Journal of Inequalities and Applications
a SpringerOpen Journal

RESEARCH Open Access

A semi-smoothing augmented Lagrange multiplier algorithm for low-rank Toeplitz matrix completion

Ruiping Wen[1*], Shuzhen Li[2] and Yonghong Duan[3]

*Correspondence: wenrp@163.com
[1]Key Laboratory of Engineering & Computing Science, Shanxi Provincial Department of Education/Department of Mathematics, Taiyuan Normal University, Jinzhong, P.R. China
Full list of author information is available at the end of the article

Abstract

The smoothing augmented Lagrange multiplier (SALM) algorithm is a generalization of the augmented Lagrange multiplier algorithm for completing a Toeplitz matrix, which saves computational cost of the singular value decomposition (SVD) and approximates well the solution. However, the communication of numerous data is computationally demanding at each iteration step. In this paper, we propose an accelerated scheme to the SALM algorithm for the Toeplitz matrix completion (TMC), which will reduce the extra load coming from data communication under reasonable smoothing. It has resulted in a semi-smoothing augmented Lagrange multiplier (SSALM) algorithm. Meanwhile, we demonstrate the convergence theory of the new algorithm. Finally, numerical experiments show that the new algorithm is more effective/economic than the original algorithm.

Keywords: Toeplitz matrix; Completion; Augmented Lagrange multiplier; Data communication

1 Introduction

Completing a low-rank matrix from a subset of its entries has been a hot problem recently, first introduced by [8], that has arisen in a wide variety of practical contexts across all disciplines of engineering and computational science such as model reduction [19], machine learning [1, 2], control [22], pattern recognition [12], imaging inpainting [3], video denoising [16], computer vision [28], and so on. Despite matrix completion (MC) requiring the global solution of a non-convex objective, there are many computational efficient algorithms which are effective for a broad class of matrices. The problem has received intensive research from both theoretical and algorithmic aspects, see, e.g., [4–11, 13–18, 21, 23, 26, 27, 29–32, 34, 35], and the references therein for partial review. It is well known that the mathematical model of the MC problem is of the following form:

$$\begin{aligned} &\min_{A\in\mathbb{R}^{m\times n}} \|A\|_*, \\ &\text{subject to } \mathcal{P}_\Omega(A) = \mathcal{P}_\Omega(M), \end{aligned} \tag{1.1}$$

where the matrix $M \in \mathbb{R}^{m\times n}$ is an underlying matrix to be completed, Ω is a random subset of indices for the available entries, and $\mathcal{P}_\Omega$ is the associated sampling orthogonal projection operator which acquires only the entries indexed by $\Omega \subset \{1,2,\dots,m\} \times \{1,2,\dots,n\}$.

In the current MC problems, the matrix M is of special structure in general. Therefore, much attention has been paid to the completion of Toeplitz and Hankel matrices in recent years [20, 24, 25, 30, 32]. Many scholars have conducted in-depth research on the special structure, property, and application of the Toeplitz and Hankel matrices; for example, nuclear norm minimization for the low-rank Hankel matrix reconstruction problem under the random Gaussian sampling model is investigated in [7]. In addition, Hankel matrix reconstitution in the sense of minimizing the nuclear norm with non-uniform sampling of entries is researched in [11]. To make full use of the special structure of a Toeplitz matrix, a mean value algorithm is presented in [30]; the modified Lagrange multiplier (MALM) algorithm [31] and the smoothing augmented Lagrange multiplier (SALM) algorithm [34] are also proposed. Therefore, the Toeplitz matrix completion (TMC) is one of the most important MC problems and has attracted a large amount of attention recently. As is well known, an $n \times n$ Toeplitz matrix is of the following form:

$$T = (t_{j-i})_{i,j=1}^{n} = \begin{pmatrix} t_0 & t_1 & \cdots & t_{n-2} & t_{n-1} \\ t_{-1} & t_0 & \cdots & t_{n-3} & t_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ t_{-n+2} & t_{-n+3} & \cdots & t_0 & t_1 \\ t_{-n+1} & t_{-n+2} & \cdots & t_{-1} & t_0 \end{pmatrix} \in \mathbb{T}^{n\times n} \subset \mathbb{R}^{n\times n}, \tag{1.2}$$

which is determined by $2n-1$ entries, say, the first row and the first column. Explicitly seeking the lowest rank Toeplitz matrix consistent with the known entries is mathematically considered as

$$\begin{aligned} &\min_{A\in\mathbb{T}^{n\times n}} \|A\|_*, \\ &\text{subject to } H \circ A = M, \end{aligned} \tag{1.3}$$

where "$\circ$" is the Hadamard product, $H = (H_{ij}) \in \mathbb{R}^{n\times n}$ is the weighted matrix with entries $H_{ij} = 1$ for $j - i \in \Omega \subset \{-n+1,\dots,n-1\}$ and $H_{ij} = 0$ for any other (i,j), $M = (M_{ij}) \in \mathbb{T}^{n\times n}$ is the underlying Toeplitz matrix to be completed, namely $M_{ij} = 0$ for $j - i \notin \Omega$.

The SALM algorithm switches the iteration matrix into the Toeplitz structure at each iteration step by the smoothing operator, which saves computational cost of the singular value decomposition and approximates well the solution. Unfortunately, numerous data have to be shifted at each iteration step in the process of implementing this algorithm. However, there is a cost, sometimes relatively great, associated with the moving of data. The control of memory traffic is crucial to performance in many computers.

These factors motivated us to reduce the traffic jam of data, resulting in a semi-smoothing augmented Lagrange multiplier (SSALM) algorithm based on the selecting technique of the optimal parameter $\omega^{(k)}$ at each of the five iteration steps in [33]. Compared with the SALM algorithm, the new algorithm either saves computation cost or reduces data communication. Two aspects are taken into account, which results in more practical or economic implementation. The new algorithm not only overcomes the slowness of the SVD for the original ALM algorithm, but also reduces the greatness of the data

communication for the ALM algorithm. We can see that the CPU of SSALM algorithm is reduced to 30.44% from the numerical experiments.

The rest of this paper is organized as follows. Some preliminaries are provided in Sect. 2. Section 3 presents the semi-smoothing augmented Lagrange multiplier (SSALM) algorithm after giving an outline of the ALM algorithm, the dual approach, and the SALM algorithm. The convergence property of the SSALM algorithm is constructed in Sect. 4. We report the numerical results to indicate the effectiveness of the SSALM algorithm in Sect. 5. Finally, we end the paper with the concluding remarks in Sect. 6.

2 Preliminaries

This section is devoted to some of the necessary notations and preliminaries. $\mathbb{R}^{m\times n}$ denotes the set of $m \times n$ real matrices, $\mathbb{T}^{n\times n}$ is the set of $n \times n$ real Toeplitz matrices. The nuclear norm of a matrix A is denoted by $\|A\|_*$, and the Frobenius norm $\|A\|_F$ is the maximum absolute value of the matrix entries of a matrix A. A^T is used to express the transpose of a matrix $A \in \mathbb{R}^{n\times n}$, $\operatorname{rank}(A)$ is equal to the rank of a matrix A, and $\operatorname{tr}(A)$ represents the trace of A. The standard inner product between two matrices is denoted by $\langle X, Y\rangle = \operatorname{tr}(X^T Y)$. For $A = (a_{ij}) \in \mathbb{R}^{m\times n}$, $B = (b_{ij}) \in \mathbb{R}^{m\times n}$, their Hadamard product $A \circ B$ is an $m \times n$ matrix whose (i,j) entry is the $a_{ij}b_{ij}$, i.e., $A \circ B = (a_{ij}b_{ij}) \in \mathbb{R}^{m\times n}$.

The singular value decomposition (SVD) of a matrix $A \in \mathbb{R}^{m\times n}$ of r-rank is defined by

$$A = U\Sigma_r V^T, \qquad \Sigma_r = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r),$$

where $U \in \mathbb{R}^{m\times r}$ and $V \in \mathbb{R}^{n\times r}$ are column orthonormal matrices, and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$.

Definition 2.1 (Singular value thresholding operator [6]) For each $\tau \geq 0$, the singular value thresholding operator $\mathcal{D}_\tau$ is defined as follows:

$$\mathcal{D}_\tau(A) := U\mathcal{D}_\tau(\Sigma)V^T, \qquad \mathcal{D}_\tau(\Sigma) = \operatorname{diag}\big(\{\sigma_i - \tau\}_+\big),$$

where

$$A = U\Sigma_r V^T \in \mathbb{R}^{m\times n}, \qquad \{\sigma_i - \tau\}_+ = \begin{cases} \sigma_i - \tau, & \text{if } \sigma_i > \tau, \\ 0, & \text{if } \sigma_i \leq \tau. \end{cases}$$

$I_n = (e_1, e_2, \ldots, e_n) \in \mathbb{R}^{n\times n}$ denotes the $n \times n$ identity matrix and $Z_n = (e_2, e_3, \ldots, e_n, 0) \in \mathbb{R}^{n\times n}$ is called the shift matrix. It is clear that

$$Z_n^r = \begin{cases} \begin{pmatrix} O & O \\ I_{n-r} & O \end{pmatrix}, & 1 < r < n, \\ O, & r \geq n, \end{cases}$$

where "O" stands for a zero-matrix. Thus, a Toeplitz matrix $T \in \mathbb{T}^{n\times n}$, shown in (1.2), can be written as a linear combination of these shift matrices, that is,

$$T = \sum_{l=1}^{n-1} t_{-l} Z_n^l + \sum_{l=0}^{n-1} t_l \big(Z_n^T\big)^l.$$

$\Omega \subset \{-n+1,\dots,n-1\}$ is an indices set of observed diagonals of a Toeplitz matrix $M \in \mathbb{T}^{n\times n}$, $\bar{\Omega}$ is the complementary set of Ω. For any Toeplitz matrix $A \in \mathbb{T}^{n\times n}$, the vector $\mathrm{vec}(A,\alpha)$ denotes the αth diagonal of A, $\alpha = -n+1,-n+2,\dots,n-1$, that is to say,

$$\mathrm{vec}(H\circ A,\alpha) = \begin{cases} \mathrm{vec}(A,\alpha), & \alpha \in \Omega, \\ \mathbf{0}, & \alpha \notin \Omega, \end{cases} \quad (\mathbf{0} \text{ is a zero-vector}).$$

Definition 2.2 (Toeplitz structure smoothing operator [34]) For any matrix $A = (a_{ij}) \in \mathbb{R}^{n\times n}$, the Toeplitz structure smoothing operator $\mathcal{T}$ is defined as follows:

$$\mathcal{T}(A) := \sum_{l=1}^{n-1} \tilde{a}_{-l} Z_n^l + \sum_{l=0}^{n-1} \tilde{a}_l \left(Z_n^T\right)^l, \tag{2.1}$$

where $\tilde{a}_\alpha = \frac{{}_\alpha A^{\min} + {}_\alpha A^{\max}}{2}$, $\alpha = -n+1,-n+2,\dots,n-1$ with

$${}_\alpha A^{\min} = \min_{i,j\in\{1,2,\dots,n\}} \{a_{ij}, i-j=\alpha\}, \quad \text{and} \quad {}_\alpha A^{\max} = \max_{i,j\in\{1,2,\dots,n\}} \{a_{ij}, i-j=\alpha\}.$$

It is clear that $\mathcal{T}(A)$ is a Toeplitz matrix derived from the matrix A. Namely, any $A \in \mathbb{R}^{n\times n}$ can be changed into a Toeplitz structure via the smoothing operator $\mathcal{T}(\cdot)$.

3 Algorithms

First of all in this section, for completeness as well as for the purpose of comparison, we briefly review and summarize some relative algorithms for approximately minimizing the nuclear norm of a matrix under convex constraints.

Since the matrix completion problem is closely connected to the robust principal component analysis (RPCA) problem, then it can be formulated in the same way as RPCA, an equivalent problem of (1.1) can be considered as follows.

As E will compensate for the unknown entries of M, the unknown entries of M are simply set as zeros. Suppose that the given data are arranged as the columns of a large matrix $M \in \mathbb{R}^{m\times n}$. The mathematical model for estimating the low-dimensional subspace is to find a low-rank matrix $A \in \mathbb{R}^{m\times n}$ such that the discrepancy between A and M is minimized, leading to the following constrained optimization:

$$\begin{aligned} &\min_{A,E\in\mathbb{R}^{m\times n}} \|A\|_*, \\ &\text{subject to } A+E=M, \mathcal{P}_\Omega(E)=0, \end{aligned} \tag{3.1}$$

where E will compensate for the unknown entries of M, the unknown entries of $M \in \mathbb{R}^{m\times n}$ are simply set as zeros. And $\mathcal{P}_\Omega : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ is a linear operator that keeps the entries in Ω unchanged and sets those outside Ω (say, in $\overline{\Omega}$) zeros.

3.1 The dual algorithm

The dual algorithm proposed in [10] tackles problem (3.1) via its dual. That is, one first solves the dual problem

$$\begin{aligned} &\max_Y \langle M, Y\rangle, \\ &\text{subject to } J(Y) \le 1, \end{aligned} \tag{3.2}$$

for the optimal Lagrange multiplier Y, where

$$J(Y) = \max\bigl(\|Y\|_2, \lambda^{-1}\|Y\|_\infty\bigr). \tag{3.3}$$

A steepest ascend algorithm constrained on the surface $\{Y|J(Y) = 1\}$ can be adopted to solve (3.2), where the constrained steepest ascend direction is obtained by projecting M onto the tangent cone of the convex body $\{Y|J(Y) \leq 1\}$. It turns out that the optimal solution to the primal problem (3.1) can be obtained during the process of finding the constrained steepest ascend direction.

3.2 The augmented Lagrange multiplier algorithm

The augmented Lagrange multiplier (ALM) algorithm was proposed in [18] for solving a convex optimization (1.1). It should be described subsequently.

It is famous that the partial augmented Lagrangian function of problem (3.1) is

$$\mathcal{L}(A, E, Y, \mu) = \|A\|_* + \langle Y, M - A - E\rangle + \frac{\mu}{2}\|M - A - E\|_F^2. \tag{3.4}$$

Hence, the augmented Lagrange multiplier algorithm is designed as follows.

Algorithm 3.1 ([18]) Given a sampled set Ω, a sampled matrix $D = \mathcal{P}_\Omega(M)$, $\mu_0 > 0$, $\rho > 1$. Given also two initial matrices $Y_0 = 0$, $E_0 = 0$. $k := 0$.

1. Compute the SVD of the matrix $(D - E_k + \mu_k^{-1}Y_k)$:

$$[U_k, \Sigma_k, V_k] = \operatorname{svd}\bigl(D - E_k + \mu_k^{-1}Y_k\bigr);$$

2. Set

$$A_{k+1} = U_k\mathcal{D}_{\mu_k^{-1}}(\Sigma_k)V_k^T,$$

solve $E_{k+1} = \arg\min_{\mathcal{P}_\Omega(E)=0}\mathcal{L}(A_{k+1}, E, Y_k, \mu_k)$,

$$E_{k+1} = \mathcal{P}_{\bar{\Omega}}\bigl(D - A_{k+1} + \mu_k^{-1}Y_k\bigr);$$

3. If $\|D - A_{k+1} - E_{k+1}\|_F/\|D\|_F < \epsilon_1$ and $\mu_k\|E_{k+1} - E_k\|_F/\|D\|_F < \epsilon_2$, stop; otherwise, go to the next step;
4. Set $Y_{k+1} = Y_k + \mu_k(D - A_{k+1} - E_{k+1})$.

If $\mu_k\|E_{k+1} - E_k\|_F/\|D\|_F < \epsilon_2$, set $\mu_{k+1} = \rho\mu_k$; otherwise, go to Step 1.

Note that the ALM algorithm performs better both in theory and practice than other algorithms with a Q-linear convergence speed globally. It is of much better numerical behavior or higher accuracy. It is reported that the ALM algorithm has been applied to MC problem (1.1). The algorithm, however, has the disadvantage of the penalty function, namely the matrix sequences $\{A_k\}$ generated by the ALM algorithm are the accepted solutions but feasible ones.

3.3 The smoothing augmented Lagrange multiplier algorithm

In this subsection, we make mention of a stepped-up scheme for the TMC problem. The smoothing augmented Lagrange multiplier (SALM) approach employs the smoothing operator $\mathcal{T}$ (see (2.1)) to approximate a matrix generated in the kth iteration so that the current approximation is of a Toeplitz structure.

Then our problem can be expressed as the following convex programming:

$$\begin{aligned} &\min_{A,E\in\mathbb{T}^{n\times n}} \|A\|_*, \\ &\text{subject to} \quad A+E=\mathcal{P}_\Omega(M), \qquad \mathcal{P}_\Omega(E)=0, \end{aligned} \tag{3.5}$$

where $M\in\mathbb{T}^{n\times n}$ is a real Toeplitz matrix, and $\Omega\subset\{-n+1,\dots,n-1\}$.

Let $D=\mathcal{P}_\Omega(M)$. Then the partial augmented Lagrangian function is

$$\mathcal{L}(A,E,Y,\mu)=\|A\|_*+\langle Y,D-A-E\rangle+\frac{\mu}{2}\|D-A-E\|_F^2, \tag{3.6}$$

where $Y\in\mathbb{R}^{n\times n}$.

Algorithm 3.2 ([34]) Given a sampled set Ω, a sampled matrix D, $\mu_0>0$, $\rho>1$. Given also two initial matrices $Y_0=0$, $E_0=0$. $k:=0$.

1. Compute the SVD of the matrix $(D-E_k+\mu_k^{-1}Y_k)$ using the Lanczos method

$$[U_k,\Sigma_k,V_k]=\operatorname{lansvd}\bigl(D-E_k+\mu_k^{-1}Y_k\bigr);$$

2. Set

$$X_{k+1}=U_k\mathcal{D}_{\mu_k^{-1}}(\Sigma_k)V_k^T,$$

compute for smoothing $\tilde{a}_\alpha=\frac{\alpha X_{k+1}^{\min}+\alpha X_{k+1}^{\max}}{2}$, $\alpha\in\{-n+1,-n+2,\dots,n-1\}$, and

$$A_{k+1}=\mathcal{T}(X_{k+1})=\sum_{l=1}^{n-1}\tilde{a}_{-l}Z_n^l+\sum_{l=0}^{n-1}\tilde{a}_l\bigl(Z_n^T\bigr)^l,$$

$$E_{k+1}=\mathcal{P}_{\bar{\Omega}}\bigl(D-A_{k+1}+\mu_k^{-1}Y_k\bigr);$$

3. If $\|D-A_{k+1}-E_{k+1}\|_F/\|D\|_F<\epsilon_1$ and $\mu_k\|E_{k+1}-E_k\|_F/\|D\|_F<\epsilon_2$, stop; otherwise, go to the next step;
4. Set $Y_{k+1}=Y_k+\mu_k(D-A_{k+1}-E_{k+1})$.

If $\mu_k\|E_{k+1}-E_k\|_F/\|D\|_F<\epsilon_2$, set $\mu_{k+1}=\rho\mu_k$; otherwise, go to Step 1.

It is reported that the convergence speed of the SALM algorithm is greater than that of the ALM and APG algorithms. A merit of smoothing is that the fast SVD procedure can be utilized to reduce the computation.

As we know, the SVD time is saved at the expense of data communication. Sometimes, this is not worth the candle. This motivated us to come up with the following algorithm.

3.4 The semi-smoothing augmented Lagrange multiplier algorithm

In this subsection, we propose a semi-smoothing augmented Lagrange multiplier algorithm based on the ALM and SALM algorithms for the TMC problem. The new algorithm consists of two stages: one is $\ell - 1$ iterations by the ALM scheme, which is free moving of data; another is the ℓth smoothing by the SALM procedure, which is keeping the iteration matrix as a Toeplitz structure.

Now, the semi-smoothing augmented Lagrange multiplier (SSALM) algorithm will be presented in the following.

Algorithm 3.3 (SSALM algorithm)

Input: A sampled set Ω, a sampled matrix D, $Y_{0,0} = 0$, $E_{0,0} = 0$; parameters $\mu_0 > 0$, $\rho > 1$, $\ell > 1$, ϵ_1, ϵ_2.

Let $k := 0$, $q := 1$, $q = 1, 2, \ldots, \ell - 1$.

Repeat:

1. $\ell - 1$ iterations.

   (1) Compute the SVD of the matrix $(D - E_{k,q} + \mu_{k,q}^{-1} Y_{k,q})$:

   $$[U_{k,q}, \Sigma_{k,q}, V_{k,q}] = \operatorname{lansvd}\big(D - E_{k,q} + \mu_{k,q}^{-1} Y_{k,q}\big);$$

   (2) Set

   $$X_{k+1,q+1} = U_{k,q} \mathcal{D}_{\mu_{k,q}^{-1}}(\Sigma_{k,q}) V_{k,q}^T,$$

   $$E_{k+1,q+1} = \mathcal{P}_{\bar{\Omega}}\big(D - X_{k+1,q+1} + \mu_{k,q}^{-1} Y_{k,q}\big);$$

   (3) If $\|D - X_{k+1,q+1} - E_{k+1,q+1}\|_F / \|D\|_F < \epsilon_1$ and $\mu_{k,q}\|E_{k+1,q+1} - E_{k,q}\|_F / \|D\|_F < \epsilon_2$, stop; otherwise, go to the next step;

   (4) Set $Y_{k+1,q+1} = Y_{k,q} + \mu_{k,q}(D - X_{k+1,q+1} - E_{k+1,q+1})$, $\mu_{k+1,q+1} = \rho \mu_{k,q}$; otherwise, go to step (1);

2. ℓth smoothing.

   (1) Compute the SVD of the matrix $(D - E_{k,\ell} + \mu_{k,\ell}^{-1} Y_{k,\ell})$:

   $$[U_{k,\ell}, \Sigma_{k,\ell}, V_{k,\ell}] = \operatorname{svd}\big(D - E_{k,\ell} + \mu_{k,\ell}^{-1} Y_{k,\ell}\big);$$

   (2) Set

   $$X_{k+1,\ell} = U_{k,\ell} \mathcal{D}_{\mu_{k,\ell}^{-1}}(\Sigma_{k,\ell}) V_{k,\ell}^T,$$

   compute the factors of smoothing $\tilde{a}_\alpha = \frac{\alpha X_{k+1,\ell}^{\min} + \alpha X_{k+1,\ell}^{\max}}{2}$, $\alpha = -n + 1, -n + 2, \ldots, n - 1$,

   and set

   $$A_{k+1,\ell} = \mathcal{T}(X_{k+1,\ell}) = \sum_{l=1}^{n-1} \tilde{a}_{-l} Z_n^l + \sum_{l=0}^{n-1} \tilde{a}_l \big(Z_n^T\big)^l.$$

   Update

   $$E_{k+1,\ell} = \mathcal{P}_{\bar{\Omega}}\big(D - A_{k+1,\ell} + \mu_{k,\ell}^{-1} Y_{k+1,\ell}\big);$$

3. If $\|D - A_{k+1,\ell} - E_{k+1,\ell}\|_F / \|D\|_F < \epsilon_1$ and $\mu_{k,\ell}\|E_{k+1,\ell} - E_{k,\ell}\|_F / \|D\|_F < \epsilon_2$, stop; otherwise, go to the next step;
4. Set $Y_{k+1,q+1} = Y_{k,q} + \mu_{k,q}(D - A_{k+1,q+1} - E_{k+1,q+1})$.

If $\mu_{k,q}\|E_{k+1,q+1} - E_{k,q}\|_F / \|D\|_F < \epsilon_2$, set $\mu_{k+1,q+1} = \rho\mu_{k,q}$; otherwise, go to Step 1.

In fact, Algorithm 3.3 includes the above Algorithm 3.2 as a special case if $\ell = 1$. Obviously, the algorithm is an acceleration of the SALM algorithm in [34].

4 Convergence analysis

This section will analyze the convergence of Algorithm 3.3.

Let $(\ddot{A}, \ddot{E})$ be the solution of model (3.5) and $\ddot{Y}$ be that of the optimal problem (3.2). We provide some lemmas in the following firstly.

Lemma 4.1 ([8]) *Let $A \in \mathbb{R}^{m\times n}$ be an arbitrary matrix and $U\Sigma V^T$ be its SVD. Then the set of subgradients of the nuclear norm of A is given by*

$$\partial\|A\|_* = \left\{UV^T + W : W \in \mathbb{R}^{m\times n}, U^T W = 0, WV = 0, \|W\|_2 \le 1\right\}.$$

Lemma 4.2 ([18]) *If μ_k is nondecreasing, then each entry of the following series is nonnegative and their sum is finite, i.e.,*

$$\sum_{k=1}^{+\infty} \mu_k{}^{-1}\left(\langle Y_{k+1} - Y_k, E_{k+1} - E_k\rangle + \langle A_{k+1} - \ddot{A}, \hat{Y}_{k+1} - \ddot{Y}\rangle + \langle E_{k+1} - \ddot{E}, Y_{k+1} - \ddot{Y}\rangle\right) < +\infty. \tag{4.1}$$

Lemma 4.3 ([18]) *The sequences $\{\ddot{Y}_k\}$, $\{Y_k\}$, and $\{\hat{Y}_k\}$ are all bounded, where $\hat{Y}_k = Y_{k+1} + \mu_{k-1}(D - A_k - E_{k-1})$.*

Lemma 4.4 *The sequence $\{Y_{k,q}\}$ generated by Algorithm 3.3 is bounded.*

Proof Let $B = \mu_{k,q}(D - E_{k,q} + \mu_{k,q}^{-1}Y_{k,q} - X_{k+1,q+1})$, $\mathcal{T}(B) = \sum_{l=1}^{n-1}\tilde{b}_{-l}Z_n^l + \sum_{l=0}^{n-1}\tilde{b}_l(Z_n^T)^l$, defined as (2.1).

First of all, we indicate that $Y_{k,q}$, $E_{k,q}$, $k = 1, 2, \ldots$, $q = 1, 2, \ldots, \ell - 1$, are all the Toeplitz matrices. Clearly, $Y_{0,0} = 0$, $E_{0,0} = 0$ are both smoothed into a Toeplitz structure. Suppose that $Y_{k,q}$, $E_{k,q}$ are both the Toeplitz matrices, so is $E_{k+1,q+1} = \mathcal{P}_{\bar{\Omega}}(D - X_{k+1,q+1} + \mu_{k,q}^{-1}Y_{k,q})$. Thus, $Y_{k+1,q+1}$ is a Toeplitz matrix also because of Step 4 in Algorithm 3.3.

$$\begin{aligned} Y_{k+1,q+1} &= Y_{k,q} + \mu_{k,q}(D - A_{k+1,q+1} - E_{k+1,q+1}) \\ &= Y_{k,q} + \mu_{k,q}(D - A_{k+1,q+1} - E_{k,q}) + \mu_{k,q}(E_{k,q} - E_{k+1,q+1}). \end{aligned}$$

Moreover,

$$\begin{aligned} \mu_{k,q}(E_{k,q} - E_{k+1,q+1}) &= \mu_{k,q}\mathcal{P}_{\bar{\Omega}}\left(E_{k,q} - \left(D - A_{k+1,q+1} + \mu_{k,q}^{-1}Y_{k,q}\right)\right) \\ &= \mu_{k,q}\mathcal{P}_{\bar{\Omega}}\left(E_{k,q} - \left(D - \mathcal{T}(X_{k+1,q+1}) + \mu_{k,q}^{-1}Y_{k,q}\right)\right) \\ &= \mu_{k,q}\mathcal{P}_{\bar{\Omega}}\left(E_{k,q} - \left(D - \mathcal{T}\left(D - E_{k,q} + \mu_{k,q}^{-1}Y_{k,q} - \mu_{k,q}^{-1}B\right) + \mu_{k,q}^{-1}Y_{k,q}\right)\right) \end{aligned}$$

$$
\begin{aligned}
&= \mu_{k,q}\mathcal{P}_{\bar{\Omega}}\big(E_{k,q} - \big(D - \big(D - E_{k,q} + \mu_{k,q}^{-1}Y_{k,q} - \mu_{k,q}^{-1}\mathcal{T}(B)\big) + \mu_{k,q}^{-1}Y_{k,q}\big)\big)\\
&= \mathcal{P}_{\bar{\Omega}}\mathcal{T}(B).
\end{aligned}
$$

It is clear that by Steps 1–2 in Algorithm 3.3 we have

$$
D - E_{k,q} + \mu_{k,q}^{-1}Y_{k,q} = \check{U}_{k,q}\check{\Sigma}_{k,q}\check{V}_{k,q}^{T} + \tilde{U}_{k,q}\tilde{\Sigma}_{k,q}\tilde{V}_{k,q}^{T},
$$

where $\check{U}_{k,q}$, $\check{V}_{k,q}$ are the singular vectors associated with singular values that are more than $\frac{1}{\mu_{k,q}}$ and $\tilde{U}_{k,q}$, $\tilde{V}_{k,q}$ are those associated with singular values that are less than or equal to $\frac{1}{\mu_{k,q}}$, the elements of the diagonal matrix $\check{\Sigma}_{k,q}$ are more than $\frac{1}{\mu_{k,q}}$, and those of the diagonal matrix $\tilde{\Sigma}_{k,q}$ are less than or equal to $\frac{1}{\mu_{k,q}}$. Hence, it is drawn that $A_{k+1,q+1} = \check{U}_{k,q}(\check{\Sigma}_{k,q} - \mu_{k,q}^{-1}I)\check{V}_{k,q}^{T}$, and

$$
\begin{aligned}
\|B\|_F &= \big\|\mu_{k,q}\big(D - E_{k,q} + \mu_{k,q}^{-1}Y_{k,q} - A_{k+1,q+1}\big)\big\|_F\\
&= \big\|\mu_{k,q}\big(\mu_{k,q}^{-1}\check{U}_{k,q}\check{V}_{k,q}^{T} + \tilde{U}_{k,q}\tilde{\Sigma}_{k,q}\tilde{V}_{k,q}^{T}\big)\big\|_F\\
&= \big\|\check{U}_{k,q}\check{V}_{k,q}^{T} + \mu_{k,q}\tilde{U}_{k,q}\tilde{\Sigma}_{k,q}\tilde{V}_{k,q}^{T}\big\|_F\\
&\le \sqrt{n}.
\end{aligned}
$$

We can obtain hence that $Y_{k,q} + \mu_{k,q}(D - A_{k+1,q+1} - E_{k,q}) \in \partial\|A_{k+1,q+1}\|_*$ from Lemmas 4.2 and 4.3.

It is known that, for $A_{k+1,q+1} = U\Sigma V^T$, by Lemma 4.1,

$$
\partial\|A_{k+1,q+1}\|_* = \big\{UV^T + W : W \in \mathbb{R}^{n\times n}, U^TW = 0, WV = 0, \|W\|_2 \le 1\big\}.
$$

We also have

$$
\big\|UV^T + W\big\|_F^2 = \operatorname{tr}\big(\big(UV^T + W\big)^T\big(UV^T + W\big)\big) = \operatorname{tr}\big(VV^T + W^TW\big) \le n.
$$

Therefore, the following inequalities can be obtained:

$$
\big\|Y_{k,q} + \mu_{k,q}(D - A_{k+1,q+1} - E_{k,q})\big\|_F \le \sqrt{n}
$$

and

$$
\big\|\mathcal{P}_{\bar{\Omega}}\big(\mathcal{T}(B)\big)\big\|_F \le \big\|\mathcal{T}(B)\big)\big\|_F \le \|B\|_F \le \sqrt{n}.
$$

It is evident that the sequence $\{Y_{k,q}\}$ is bounded. □

Theorem 4.1 *Suppose that* $\langle A_{k+1,q+1} - A_{k,q}, D - A_{k+1,q+1} - E_{k,q}\rangle \ge 0$, *then the sequence* $\{A_{k,q}\}$ *converges to the solution of* (3.5) *when* $\mu_{k,q} \to \infty$ *and* $\sum_{k=1}^{+\infty}\mu_{k,q}^{-1} = +\infty$.

Proof It is true that

$$
\lim_{k\to\infty}(D - A_{k+1} - E_{k+1}) = 0
$$

since $\mu_{k,q}^{-1}(Y_{k+1,q+1} - Y_{k,q}) = D - A_{k+1,q+1} - E_{k+1,q+1}$ and Lemma 4.4.

Let $(\ddot{A}, \ddot{E})$ be the solution of (3.5). Then $A_{k+1,q+1}, Y_{k+1,q+1}, E_{k+1,q+1}$, $k = 1, 2, \ldots$, are all Toeplitz matrices since $\ddot{A} + \ddot{E} = D$. We prove first that

$$\begin{aligned}
&\|E_{k+1,q+1} - \ddot{E}\|_F^2 + \mu_{k,q}^{-2}\|Y_{k+1,q+1} - \ddot{Y}\|_F^2 \\
&\quad = \|E_{k,q} - \ddot{E}\|_F^2 - \|E_{k+1,q+1} - E_{k,q}\|_F^2 + \mu_{k,q}^{-2}\|Y_{k,q} - \ddot{Y}\|_F^2 \\
&\qquad - \mu_{k,q}^{-2}\|Y_{k+1,q+1} - Y_{k,q}\|_F^2 - 2\mu_{k,q}^{-1}\langle A_{k+1,q+1} - \ddot{A}, \hat{Y}_{k+1,q+1} - \ddot{Y}\rangle,
\end{aligned} \tag{4.2}$$

where $\hat{Y}_{k+1,q+1} = Y_{k,q} + \mu_{k,q}(D - A_{k+1,q+1} - E_{k,q})$, $\ddot{Y}$ is the optimal solution to the dual problem (3.2).

$$\begin{aligned}
\|E_{k,q} - \ddot{E}\|_F^2 &= \left\|\mathcal{P}_{\bar{\Omega}}(E_{k,q} - \ddot{E})\right\|_F^2 \\
&= \left\|\mathcal{P}_{\bar{\Omega}}(E_{k+1,q+1} - \ddot{E} - E_{k+1,q+1} + E_{k,q})\right\|_F^2 \\
&= \left\|\mathcal{P}_{\bar{\Omega}}(E_{k+1,q+1} - \ddot{E})\right\|_F^2 + \left\|\mathcal{P}_{\bar{\Omega}}(E_{k+1,q+1} - E_{k,q})\right\|_F^2 \\
&\quad - 2\left\langle\mathcal{P}_{\bar{\Omega}}(E_{k+1,q+1} - \ddot{E}), \mathcal{P}_{\bar{\Omega}}(E_{k+1,q+1} - E_{k,q})\right\rangle \\
&= \|E_{k+1,q+1} - \ddot{E}\|_F^2 + \|E_{k+1,q+1} - E_{k,q}\|_F^2 \\
&\quad + 2\mu_{k,q}^{-1}\left\langle\mathcal{P}_{\bar{\Omega}}(A_{k+1,q+1} - \ddot{A}), \hat{Y}_{k+1,q+1} - \ddot{Y}\right\rangle.
\end{aligned}$$

We obtain the following result with the same analysis:

$$\begin{aligned}
\mu_{k,q}^{-2}\|Y_{k,q} - \ddot{Y}\|_F^2 &= \mu_{k,q}^{-2}\left\|\mathcal{P}_{\Omega}(Y_{k,q} - \ddot{Y})\right\|_F^2 \\
&= \mu_{k,q}^{-2}\left\|\mathcal{P}_{\Omega}(Y_{k+1,q+1} - \ddot{Y})\right\|_F^2 + \mu_{k,q}^{-2}\left\|\mathcal{P}_{\Omega}(Y_{k+1,q+1} - Y_{k,q})\right\|_F^2 \\
&\quad - 2\mu_{k,q}^{-1}\left\langle\mathcal{P}_{\Omega}(Y_{k+1,q+1} - \ddot{Y}), \mathcal{P}_{\Omega}(Y_{k+1,q+1} - Y_{k,q})\right\rangle \\
&= \mu_{k,q}^{-2}\|Y_{k+1,q+1} - \ddot{Y}\|_F^2 + \mu_{k,q}^{-2}\|Y_{k+1,q+1} - Y_{k,q}\|_F^2 \\
&\quad + 2\mu_{k,q}^{-1}\left\langle\mathcal{P}_{\Omega}(A_{k+1,q+1} - \ddot{A}), \hat{Y}_{k+1,q+1} - \ddot{Y}\right\rangle.
\end{aligned}$$

Then

$$\min_{A\in\mathbb{T}^{m\times n}} \mu\|A\|_* + \frac{1}{2}\left\|\mathcal{P}_{\Omega}(A - M)\right\|_F^2 \tag{4.3}$$

holds true.

The sum $\sum_{k=1}^{\infty}\mu_{k,q}^{-1}\langle A_{k,q} - \ddot{A}, \hat{Y}_{k,q} - \ddot{Y}\rangle < +\infty$ and $\|E_{k,q} - \ddot{E}\|^2 + \mu_{k,q}^{-2}\|Y_{k,q} - \ddot{Y}\|^2$ is nonincreasing since $\|A\|_*$ is a convex function and $\hat{Y}_{k+1,q+1} \in \partial\|A\|_*$, $\langle A_{k+1,q+1} - \ddot{A}, \hat{Y}_{k+1,q+1} - \ddot{Y}\rangle \geq 0$. On the other hand, the following are true by Algorithm 3.3:

$$\langle Y_{k,q}, E_{k,q} - \ddot{E}\rangle = 0 \quad \text{and} \quad \langle \ddot{Y}, E_{k,q} - \ddot{E}\rangle = 0.$$

Therefore, by the same idea of Theorem 2 in [18], it is obtained that $\ddot{A}$ is the solution of (3.5). □

Theorem 4.2 *Let $X = (x_{ij}) \in \mathbb{R}^{n\times n}$, $\mathcal{T}(X) = (\tilde{x}_{ij}) \in \mathbb{T}^{n\times n}$ be the Toeplitz matrix derived from X, introduced in (2.1). Then, for any Toeplitz matrix $Y = (y_{ij}) \in \mathbb{T}^{n\times n}$, $\langle X - \mathcal{T}(X), Y\rangle = 0$.*

Proof By the definition of $\mathcal{T}(X)$, we have $\sum_{i,j}(x_{ij} - \tilde{x}_{ij}) = 0$, $i,j = 1,2,\ldots,n$. Since Y is a Toeplitz matrix and $y_\alpha = y_{ij}$, $\alpha = i - j$, $i,j = 1,2,\ldots,n$, then

$$\begin{aligned}\langle X - \mathcal{T}(X), Y\rangle &= \sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ij} - \tilde{x}_{ij})y_{ji} \\ &= \sum_{l=1}^{n-1} y_{-l}(x_{ij} - \tilde{x}_{ij}) + \sum_{l=0}^{n-1} y_l(x_{ij} - \tilde{x}_{ij}) \\ &= 0.\end{aligned}$$

□

Theorem 4.3 *In Algorithm 3.3, $A_{k,q}$ is a Toeplitz matrix generated by $X_{k,q}$, then*

$$\|A_{k,q} - \ddot{A}\|_F < \|X_{k,q} - \ddot{A}\|_F$$

is true with $\ddot{A}$ being the solution of (3.5).

Proof

$$\begin{aligned}\|X_{k,q} - \ddot{A}\|_F^2 &= \|X_{k,q} - A_{k,q} + A_{k,q} - \ddot{A}\|_F^2 \\ &= \langle X_{k,q} - A_{k,q}, X_{k,q} - A_{k,q}\rangle + 2\langle X_{k,q} - A_{k,q}, A_{k,q} - \ddot{A}\rangle \\ &\quad + \langle A_{k,q} - \ddot{A}, A_{k,q} - \ddot{A}\rangle \\ &= \|X_{k,q} - A_{k,q}\|_F^2 + \|A_{k,q} - \ddot{A}\|_F^2.\end{aligned}$$

Thus, $\|A_{k,q} - \ddot{A}\|_F < \|X_{k,q} - \ddot{A}\|_F$ holds true. □

5 Numerical experiments

In this section we report some original numerical results of two algorithms (SALM, SSALM) for some $n \times n$ matrices with different ranks. All the experiments are conducted on the same workstation with an Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz that has 16 GB memory and 16-bit operating system, running Windows 7 and Matlab (vision 2016a). We analyze and compare iteration numbers (IT), computing time in seconds (TIME(s)), deviation (ERROR 1, ERROR 2), and RATIO which are defined in the following. It can be seen that the SSALM algorithm proposed in this study is highly effective compared with the SALM algorithm.

$$\text{ERROR 1} = \frac{\|A + E - D\|_F}{\|D\|_F}, \qquad \text{ERROR 2} = \frac{\|A - M\|_F}{\|M\|_F},$$

$$\text{RATIO} = \frac{\text{the CPU of the SSALM algorithm}}{\text{the CPU of the SALM algorithm}} \times 100\%.$$

In the experiments, $M \in \mathbb{T}^{n\times n}$ represents an uncompleted Toeplitz matrix, the sampling density $p = m/(2n-1)$, where m is the number of the known diagonal entries. By the way, $p \in \{0.3, 0.4, 0.5, 0.6\}$ here. Due to the special structure of a Toeplitz matrix, we have $0 \le m \le 2n-1$. The SALM and SSALM algorithms share the same values of all parameters, say, $\tau_0 = 1/\|D\|_2$, $\delta = 1.2172 + 1.8588m/n^2$, $\epsilon_1 = 10^{-9}$, $\epsilon_2 = 5 \times 10^{-6}$. In the test of the SSALM algorithm, $\ell = 2$ or $\ell = 3$ as a rule of semi-smoothing.

Table 1 Comparison between SALM and SSALM for $p = 0.6$

n	rank(M)	Algorithm	IT	TIME (s)	ERROR 1	ERROR 2
500	10	SALM	44	2.6563	8.2372e–10	1.0363e–07
		SSALM	42	1.8853	9.2608e–10	1.9658e–09
800	10	SALM	54	5.9456	8.1138e–10	8.2779e–09
		SSALM	52	5.0976	5.6384e–10	7.6367e–09
1000	10	SALM	57	8.2474	8.9636e–10	4.1947e–09
		SSALM	58	6.3207	7.9333e–10	1.9217e–09
1500	10	SALM	64	16.7198	9.5636e–10	2.1154e–09
		SSALM	64	15.0399	9.1550e–10	1.0551e–09
2000	10	SALM	71	31.7924	9.4056e–10	7.5879e–08
		SSALM	68	26.8256	8.1847e–10	2.3201e–08
2500	10	SALM	71	47.3399	7.0194e–10	1.5048e–08
		SSALM	72	42.5214	9.8809e–10	2.6892e–09
3000	10	SALM	77	70.2107	6.7658e–10	2.4626e–09
		SSALM	76	60.6145	6.9232e–10	1.1348e–09
4000	20	SALM	74	146.3660	6.5154e–10	5.1684e–05
		SSALM	72	129.2216	5.3346e–10	1.4304e–09
5000	20	SALM	74	220.6386	9.9475e–10	2.1943e–08
		SSALM	74	198.3210	5.5006e–10	1.8351e–09
8000	25	SALM	78	550.5427	9.2197e–10	3.6735e–09
		SSALM	80	471.4695	5.5647e–10	1.5932e–09

Table 2 Comparison between SALM and SSALM for $p = 0.5$

n	rank(M)	Algorithm	IT	TIME (s)	ERROR 1	ERROR 2
500	10	SALM	44	2.8955	9.5801e–10	2.2756e–08
		SSALM	44	2.1933	4.4278e–10	2.3334e–09
800	10	SALM	56	8.6741	7.0000e–10	2.6659e–06
		SSALM	52	5.4772	9.2740e–10	3.3480e–09
1000	10	SALM	60	11.0450	8.2993e–10	6.6858e–08
		SSALM	56	6.7779	8.6651e–10	1.8478e–09
1500	10	SALM	68	20.6328	8.45398e–10	9.6939e–07
		SSALM	64	15.6997	9.3584e–10	1.5261e–09
2000	10	SALM	71	44.8591	9.9033e–10	7.2479e–08
		SSALM	68	27.9845	9.6632e–10	2.8679e–09
2500	10	SALM	78	67.4042	7.3828e–10	1.2312e–07
		SSALM	74	45.3268	7.1109e–10	2.5905e–09
3000	10	SALM	77	74.2323	7.2763e–10	9.9884e–09
		SSALM	76	64.4300	8.2104e–10	1.2376e–08
4000	20	SALM	73	155.9955	9.4452e–10	1.8879e–06
		SSALM	72	128.0443	7.1470e–10	1.8278e–09
5000	20	SALM	74	235.1514	9.1421e–10	7.9369e–06
		SSALM	76	200.7122	6.2859e–10	3.5799e–09
8000	25	SALM	80	622.5518	9.5065e–10	5.5408e–06
		SSALM	80	498.8602	5.1021e–10	3.0270e–08

The experimental results of two algorithms are presented in Tables 1–6. From the tables, two algorithms can successfully compute an approximate solution of the prescriptive stop condition for all the test matrix M. And computing time of our SSALM algorithm in far less than that of the SALM algorithm. In particular, compared with the cost of the SALM algorithm, we can find that the cost of the SSALM algorithm is decreased up to 30.44%. The "RATIO" in Tables 5–6 can show this effectiveness.

6 Concluding remarks

As is known to all, matrix completion usually means to reconstruct a matrix from a subset of the items of a matrix by taking advantage of low-rank structure matrix interdepen-

Table 3 Comparison between SALM and SSALM for $p = 0.4$

n	rank(M)	Algorithm	IT	TIME (s)	ERROR 1	ERROR 2
500	10	SALM	45	7.8497	6.8884e–10	5.3595e–06
		SSALM	44	2.3892	5.3103e–10	2.3267e–09
800	10	SALM	54	7.4948	9.3427e–10	2.1464e–06
		SSALM	54	5.9618	8.9128e–10	1.5780e–08
1000	10	SALM	58	9.9528	9.7372e–10	3.2663e–05
		SSALM	56	7.2641	9.7968e–10	2.4662e–09
1500	10	SALM	67	25.3226	9.8741e–10	2.7106e–06
		SSALM	64	16.1484	9.0982e–10	1.5210e–09
2000	10	SALM	70	34.6083	8.4557e–10	1.8757e–07
		SSALM	70	29.7298	6.4346e–10	1.7874e–09
2500	10	SALM	76	54.5748	7.1288e–10	8.4514e–08
		SSALM	74	46.5716	4.9981e–10	1.6792e–09
3000	10	SALM	78	84.9121	5.3019e–10	2.2285e–06
		SSALM	76	66.0761	9.2876e–10	1.1697e–09
4000	20	SALM	73	164.7655	8.6946e–10	7.3598e–06
		SSALM	72	144.5329	5.1358e–10	1.4439e–09
5000	20	SALM	75	251.7641	8.9821e–10	7.9543e–08
		SSALM	76	203.3167	8.3168e–10	3.4049e–08
8000	25	SALM	79	598.8095	9.5617e–10	3.7230e–08
		SSALM	80	530.8533	4.9436e–10	1.2710e–09

Table 4 Comparison between SALM and SSALM for $p = 0.3$

n	rank(M)	Algorithm	IT	TIME (s)	ERROR 1	ERROR 2
500	10	SALM	46	11.3753	9.3029e–10	3.2509e–06
		SSALM	46	5.1926	6.5512e–10	1.8892e–08
800	10	SALM	57	17.3212	9.0562e–10	1.0264e–04
		SSALM	58	13.5300	5.7672e–10	1.9937e–05
1000	10	SALM	59	12.2900	9.8849e–10	2.9149e–07
		SSALM	58	7.9075	8.0278e–10	6.0819e–09
1500	10	SALM	69	38.0422	8.9932e–10	6.4286e–05
		SSALM	66	17.6928	5.5536e–10	4.7554e–08
2000	10	SALM	72	39.9752	9.1752e–10	3.8681e–07
		SSALM	70	30.0543	8.0322e–10	2.2090e–09
2500	10	SALM	78	92.3154	9.8020e–10	6.9561e–06
		SSALM	74	47.8242	9.2726e–10	6.3568e–09
3000	10	SALM	80	119.9347	8.6992e–10	6.7548e–06
		SSALM	80	84.1827	7.6381e–10	1.6034e–07
4000	20	SALM	74	187.0096	7.3863e–10	2.0127e–06
		SSALM	72	142.3911	8.5136e–10	3.0778e–08
5000	20	SALM	74	246.3862	9.4517e–10	4.4710e–07
		SSALM	78	219.6853	6.8201e–10	2.8025e–06
8000	25	SALM	83	1.4409e+03	9.9003e–10	2.7735e–04
		SSALM	84	1.1312e+03	8.0522e–10	3.7995e–04

dencies between the entries. It is well known but NP-hard in general. In recent years, the Toeplitz matrix completion has attracted widespread attention and has become one of the most important completion problems. In order to solve such problems, we proposed an accelerated technique of the SALM algorithm in this study, namely the SSALM algorithm, and the theory of the SSALM algorithm convergence is established. Theoretical analysis and numerical results have shown that the SSALM scheme is an effective algorithm for solving the TMC problem. In particular, the CPU of the SSALM algorithm is consistently reduced up to 30.44% in all cases. The SSALM algorithm overcomes the original ALM algorithm both complexity of the singular value decomposition and surmounts the property of the extra load of the SALM algorithm. The reason is that data communication conges-

Table 5 The values of RATIO for $\ell = 2$

$p = 0.6$	n	500	800	1000	1500	2000	2500	3000	4000	5000	8000
	rank(M)	10	10	10	10	10	10	10	20	20	25
	RATIO (%)	70.97	85.74	76.64	89.95	84.38	89.82	86.33	88.29	89.88	85.64
$p = 0.5$	n	500	800	1000	1500	2000	2500	3000	4000	5000	8000
	rank(M)	10	10	10	10	10	10	10	20	20	25
	RATIO (%)	75.75	63.41	61.37	76.09	62.38	67.25	86.80	82.08	85.35	80.13
$p = 0.4$	n	500	800	1000	1500	2000	2500	3000	4000	5000	8000
	rank(M)	10	10	10	10	10	10	10	20	20	25
	RATIO (%)	30.44	79.55	72.99	63.77	85.90	85.34	77.82	87.72	80.76	88.65
$p = 0.3$	n	500	800	1000	1500	2000	2500	3000	4000	5000	8000
	rank(M)	10	10	10	10	10	10	10	20	20	25
	RATIO (%)	45.65	78.11	64.34	46.51	75.18	51.81	70.19	76.14	89.16	78.51

Table 6 Comparison between SALM and SSALM for $\ell = 3$

n	rank(M)	p	Algorithm	IT	TIME (s)	ERROR 1	ERROR 2	RATIO (%)
500	10	0.4	SALM	47	7.5689	5.2053e–10	3.9970e–06	71.71
			SSALM	47	5.4276	3.8663e–10	3.2564e–06	
1000	10	0.4	SALM	60	13.9741	8.8726e–10	2.6235e–07	61.58
			SSALM	59	8.6048	9.8580e–10	9.1642e–09	
1500	10	0.5	SALM	68	29.6545	9.1634e–10	4.5666e–05	63.99
			SSALM	68	18.9748	8.6885e–10	4.3619e–09	
1500	10	0.3	SALM	69	47.1327	9.3547e–10	6.8306e–06	65.78
			SSALM	68	31.0025	9.0834e–10	2.2110e–06	
2000	10	0.5	SALM	73	50.5371	8.4794e–10	3.1914e–05	59.00
			SSALM	71	29.8151	9.8594e–10	8.2954e–09	
2000	10	0.4	SALM	72	38.5308	6.6310e–10	9.5342e–07	75.51
			SSALM	69	29.0941	9.3350e–10	3.5384e–09	
3000	10	0.5	SALM	79	83.3712	9.9293e–10	1.2756e–05	76.68
			SSALM	78	63.9294	6.5972e–10	2.8985e–09	
3000	10	0.4	SALM	77	80.7144	8.9678e–10	2.0862e–07	83.38
			SSALM	78	67.2993	9.9493e–10	7.3612e–09	
3000	10	0.3	SALM	79	118.3675	9.8435e–10	9.6348e–05	81.24
			SSALM	80	96.1642	7.0585e–10	1.4254e–06	
4000	20	0.6	SALM	73	152.4618	8.2869e–10	1.7109e–06	75.89
			SSALM	72	115.7083	8.2721e–10	2.4255e–09	
4000	20	0.5	SALM	72	166.5827	9.3233e–10	1.0321e–07	74.92
			SSALM	72	124.8054	7.5040e–10	2.4066e–09	
4000	20	0.4	SALM	75	186.7383	9.0624e–10	7.5345e–07	73.03
			SSALM	74	136.3765	8.7658e–10	1.0216e–07	
5000	20	0.5	SALM	75	232.0879	9.8661e–10	5.7423e–08	80.86
			SSALM	75	187.6559	9.6715e–10	2.2527e–08	
5000	20	0.4	SALM	76	250.9495	8.5572e–10	1.1830e–06	83.20
			SSALM	77	208.7798	9.7333e–10	7.8349e–07	
5000	20	0.3	SALM	76	273.7735	9.4519e–10	6.4327e–06	86.76
			SSALM	80	237.5358	5.0062e–10	8.2082e–06	
8000	25	0.5	SALM	80	575.2382	9.1308e–10	5.0447e–05	82.81
			SSALM	81	476.3530	7.4764e–10	4.6774e–08	
8000	25	0.4	SALM	80	512.3396	9.7395e–10	6.3150e–06	97.00
			SSALM	81	496.9932	7.2501e–10	9.3571e–09	
8000	25	0.3	SALM	83	579.2973	7.1372e–10	1.3349e–06	90.08
			SSALM	80	521.8350	9.4433e–10	6.5484e–09	

tion is far more expensive than computing in many computers. Therefore, the SSALM algorithm has better numerical behavior for solving the TMC problem than the SALM algorithm (Tables 1–6).

Acknowledgements
The authors are very much indebted to the editor and anonymous referees for their helpful comments and suggestions which greatly improved the original manuscript of this paper. The authors also are thankful for the support from the NSF of Shanxi Province (201601D011004) and the SYYJSKC-1803.

Funding
It is not applicable.

Abbreviations
MC, matrix completion; TMC, Toeplitz matrix completion; SVD, singular value decomposition; SVT, singular value thresholding; APG, accelerated proximal gradient; ALM, augmented Lagrange multiplier; MALM, modified augmented Lagrange multiplier; SALM, soothing augmented Lagrange multiplier; SSALM, semi-soothing augmented Lagrange multiplier; RPCA, robust principal component analysis; IT, iteration number; CPU, computing time.

Availability of data and materials
Please contact author for data requests.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All authors contributed equally to the writing of this paper. All authors read and approved the final manuscript.

Author details
[1]Key Laboratory of Engineering & Computing Science, Shanxi Provincial Department of Education/Department of Mathematics, Taiyuan Normal University, Jinzhong, P.R. China. [2]Department of Mathematics, Taiyuan Normal University, Jinzhong, P.R. China. [3]Department of Mathematics, Taiyuan University, Taiyuan, P.R. China.

Publisher's Note

Received: 29 January 2019 Accepted: 19 March 2019 Published online: 28 March 2019

References

1. Amit, Y., Fink, M., Srebro, N., Ullman, S.: Uncovering shared structures in multiclass classification. In: Proceeding of the 24th International Conference on Machine Learning, pp. 17–24. ACM, New York (2007)
2. Argyriou, A., Evgeniou, T., Pontil, M.: Multi-task feature learning. Adv. Neural Inf. Process. Syst. **19**, 41–48 (2007)
3. Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C.: Multi-task feature learning, image inpainting. Comput. Graph. **34**, 417–424 (2000)
4. Blanchard, J., Tanner, J., Wei, K.: CGIHT: conjugate gradient iterative hard thresholding for compressed sensing and matrix completion. In: Numerical Analysis Group (2014) Preprint 14/08
5. Boumal, N., Absil, P.A.: RTRMC: a Riemannian trust-region method for low-rank matrix completion. In: Shawe-Taylor, J., Zemel, R.S., Bartlett, P., Pereira, F.C.N., Weinberger, K.Q. (eds.) Advances in Neural Inf. Processing Systems, NIPS, vol. 24, pp. 406–414 (2011)
6. Cai, J.F., Candès, E.J., Shen, Z.: A singular value thresholding method for matrix completion. SIAM J. Optim. **20**(4), 1956–1982 (2010)
7. Cai, J.F., Qu, X., Xu, W., Ye, G.B.: Robust recovery of complex exponential signals from random Gaussian projections via low rank Hankel matrix reconstruction. Appl. Comput. Harmon. Anal. **41**(2), 470–490 (2016)
8. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. Found. Comput. Math. **9**(6), 717–772 (2009)
9. Chen, C., He, B., Yuan, X.: Matrix completion via an alternating direction method. IMA J. Numer. Anal. **32**, 227–245 (2012)
10. Chen, M., Ganesh, A., Lin, Z., Ma, Y., Wright, J., Wu, L.: Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. J. Mar. Biol. Assoc. UK **56**(3), 707–722 (2009)
11. Chen, Y., Chi, Y.: Robust spectral compressed sensing via structured matrix completion. IEEE Trans. Inf. Theory **60**(10), 6576–6601 (2014)
12. Eldén, L.: Matrix Methods in Data Mining and Pattern Recognition. Society for Industrial and Applied Mathematics, Philadelphia (2007)
13. Hu, Y., Zhang, D.B., Liu, J., Ye, J.P., He, X.F.: Accelerated singular value thresholding for matrix completion. In: KDD'12, Beijing, August 12-16, 2012 (2012)
14. Jain, P., Meka, R., Dhillon, I.: Guaranteed rank minimization via singular value projection. In: Proceeding of the Neural Information Processing Systems Conf., NIPS, pp. 937–945 (2010)
15. Jain, P., Netrapalli, P., Sanghavi, S.: Low-rank matrix completion using alternating minimization. In: Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC), pp. 665–674 (2013)
16. Ji, H., Liu, C., Shen, Z., Xu, Y.: Robust video denoising using low rank matrix completion. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition **CVPR** (2010)
17. Kyrillidis, A., Cevher, V.: Matrix recipes for hard thresholding methods. J. Math. Imaging Vis. **48**(2), 235–265 (2013)
18. Lin, Z., Chen, M., Wu, L., Ma, Y.: The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. UIUC Technicial Report UIUL-ENG-09-2214, (2010)
19. Liu, Z., Vandenberghe, L.: Interior-point method for nuclear norm approximation with application to system identification. SIAM J. Matrix Anal. Appl. **31**(3), 1235–1256 (2009)

20. Luk, F.T., Qiao, S.: A fast singular value algorithm for Hankel matrices. In: Olshevsky, V. (ed.) Fast Algorithms for Structured Matrices: Theory and Applications. Contemporary Mathematics, vol. 323. Am. Math. Soc., Providence (2003)
21. Ma, S., Goldfarb, D., Chen, L.: Fixed point and Bregman iterative methods for matrix rank minimization. Math. Program. **128**, 321–353 (2011)
22. Mesbahi, M., Papavassilopoulos, G.P.: On the rank minimization problem over a positive semidefinite linear matrix inequality. IEEE Trans. Autom. Control **42**(2), 239–243 (1997)
23. Mishra, B., Apuroop, K.A., Sepulchre, R.: A Riemannian geometry for low-rank matrix completion (2013) arXiv:1306.2672
24. Sebert, F., Zou, Y.M., Ying, L.: Toeplitz block matrices in compressed sensing and their applications in imaging. IEEE, Inf. Tech. Appl. in Biomedicine, 47–50 (2008)
25. Shaw, A.K., Pokala, S., Kumaresan, R.: Toeplitz and Hankel approximation using structured approach. Acoustics. Speech and signal processing. Proc. IEEE Int. Conf. **2349**, 12–15 (1998)
26. Tanner, J., Wei, K.: Low rank matrix completion by alternating steepest descent methods. Appl. Comput. Harmon. Anal. **40**, 417–429 (2016)
27. Toh, K.C., Yun, S.: An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. Pac. J. Optim. **6**, 615–640 (2010)
28. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: a factorization method. Int. J. Comput. Vis. **9**(2), 137–154 (1992)
29. Vandereycken, B.: Low rank matrix completion by Riemannian optimization. SIAM J. Optim. **23**(2), 1214–1236 (2013)
30. Wang, C.L., Li, C.: A mean value algorithm for Toeplitz matrix completion. Appl. Math. Lett. **41**, 35–40 (2015)
31. Wang, C.L., Li, C., Wang, J.: A modified augmented Lagrange multiplier algorithm for Toeplitz matrix completion. Adv. Comput. Math. **42**, 1209–1224 (2016)
32. Wang, C.L., Li, C., Wang, J.: Comparisons of several algorithms for Toeplitz matrix recovery. Comput. Math. Appl. **71**(1), 133–146 (2016)
33. Wen, R.P., Li, S.D., Meng, G.Y.: SOR-like methods with optimization model for augmented linear systems. East Asian J. Appl. Math. **7**(1), 101–115 (2017)
34. Wen, R.P., Li, S.Z., Zhou, F.: Toeplitz matrix completion via smoothing augmented Lagrange multiplier algorithm. Appl. Math. Comput. **355**, 299–310 (2019)
35. Wen, R.P., Yan, X.H.: A new gradient projection method for matrix completion. Appl. Math. Comput. **258**, 537–544 (2015)