \geq \cdots \geq \sigma_r(X^*) > 0 = \sigma_{r+1}(X^*) = \cdots = \sigma_m(X^*).$$

Furthermore, X^* has the following SVD:

$$X^* = U^*[\Sigma^*\ 0](V^*)^T, \quad \Sigma^* = \mathbf{Diag}[(\sigma_1(X^*), \sigma_2(X^*), \cdots, \sigma_r(X^*), \underbrace{0, \cdots, 0}_{m-r})^T].$$

and

$$\mathcal{O}^{m,n}(X^*) = \left\{(U,V) : X^* = U[\Sigma^*\ 0]V^T\right\}. \tag{2.3}$$

Now we give an important lemma in this paper, from which a bridge between (**P**) and the auxiliary vector problem defined below will be built.

Lemma 2.1 *For any pair $(U^*, V^*) \in \mathcal{O}^{m,n}(X^*)$, where $\mathcal{O}^{m,n}(X^*)$ is defined by (2.3), the vector $z^* := (\sigma_1(X^*), \sigma_2(X^*), \cdots, \sigma_r(X^*))^T \in \mathcal{R}^r$ is a local minimizer of the following problem*

$$\begin{array}{ll} \textit{minimize} & \varphi(z) := F(U^*[\boldsymbol{Diag}(z, 0_{m-r})\ 0](V^*)^T) \\ \textit{subject to} & z \geq 0. \end{array} \tag{2.4}$$

Moreover, the first and second order necessary conditions of (2.4) at z^* are respectively given by

$$\nabla_x \varphi(z^*) = 0 \tag{2.5}$$

and

$$\nabla_x^2 \varphi(z^*) \succeq 0, \tag{2.6}$$

where $\nabla_x^2 \varphi(z^*) \succeq 0$ means that the matrix $\nabla_x^2 \varphi(z^*)$ is positive semidefinite.

Proof. Since $\sigma_i(X^*) > 0 (i = 1, 2, \cdots, r)$, z^* is a feasible point of the above problem (2.4). It is easy to show that

$$\begin{array}{rcl} \varphi(z^*) & = & F(U^*[\mathbf{Diag}(z^*, 0_{m-r})\ 0](V^*)^T) \\ & = & F(X^*) \\ & \leq & \min\left\{F(X) :\ X = U^*[\mathbf{Diag}(z, 0_{m-r})\ 0](V^*)^T,\ z \geq 0\right\} \\ & = & \min\left\{\varphi(z) : z \geq 0\right\}, \end{array}$$

where the third inequality uses the fact that X^* is a local minimizer of Problem (**P**), which proves the first part of the lemma.

Since z^* lies in the interior of the feasible set $\{z : z \geq 0\}$, the constraint $z \geq 0$ is inactive at z^*. Hence, the problem (2.4) becomes a unconstrained optimization problem at the neighborhood of z^*. By the first-order and second-order necessary optimality conditions for the unconstrained optimization problem, we obtain (2.5) and (2.6). □

Note that Lemma 2.1 gives the optimality condition for the auxiliary vector problem (2.4), which will be used to analyze the lower bound for the nonzero singular value of the local minimization point X^* of the low rank problem (**P**).

If we denote the matrix $\tilde{A}$ as follows

$$\tilde{A} := \begin{pmatrix} \tilde{a}_{11}^1 & \tilde{a}_{22}^1 & \cdots & \tilde{a}_{rr}^1 \\ \tilde{a}_{11}^2 & \tilde{a}_{22}^2 & \cdots & \tilde{a}_{rr}^2 \\ \vdots & \vdots & & \vdots \\ \tilde{a}_{11}^q & \tilde{a}_{22}^q & \cdots & \tilde{a}_{rr}^q \end{pmatrix},$$

where $\tilde{a}_{jj}^i := [(U^*)^T A_i V^*]_{jj}$ ($i = 1, 2, \cdots, q$ and $j = 1, 2, \cdots, r$). Problem (2.4) is expressed as a simple bound-constrained optimization problem (see Appendix A)

$$\begin{array}{ll} \text{minimize} & \varphi(z) = \frac{1}{2}\|\tilde{A}z - b\|_2^2 + \frac{\tau}{p}\sum_{i=1}^{r} z_i^p \\ \text{subject to} & z \geq 0. \end{array} \tag{2.7}$$

Note that from the original Problem (**P**) to Problem (2.7), the number of variables is tremendously decreased and the low rank problem is converted into the corresponding vector problem. Problem (2.7) is similar to Problem (1.10), but it has nonnegative variable constraints.

Corollary 2.2 *The first order necessary condition of Problem (2.7) at $z^* \in \mathcal{R}^r$ can be written as follows*

$$\tilde{A}^T(\tilde{A}z^* - b) + \tau(z^*)^{p-1} = 0, \tag{2.8}$$

where $(z^*)^{p-1} = \left(\sigma_1^{p-1}(X^*), \sigma_2^{p-1}(X^*), \cdots, \sigma_r^{p-1}(X^*)\right)^T$.

Proof. The conclusion is easy to obtain from Lemma 2.1 to obtain the conclusion. □

Finally, we use the equality (2.8) of Corollary 2.2 to give the lower bound for the singular values of the optimal solution X^* to Problem (**P**). Let X^0 be a given matrix.

Theorem 2.1 *Let* $\tilde{L} := \kappa\left(\frac{\tau}{\|\tilde{A}\|\sqrt{2F(X^0)}}\right)^{\frac{1}{1-p}}$ *and rank*$(X^*) = r$, *where X^* is a local minimizer of Problem* (**P**) *with $F(X^*) \leq F(X^0)$. Then*

$$\text{for any } i \in \{1, 2, \cdots, m\}, \quad \sigma_i(X^*) < \tilde{L} \quad \Rightarrow \quad \sigma_i(X^*) = 0,$$

where $\kappa := \left(\frac{1}{2}\right)^{\frac{1}{1-p}} < 1$. *Meanwhile, the rank of X^* is bounded by*

$$\text{rank}(X^*) \leq \min\left(m, \frac{pF(X^0)}{\tau(\tilde{L})^p}\right).$$

Proof. By the fact that $F(X^*) \leq F(X^0)$, we have

$$\begin{aligned} \|\tilde{A}^T\left(\mathcal{A}(X^*) - b\right)\|_2^2 &\leq \|\tilde{A}\|^2\left(\|\mathcal{A}(X^*) - b\|_2^2 + \frac{2\tau}{p}\|X\|_p^p\right) \\ &= 2\|\tilde{A}\|^2 F(X^*) \\ &\leq 2\|\tilde{A}\|^2 F(X^0). \end{aligned} \tag{2.9}$$

By the equation (2.8) in Corollary 2.2 and (2.9), we have

$$\begin{aligned} \tau\sigma_r^{p-1}(X^*) &\leq \tau\left(\sum_{j=1}^{r}\left[\sigma_j^{p-1}(X^*)\right]^2\right)^{\frac{1}{2}} \\ &= \tau\|(z^*)^{p-1}\|_2 \\ &= \|\tilde{A}^T(\tilde{A}z^* - b)\|_2 \\ &= \|\tilde{A}^T\left(\mathcal{A}(X^*) - b\right)\|_2 \\ &\leq \|\tilde{A}\|\sqrt{2F(X^0)}. \end{aligned}$$

Note that $p \in (0,1)$ implying

$$\sigma_r(X^*) \geq \left(\frac{\tau}{\|\tilde{A}\|\sqrt{2F(X^0)}}\right)^{\frac{1}{1-p}} > \kappa\left(\frac{\tau}{\|\tilde{A}\|\sqrt{2F(X^0)}}\right)^{\frac{1}{1-p}} = \tilde{L}.$$

Hence, all nonzero singular values of X^* are no less than $\tilde{L}$. That is, if $\sigma_i(X^*) < \tilde{L}$, we have $\sigma_i(X^*) = 0$.

Now we turn to the proof of the second part. By the definition of $F(X)$, it follows that

$$\frac{r\tau}{p}\tilde{L}^p \leq \frac{\tau}{p}\|X^*\|_p^p \leq \frac{1}{2}\|\mathcal{A}(X^*) - b\|_2^2 + \frac{\tau}{p}\|X^*\|_p^p = F(X^*) \leq F(X^0),$$

which implies

$$\text{rank}(X^*) = r \leq \frac{pF(X^0)}{\tau\tilde{L}^p} \quad \text{or} \quad \text{rank}(X^*) \leq \min\left(m, \frac{pF(X^0)}{\tau\tilde{L}^p}\right).$$

□

3. The Smoothing Function Method

3.1 Differential properties of the smoothing function

In this subsection, we focus on the properties of smoothing function for the objective function $F(X)$ defined in Problem (**P**). Before that, we firstly review some basics of spectral functions for symmetric matrices.

Let $\mathcal{S}^n$ denote the space of n dimensional symmetric matrices equipped with the inner product $\langle A, B\rangle := \textbf{trace}(AB)$ for $A, B \in \mathcal{S}^n$. Let $\mathcal{S}_+^n$ and $\mathcal{S}_{++}^n$ denote the cone of n dimensional positive semidefinite symmetric matrices and the cone of n dimensional positive definite symmetric matrices, respectively. A real-valued function $F = f \circ \lambda$ is called *spectral function*, if it is defined on a subset of $\mathcal{S}^n$ and invariant under orthogonal similarity transformation:

$$F(U^TMU) = F(M), \quad \forall\ M \in \mathcal{O}^n \text{ for } X \in \mathcal{S}^n.$$

The function $f : \mathcal{R}^n \to \mathcal{R}$ is a real-valued function and satisfies the following *symmetric* property

$$f(\nu) = f(P\nu), \quad \forall\ P \in \mathcal{P}^n, \nu \in \Re^n,$$

where $\mathcal{P}^n$ is the set of all $n \times n$ permutation matrices. A set Ω in $\mathcal{R}^n$ is *symmetric* if

$$P\Omega = \Omega, \quad \forall\ P \in \mathcal{P}^n.$$

There are some important spectral functions below:

(a) The nuclear norm of $M \in \mathcal{S}^n$, which is defined by

$$\|M\|_* := \sum_{i=1}^n |\lambda_i(M)|.$$

If we choose $f : \mathcal{R}^n \to \mathcal{R}$ as follows

$$f(\eta) := |\eta_1| + |\eta_2| + \cdots + |\eta_n|,$$

then

$$\|M\|_* = [f \circ \lambda](M).$$

(b) The spectral norm of $M \in \mathcal{S}^n$, which is defined by

$$\|M\|_2 := \max_{1\le i\le n} |\lambda_i(M)|.$$

If we choose $f : \mathcal{R}^n \to \mathcal{R}$ as follows

$$f(\eta) := \max_{1\le i\le n} |\eta_i|,$$

then

$$\|M\|_2 = [f \circ \lambda](M).$$

(c) The function $-\text{logdet}(M)$ ($M \in \mathcal{S}^n_{++}$), which is used to define the merit function in the semidefinite programming. If we choose $f : \Omega \to \mathcal{R}$ as follows

$$f(\eta) := -\sum_{i=1}^{n} \log \eta_i,$$

where $\Omega := \{\eta : \eta_i > 0,\ i = 1, 2, \cdots, n\}$, then

$$-\text{logdet}(M) = [f \circ \lambda](M).$$

There are many publications about the study of spectral functions. Lewis et al. [22, 23] investigate the explicit expressions of the gradient and the Hessian for spectral function $f \circ \lambda$. Qi et al. [24] study the semismoothness of spectral function. Li et al. [25] give the explicit expressions of second-order directional derivatives of spectral function when f is $C^{1,1}$ and $\nabla f(\cdot)$ is semidifferentiable at $\lambda(X)$. Sun and Sun [26] study the differentiability and the semismoothness of Löwner's operator and spectral function under the framework of Euclidean Jordan algebras.

In this paper, we define the symmetric function $f : \mathcal{R}^{m+n} \to \mathcal{R}$ by

$$f(\xi) := |\xi_1|^p + |\xi_2|^p + \cdots + |\xi_{m+n}|^p.$$

Let $\Xi : \mathcal{M}_{m\times n} \to \mathcal{S}^{m+n}$ be the linear operator defined by,

$$\Xi(X) := \begin{pmatrix} 0 & X \\ X^T & 0 \end{pmatrix}.$$

It follows from [27, 28] that $\Xi(X)$ has the following eigenvalue decomposition:

$$\Xi(X) = Q(X) \begin{pmatrix} \Sigma(X) & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -\Sigma(X)^{\uparrow} \end{pmatrix} Q(X)^T,$$

where

$$Q(X) := \frac{1}{\sqrt{2}} \begin{pmatrix} U(X) & 0 & U(X)^{\uparrow} \\ V_1(X) & \sqrt{2}V_2(X) & -V_1(X)^{\uparrow} \end{pmatrix},$$

and

$$V(X) := \left[\overbrace{V_1(X)}^{m} \ \vdots\ \overbrace{V_2(X)}^{n-m} \right], \quad U(X)^{\uparrow} := U(X) I_m^{\uparrow}, \quad V_1(X)^{\uparrow} := V_1(X) I_m^{\uparrow},$$

$$\Sigma(X)^{\uparrow} := \begin{pmatrix} \sigma_m(X) & & & \\ & \sigma_{m-1}(X) & & \\ & & \ddots & \\ & & & \sigma_1(X) \end{pmatrix}, \quad I_m^{\uparrow} := \begin{pmatrix} & & & & 1 \\ & & & 1 & \\ & & \iddots & & \\ & 1 & & & \\ 1 & & & & \end{pmatrix} \in \mathcal{S}^m.$$

It is easy to check that

$$\lambda_i(\Xi(X)) = \begin{cases} \sigma_i(X) & \text{if } i = 1, 2, \cdots, m, \\ 0 & \text{if } i = m+1, \cdots, n, \\ -\sigma_{m+n+1-i}(X) & \text{if } i = n+1, n+2, \cdots, n+m. \end{cases} \tag{3.1}$$

From (3.1), we have

$$\begin{aligned}[f \circ \lambda](\Xi(X)) &= \sum_{i=1}^{m} |\sigma_i(X)|^p + 0 \cdot (n-m) + \sum_{i=1}^{m} |-\sigma_i(X)|^p \\ &= 2\sum_{i=1}^{m} \sigma_i^p(X) \\ &= 2\|X\|_p^p.\end{aligned}$$

Hence, we can rewrite $F(X)$ as

$$F(X) = \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2 + \frac{\tau}{2p}[f \circ \lambda](\Xi(X)).$$

The following result taken from [22] provides a convenient tool for computing the gradient of spectral function.

Lemma 3.1 *Let $G \in \mathcal{S}^n$ and suppose the vector $\lambda(G)$ belongs to the domain of the symmetric function $f : \mathcal{R}^n \to \mathcal{R}$. Then f is differentiable at $\lambda(G)$ if and only if the spectral function $f \circ \lambda$ is differentiable at G. In this case the derivative of $f \circ \lambda$ at G is*

$$\mathrm{D}(f \circ \lambda)(G) = U(G)(\boldsymbol{Diag}\nabla f(\lambda(G)))U(G)^T,$$

for any orthogonal matrix $U(G)$ satisfying $G = U(G)(\boldsymbol{Diag}\lambda(G))U(G)^T$, where $\lambda(G)$ is a vector whose entries are the eigenvalues of the matrix G.

Let $F_1(X) := \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2$ and $F_2(X) := \frac{\tau}{2p}[f \circ \lambda](\Xi(X))$. Define $f_\epsilon(\eta) : \mathcal{R}\backslash\{0\} \times \mathcal{R}^{m+n} \to \mathcal{R}$ as follows

$$f_\epsilon(\eta) := (\eta_1^2 + \epsilon^2)^{\frac{p}{2}} + (\eta_2^2 + \epsilon^2)^{\frac{p}{2}} + \cdots + (\eta_{m+n}^2 + \epsilon^2)^{\frac{p}{2}},$$

then

$$\begin{aligned}[f_\epsilon \circ \lambda](\Xi(X)) &= \sum_{i=1}^{m}(\sigma_i^2(X) + \epsilon^2)^{\frac{p}{2}} + (n-m)|\epsilon|^p + \sum_{i=1}^{m}([-\sigma_i(X)]^2 + \epsilon^2)^{\frac{p}{2}} \\ &= 2\sum_{i=1}^{m}(\sigma_i^2(X) + \epsilon^2)^{\frac{p}{2}} + (n-m)|\epsilon|^p. \qquad (3.2)\end{aligned}$$

Set

$$F_2(\epsilon, X) := \frac{\tau}{2p}\left([f_\epsilon \circ \lambda](\Xi(X)) - (n-m)|\epsilon|^p\right). \qquad (3.3)$$

From (3.2) and (3.3), we have

$$\lim_{\epsilon \downarrow 0} F_2(\epsilon, X) = F_2(X).$$

For fixed $\epsilon \neq 0$, $f_\epsilon(\eta)$ is a continuous differentiable function of η. The gradient of $f_\epsilon(\eta)$ with respect to η is given by

$$\nabla_\eta f_\epsilon(\eta) = \left(p\eta_1(\eta_1^2 + \epsilon^2)^{\frac{p}{2}-1}, p\eta_2(\eta_2^2 + \epsilon^2)^{\frac{p}{2}-1}, \cdots, p\eta_{m+n}(\eta_{m+n}^2 + \epsilon^2)^{\frac{p}{2}-1}\right)^T. \qquad (3.4)$$

Lemma 3.2 *Let $F(\epsilon, X)$ be defined by*

$$F(\epsilon, X) := F_1(X) + F_2(\epsilon, X). \qquad (3.5)$$

Then $F(\epsilon, X)$ is a smoothing function for $F(X)$.

Proof. It follows from Lemma 3.1 that $[f_\epsilon \circ \lambda](\Xi(X))$ is a continuous differentiable function of $\Xi(X)$. Combining with the fact that $\Xi(X)$ is a linear operator of X, $F_2(\epsilon, X)$ is a continuous differentiable function of X. Because $F_1(X)$ is also a continuous differentiable function of X, we obtain the conclusion. □

It is easy to show that for any $H \in \mathcal{M}_{m\times n}$,

$$\mathrm{D}F_1(X)[H] = \mathcal{A}(H)^T \left(\mathcal{A}(X) - b\right), \tag{3.6}$$

where $\mathrm{D}F_1(\cdot)$ denotes the first order directional derivative of $F_1(\cdot)$.

Lemma 3.3 *For any $H \in \mathcal{M}_{m\times n}$ and fixed $\epsilon \neq 0$, let the function $F_2(\epsilon, X)$ be defined as (3.3). Then,*

$$\mathrm{D}F_2(\epsilon, X)[H] = \tau(\boldsymbol{diag}A)^T \begin{pmatrix} \sigma_1(X)(\sigma_1^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \sigma_2(X)(\sigma_2^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \vdots \\ \sigma_{m-1}(X)(\sigma_{m-1}^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \sigma_m(X)(\sigma_m^2(X)+\epsilon^2)^{\frac{p}{2}-1} \end{pmatrix},$$

where $A := U(X)^T H V_1(X)$ and $\boldsymbol{diag}A := (A_{11}, A_{22}, \cdots, A_{mm})^T$, A_{ii} is the (i,i) entry of A.

Proof. From Lemma 3.1 and (3.4), we obtain that

$$\begin{aligned} \mathrm{D}[f_\epsilon \circ \lambda](\Xi(X)) &= Q(X)\left[\mathbf{Diag}\nabla_{\Xi(X)} f_\epsilon(\lambda(\Xi(X)))\right] Q(X)^T \\ &= Q(X)\mathbf{Diag}\left[(w_1, w_2, \cdots, w_{m+n})^T\right] Q(X)^T, \end{aligned}$$

where the definition of w_i is given by

$$w_i = \begin{cases} p\sigma_i(X)(\sigma_i^2(X)+\epsilon^2)^{\frac{p}{2}-1} & \text{if } i = 1, 2, \cdots, m, \\ 0 & \text{if } i = m+1, m+2, \cdots, n, \\ -p\sigma_{m+n+1-i}(X)(\sigma_{m+n+1-i}^2(X)+\epsilon^2)^{\frac{p}{2}-1} & \text{if } i = n+1, n+2, \cdots, n+m. \end{cases}$$

Because $\Xi(X)$ is a linear operator of X, from Proposition 2.47 in [29], we obtain that for any $H \in \mathcal{M}_{m\times n}$ and fixed $\epsilon \neq 0$ that

$$\begin{aligned} \mathrm{D}F_2(\epsilon, X)[H] &= F_2'(\epsilon, X)[H] \\ &= \frac{\tau}{2p}[f_\epsilon \circ \lambda]'(\Xi(X); \Xi(H)) \\ &= \frac{\tau}{2p}\langle \nabla[f_\epsilon \circ \lambda](\Xi(X)), \Xi(H)\rangle \\ &= \frac{\tau}{2p}\langle [\mathbf{Diag}\nabla f_\epsilon(\lambda(\Xi(X)))], Q(X)^T \Xi(H) Q(X)\rangle, \end{aligned}$$

where the second and the third equations use Proposition 2.47 in [29] and the fact that $[f_\epsilon \circ \lambda](\Xi(X))$ is a continuous differentiable function of $\Xi(X)$, respectively. Therefore, (see Appendix B)

$$\mathrm{D}F_2(\epsilon, X)[H] = \tau(\mathbf{diag}A)^T \begin{pmatrix} \sigma_1(X)(\sigma_1^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \sigma_2(X)(\sigma_2^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \vdots \\ \sigma_{m-1}(X)(\sigma_{m-1}^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \sigma_m(X)(\sigma_m^2(X)+\epsilon^2)^{\frac{p}{2}-1} \end{pmatrix}.$$

□

Theorem 3.1 *For any $H \in \mathcal{M}_{m\times n}$, fixed $\epsilon \neq 0$ and let the function $F(\epsilon, X)$ be defined as (3.5). Then,*

$$\mathrm{D}F(\epsilon, X)[H] = \mathcal{A}(H)^T \left(\mathcal{A}(X) - b\right) + \tau(\boldsymbol{diag}A)^T \begin{pmatrix} \sigma_1(X)(\sigma_1^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \sigma_2(X)(\sigma_2^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \vdots \\ \sigma_{m-1}(X)(\sigma_{m-1}^2(X)+\epsilon^2)^{\frac{p}{2}-1} \\ \sigma_m(X)(\sigma_m^2(X)+\epsilon^2)^{\frac{p}{2}-1} \end{pmatrix},$$

where $A := U(X)^T H V_1(X)$.

Proof. It is obvious from (3.6) and Lemma 3.3. □

3.2 Approximation by the smoothing model

In this subsection, we establish the smoothing model to Problem (**P**) and analyze the corresponding first order necessary condition. We define the smoothing model ($\mathbf{P}_\epsilon$) as follows

$$(\mathbf{P}_\epsilon) \quad \begin{cases} \text{minimize} & F(\epsilon, X) \\ \text{subject to} & X \in \mathcal{M}_{m\times n}. \end{cases}$$

For simplify, we denote

$$\begin{aligned} \mathbf{Sol}(\mathbf{P}) \quad &:= \quad \text{the set of local minimizer of } (\mathbf{P}), \\ \mathbf{Sol}(\mathbf{P}_\epsilon) \quad &:= \quad \text{the set of local minimizer of } (\mathbf{P}_\epsilon). \end{aligned}$$

From Corollary 2.2, we have

$$\tilde{A}^T(\tilde{A}z^* - b) + \tau (z^*)^{p-1} = 0,$$

where $z^* := (\sigma_1(X^*), \sigma_2(X^*), \cdots, \sigma_r(X^*))^T \in \mathcal{R}^r$. Since $\sigma_i(X^*) > 0$ $(i = 1, 2, \cdots, r)$, we obtain that

$$(z^*)^T \tilde{A}^T(\tilde{A}z^* - b) + \tau \|X^*\|_p^p = 0.$$

As $F(X)$ is not Lipschitz continuous, it is not convenient to give a necessary condition for Problem (**P**) in terms of Clarke subdifferential notion, here we introduce the following definition of the first order necessary condition for Problem (**P**).

Definition 3.4 *For $X \in M_{m\times n}$ and $p \in (0, 1)$, X is said to satisfy the first order necessary condition of Problem* (**P**) *if*

$$\mathcal{A}(X)^T (\mathcal{A}(X) - b) + \tau \|X\|_p^p = 0. \tag{3.7}$$

Obviously, if X^* is a local minimizer of Problem (**P**), X^* satisfies the above conclusion (3.7).

Note that Problem ($\mathbf{P}_\epsilon$) is an unconstrained optimization problem. Let X_ϵ^* be a local minimizer of ($\mathbf{P}_\epsilon$). Then for any $H \in \mathcal{M}_{m\times n}$,

$$\mathrm{D}F(\epsilon, X_\epsilon^*)[H] = 0. \tag{3.8}$$

By the definitions of $F(X)$ and $F(\epsilon, X)$, we obtain that

$$0 \le F(\epsilon, X) - F(X) \le \frac{\tau m |\epsilon|^p}{p}. \tag{3.9}$$

The inequality (3.9) gives the bound of the difference between the original objective function $F(X)$ and the smoothing function $F(\epsilon, X)$, which also implies that $\lim_{\epsilon \downarrow 0} F(\epsilon, X) = F(X)$.

Let $\{X_{\epsilon_k}\}$ denote the sequence with $\epsilon_k \neq 0,\ k = 1, 2, \cdots$ and $\epsilon_k \to 0$ as $k \to \infty$.

Theorem 3.2 *We have the following conclusions:*

(1) *Let $\{X_{\epsilon_k}^*\}$ be a sequence of matrices satisfying (3.8) with $\epsilon = \epsilon_k$. Then any accumulation of $\{X_{\epsilon_k}^*\}$ satisfies the first order necessary condition of Problem* (**P**).

(2) *Let $\{X_{\epsilon_k}^*\}$ be a sequence of matrices being global minimizer of* ($\mathbf{P}_{\epsilon_k}$). *Then any accumulation of $\{X_{\epsilon_k}^*\}$ is the global minimizer of Problem* (**P**).

Proof. Let X^* be an accumulation point of $\{X^*_{\epsilon_k}\}$. Then $X^*_{\epsilon_k} \to X^*$ as $k \to \infty$.

(1) From (3.8), we have

$$\mathcal{A}(X^*_{\epsilon_k})^T \left(\mathcal{A}(X^*_{\epsilon_k}) - b\right) + \tau(\mathbf{diag} A_{\epsilon_k})^T \begin{pmatrix} \sigma_1(X^*_{\epsilon_k})(\sigma_1^2(X^*_{\epsilon_k}) + \epsilon_k^2)^{\frac{p}{2}-1} \\ \sigma_2(X^*_{\epsilon_k})(\sigma_2^2(X^*_{\epsilon_k}) + \epsilon_k^2)^{\frac{p}{2}-1} \\ \vdots \\ \sigma_{m-1}(X^*_{\epsilon_k})(\sigma_{m-1}^2(X^*_{\epsilon_k}) + \epsilon_k^2)^{\frac{p}{2}-1} \\ \sigma_m(X^*_{\epsilon_k})(\sigma_m^2(X^*_{\epsilon_k}) + \epsilon_k^2)^{\frac{p}{2}-1} \end{pmatrix} = 0, \tag{3.10}$$

where $A_{\epsilon_k} := U(X^*_{\epsilon_k})^T X^*_{\epsilon_k} V_1(X^*_{\epsilon_k})$. The SVD of $X^*_{\epsilon_k}$ is given by

$$X^*_{\epsilon_k} = U(X^*_{\epsilon_k})[\Sigma(X^*_{\epsilon_k})\ 0]V(X^*_{\epsilon_k})^T,$$

where

$$V(X^*_{\epsilon_k}) := \left[\overbrace{V_1(X^*_{\epsilon_k})}^{m} \vdots \overbrace{V_2(X^*_{\epsilon_k})}^{n-m}\right] \quad \text{and} \quad \Sigma(X^*_{\epsilon_k}) := \begin{pmatrix} \sigma_1(X^*_{\epsilon_k}) & & & \\ & \sigma_2(X^*_{\epsilon_k}) & & \\ & & \ddots & \\ & & & \sigma_m(X^*_{\epsilon_k}) \end{pmatrix},$$

which implies

$$A_{\epsilon_k} = U(X^*_{\epsilon_k})^T X^*_{\epsilon_k} V_1(X^*_{\epsilon_k}) = \Sigma(X^*_{\epsilon_k}). \tag{3.11}$$

From (3.10) and (3.11), we have

$$\left(\mathcal{A}(X^*_{\epsilon_k}) - b\right)^T \mathcal{A}(X^*_{\epsilon_k}) + \tau \sum_{i=1}^{m} \sigma_i^2(X^*_{\epsilon_k})(\sigma_i^2(X^*_{\epsilon_k}) + \epsilon_k^2)^{\frac{p}{2}-1} = 0.$$

When $k \to \infty$, we obtain that $X^*_{\epsilon_k} \to X^*$ and $\sigma_i(X^*_{\epsilon_k}) \to \sigma_i(X^*)$ $(i = 1, 2, \cdots, m)$. Hence,

$$\mathcal{A}(X^*)^T \left(\mathcal{A}(X^*) - b\right) + \tau \sum_{i=1}^{m} \sigma_i^p(X^*) = 0,$$

i.e.

$$\mathcal{A}(X^*)^T \left(\mathcal{A}(X^*) - b\right) + \tau \|X^*\|_p^p = 0,$$

and X^* satisfies the first order necessary condition of Problem ($\mathbf{P}$).

(2) Let $\hat{X}^*$ be the global minimizer of Problem ($\mathbf{P}$). Then from the following three inequalies,

$$F(X^*_{\epsilon_k}) \le F(\epsilon_k, X^*_{\epsilon_k}) \le F(\epsilon_k, \hat{X}^*) \le F(\hat{X}^*) + \frac{\tau m |\epsilon_k|^p}{p}.$$

When $k \to \infty$, implies that $X^*_{\epsilon_k} \to X^*$ and $F(X^*) \le F(\hat{X}^*)$. Hence, X^* is the global minimizer of Problem ($\mathbf{P}$). $\square$

Let us introduce $\bar{A}(\epsilon)$ as

$$\bar{A}(\epsilon) = \begin{pmatrix} \bar{a}^1_{11}(\epsilon) & \bar{a}^1_{22}(\epsilon) & \cdots & \bar{a}^1_{mm}(\epsilon) \\ \bar{a}^2_{11}(\epsilon) & \bar{a}^2_{22}(\epsilon) & \cdots & \bar{a}^2_{mm}(\epsilon) \\ \vdots & \vdots & & \vdots \\ \bar{a}^q_{11}(\epsilon) & \bar{a}^q_{22}(\epsilon) & \cdots & \bar{a}^q_{mm}(\epsilon) \end{pmatrix},$$

where

$$\bar{a}^i_{jj}(\epsilon) := \left[U(X^*_\epsilon)^T A_i V(X^*_\epsilon)\right]_{ii} \ (i = 1, 2, \cdots, q \text{ and } j = 1, 2, \cdots, m).$$

The following theorem presents the lower bound for the singular values at the solution of the smoothing model ($\mathbf{P}_\epsilon$).

Theorem 3.3 *Let* $\tilde{L}_\epsilon := \kappa \left(\dfrac{\tau}{\|\bar{A}(\epsilon)\| \sqrt{2F(X^0) + \dfrac{2\tau m|\epsilon|^{p})}{p}}} \right)^{\frac{1}{1-p}}$ *and fixed* $\epsilon \neq 0$. *Suppose* X_ϵ^* *is a local miminizer of Problem* $(\mathbf{P}_\epsilon)$ *and* $F(X_\epsilon^*) \leq F(X^0)$, *we have*

$$\text{for any } i \in \{1, 2, \cdots, m\}, \quad \sigma_i(X_\epsilon^*) < \tilde{L}_\epsilon \quad \Rightarrow \quad \sigma_i(X_\epsilon^*) \leq |\epsilon|, \tag{3.12}$$

where $\kappa := \left(\dfrac{1}{2}\right)^{\frac{1}{1-p}} < 1$.

Proof. By the definition of $F(\epsilon, \cdot)$, we have

$$\begin{aligned}
\left(\mathcal{A}(H)^T \left(\mathcal{A}(X_\epsilon^*) - b\right)\right)^2 &\leq \|\mathcal{A}(H)\|_2^2 \cdot \|\mathcal{A}(X_\epsilon^*) - b\|_2^2 \\
&\leq \|\mathcal{A}(H)\|_2^2 \left(\|\mathcal{A}(X_\epsilon^*) - b\|_2^2 + \frac{\tau}{p} \sum_{i=1}^{m} (\sigma_i^2(X_\epsilon^*) + \epsilon^2)^{\frac{p}{2}} \right) \\
&= 2\|\mathcal{A}(H)\|_2^2 F(\epsilon, X_\epsilon^*) \\
&\leq \|\mathcal{A}(H)\|_2^2 \left(2F(X_\epsilon^*) + \frac{2\tau m|\epsilon|^p}{p} \right) \\
&\leq \|\mathcal{A}(H)\|_2^2 \left(2F(X^0) + \frac{2\tau m|\epsilon|^p}{p} \right).
\end{aligned}$$

Since $X_\epsilon^* \in \mathbf{Sol}(\mathbf{P}_\epsilon)$, the equation (3.8) implies

$$\mathcal{A}(H)^T \left(\mathcal{A}(X_\epsilon^*) - b\right) + \tau(\mathbf{diag} A(\epsilon))^T \begin{pmatrix} \sigma_1(X_\epsilon^*)(\sigma_1^2(X_\epsilon^*) + \epsilon^2)^{\frac{p}{2}-1} \\ \sigma_2(X_\epsilon^*)(\sigma_2^2(X_\epsilon^*) + \epsilon^2)^{\frac{p}{2}-1} \\ \vdots \\ \sigma_{m-1}(X_\epsilon^*)(\sigma_{m-1}^2(X_\epsilon^*) + \epsilon^2)^{\frac{p}{2}-1} \\ \sigma_m(X_\epsilon^*)(\sigma_m^2(X_\epsilon^*) + \epsilon^2)^{\frac{p}{2}-1} \end{pmatrix} = 0,$$

where $A(\epsilon) := U(X_\epsilon^*) H V_1^T(X_\epsilon^*)$. Set

$$H(\epsilon) := U(X_\epsilon^*) \left(\begin{array}{ccccccc|c} 0 & & & & & & & \\ & \ddots & & & & & & \\ & & 0 & & & & & \\ & & & 1 & & & & \mathbf{0} \\ & & & & 0 & & & \\ & & & & & \ddots & & \\ & & & & & & 0 & \end{array} \right) V(X_\epsilon^*)^T,$$

then

$$\mathcal{A}(H(\epsilon)) = \begin{pmatrix} \bar{a}_{ii}^1(\epsilon) \\ \bar{a}_{ii}^2(\epsilon) \\ \vdots \\ \bar{a}_{ii}^q(\epsilon) \end{pmatrix} = \bar{A}(\epsilon) e_i^m,$$

where e_i^m is the ith column of the identity matrix I_m. Hence,we obtain

$$\begin{aligned}
\tau\sigma_i(X_\epsilon^*)(\sigma_i^2(X_\epsilon^*) + \epsilon^2)^{\frac{p}{2}-1} &\leq \left|\mathcal{A}(H(\epsilon))^T \left(\mathcal{A}(X_\epsilon^*) - b\right)\right| \\
&\leq \|\mathcal{A}(H(\epsilon))\|_2 \sqrt{2F(X^0) + \frac{2\tau m|\epsilon|^p}{p}} \\
&\leq \|\bar{A}(\epsilon)\| \sqrt{2F(X^0) + \frac{2\tau m|\epsilon|^p}{p}}.
\end{aligned}$$

Suppose $\sigma_i(X_\epsilon^*) > |\epsilon|$, then we have

$$2\sigma_i^2(X_\epsilon^*) > \sigma_i^2(X_\epsilon^*) + \epsilon^2 \Rightarrow (2\sigma_i^2(X_\epsilon^*))^{\frac{p}{2}-1} < (\sigma_i^2(X_\epsilon^*) + \epsilon^2)^{\frac{p}{2}-1}.$$

Therefore,

$$\tau(2^{\frac{p}{2}-1})\sigma_i^{p-1}(X_\epsilon^*) \leq \|\bar{A}(\epsilon)\|\sqrt{2F(X^0) + \frac{2\tau m|\epsilon|^p)}{p}},$$

which implies

$$\sigma_i(X_\epsilon^*) \quad \geq \quad \left(\frac{\tau(2^{\frac{p}{2}-1})}{\|\bar{A}(\epsilon)\|\sqrt{2F(X^0) + \frac{2\tau m|\epsilon|^p)}{p}}}\right)^{\frac{1}{1-p}} > \kappa\left(\frac{\tau}{\|\bar{A}(\epsilon)\|\sqrt{2F(X^0) + \frac{2\tau m|\epsilon|^p)}{p}}}\right)^{\frac{1}{1-p}} = \tilde{L}_\epsilon.$$

Hence we can claim that, for $i \in \{1, 2, \cdots, m\}$, if $\sigma_i(X_\epsilon^*) < \tilde{L}_\epsilon$ then $\sigma_i(X_\epsilon^*) \leq |\epsilon|$. □

Theorem 3.4 *Let $X_{\epsilon_k}^*$ be a local minimizer of Problem $(\mathbf{P}_{\epsilon_k})$ with $F(X_{\epsilon_k}^*) \leq F(X^0)$ and $\{X_{\epsilon_k}^*\}$ be a convergent sequence. Then there is an integer $K > 0$ such that for any $k \geq K$, there is $X^* \in \mathbf{Sol}(\mathbf{P})$ such that*

$$\begin{aligned}\mathbf{\Gamma}_{\epsilon_k} \quad &:= \quad \{i \in \{1, 2, \cdots, m\} \mid \sigma_i(X_{\epsilon_k}^*) \leq |\epsilon_k|\} \\ &= \quad \{i \in \{1, 2, \cdots, m\} \mid \sigma_i(X^*) = 0\} =: \mathbf{\Gamma}.\end{aligned} \tag{3.13}$$

Proof. Since the level set $\{X : F(X) \leq F(X^0)\}$ is bounded, the sequence $\{X_{\epsilon_k}^*\}$ is bounded. From (1) of Theorem 3.2, any accumulation point of $\{X_{\epsilon_k}^*\}$ lies in the set $\mathbf{Sol}(\mathbf{P})$, it follows from the assumptions in the theorem that

$$\lim_{k\to\infty} \operatorname{dist}(X_{\epsilon_k}^*, \mathbf{Sol}(\mathbf{P})) = 0,$$

which implies that there exists $X^* \in \mathbf{Sol}(\mathbf{P})$ such that $\lim_{k\to\infty} X_{\epsilon_k}^* = X^*$ and there exists an integer $K > 0$ such that for $k \geq K$, $|\epsilon_k| < \dfrac{\tilde{L}}{2} < \tilde{L}_{\epsilon_k}$,

$$\operatorname{dist}(X_{\epsilon_k}^*, X^*) = \|X_{\epsilon_k}^* - X^*\| \leq \frac{\tilde{L}}{2},$$

and $F(X^*) \leq F(X^0)$ hold. Then we have $\sigma_i(X^*) - \sigma_i(X_{\epsilon_k}^*) \leq \left|\sigma_i(X^*) - \sigma_i(X_{\epsilon_k}^*)\right| \leq \|X^* - X_{\epsilon_k}^*\| < \dfrac{\tilde{L}}{2}$.

If $i \in \mathbf{\Gamma}_{\epsilon_k}$, we have

$$\sigma_i(X^*) \leq \sigma_i(X_{\epsilon_k}^*) + \frac{\tilde{L}}{2} < \tilde{L}.$$

Assume that $\sigma_i(X^*) \neq 0$, from Corollary 2.2, we have

$$\tau\tilde{L}^{p-1} < \tau\sigma_i^{p-1}(X^*) \leq |\tilde{A}^T(\tilde{A}z^* - b)|_i \leq \|\tilde{A}\| \cdot \|\tilde{A}z^* - b\|_2.$$

where $z^* := (\sigma_1(X^*), \sigma_2(X^*), \cdots, \sigma_r(X^*))^T \in \mathcal{R}^r$. By the fact that $\mathcal{A}(X^*) = \tilde{A}z^*$ (see Appendix A), we have

$$\tau\tilde{L}^{p-1} < \|\tilde{A}\| \cdot \|\tilde{A}z^* - b\|_2 = \|\tilde{A}\| \cdot \|\mathcal{A}(X^*) - b\|_2 \leq \|\tilde{A}\|\sqrt{2F(X^0)},$$

which leads to the following contradiction:

$$\tilde{L} > \left(\frac{\tau}{\|\tilde{A}\|\sqrt{2F(X^0)}}\right)^{\frac{1}{1-p}} > \kappa\left(\frac{\tau}{\|\tilde{A}\|\sqrt{2F(X^0)}}\right)^{\frac{1}{1-p}} = \tilde{L}.$$

where $\kappa := \left(\dfrac{1}{2}\right)^{\frac{1}{1-p}} < 1$. Therefore, we obtain that $\sigma_i(X^*) = 0$, which means that $\mathbf{\Gamma}_{\epsilon_k} \subset \mathbf{\Gamma}$.

On the other hand, if $i \in \mathbf{\Gamma}$ then $\sigma_i(X^*) = 0$ and

$$\sigma_i(X^*_{\epsilon_k}) = \sigma_i(X^*_{\epsilon_k}) - \sigma_i(X^*) \le \|X^*_{\epsilon_k} - X^*\| \le \frac{\tilde{L}}{2} < \tilde{L}_{\epsilon_k}.$$

From (3.12), we can deduce that $\sigma_i(X^*_{\epsilon_k}) \le |\epsilon_k|$. Hence, $\mathbf{\Gamma} \subset \mathbf{\Gamma}_{\epsilon_k}$, i.e. the conclusion (3.13) is true. □

4. The Majorization Method for the Smoothing Model

The purpose of this section is to introduce the majorization method for solving Problem $(\mathbf{P}_\epsilon)$. For notational convenience, we define $W(\epsilon, X)$ by

$$W(\epsilon, X) := U(X)\mathbf{Diag}\left((\sigma_1^2(X) + \epsilon^2)^{\frac{p}{2}-1}, \cdots, (\sigma_m^2(X) + \epsilon^2)^{\frac{p}{2}-1}\right) U(X)^T,$$

where $U(X)$ is the left singular matrix of X. From Theorem 3.1, the first order directional derivative of $F(\epsilon, X)$ with respect to the variable X along the direction H can be written as

$$\mathrm{D}F(\epsilon, X)[H] = \langle \mathcal{A}^*(\mathcal{A}(X) - b) + \tau W(\epsilon, X)X, H\rangle,$$

which implies

$$\mathrm{D}_X F(\epsilon, X) = \mathcal{A}^*(\mathcal{A}(X) - b) + \tau W(\epsilon, X)X,$$

where $\mathcal{A}^*$ is the adjoint of $\mathcal{A}$ defined by

$$\mathcal{A}^*(y) := y_1 A_1 + y_2 A_2 + \cdots + y_q A_q = \sum_{j=1}^{q} y_j A_j.$$

In the sequel, we treat ϵ as a decision variable of $F(\epsilon, X)$. Then

$$\mathrm{D}_\epsilon F(\epsilon, X) = \tau\epsilon\ \mathbf{trace} W(\epsilon, X), \quad \text{if } \epsilon \neq 0.$$

Since $F(\epsilon, X)$ is a nonconvex function, we shall construct the following problem

$$(\mathbf{SP}_{\epsilon^k}) \quad \begin{cases} \text{minimize} & \hat{F}^k(\epsilon, X) \\ \text{subject to} & X \in \mathcal{M}_{m\times n}, \end{cases}$$

to obtain the next iterate in the kth iteration, where

$$\hat{F}^k(\epsilon, X) := F_1(X) + \hat{F}_2^k(\epsilon, X),$$

and

$$\begin{aligned} \hat{F}_2^k(\epsilon, X) \quad := \quad & F_2(\epsilon^k, X^k) + \langle \mathrm{D}_X F_2(\epsilon^k, X^k), X - X^k\rangle + \mathrm{D}_\epsilon F_2(\epsilon^k, X^k)\cdot(\epsilon - \epsilon^k) \\ & + \frac{\tau\rho^k}{2}\left[\|X - X^k\|_F^2 + (\epsilon - \epsilon^k)^2\right]. \end{aligned}$$

It is not difficult to find that

$$\hat{F}_2^k(\epsilon^k, X^k) = F_2(\epsilon^k, X^k). \tag{4.1}$$

Meanwhile, we wish that $\hat{F}_2^k(\epsilon, X)$ may satisfy the following condition by choosing the parameter ρ^k dynamically:

$$\hat{F}_2^k(\epsilon, X) \ge F_2(\epsilon, X), \quad \text{for all } X \in \mathcal{M}_{m\times n} \text{ and } \epsilon \neq 0. \tag{4.2}$$

If (4.1) and (4.2) hold, the function $\hat{F}_2^k(\epsilon, X)$ is called a *majorization* function of $F_2(\epsilon, X)$ at X^k.

Note that the idea of using *majorization* function in optimization was dated back to Ortega and Rheinboldt [30] for studying the line search strategy. After that, many contributions for majorization methods have been made. For instance, Leeuw et al. [31, 32, 33, 34, 35] proposed many algorithms for multidimensional scaling problems. And recently, Gao and Sun [36] design a majorized penalty approach for the *calibrating rank constrained correlation matrix problems.*

If we choose ρ^k sufficiently large, Problem $(\mathbf{SP}_{\epsilon^k})$ becomes a continuous convex programming problem. Let the pair $(\hat{X}_k^*, \hat{\epsilon}_k^*)$ be a optimal solution of Problem $(\mathbf{SP}_{\epsilon^k})$, then we have

$$\left\{\begin{array}{l} \mathrm{D}_X \hat{F}^k(\hat{\epsilon}_k^*, \hat{X}_k^*) = 0, \\ \mathrm{D}_\epsilon \hat{F}^k(\hat{\epsilon}_k^*, \hat{X}_k^*) = 0, \end{array}\right.$$

which implies

$$\left\{\begin{array}{l} \mathcal{A}^*(\mathcal{A}(\hat{X}_k^*) - b) + \mathrm{D}_X F_2(\epsilon^k, X^k) + \tau\rho^k(\hat{X}_k^* - X^k) = 0, \\ \\ \mathrm{D}_\epsilon F_2(\epsilon^k, X^k) + \tau\rho^k(\hat{\epsilon}_k^* - \epsilon^k) = 0, \end{array}\right.$$

i.e.

$$\left\{\begin{array}{l} \mathcal{A}^*(\mathcal{A}(\hat{X}_k^*)) + \tau\rho^k \hat{X}_k^* = \tau\rho^k X^k - \tau W(\epsilon^k, X^k) X^k + \mathcal{A}^*(b), \\ \\ \hat{\epsilon}_k^* = \epsilon^k \left(1 - \dfrac{1}{\rho^k} \mathbf{trace} W(\epsilon^k, X^k)\right). \end{array}\right. \tag{4.3}$$

We set the pair $(\hat{X}_k^*, \hat{\epsilon}_k^*)$ satisfied the above relationship (4.3) to be the next iterate pair $(X^{k+1}, \epsilon^{k+1})$. Now, the method for solving Problem $(\mathbf{P}_\epsilon)$ is summarized as follows:
The majorization method

Step 0 Choose the tradeoff parameter τ, the initial $\epsilon^0 \neq 0$ and the initial X^0. Construct Problem $(\mathbf{P}_{\epsilon^0})$ and set $k := 0$.

Step 1 Select ρ_k satisfying (4.1) and (4.2). Construct the majorization function $\hat{F}_2^k(\epsilon, X)$ and solve the following $(\mathbf{SP}_{\epsilon^k})$ problem

$$\left\{\begin{array}{ll} \text{minimize} & \hat{F}^k(\epsilon, X) \\ \text{subject to} & X \in \mathcal{M}_{m\times n}, \end{array}\right.$$

to get the new iterate pair $(X^{k+1}, \epsilon^{k+1})$.

Step 2 Set $k := k+1$ and goto Step 1.

Theorem 4.1 *Let the pairs $\{(X^k, \epsilon^k)\}$ be the sequence generated by the above majorization method. Then*

(1) $\{F(\epsilon^k, X^k)\}$ *is a monotonically decreasing sequence.*

(2) *The difference between $F(\epsilon^k, X^k)$ and $F(\epsilon^{k+1}, X^{k+1})$ is given as follows*

$$F(\epsilon^k, X^k) - F(\epsilon^{k+1}, X^{k+1}) \geq \frac{1}{2}\|\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\|_2^2 + \frac{\tau\rho^k}{2}\left(\|X^{k+1} - X^k\|_F^2 + (\epsilon^{k+1} - \epsilon^k)^2\right).$$

(3) *Let $\omega_k := \left(1 - \dfrac{1}{\rho^k} \textbf{\textit{trace}} W(\epsilon^k, X^k)\right)$ and $|\omega_k| \in (0,1)$. Then any accumulation point of the bounded sequence $\{X^k\}$ contained in the level set $\{X : F(X) \leq F(X^0)\}$ satisfies the necessary condition of Problem **(P)**.*

Proof. (1) By the definition of F, we have

$$\begin{array}{rcl} F(\epsilon^{k+1}, X^{k+1}) & = & F_1(X^{k+1}) + F_2(\epsilon^{k+1}, X^{k+1}) \\ & \leq & F_1(X^{k+1}) + \hat{F}_2^k(\epsilon^{k+1}, X^{k+1}) \\ & = & \hat{F}^k(\epsilon^{k+1}, X^{k+1}) \\ & \leq & \hat{F}^k(\epsilon^k, X^k) \\ & = & F_1(X^k) + \hat{F}_2^k(\epsilon^k, X^k) \\ & = & F_1(X^k) + F_2(\epsilon^k, X^k) \\ & = & F(\epsilon^k, X^k), \end{array}$$

where the first inequality uses the property of the majorization function given in (4.2), the second one uses the fact that $(X^{k+1}, \epsilon^{k+1})$ is the solution of Problem $(\mathbf{SP}_{\epsilon^k})$, the fourth equation uses the property of the majorzation function given in (4.1).

(2) The difference between $F(\epsilon^k, X^k)$ and $F(\epsilon^{k+1}, X^{k+1})$ satisfies

$$\begin{aligned}
& F(\epsilon^{k+1}, X^{k+1}) - F(\epsilon^k, X^k) \\
\leq\ & \hat{F}^k(\epsilon^{k+1}, X^{k+1}) - F(\epsilon^k, X^k) \\
=\ & F_1(X^{k+1}) + \hat{F}_2^k(\epsilon^{k+1}, X^{k+1}) - F_1(X^k) - F_2(\epsilon^k, X^k) \\
=\ & \frac{1}{2}\|\mathcal{A}(X^{k+1}) - b\|_2^2 - \frac{1}{2}\|\mathcal{A}(X^k) - b\|_2^2 \\
& + \langle \mathrm{D}_X F_2(\epsilon^k, X^k), X^{k+1} - X^k \rangle + \mathrm{D}_\epsilon F_2(\epsilon^k, X^k) \cdot (\epsilon^{k+1} - \epsilon^k) \\
& + \frac{\tau\rho^k}{2}\left(\|X^{k+1} - X^k\|_F^2 + (\epsilon^{k+1} - \epsilon^k)^2\right). \qquad (4.4)
\end{aligned}$$

For simplicity, we divide the right hand term in (4.4) into the following two parts

$$L_1 := \frac{1}{2}\|\mathcal{A}(X^{k+1}) - b\|_2^2 - \frac{1}{2}\|\mathcal{A}(X^k) - b\|_2^2 + \langle \mathrm{D}_X F_2(\epsilon^k, X^k), X^{k+1} - X^k \rangle + \frac{\tau\rho^k}{2}\|X^{k+1} - X^k\|_F^2,$$

and

$$L_2 := \mathrm{D}_\epsilon F_2(\epsilon^k, X^k) \cdot (\epsilon^{k+1} - \epsilon^k) + \frac{\tau\rho^k}{2}(\epsilon^{k+1} - \epsilon^k)^2.$$

Then (4.4) can be rewritten as

$$F(\epsilon^{k+1}, X^{k+1}) - F(\epsilon^k, X^k) \leq L_1 + L_2. \qquad (4.5)$$

From (4.3) and Step 1 in the *majorization method*, we have

$$\mathcal{A}^*(\mathcal{A}(X^{k+1})) + \tau\rho^k X^{k+1} = \tau\rho^k X^k - \tau W(\epsilon^k, X^k) X^k + \mathcal{A}^*(b)$$

and

$$\mathrm{D}_\epsilon F_2(\epsilon^k, X^k) + \tau\rho^k(\epsilon^{k+1} - \epsilon^k) = 0.$$

Hence,

$$\begin{aligned}
(\mathcal{A}(X^{k+1}) - b)^T(\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)) + \tau\rho^k\|X^{k+1} - X^k\|_F^2 + \langle \mathrm{D}_X F_2(\epsilon^k, X^k), X^{k+1} - X^k \rangle = 0, \\
\mathrm{D}_\epsilon F_2(\epsilon^k, X^k) \cdot (\epsilon^{k+1} - \epsilon^k) + \tau\rho^k(\epsilon^{k+1} - \epsilon^k)^2 = 0. \qquad (4.6)
\end{aligned}$$

By the equation (4.6), we can obtain that (for details, see Appendix C)

$$L_1 = -\frac{1}{2}\|\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\|_2^2 - \frac{\tau\rho^k}{2}\|X^{k+1} - X^k\|_F^2, \qquad (4.7)$$

and

$$L_2 = -\frac{\tau\rho^k}{2}(\epsilon^{k+1} - \epsilon^k)^2. \qquad (4.8)$$

From the inequality (4.5) and the equations (4.7),(4.8), we have

$$F(\epsilon^k, X^k) - F(\epsilon^{k+1}, X^{k+1}) \geq \frac{1}{2}\|\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\|_2^2 + \frac{\tau\rho^k}{2}\left(\|X^{k+1} - X^k\|_F^2 + (\epsilon^{k+1} - \epsilon^k)^2\right).$$

(3) To prove the remaining part of this theorem, we assume that X^* is an accumulation point of $\{X^k\}$, then exists an index set $\{k_s\} \subseteq \{1, 2, \cdots, \}$ such that

$$\lim_{s \to +\infty} X^{k_s} = X^*. \qquad (4.9)$$

From the second part of the theorem, we obtain

$$\begin{array}{rl} & \lim\limits_{s\to+\infty}\sum\limits_{i=0}^{k_s}\left(\frac{1}{2}\|\mathcal{A}(X^{i+1})-\mathcal{A}(X^i)\|_2^2+\frac{\tau\rho^i}{2}\left(\|X^{i+1}-X^i\|_F^2+(\epsilon^{i+1}-\epsilon^i)^2\right)\right) \\ \leq & \liminf\limits_{s\to+\infty}\left(F(\epsilon^0,X^0)-F(\epsilon^{k_s+1},X^{k_s+1})\right) \\ \leq & F(\epsilon^0,X^0)<+\infty, \end{array}$$

which implies

$$\lim_{i\to\infty}\|X^{i+1}-X^i\|_F=0,\quad \lim_{i\to\infty}\|\mathcal{A}(X^{i+1})-\mathcal{A}(X^i)\|_2=0,\quad \lim_{i\to\infty}|\epsilon^{i+1}-\epsilon^i|=0.$$

From the equation (4.9), the relationship $\epsilon^{k+1}=\epsilon^k\cdot\omega_k$ and $0<|\omega_k|<1$, we have

$$\lim_{i\to\infty}X^{i+1}=\lim_{i\to\infty}X^i=X^*\quad\text{and}\quad\lim_{i\to\infty}\epsilon^i=0.$$

By the first equation of (4.3) and Step 1 of the *majorization method*,

$$\mathcal{A}^*(\mathcal{A}(X^{k+1}))+\tau\rho^kX^{k+1}=\tau\rho^kX^k-\tau W(\epsilon^k,X^k)X^k+\mathcal{A}^*(b).$$

Then

$$\langle\mathcal{A}^*(\mathcal{A}(X^{k+1})-b),X^k\rangle+\langle\tau\rho^k(X^{k+1}-X^k),X^k\rangle=\langle-\tau W(\epsilon^k,X^k)X^k,X^k\rangle,$$

i.e.

$$\mathcal{A}(X^k)^T\left(\mathcal{A}(X^{k+1})-b\right)+\langle\tau\rho^k(X^{k+1}-X^k),X^k\rangle+\tau\sum_{i=1}^m(\sigma_i^2(X^k)+\epsilon^k)^{\frac{p}{2}-1}\sigma_i^2(X^k)=0.$$

Hence, when $k\to+\infty$, we have

$$\mathcal{A}(X^*)^T\left(\mathcal{A}(X^*)-b\right)+\tau\sum_{i=1}^m\sigma_i^p(X^*)=0,$$

which implies that X^* satisfies the necessary condition of Problem **(P)** such that

$$\mathcal{A}(X^*)^T\left(\mathcal{A}(X^*)-b\right)+\tau\|X^*\|_p^p=0.$$

□

5. An Application to Matrix Completion Problem

5.1 Smoothing majorization algorithm

This subsection presents implementation details of the smoothing majorization algorithm applied to the *matrix completion* (**MC**) problem

$$(\mathbf{MC})\qquad\left\{\begin{array}{ll}\text{minimize} & \frac{1}{2}\|\mathcal{P}_\Omega(X-X_R)\|_2^2+\frac{\tau}{p}\|X\|_p^p\\ \text{subject to} & X\in\mathcal{M}_{m\times n},\end{array}\right.\tag{5.1}$$

where the operator $\mathcal{A}:=\mathcal{P}_\Omega$ and $b:=\mathcal{P}_\Omega(X_R)$, Ω is the observation index of the given recovery matrix X_R with the known rank r and the size of Ω is q. We denote Ω as follows

$$\Omega:=\{(i_1,j_1),(i_2,j_2),\cdots,(i_q,j_q)\mid i_1,i_2,\cdots,i_q\in[1,2,\cdots,m],j_1,j_2,\cdots,j_q\in[1,2,\cdots,n]\}.$$

The corresponding A_s for the operator $\mathcal{A}(\cdot)$ are given by

$$A_s:=E(i_s,j_s),\quad s=1,2,\cdots,q,$$

where $E(i_s, j_s)$ denotes the matrix whose (i_s, j_s) entry equals 1 and all others are 0. Hence,

$$\mathcal{P}_\Omega(X) = \left(X_{i_1j_1}, X_{i_2j_2}, \cdots, X_{i_qj_q}\right)^T \in \mathcal{R}^q,$$

where $X_{i_sj_s}$ denotes the (i_s, j_s) entry of the matrix X.

As mentioned previously, some authors have also conducted the research on **(MC)** problem via l_p regularization. Mohan et al. [19] studied the constrained version of Problem (5.1), i.e.

$$\begin{cases} \text{minimize} & f_p(X) = \mathbf{trace}(X^TX + \epsilon I)^{\frac{p}{2}} \\ \text{subject to} & \mathcal{P}_\Omega(X - X_R) = 0. \end{cases}$$

The update strategy of [19] may be expressed as

$$\begin{cases} W_p^k = ((X^k)^TX^k + \epsilon^k I)^{\frac{p}{2}-1}, \\ X^{k+1} = (X^k - s^kX^kW_p^k)_{\Omega^c} + (X_R)_\Omega, \\ \epsilon^k = \epsilon^0/(\eta)^k, \\ s^{k+1} = (\epsilon^{k+1})^{1-\frac{p}{2}}. \end{cases} \tag{5.2}$$

Lai et al. [20] considered the following problem

$$\begin{cases} \text{minimize} & \mathbf{trace}(X^TX + \epsilon^2 I)^{\frac{p}{2}} + \dfrac{1}{2\tau}\|\mathcal{P}_\Omega(X - X_R)\|_2^2 \\ \text{subject to} & X \in \mathcal{M}_{m\times n}. \end{cases}$$

The update strategy of [20] was given by

$$\begin{cases} X^k = \mathbf{Sol}\left(\tau pXW_p^{k-1} + \mathcal{A}^*(\mathcal{A}(X) - b) = 0\right), \\ (X^k)^TX^k = V^k(\Sigma^k)^2(V^k)^T, \\ \epsilon^k = \min\{\epsilon^{k-1}, \epsilon_s\sigma_{r+1}(X^k)\}, \epsilon_s := 0.9, \\ W_p^k = V^k[(\Sigma^k)^2 + (\epsilon^k)^2 I]^{\frac{p}{2}-1}(V^k)^T. \end{cases}$$

It is not difficult to find that the above update strategies of regularization parameter ϵ are based on the computational experience from the authors. As Mohan et al. pointed out in the section 6.1 of [19], *the regularization parameter ϵ^k plays an important role in the recovery.* Hence, we have the reason for studying the self-adaptive update scheme for the parameter ϵ^k. From (4.3), we have

$$\mathcal{A}^*(\mathcal{A}(X^{k+1})) + \tau\rho^kX^{k+1} = \tau\rho^kX^k - \tau W(\epsilon^k, X^k)X^k + \mathcal{A}^*(b),$$

$$\mathrm{D}_\epsilon F_2(\epsilon^k, X^k) + \tau\rho^k(\epsilon^{k+1} - \epsilon^k) = 0.$$

Since the operator $\mathcal{A}(\cdot)$ equals $\mathcal{P}_\Omega(\cdot)$, the above update strategy can be rewritten as

$$\begin{cases} (X^{k+1})_\Omega = \left(\dfrac{\tau\rho^k}{1+\tau\rho^k}X^k - \dfrac{\tau}{1+\tau\rho^k}W(\epsilon^k, X^k)X^k + \dfrac{1}{1+\tau\rho^k}\mathcal{A}^*(b)\right)_\Omega, \\ (X^{k+1})_{\Omega^c} = \left(X^k - \dfrac{1}{\rho^k}W(\epsilon^k, X^k)X^k\right)_{\Omega^c}, \\ \epsilon^{k+1} = \epsilon^k\left(1 - \dfrac{1}{\rho^k}\mathbf{trace}W(\epsilon^k, X^k)\right), \end{cases}$$

where $(X^{k+1})_\Omega$ denotes the matrix whose (i, j) entry equals $(X^{k+1})_{ij}$ if the index (i, j) lies in Ω and zero otherwise. The definition of $(X^{k+1})_{\Omega^c}$ is similar as $(X^{k+1})_\Omega$. Obviously, if we set $\tau = 0$, we have

$$\begin{cases} (X^{k+1})_\Omega = \mathcal{A}^*(b) = (X_R)_\Omega, \\ (X^{k+1})_{\Omega^c} = \left(X^k - \dfrac{1}{\rho^k}W(\epsilon^k, X^k)X^k\right)_{\Omega^c}, \\ \epsilon^{k+1} = \epsilon^k\left(1 - \dfrac{1}{\rho^k}\mathbf{trace}W(\epsilon^k, X^k)\right). \end{cases} \tag{5.3}$$

Note that, comparing with Mohan's update strategy, the parameter ρ^k plays a similar role as s^k in (5.2). However, the self-adaptive update in our strategy is displayed in the updating formula of the variable ϵ.

Next, we will discuss how to choose the majorization parameter ρ^k and the stop criterion, which are the important issues both in algorithm and in theory. From (3) of Theorem 4.1, we know that $\epsilon^k \to 0$ as $k \to +\infty$, which is guaranteed by choosing $|\omega_k| \in (0,1)$, i.e.

$$-1 < 1 - \frac{1}{\rho^k}\mathbf{trace}W(\epsilon^k, X^k) < 1 \quad \Rightarrow \quad \rho^k > \frac{1}{2}\mathbf{trace}W(\epsilon^k, X^k).$$

In addition, the majorization parameter ρ^k should make

$$\hat{F}_2^k(\epsilon, X) \geq F_2(\epsilon, X), \quad \text{for all } X \in \mathcal{M}_{m\times n} \text{ and } \epsilon \neq 0,$$

be satisfied, so that ρ^k should be sufficiently large. Hence, in the numerical experiments, we set

$$\rho^k := 10^6 \cdot \mathbf{trace}W(\epsilon^k, X^k).$$

Next, the stop criterion for (5.1) will be present. There are many types of stop criterions for (**MC**) problem such as

(1) The SVT algorithm [8]:

$$\frac{\|\mathcal{P}_\Omega(X^k - X_R)\|_F}{\|\mathcal{P}_\Omega(X_R)\|_F} < 10^{-4}.$$

(2) The IRLS slgorithm [19]:

$$\frac{\|X^k - X_R\|_F}{\|X_R\|_F} < 10^{-3}. \tag{5.4}$$

(3) The IRucLq algorithm [20]:

$$\frac{|\epsilon_k - \epsilon_{k-1}|}{\max\{1, \epsilon_{k-1}\}} < 10^{-5} \quad \text{or} \quad |\epsilon_k| < 10^{-5}.$$

In this paper, we set (5.4) to be the stop criterion of our algorithm. Now, we present the smoothing majorization algorithm for matrix completion problem (5.1).

Algorithm (MAMC)

Step 0. (Initial) Choose the tradeoff parameter τ, the initial $\epsilon^0 \neq 0$ and the initial X^0. Construct the smoothing problem $(\mathbf{P}_{\epsilon^0})$ and set $k := 0$.

Step 1. (Test stop criterion) If

$$\frac{\|X^k - X_R\|_F}{\|X_R\|_F} < tol := 10^{-3},$$

is satisfied, stop; else, goto Step 2.

Step 2. (Update the iterate) Set the majorization parameter ρ^k as follows

$$\rho^k = 10^6 \cdot \mathbf{trace}W(\epsilon^k, X^k).$$

Compute the next iterate $(X^{k+1}, \epsilon^{k+1})$ via the following equations

$$\begin{cases} (X^{k+1})_\Omega = \left(\dfrac{\tau\rho^k}{1+\tau\rho^k}X^k - \dfrac{\tau}{1+\tau\rho^k}W(\epsilon^k, X^k)X^k + \dfrac{1}{1+\tau\rho^k}\mathcal{A}^*(b)\right)_\Omega, \\ (X^{k+1})_{\Omega^c} = \left(X^k - \dfrac{1}{\rho^k}W(\epsilon^k, X^k)X^k\right)_{\Omega^c}, \\ \epsilon^{k+1} = \epsilon^k\left(1 - \dfrac{1}{\rho^k}\mathbf{trace}W(\epsilon^k, X^k)\right). \end{cases}$$

Step 3. (Update the counter) Set $k := k+1$ and goto Step 1.

5.2 Numerical experiments

In this subsection, we report numerical results on a series of matrix completion problems to show the effectiveness and robustness of Algorithm MAMC. All tests were performed in the double precision on an Aoc desktop computer (3.3GHz, Intel double-cores processor, 4096Mbyte of RAM) under Ubuntu 12.04 operation system and Matlab 2010a Unix version. The recovery matrix X_R was exactly of low rank with the form

$$X_R = (ML * MR)/\|ML * MR\|_F,$$

where ML, MR were generated by Matlab's command randn(m, r) and randn(r, n), $ML \in \mathcal{M}_{m\times r}$ and $MR \in \mathcal{M}_{r\times n}$. The largest number of iterate was set to 10000. The initial pair (X^0, ϵ^0) was given as follows

$$X^0 = P_\Omega(X_R), \quad \epsilon^0 = 1.$$

The labels in the following tables are list as follows:

(m,n)	:	the size of the matrix X^k,
p	:	the regularization parameter,
r	:	the rank of the matrix X_R,
SR	:	the sampling ratio,
FR	:	the degrees of freedom in a rank r matrix,
iter	:	the number of iterations for MAMC in the section 5.2.1,
iter1, iter2, iter3, iter4	:	the number of iterations for MAMC, sIRLS, SVT and FPC,
time	:	the cputime of iterations for MAMC in the section 5.2.1,
time1, time2, time3, time4	:	the cputime for MAMC, sIRLS, SVT and FPC,
res	:	the residual of the final iterate in the section 5.2.1,
res1, res2, res3, res4	:	the residual for MAMC, sIRLS, SVT and FPC.

5.2.1 Test for MAMC

We consider four groups of tests: (**A**),(**B**)(**C**) and (**D**).

(**A**) The first group test is focusing on the sensitivity of MAMC algorithm to SR. SR denotes the sampling ratio and SR:= $q/(mn)$, where q is the size of Ω. Table 5.1 gives the numerical results for MAMC algorithm under SR= 0.57 and SR= 0.39. Figure 5.1 and Figure 5.2 show that with the size of the matrix increased, the number of iterations is decreased but the cputime is increased. There exist two main reasons to explain this phenomenon, one is the complexity of the singular value decomposition (SVD); the other is the multiply operation for the large-scale matrices. When the size of the matrix grows up, computing the (SVD) in each iteration will become the bottleneck of the algorithm. Meanwhile, a large quantity of matrix multiplication calculations in each iteration must be executed. In both [19] and [20], the truncated SVD for weighting matrix computation is used. In Algorithm MAMC here, we use the svds in the matlab solver and run a further procedure that makes the singular values after $\sigma_r(X^k)$ become zero.

TABLE 5.1(a): Numerical Results for SR= 0.57.

m	n	p	r	SR	FR	iter	time	res
100	100	0.1	10	0.57	0.33	38	0.493186	9.67×1E-4
200	200	0.1	10	0.57	0.17	19	0.395697	9.33×1E-4
300	300	0.1	10	0.57	0.11	16	0.557317	7.39×1E-4
400	400	0.1	10	0.57	0.08	13	0.793902	9.99×1E-4
500	500	0.1	10	0.57	0.06	13	1.469702	7.45×1E-4
600	600	0.1	10	0.57	0.058	12	2.055558	8.00×1E-4
700	700	0.1	10	0.57	0.049	12	2.637797	6.25×1E-4
800	800	0.1	10	0.57	0.044	11	3.135172	8.72×1E-4
900	900	0.1	10	0.57	0.039	11	4.029796	7.37×1E-4
1000	1000	0.1	10	0.57	0.035	11	4.970550	6.48×1E-4
2000	2000	0.1	10	0.57	0.017	10	20.145275	6.09×1E-4
3000	3000	0.1	10	0.57	0.011	9	44.749802	9.54×1E-4

TABLE 5.1(b): Numerical Results for SR= 0.39.

m	n	p	r	SR	FR	iter	time	res
100	100	0.1	10	0.39	0.49	114	1.322021	9.75×1E-4
200	200	0.1	10	0.39	0.25	46	0.842838	9.10×1E-4
300	300	0.1	10	0.39	0.17	33	1.279406	9.11×1E-4
400	400	0.1	10	0.39	0.13	27	1.884999	9.73×1E-4
500	500	0.1	10	0.39	0.10	25	2.720678	9.04×1E-4
600	600	0.1	10	0.39	0.084	23	3.778318	9.02×1E-4
700	700	0.1	10	0.39	0.073	21	4.589086	9.67×1E-4
800	800	0.1	10	0.39	0.064	21	6.070084	8.15×1E-4
900	900	0.1	10	0.39	0.057	20	7.578933	9.49×1E-4
1000	1000	0.1	10	0.39	0.051	20	9.090852	7.60×1E-4
2000	2000	0.1	10	0.39	0.026	17	33.955004	8.72×1E-4
3000	3000	0.1	10	0.39	0.017	16	79.631080	9.36×1E-4

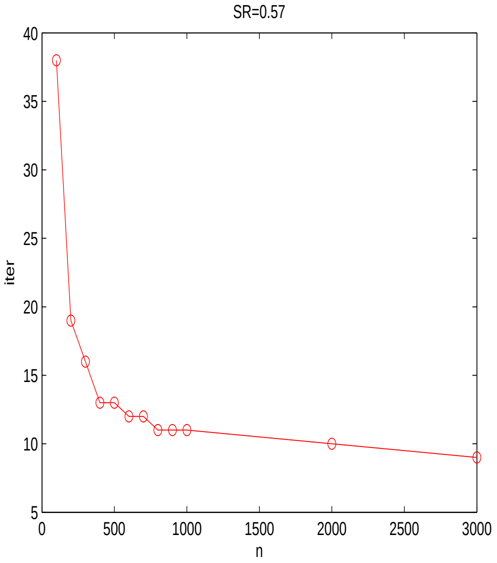

FIG. 5.1(a): The number of iterations when SR= 0.57.

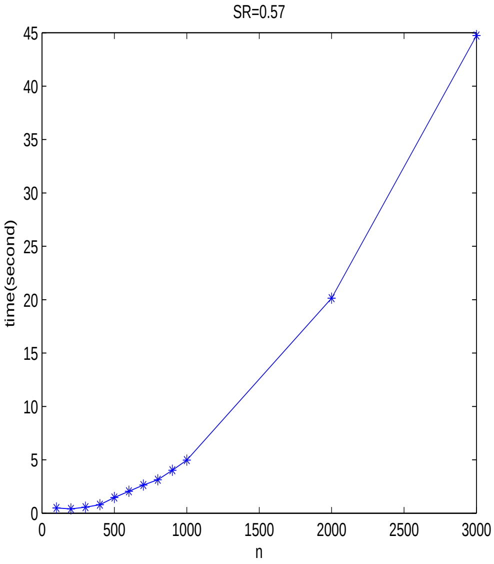

FIG. 5.1(b): The cputime of iterations when SR= 0.57.

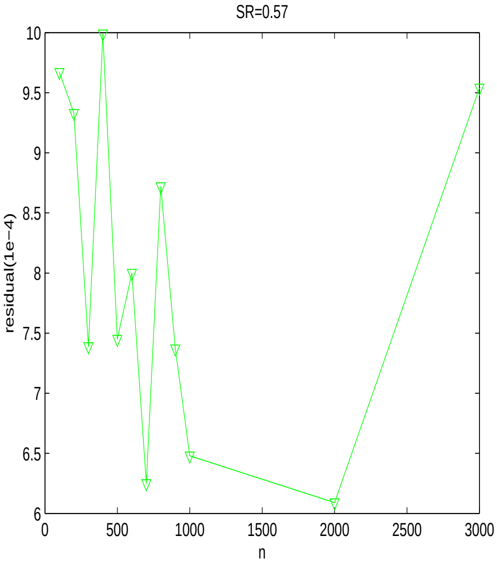

FIG. 5.1(c): The residual of the final iterate when SR= 0.57.

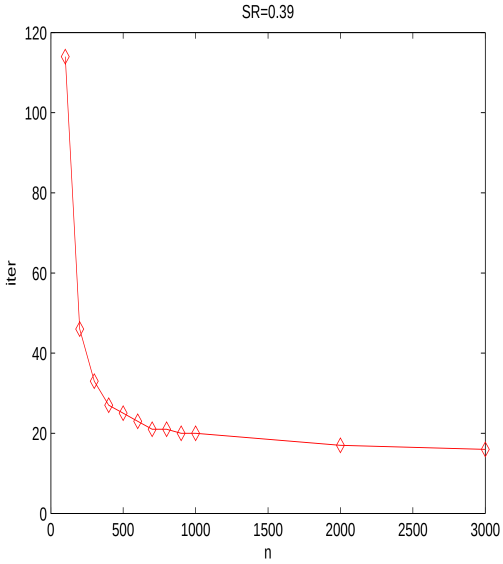

FIG. 5.2(a): The number of iterations when SR= 0.39.

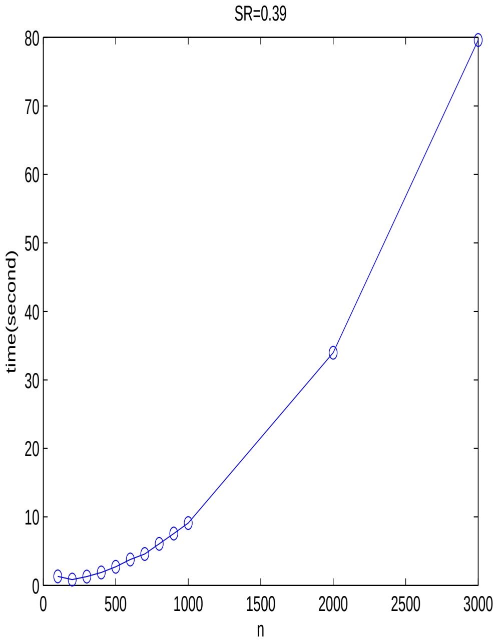

FIG. 5.2(b): The cputime of iterations when SR= 0.39.

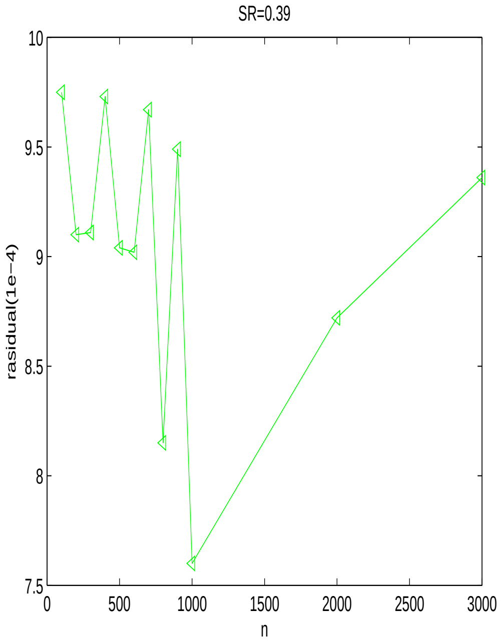

FIG. 5.2(c): The residual of the final iterate when SR= 0.39.

(**B**) The second group test reports the performance for Hard problem that is defined in the paper [19], i.e. FR:= $r(2n-r)/q > 0.4$. Table 5.2 shows that our algorithm can also solve Hard problem efficiently and has the high successful probability that is showed in the column of "ratio".

TABLE 5.2: Numerical results for hard problem (FR > 0.4).

m	n	p	r	SR	FR	ratio	iter	time	res
40	40	0.1	9	0.5	0.82	70%	1907	15.289274	9.99×1E-4
50	50	0.1	5	0.3	0.62	80%	1099	6.372210	9.98×1E-4
50	50	0.1	10	0.5	0.69	90%	335	3.245435	9.97×1E-4
100	100	0.1	14	0.3	0.87	70%	3598	46.151422	9.99×1E-4
100	100	0.1	16	0.35	0.83	80%	2277	31.450614	9.99×1E-4
200	200	0.1	10	0.2	0.49	80%	348	6.445658	9.90×1E-4
300	300	0.1	20	0.2	0.64	80%	588	39.801940	9.97×1E-4
400	400	0.1	30	0.2	0.72	80%	932	116.129172	9.96×1E-4
500	500	0.1	15	0.1	0.60	80%	1249	157.815187	9.97×1E-4
1000	1000	0.1	20	0.1	0.40	80%	363	223.761061	9.90×1E-4

(**C**) The third group test gives the numerical results for different rank r on the 200×200 matrices. In Table 5.3, we choose five different values of $p(p = 0.1, 0.3, 0.5, 0.7, 0.9)$ to test the efficiency of the algorithm under different rank $r(r = 10, 12, 14, 16, 18, 20)$. From Figure 5.3, it can be observed that no matter which p you choose, the cputime of iterations increases with the enhancement of the rank r.

TABLE 5.3: Numerical results for different r.

m	n	p	r	SR	FR	iter	time	res
200	200	0.1 0.3 0.5 0.7 0.9	10	0.5	0.19	25	0.512631 0.517313 0.502533 0.494348 0.514325	9.94×1E-4
200	200	0.1 0.3 0.5 0.7 0.9	12	0.5	0.23	29	0.622265 0.645685 0.631189 0.665399 0.628557	9.85×1E-4
200	200	0.1 0.3 0.5 0.7 0.9	14	0.5	0.27	30	0.707813 0.707156 0.708456 0.691847 0.694773	9.79×1E-4
200	200	0.1 0.3 0.5 0.7 0.9	16	0.5	0.31	38	0.916020 0.936906 0.934552 0.920090 0.923550	8.94×1E-4
200	200	0.1 0.3 0.5 0.7 0.9	18	0.5	0.34	40	1.056108 1.024769 1.054809 1.018782 1.016996	9.79×1E-4
200	200	0.1 0.3 0.5 0.7 0.9	20	0.5	0.38	47	1.284547 1.281448 1.315969 1.300480 1.318522	9.92×1E-4

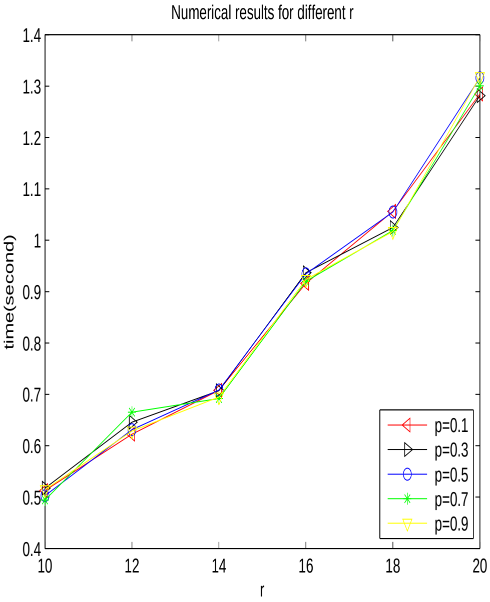

FIG. 5.3: Numerical results for different r.

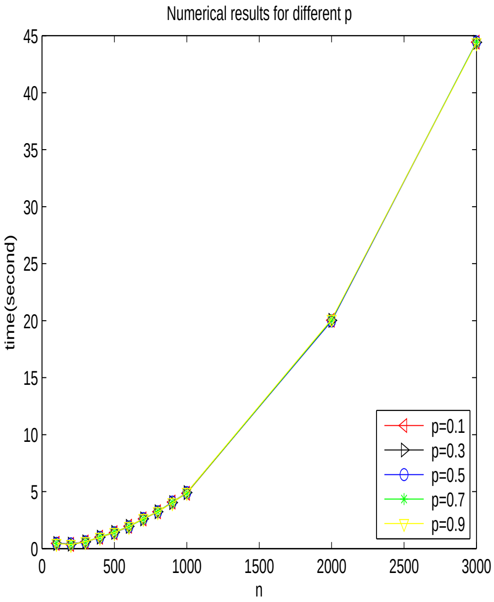

FIG. 5.4: Numerical results for different p.

(**D**) The last group test gives the numerical results for different p. Table 5.4 manifests the numerical results on the different size of matrices. Similar as Figure 5.3, our algorithm seems to be insensitive to the value of p, which is shown in Figure 5.4.

TABLE 5.4: Numerical results for different p ($p = 0.1, 0.3, 0.5, 0.7, 0.9$).

m	n	p	r	SR	FR	iter	time	res
100	100	0.1	10	0.57	0.33	37	0.456394	8.99×1E-4
		0.3					0.452486	
		0.5					0.453866	
		0.7					0.445611	
		0.9					0.438291	
200	200	0.1	10	0.57	0.17	19	0.372217	8.25×1E-4
		0.3					0.393038	
		0.5					0.418714	
		0.7					0.368374	
		0.9					0.364371	
300	300	0.1	10	0.57	0.11	15	0.579001	9.06×1E-4
		0.3					0.584276	
		0.5					0.590302	
		0.7					0.584102	
		0.9					0.608856	
400	400	0.1	10	0.57	0.08	14	1.005568	7.37×1E-4
		0.3					1.000946	
		0.5					0.989718	
		0.7					1.008598	
		0.9					0.990242	
500	500	0.1	10	0.57	0.06	13	1.396654	6.22×1E-4
		0.3					1.417247	
		0.5					1.399342	
		0.7					1.435313	
		0.9					1.406271	
600	600	0.1	10	0.57	0.058	12	1.958660	7.29×1E-4
		0.3					1.942243	
		0.5					1.910733	
		0.7					1.941489	
		0.9					1.935927	
700	700	0.1	10	0.57	0.049	12	2.599485	6.28×1E-4
		0.3					2.599628	
		0.5					2.631520	
		0.7					2.592599	
		0.9					2.629972	
800	800	0.1	10	0.57	0.044	11	3.236286	8.46×1E-4
		0.3					3.234253	
		0.5					3.235572	
		0.7					3.264344	
		0.9					3.228259	
900	900	0.1	10	0.57	0.039	11	4.074461	7.06×1E-4
		0.3					4.034496	
		0.5					4.079118	
		0.7					4.038276	
		0.9					4.019235	
1000	1000	0.1	10	0.57	0.035	11	4.858736	6.33×1E-4
		0.3					4.918483	
		0.5					4.916488	
		0.7					4.891239	
		0.9					4.931551	
2000	2000	0.1	10	0.57	0.017	10	20.040831	6.05×1E-4
		0.3					20.028482	
		0.5					19.964494	
		0.7					20.048369	
		0.9					20.107799	
3000	3000	0.1	10	0.57	0.011	9	44.434241	9.45×1E-4
		0.3					44.428431	
		0.5					44.456686	
		0.7					44.438987	
		0.9					44.375215	

5.2.2 Test for different algorithms without noisy under the known rank

In this subsection, we compare Algorithm MAMC with sIRLS [19], SVT [8] and FPCA [9]. The true rank of X_R is known priori and set the payoff parameter $\tau := 0$, i.e use the update strategy (5.3). In

Table 5.5, we list the numerical results for the algorithms on the different size of matrices. Numerical results show that our algorithm is very efficient, which can be seen in Figure 5.5.

TABLE 5.5: Numerical results for different algorithms.

m	100	200	300	400	500	600
n	100	200	300	400	500	600
p	0.9	0.9	0.9	0.9	0.9	0.9
r	10	10	10	10	10	10
SR	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.34	0.17	0.12	0.08	0.07	0.058
iter1	56	26	22	19	18	17
iter2	160	160	160	160	160	160
iter3	181	88	71	59	55	51
iter4	307	174	133	115	103	99
time1	0.69	0.52	0.77	1.15	2.01	2.68
time2	2.04	2.37	3.01	3.96	5.28	6.75
time3	4.18	2.00	2.20	2.09	3.36	3.96
time4	6.56	5.24	5.80	6.94	8.18	10.25
res1	9.9E-5	8.8E-5	9.1E-5	8.4E-5	7.8E-5	7.4E-5
res2	9.2E-5	4.6E-5	4.1E-5	3.5E-5	3.4E-5	3.3E-5
res3	2.5E-4	1.5E-4	1.3E-4	1.2E-4	1.2E-4	1.2E-4
res4	2.6E-5	9.9E-5	6.5E-5	4.7E-5	3.8E-5	3.1E-5
m	700	800	900	1000	2000	3000
n	700	800	900	1000	2000	3000
p	0.9	0.9	0.9	0.9	0.9	0.9
r	10	10	10	10	10	10
SR	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.050	0.044	0.039	0.035	0.018	0.012
iter1	16	15	15	15	13	13
iter2	160	160	160	160	160	160
iter3	51	48	46	45	37	34
iter4	93	89	86	81	81	73
time1	3.45	4.27	5.46	6.59	25.54	64.54
time2	8.39	10.42	12.60	14.99	52.93	127.19
time3	4.30	6.17	8.22	9.44	36.35	82.58
time4	10.29	11.72	17.04	23.39	66.42	148.14
res1	7.3E-5	9.7E-5	7.5E-5	6.2E-5	8.4E-5	5.6E-5
res2	3.3E-5	3.2E-5	3.1E-5	3.0E-5	2.8E-5	2.8E-5
res3	1.0E-4	1.0E-4	1.1E-4	1.0E-4	1.0E-4	1.0E-4
res4	2.7E-5	2.3E-5	2.0E-5	1.8E-5	8.9E-6	5.9E-6

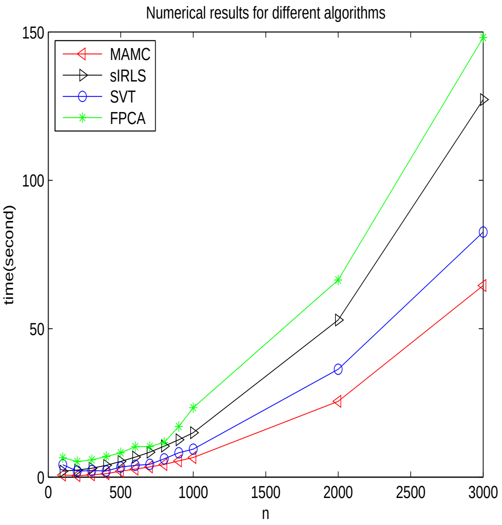

FIG. 5.5(a): The cputime for different algorithms.

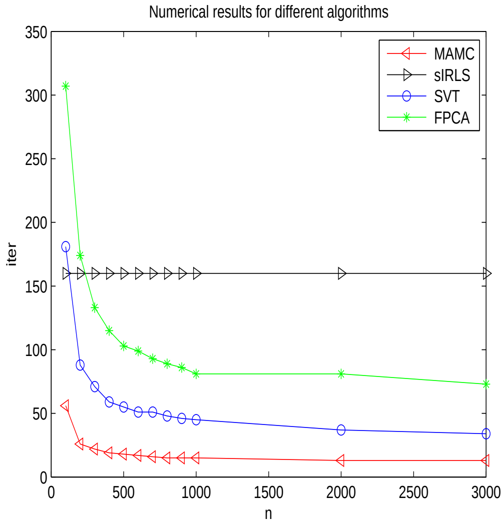

FIG. 5.5(b): The number of iterations for different algorithms.

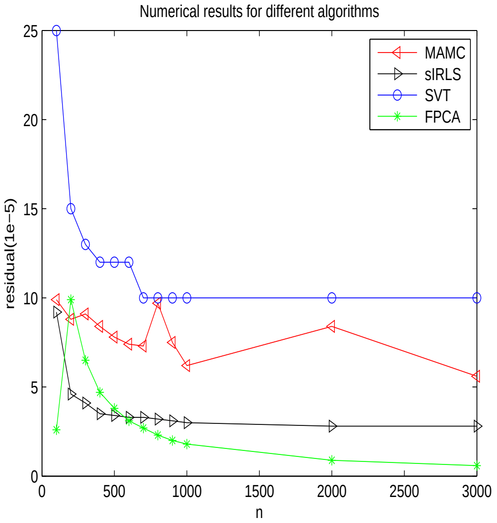

FIG. 5.5(c): The residual for different algorithms.

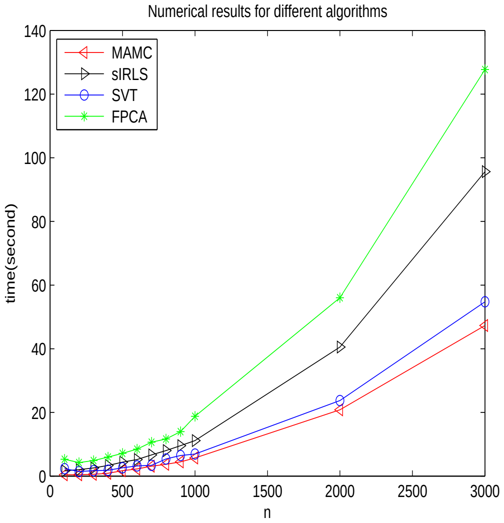

FIG. 5.6(a): The cputime for different algorithms.

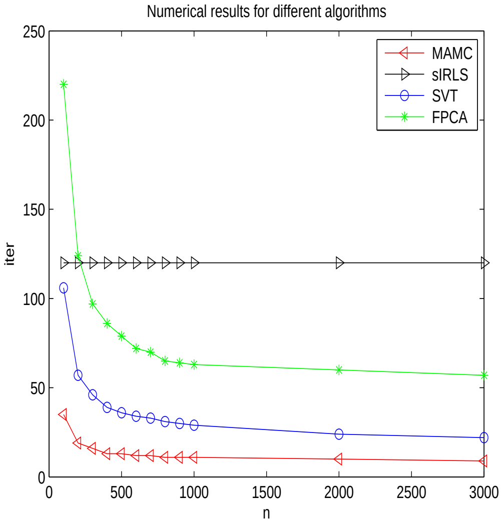

FIG. 5.6(b): The number of iterations for different algorithms.

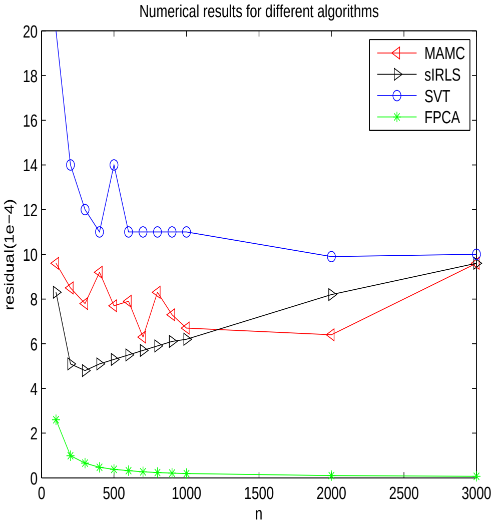

FIG. 5.6(c): The residual for different algorithms.

5.2.3 Test for different algorithms with noisy under the known rank

The numerical results for the above mentioned algorithms on randomly noisy matrix completion problems is demonstrated in this subsection. Now, we consider the following *noisy matrix completion problem*

$$\begin{array}{ll}\text{minimize} & \operatorname{rank}(X)\\ \text{subject to} & P_\Omega(X) = P_\Omega(X_R) + N,\end{array}$$

where N denotes the noise. We denote $NM := \text{rankn}(m, n)$ and set $N := (1e-4) * NM/\text{norm}(NM)$. The true rank of X_R is known priori. The tolerance of stop criterion and the payoff parameter are set to $tol := 1e-3, \tau := 1e-6$, respectively. Table 5.6 shows that MAMC algorithm has successful recovery for noisy matrix completion problem efficiently. Compared with other algorithms, Algorithm MAMC has a good performance as a whole, which can be seen in Figure 5.6.

TABLE 5.6: Numerical results for different algorithms.

m	100	200	300	400	500	600
n	100	200	300	400	500	600
p	0.9	0.9	0.9	0.9	0.9	0.9
r	10	10	10	10	10	10
SR	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.34	0.17	0.12	0.08	0.07	0.058
iter1	35	19	16	13	13	12
iter2	120	120	120	120	120	120
iter3	106	57	46	39	36	34
iter4	220	124	97	86	79	72
time1	0.44	0.41	0.56	0.87	1.69	2.26
time2	1.68	2.02	2.56	3.35	4.40	5.21
time3	2.34	1.44	1.69	1.83	2.63	3.19
time4	5.28	4.25	4.89	5.98	7.15	8.49
res1	9.6E-4	8.5E-4	7.8E-4	9.2E-4	7.7E-4	7.9E-4
res2	8.3E-4	5.1E-4	4.8E-4	5.1E-4	5.3E-4	5.5E-4
res3	2.0E-3	1.4E-3	1.2E-3	1.1E-3	1.4E-3	1.1E-3
res4	2.6E-4	9.9E-5	6.6E-5	4.7E-5	3.8E-5	3.2E-5
m	700	800	900	1000	2000	3000
n	700	800	900	1000	2000	3000
p	0.9	0.9	0.9	0.9	0.9	0.9
r	10	10	10	10	10	10
SR	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.050	0.044	0.039	0.035	0.018	0.012
iter1	12	11	11	11	10	9
iter2	120	120	120	120	120	120
iter3	33	31	30	29	24	22
iter4	70	65	64	63	60	57
time1	3.11	3.71	4.48	5.68	20.83	47.33
time2	6.67	8.04	9.54	11.13	40.54	95.64
time3	3.39	5.37	6.46	6.91	23.72	54.75
time4	10.64	11.64	13.98	18.79	56.01	127.69
res1	6.3E-4	8.3E-4	7.3E-4	6.7E-4	6.4E-4	9.6E-4
res2	5.7E-4	5.9E-4	6.1E-4	6.2E-4	8.2E-4	9.6E-4
res3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	9.9E-4	1.0E-3
res4	2.7E-5	2.4E-5	2.1E-5	1.9E-5	9.8E-6	6.9E-6

5.2.4 Test for different algorithms without noisy under the unknown rank

In this subsection, we report the numerical results of the above mentioned algorithms for exact matrix completion problems when the true rank of X_R is unknown. τ is set to $1e-6$. Similar as [19], we choose r to be $\min\{r_{\max}, \hat{r}\}$ and $\hat{r}$ is the largest integer such that $\sigma_{\hat{r}}(X^k) > 0.2\sigma_1(X^k)$. The numerical results are displayed in Table 5.7. In Table 5.8, we report the results for different r on the 1000×1000 matrices. From Figure 5.7 and Figure 5.8, we can see that our algorithm returns solutions of high quality.

TABLE 5.7: Numerical results for different algorithms.

m	100	200	300	400	500	600	700	800	900	1000	2000
n	100	200	300	400	500	600	700	800	900	1000	2000
p	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
r	10	10	10	10	10	10	10	10	10	10	10
SR	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.34	0.17	0.12	0.08	0.07	0.058	0.050	0.044	0.039	0.035	0.018
iter1	46	21	18	15	13	12	12	11	11	11	10
iter2	120	120	120	120	120	120	120	120	120	120	120
iter3	106	57	46	39	36	34	33	31	30	29	24
iter4	220	124	98	87	79	73	71	66	65	65	63
time1	0.87	0.69	1.15	1.59	2.52	2.67	3.19	3.44	4.33	5.34	21.55
time2	1.65	2.09	3.03	4.42	6.48	9.12	12.49	16.84	23.00	30.10	65.94
time3	2.33	1.44	1.69	1.77	2.66	3.24	4.07	4.63	6.32	6.45	23.63
time4	5.32	4.20	4.92	5.99	7.08	8.96	9.37	9.63	13.39	18.28	53.62
res1	9.8E-4	8.9E-4	8.3E-4	6.6E-4	10.0E-4	10.0E-4	6.6E-4	8.9E-4	7.2E-4	6.4E-4	6.2E-4
res2	8.8E-4	4.0E-4	3.3E-4	2.9E-4	2.8E-4	2.7E-4	2.6E-4	2.6E-4	2.6E-4	2.5E-4	2.3E-4
res3	2.0E-3	1.4E-3	1.2E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	9.9E-4
res4	2.6E-4	9.9E-5	6.5E-5	4.7E-5	3.8E-5	3.1E-5	2.7E-5	2.3E-5	2.0E-5	1.8E-5	8.9E-6

TABLE 5.8: Numerical results for different r.

m	1000	1000	1000	1000	1000	1000
n	1000	1000	1000	1000	1000	1000
p	0.9	0.9	0.9	0.9	0.9	0.9
r	10	12	14	16	18	20
SR	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.035	0.042	0.049	0.056	0.063	0.069
iter1	11	11	11	12	12	13
iter2	120	120	120	120	120	120
iter3	29	30	31	32	33	34
iter4	65	66	71	73	75	79
time1	5.31	6.22	6.69	8.40	9.23	11.76
time2	29.15	29.70	30.35	32.78	33.62	34.24
time3	6.26	7.66	8.80	9.33	9.64	10.44
time4	19.68	21.16	24.93	26.21	32.43	46.59
res1	6.6E-4	7.5E-4	9.3E-4	6.7E-4	8.2E-4	7.3E-4
res2	2.6E-4	2.5E-4	2.6E-4	2.6E-4	2.8E-4	2.9E-4
res3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3
res4	1.8E-5	1.8E-5	1.8E-5	1.8E-5	1.8E-5	1.8E-5

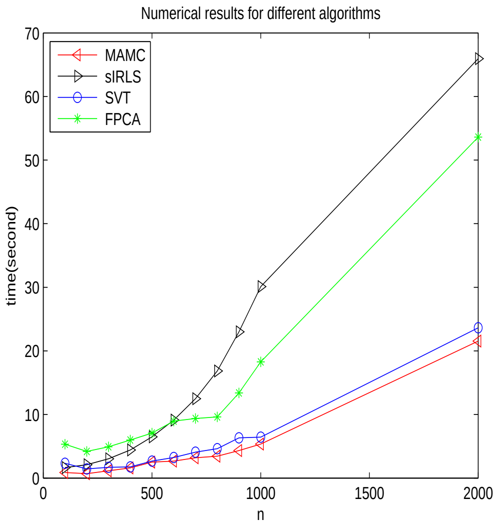

FIG. 5.7(a): The cputime for different algorithms.

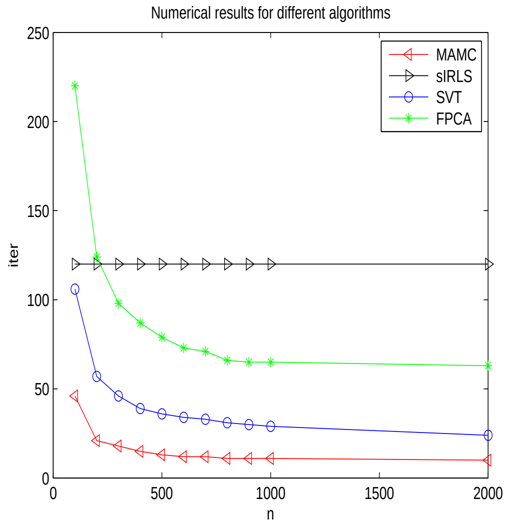

FIG. 5.7(b): The number of iterations for different algorithms.

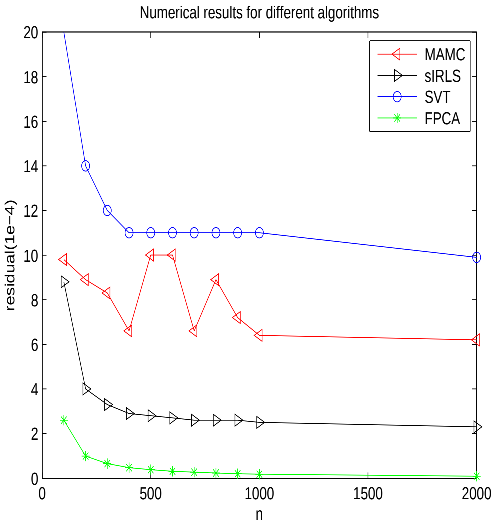

FIG. 5.7(c): The residual for different algorithms.

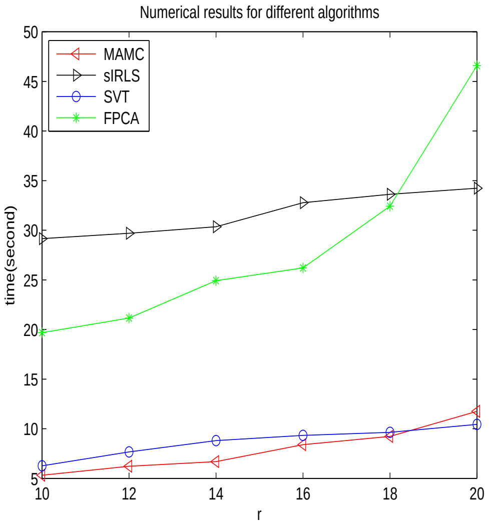

FIG. 5.8(a): The cputime for different r.

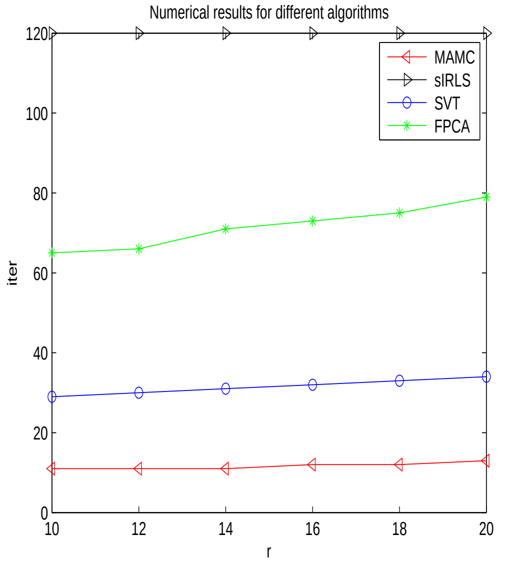

FIG. 5.8(b): The number of iterations for different r.

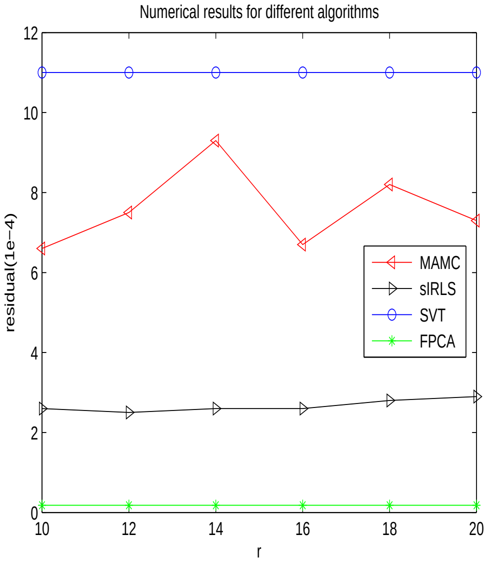

FIG. 5.8(c): The residual for different r.

5.2.5 Test for different algorithms with noisy under the unknown rank

We demonstrate the numerical results of the above mentioned algorithms for noisy matrix completion problems in this subsection. In this case, the true rank of X_R is unknown and the payoff parameter τ is set to $1e-6$. The strategy of estimating the true rank is same as that in the subsection 5.2.4. The results are displayed in Table 5.9 and Table 5.10. From Figure 5.9 and Figure 5.10, our algorithm MAMC is still more powerful than sIRLS, SVT and FPCA.

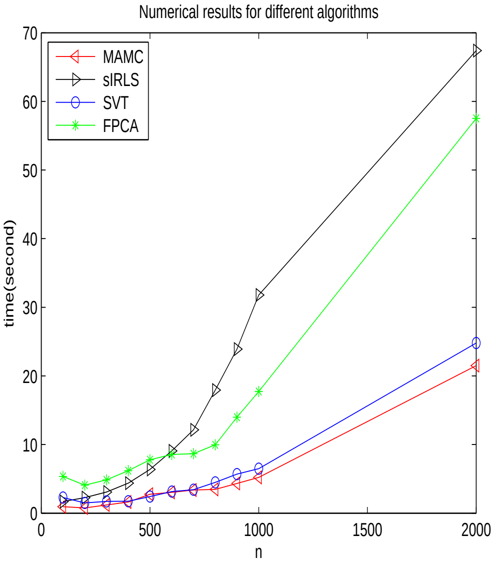

FIG. 5.9(a): The cuptime for different algorithms.

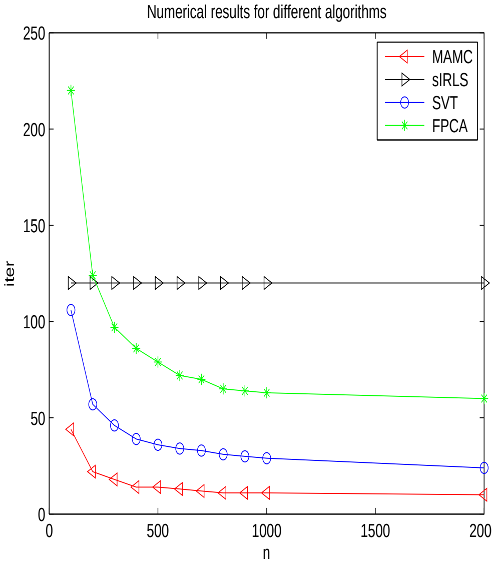

FIG. 5.9(b): The number of iterations for different algorithms.

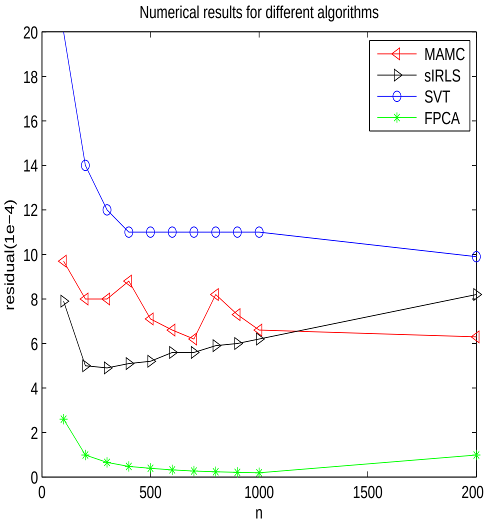

FIG. 5.9(c): The residual for different algorithms.

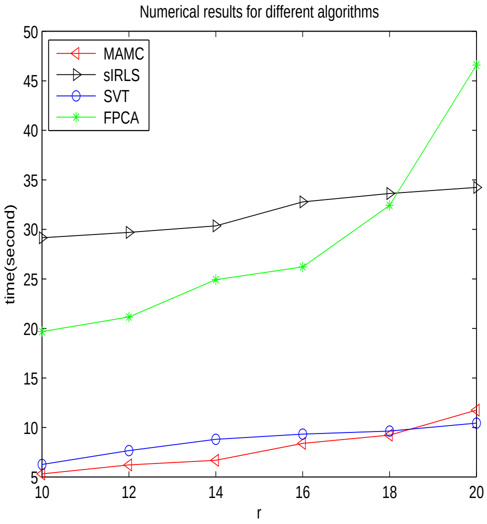

FIG. 5.10(a): The cputime for different r.

TABLE 5.9: Numerical results for different algorithms.

m	100	200	300	400	500	600	700	800	900	1000	2000
n	100	200	300	400	500	600	700	800	900	1000	2000
p	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
r	10	10	10	10	10	10	10	10	10	10	10
SR	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.34	0.17	0.12	0.08	0.07	0.058	0.050	0.044	0.039	0.035	0.018
iter1	44	22	18	14	14	13	12	11	11	11	10
iter2	120	120	120	120	120	120	120	120	120	120	120
iter3	106	57	46	39	36	34	33	31	30	29	24
iter4	220	124	97	86	79	72	70	65	64	63	60
time1	0.94	0.76	1.19	1.63	2.76	3.03	3.37	3.47	4.32	5.22	21.50
time2	1.70	2.26	3.08	4.38	6.37	9.07	12.15	17.93	23.90	31.79	67.41
time3	2.29	1.49	1.72	1.74	2.41	3.15	3.42	4.49	5.68	6.49	24.78
time4	5.35	4.08	4.86	6.20	7.79	8.54	8.67	9.97	13.98	17.73	57.52
res1	9.7E-4	8.0E-4	8.0E-4	8.8E-4	7.1E-4	6.6E-4	6.2E-4	8.2E-4	7.3E-4	6.6E-4	6.3E-4
res2	7.9E-4	5.0E-4	4.9E-4	5.1E-4	5.2E-4	5.6E-4	5.6E-4	5.9E-4	6.0E-4	6.2E-4	8.2E-4
res3	2.0E-3	1.4E-3	1.2E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	9.9E-4
res4	2.6E-4	9.9E-5	6.6E-5	4.8E-5	3.9E-5	3.2E-5	2.7E-5	2.4E-5	2.1E-5	1.9E-5	9.9E-5

TABLE 5.10: Numerical results for different r.

m	1000	1000	1000	1000	1000	1000
n	1000	1000	1000	1000	1000	1000
p	0.9	0.9	0.9	0.9	0.9	0.9
r	10	12	14	16	18	20
SR	0.57	0.57	0.57	0.57	0.57	0.57
FR	0.035	0.042	0.049	0.056	0.063	0.069
iter1	11	11	11	12	12	13
iter2	120	120	120	120	120	120
iter3	29	30	31	32	33	34
iter4	63	65	68	72	74	76
time1	5.25	5.63	7.26	8.59	8.99	11.89
time2	29.71	29.99	32.70	33.21	34.72	35.30
time3	6.51	7.12	9.16	9.20	9.55	10.37
time4	17.84	21.82	25.52	27.67	34.82	45.37
res1	7.0E-4	7.7E-4	6.1E-4	7.2E-4	8.2E-4	7.0E-4
res2	6.2E-4	5.9E-4	5.6E-4	5.3E-4	5.3E-4	5.2E-4
res3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.2E-3
res4	1.9E-5	1.9E-5	1.9E-5	1.9E-5	1.9E-5	1.9E-5

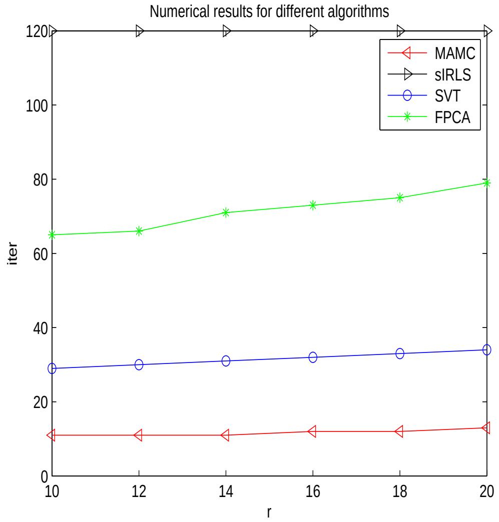

FIG. 5.10(b): The number of iterations for different r.

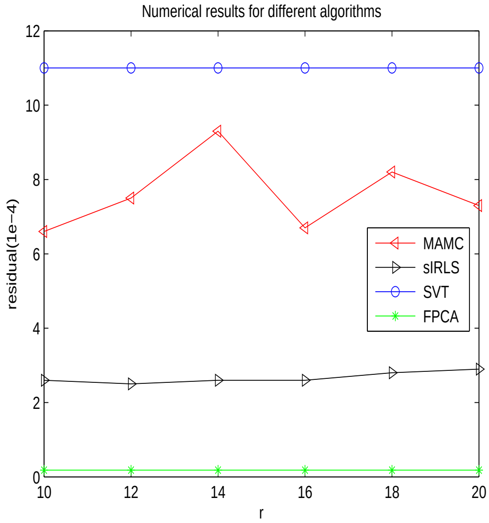

FIG. 5.10(c): The residual for different r.

5.2.6 Test for real problems

In this subsection, we implement our algorithm and sIRLS to tackle the matrix completion problem whose data taken from the well-known MovieLens data sets [37]. MovieLens data sets were collected by the GroupLens Research Project at the University of Minnesota. This data set consists of

(1) 100,000 ratings (1-5) from 943 users on 1682 movies.

(2) Each user has rated at least 20 movies.

(3) Simple demographic info for the users (age, gender, occupation, zip)

We consider MovieLens 100k data set for our numerical experiments. The MovieLens 100k data set includes four small splits such as (u1.base,u1.test), (u2.base,u2.test), (u3.base,u3.test), (u4.base,u4.test) and two medium splits such as (ua.base,ua.test), (ub.base,ub.test). To measure the accuracy of the completed matrix, as in Goldberg et al. [38] , we define the mean absolute error (MAE) of the output matrix X generated by the algorithm as follows

$$\text{MAE} := \frac{\sum_{(i,j)\in\Omega} |X_{ij} - M_{ij}|}{|\Omega|},$$

where Ω is the support set of M, $|\Omega|$ is the cardinality of Ω and M is generated by u*.test. The size of M equals $(\max(\text{u*.test}(:,1)), \max(\text{u*.test}(:,2)))$ and $M(\text{u*.test}(i,1), \text{u*.test}(i,2)) := \text{u*.test}(i,3)$. Similar as M, the initial matrix X^0 is generated by the data set u*.base.

The normalized mean absolute error (NMAE) is used to measure the accuracy of the approximated completion X,

$$\text{NMAE} := \frac{\text{MAE}}{r_{\max} - r_{\min}},$$

where $r_{\max}, r_{\min}$ denote the upper and the lower bounds of the ratings respectively, i.e.

$$r_{\max} = \max\{\text{u*.test}(:,3)\} \quad \text{and} \quad r_{\min} = \min\{\text{u*.test}(:,3)\}.$$

Furthermore, the stop criterion in this test is given by

$$\frac{\|X^k - X^{k-1}\|_F}{\|X^k\|_F} < tol := 1e - 3.$$

We choose $p = 0.1$ and set the estimate rank of X to be equal to 5. In Table 5.11, we report the NAME for our algorithm MAMC and sIRLS for different splits of MovieLens 100k data set.

TABLE 5.11: NAME for MAMC and sIRLS.

SPLITS	MAMC	sIRLS
(u1.base, u1.test)	0.186715	0.191970
(u2.base, u2.test)	0.184374	0.186562
(u3.base, u3.test)	0.184691	0.186755
(u4.base, u4.test)	0.185341	0.188907
(ua.base, ua.test)	0.193442	0.198626
(ub.base, ub.test)	0.196911	0.200994

From Table 5.11, our algorithm MAMC has a better NAME than sIRLS.

6. Concluding Remarks

In this paper, we propose a smoothing majorization method for solving the l_2-l_p matrix minimization problem, which is an approximation optimization model for the low rank recovery problem. The lower bound for nonzero singular values in any local optimal solution of the l_2-l_p problem is established. A smoothing function for the l_2-l_p objective function is used to design a smoothing majorization method for solve the l_2-l_p problem, in which the smoothing parameter is treated as a variable. Because our objective function is non-smooth, non-Lipschitz, non-convex function, the smoothing techniques and the idea of majorization are applied to alleviate these difficulties. The convergence theorem indicates that any accumulation point of the sequence generated by the smoothing majorization method satisfies the necessary optimality condition for the l_2-l_p problem. As an application, a smoothing majorization algorithm MAMC is present for solving the well-known matrix completion problem. Numerical experiments show that our algorithm can provide a high quality recovery solution more efficient compared with several well-known methods using nuclear norm regularization and seems to be insensitive to the choice of p, which is apparently different from sIRLS algorithm in this regard.

References

[1] C. Tomasi and T. Kanade, Shape and motion from image streams under orthography: a factorization method, Int. J. Comput. Vision, 9 (1992), pp. 137-154.

[2] N. Linial, E. London, and Y. Rabinovich, The geometry of graphs and some of its algorithmic applications, Combinatorica, 15 (1995), pp. 215-245.

[3] J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert, Low-rank matrix factorization with attributes, Technical Report N24/06/MM, Ecole des Mines de Paris, 2006.

[4] Y. Amit, M. Fink, N. Srebro, and S. Ullman, Uncovering shared structures in multiclass classification, In Proceedings of the 24th International Conference on Machine Learning, ACM, Providence, RI, 2007, pp. 17-24.

[5] A. Argyriou, T. Evgeniou, and M. Pontil, Multi-task feature learning, Adv. Neural Inform. Process. Syst., 19 (2007), pp. 41-48.

[6] M. Fazel, Matrix rank minimization with applications, Ph.D. thesis, Stanford University, 2002.

[7] M. Fazel, H. Hindi, and S. Boyd, A rank minimization heuristic with application to minimum order system approximation, in Proceedings of the American Control Conference, IEEE, 2001, pp. 4734-4739.

[8] J.F. Cai, E. J. Candes, and Z. Shen, A singular value thresholding algorithm for matrix completion, SIAM J. Optim., 20 (2010), pp. 1956-1982.

[9] S. Ma, D. Goldfarb, and L. Chen. Fixed point and Bregman iterative methods for matrix rank minimization, Math. Program., 128 (2011), pp. 321-353.

[10] K.C. Toh and S. Yun, An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems, Pac. J. Optim., 6 (2010), pp. 615-640.

[11] D. Gabay and B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation, Comput. Math. Appl., 2 (1976), pp. 17-40.

[12] E.J. Candès and T. Tao, Nearly optimal signal recovery from random projections: Universal encoding strategies, IEEE Trans. Info. Theory , 52 (2006), pp. 5406-5425.

[13] D. Donoho, Compressed sensing, IEEE Trans. Info. Theory, 52 (2006), pp. 1289-1306.

[14] X.J. Chen, F. Xu, and Y.Y. Ye, Lower bound theory of nonzero entries in solutions of l_2-l_p minimization, SIAM J. Sci. Comput., 32 (2011), pp. 2832-2852.

[15] X.J. Chen, Smoothing Methods for Nonsmooth, Nonconvex Minimization, Math. Program., 134 (2012), pp. 71-99.

[16] E.J. Candès, M.B. Wakin, and S.P. Boyd. Enhancing sparsity by reweighted l_1 minimization, J. Fourier Anal. and Appl., 14 (2008), pp. 877-905.

[17] R. Chartrand and W. Yin, Iteratively reweighted algorithms for compressive sensing, International Conference on Acoustics, Speech and Signal Processing, 2008, pp. 3869-3872.

[18] I. Daubechies, R. DeVore, M. Fornasier, and C.S. Gntrk. Iteratively reweighted least squares minimization for sparse recovery, Commun. Pur. Appl. Math., 63 (2010), pp. 1-38.

[19] K. Mohan and M. Fazel, Iterative reweighted least squares for matrix rank minimization, Proc. Allerton Conference on Communications, Control, and Computing, 2010, pp. 653-661.

[20] M.J. Lai, Y. Xu, and W. Yin, Low-rank matrix recovery using unconstrained smoothed-l_q minimization, Technical Report, Rice University, 2011.

[21] R. Chartrand and V. Staneva, Restricted isometry properties and nonconvex compressive sensing, Inverse Problems, 24 (2008), pp. 1-14.

[22] A.S. Lewis, Derivatives of spectral functions, Math. Oper. Res., 21 (1996), pp. 576-588.

[23] A.S. Lewis and H.S. Sendov, Twice differentiable spectral functions, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 368-386.

[24] H.D. Qi and X.Q. Yang, Semismoothness of spectral functions, SIAM J. Matrix Anal. Appl., 25 (2004), pp. 766-783.

[25] S.J. Li, K.L. Teo and X.Q. Yang, Second-order Directional derivatives of spectral function, Comput. Math. Appl., 50 (2005), pp. 947-955.

[26] D.F. Sun and J.Sun, Löwner's operator and spectral functions in Euclidean Jordan algebras, Math. Oper. Res., 33(2008), pp. 421-445.

[27] G.H. Golub and C.F. Van Loan, Matrix Computations. 3rd edition, The Johns Hopkins University Press, Baltimore, USA, 1996.

[28] C. Ding, D.F. Sun and K.C. Toh, An introduction to a class of matrix cone programming, Department of mathematics, National University of Singapore, preprint, 2010, http://www.optimization-online.org/DB_HTML/2010/09/2746.html.

[29] J.F. Bonnans and A. Shapiro, Perturbation Analysis of Optimization Problems, Springer, New York, 2000.

[30] J.M. Otega and W.C. Rheinboldt, Iterative solutions of nonlinear equations in several variables, Academic Press, New York, 1970.

[31] J. de Leeuw, Applications of convex analysis to multidimensional scaling, In J. R. Barra, F. Brodeau, G. Romier, and B. van Cutsem (Eds.), Recent developments in statistics, Amsterdam, The Netherlands, 1977, pp. 133-145.

[32] J. de Leeuw, Convergence of the majorization method for multidimensional scaling, Journal of classification, 5 (1988), pp. 163-180.

[33] J. de Leeuw, Block relaxation algorithms in statistics. In H. H. Bock, W. Lenski and M. M. Richter (Eds.), Information Systems and Data Analysis, Springer-Verlag., Berlin, 1994, pp. 308-325.

[34] J. de Leeuw, A decomposition method for weighted least squares low-rank approximation of symmetric matrices, Department of Statistics, UCLA, April 2006. Available at http://repositories.cdlib.org/uclastat/papers/2006041602.

[35] J. de Leeuw and W.J. Heiser, Convergence of correction matrix algorithms for multidimensional scaling. In J. C. Lingoes, I. Borg and E. E. C. I. Roskam (Eds.), Geometric Representations of Relational Data, Mathesis Press, 1977, pp. 735-752.

[36] Y. Gao and D. F. Sun, A majorized penalty approach for calibrating rank constrained correlation matrix problems, Preprint available at http://www.math.nus.edu.sg/∼matsundf/MajorPen.pdf.

[37] ”Moive Lens data”, http://www.grouplens.org/node/73.

[38] K. Goldberg, T. Roeder, D. Gupta and C. Perkins, Eigentaste, A constant time collaborative filtering algorithm. Inf. Retr., 4 (2001), pp. 133-151.

Appendix A.

We simplify the function $\varphi(\cdot)$ as follows

$$\begin{aligned}\varphi(z) &= F(U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T)\\ &= \frac{1}{2}\|\mathcal{A}(U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T)-b\|_2^2+\frac{\tau}{p}\|U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T\|_p^p\\ &= \varphi_1(z)+\frac{\tau}{p}\sum_{i=1}^r z_i^p,\end{aligned}$$

where $z := (z_1, z_2, \cdots, z_r)^T$ and $\varphi_1(z) := \frac{1}{2}\|\mathcal{A}(U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T)-b\|_2^2$. By the definition of the operator $\mathcal{A}$, we have

$$\mathcal{A}(U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T)=\begin{pmatrix}\langle A_1,U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T\rangle\\ \langle A_2,U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T\rangle\\ \vdots\\ \langle A_q,U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T\rangle\end{pmatrix}.$$

By the rule of trace operation for nonsymmetric matrices, we have

$$\begin{aligned}\langle A_i,U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T\rangle &= \mathbf{trace}(A_i^TU^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T)\\ &= \mathbf{trace}((V^*)^TA_i^TU^*[\mathbf{Diag}(z,0_{m-r})\ 0])\\ &= \langle (U^*)^TA_iV^*,[\mathbf{Diag}(z,0_{m-r})\ 0]\rangle.\end{aligned}$$

Denote $\tilde{A}_i := (U^*)^TA_iV^*(i=1,2,\cdots,q)$, we obtain that

$$\langle A_i,U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T\rangle=\langle \tilde{A}_i,[\mathbf{Diag}(z,0_{m-r})\ 0]\rangle, \tag{0.1}$$

where $\tilde{A}_i$ and $[\mathbf{Diag}(z,0_{m-r})\ 0]$ have the following form respectively,

$$\tilde{A}_i=\begin{pmatrix}\tilde{a}_{11}^i & \tilde{a}_{12}^i & \cdots & \tilde{a}_{1n}^i\\ \tilde{a}_{21}^i & \tilde{a}_{22}^i & \cdots & \tilde{a}_{2n}^i\\ \vdots & \vdots & & \vdots\\ \tilde{a}_{m1}^i & \tilde{a}_{m2}^i & \cdots & \tilde{a}_{mn}^i\end{pmatrix},\quad [\mathbf{Diag}(z,0_{m-r})\ 0]=\left(\begin{array}{ccccc|c} z_1 & & & & & \\ & \ddots & & & & \\ & & z_r & & & \mathbf{0}\\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{array}\right).$$

Hence, we have

$$
\begin{aligned}
\tilde{A}_i^T[\mathbf{Diag}(z,0_{m-r})\ 0] &= \begin{pmatrix} \tilde{a}_{11}^i & \tilde{a}_{21}^i & \cdots & \tilde{a}_{m1}^i \\ \tilde{a}_{12}^i & \tilde{a}_{22}^i & \cdots & \tilde{a}_{m2}^i \\ \vdots & \vdots & & \vdots \\ \tilde{a}_{1n}^i & \tilde{a}_{2n}^i & \cdots & \tilde{a}_{mn}^i \end{pmatrix} \left(\begin{array}{cccccc|c} z_1 & & & & & & \\ & \ddots & & & & & \\ & & z_r & & & & \\ & & & 0 & & & \mathbf{0} \\ & & & & \ddots & & \\ & & & & & 0 & \end{array}\right) \\
&= \left(\begin{array}{cccc|c} z_1\tilde{a}_{11}^i & z_2\tilde{a}_{21}^i & \cdots & z_r\tilde{a}_{r1}^i & \\ z_1\tilde{a}_{12}^i & z_2\tilde{a}_{22}^i & \cdots & z_r\tilde{a}_{r2}^i & \\ \vdots & \vdots & & \vdots & \mathbf{0} \\ z_1\tilde{a}_{1n}^i & z_2\tilde{a}_{2n}^i & \cdots & z_r\tilde{a}_{rn}^i & \end{array}\right)_{n\times n}. \qquad (0.2)
\end{aligned}
$$

By the relationship (0.1) and (0.2), we obtain that

$$
\begin{aligned}
\langle A_i, U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T\rangle &= \langle \tilde{A}_i, [\mathbf{Diag}(z,0_{m-r})\ 0]\rangle \\
&= \mathbf{trace}(\tilde{A}_i^T[\mathbf{Diag}(z,0_{m-r})\ 0]) \\
&= z_1\tilde{a}_{11}^i + z_2\tilde{a}_{22}^i + \cdots + z_r\tilde{a}_{rr}^i \\
&= \begin{pmatrix} \tilde{a}_{11}^i\ \tilde{a}_{22}^i\ \cdots\ \tilde{a}_{rr}^i \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_r \end{pmatrix}.
\end{aligned}
$$

If we denote the matrix $\tilde{A}$ as follows,

$$
\tilde{A} := \begin{pmatrix} \tilde{a}_{11}^1 & \tilde{a}_{22}^1 & \cdots & \tilde{a}_{rr}^1 \\ \tilde{a}_{11}^2 & \tilde{a}_{22}^2 & \cdots & \tilde{a}_{rr}^2 \\ \vdots & \vdots & & \vdots \\ \tilde{a}_{11}^q & \tilde{a}_{22}^q & \cdots & \tilde{a}_{rr}^q \end{pmatrix},
$$

then we have

$$
\mathcal{A}(U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T) = \tilde{A}z.
$$

Hence,

$$
\varphi_1(z) = \frac{1}{2}\|\mathcal{A}(U^*[\mathbf{Diag}(z,0_{m-r})\ 0](V^*)^T) - b\|_2^2 = \frac{1}{2}\|\tilde{A}z - b\|_2^2.
$$

and

$$
\varphi(z) = \varphi_1(z) + \frac{\tau}{p}\sum_{i=1}^r z_i^p = \frac{1}{2}\|\tilde{A}z - b\|_2^2 + \frac{\tau}{p}\sum_{i=1}^r z_i^p.
$$

Appendix B.

By the definitions of $Q(X)$ and $\Xi(H)$, we have

$$
\begin{aligned}
& Q(X)^T\Xi(H)Q(X) \\
={}& \frac{1}{2}\begin{pmatrix} U(X)^T & V_1(X)^T \\ 0 & \sqrt{2}V_2(X)^T \\ (U(X)^\uparrow)^T & (-V_1(X)^\uparrow)^T \end{pmatrix} \begin{pmatrix} 0 & H \\ H^T & 0 \end{pmatrix} \begin{pmatrix} U(X) & 0 & U(X)^\uparrow \\ V_1(X) & \sqrt{2}V_2(X) & -V_1(X)^\uparrow \end{pmatrix} \\
={}& \frac{1}{2}\begin{pmatrix} V_1(X)^TH^T & U(X)^TH \\ \sqrt{2}V_2(X)^TH^T & 0 \\ (-V_1(X)^\uparrow)^TH^T & (U(X)^\uparrow)^TH \end{pmatrix} \begin{pmatrix} U(X) & 0 & U(X)^\uparrow \\ V_1(X) & \sqrt{2}V_2(X) & -V_1(X)^\uparrow \end{pmatrix} \\
={}& \frac{1}{2}\begin{pmatrix} A + A^T & \sqrt{2}B & (A^T - A)I_m^\uparrow \\ \sqrt{2}B^T & 0 & \sqrt{2}B^TI_m^\uparrow \\ I_m^\uparrow(A - A^T) & \sqrt{2}I_m^\uparrow B & -I_m^\uparrow(A + A^T)I_m^\uparrow \end{pmatrix},
\end{aligned}
$$

where $A := U(X)^T HV_1(X),\ B := U(X)^T HV_2(X)$. For simplify, we divide $\mathbf{Diag}\mathrm{D}f_\epsilon(\lambda(\Xi(X))$ into three blocks as follows

$$\mathbf{Diag}\mathrm{D}f_\epsilon(\lambda(\Xi(X)) = \begin{pmatrix} W_1 & & \\ & W_2 & \\ & & W_3 \end{pmatrix},$$

where

$$\begin{array}{rcl} W_1 & := & \mathbf{Diag}[(w_1, w_2, \cdots, w_m)^T], \\ W_2 & := & \mathbf{Diag}[(w_{m+1}, w_{m+2}, \cdots, w_n)^T], \\ W_3 & := & \mathbf{Diag}[(w_{n+1}, w_{n+2}, \cdots, w_{n+m})^T]. \end{array}$$

So we have

$$\begin{array}{rl} & \langle [\mathbf{Diag}\mathrm{D}f_\epsilon(\lambda(\Xi(X)))] \, , Q(X)^T \Xi(H) Q(X) \rangle \\ = & \left\langle \begin{pmatrix} W_1 & & \\ & W_2 & \\ & & W_3 \end{pmatrix}, \frac{1}{2} \begin{pmatrix} A + A^T & \sqrt{2}B & (A^T - A)I_m^\uparrow \\ \sqrt{2}B^T & 0 & \sqrt{2}B^T I_m^\uparrow \\ I_m^\uparrow (A - A^T) & \sqrt{2} I_m^\uparrow B & -I_m^\uparrow (A + A^T) I_m^\uparrow \end{pmatrix} \right\rangle \\ = & \frac{1}{2}\mathbf{trace}\left(\begin{pmatrix} W_1(A + A^T) & \sqrt{2}W_1 B & W_1(A^T - A)I_m^\uparrow \\ \sqrt{2}W_2 B^T & 0 & \sqrt{2}W_2 B^T I_m^\uparrow \\ W_3 I_m^\uparrow (A - A^T) & \sqrt{2}W_3 I_m^\uparrow B & -W_3 I_m^\uparrow (A + A^T) I_m^\uparrow \end{pmatrix} \right) \\ = & \frac{1}{2}\mathbf{trace}(W_1(A + A^T)) - \frac{1}{2}\mathbf{trace}(W_3 I_m^\uparrow (A + A^T) I_m^\uparrow) \\ = & (\mathbf{diag}A)^T \begin{pmatrix} w_1 - w_{n+m} \\ w_2 - w_{n+m-1} \\ \vdots \\ w_{m-1} - w_{n+2} \\ w_m - w_{n+1} \end{pmatrix} = 2(\mathbf{diag}A)^T \begin{pmatrix} p\sigma_1(X)(\sigma_1^2(X) + \epsilon^2)^{\frac{p}{2}-1} \\ p\sigma_2(X)(\sigma_2^2(X) + \epsilon^2)^{\frac{p}{2}-1} \\ \vdots \\ p\sigma_{m-1}(X)(\sigma_{m-1}^2(X) + \epsilon^2)^{\frac{p}{2}-1} \\ p\sigma_m(X)(\sigma_m^2(X) + \epsilon^2)^{\frac{p}{2}-1} \end{pmatrix}. \end{array}$$

Appendix C.

By the equation (4.6), we obtain that

$$\begin{array}{rcl} L_1 & = & \frac{1}{2}\left(\mathcal{A}(X^{k+1}) + \mathcal{A}(X^k) - 2b\right)^T \left(\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\right) \\ & & + \langle \mathrm{D}_X F_2(\epsilon^k, X^k), X^{k+1} - X^k \rangle + \frac{\tau\rho^k}{2}\|X^{k+1} - X^k\|_F^2. \\ & = & \frac{1}{2}\left(\mathcal{A}(X^{k+1}) - b\right)^T \left(\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\right) + \frac{1}{2}\left(\mathcal{A}(X^k) - b\right)^T \left(\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\right) \\ & & - \left(\mathcal{A}(X^{k+1}) - b\right)^T \left(\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\right) - \tau\rho^k \|X^{k+1} - X^k\|_F^2 \\ & & + \frac{\tau\rho^k}{2}\|X^{k+1} - X^k\|_F^2 \\ & = & -\frac{1}{2}\left(\mathcal{A}(X^{k+1}) - b\right)^T \left(\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\right) + \frac{1}{2}\left(\mathcal{A}(X^k) - b\right)^T \left(\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\right) \\ & & - \frac{\tau\rho^k}{2}\|X^{k+1} - X^k\|_F^2 \\ & = & -\frac{1}{2}\|\mathcal{A}(X^{k+1}) - \mathcal{A}(X^k)\|_2^2 - \frac{\tau\rho^k}{2}\|X^{k+1} - X^k\|_F^2. \end{array}$$

and

$$\begin{array}{rcl} L_2 & = & \mathrm{D}_\epsilon F_2(\epsilon^k, X^k) \cdot (\epsilon^{k+1} - \epsilon^k) + \frac{\tau\rho^k}{2}(\epsilon^{k+1} - \epsilon^k)^2 \\ & = & -\tau\rho^k(\epsilon^{k+1} - \epsilon^k)^2 + \frac{\tau\rho^k}{2}(\epsilon^{k+1} - \epsilon^k)^2 = -\frac{\tau\rho^k}{2}(\epsilon^{k+1} - \epsilon^k)^2. \end{array}$$