Fast and Accurate Matrix Completion via Truncated Nuclear Norm Regularization

Yao Hu, Debing Zhang, Jieping Ye, *Senior Member*, *IEEE*,
Xuelong Li, *Fellow*, *IEEE*, and Xiaofei He, *Senior Member*, *IEEE*

Abstract—Recovering a large matrix from a small subset of its entries is a challenging problem arising in many real applications, such as image inpainting and recommender systems. Many existing approaches formulate this problem as a general low-rank matrix approximation problem. Since the rank operator is nonconvex and discontinuous, most of the recent theoretical studies use the nuclear norm as a convex relaxation. One major limitation of the existing approaches based on nuclear norm minimization is that all the singular values are simultaneously minimized, and thus the rank may not be well approximated in practice. In this paper, we propose to achieve a better approximation to the rank of matrix by truncated nuclear norm, which is given by the nuclear norm subtracted by the sum of the largest few singular values. In addition, we develop a novel matrix completion algorithm by minimizing the Truncated Nuclear Norm. We further develop three efficient iterative procedures, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP, to solve the optimization problem. TNNR-ADMM utilizes the alternating direction method of multipliers (ADMM), while TNNR-AGPL applies the accelerated proximal gradient line search method (APGL) for the final optimization. For TNNR-ADMMAP, we make use of an adaptive penalty according to a novel update rule for ADMM to achieve a faster convergence rate. Our empirical study shows encouraging results of the proposed algorithms in comparison to the state-of-the-art matrix completion algorithms on both synthetic and real visual datasets.

Index Terms—Matrix completion, nuclear norm minimization, alternating direction method of multipliers, accelerated proximal gradient method

1 Introduction

Estimating missing values from very limited information of an unknown matrix has attracted considerable interest recently [1], [2], [3], [4], [5], [6], [7]. This problem occurs in many real applications in computer vision and pattern recognition, such as image inpainting [8], [9], video denoising [10], and recommender systems [11], [12]. Obviously, the completion of arbitrary matrices is an ill-posed problem. A commonly adopted assumption is that the underlying matrix comes from a restricted class. For visual data such as an image, it is probably of low rank structure, as shown in Fig. 1. Based on this observation, it is natural to assume that the underlying matrix has a low rank or approximately low rank structure. Specifically, given the incomplete data matrix $M \in \mathbb{R}^{m\times n}$ of low rank, the matrix completion problem can be formulated as follows:

$$\begin{aligned} \min_{X} \quad & rank(X) \\ s.t. \quad & X_{ij} = M_{ij}, (i,j) \in \Omega, \end{aligned} \tag{1}$$

where $X \in \mathbb{R}^{m\times n}$ and Ω is the set of locations corresponding to the observed entries.

Unfortunately, the above rank minimization problem is NP-hard in general due to the nonconvexity and discontinuous nature of the rank function. The existing algorithms cannot directly solve the rank minimization problem efficiently. Theoretical studies show that the nuclear norm, i.e., the sum of singular values of a matrix, is the tightest convex lower bound of the rank function of matrices [13]. The relationship between nuclear norm and the rank of matrices is similar to the relationship between the l_1 norm and l_0 norm of vectors [13]. Therefore, a widely used approach is to apply the nuclear norm as a convex surrogate of the nonconvex matrix rank function [14]. Recent progress in matrix completion shows that, under some general constraints, nuclear norm minimization can recover this incomplete matrix from sufficient observed entries [1], [15], [16], [2].

The existing approaches based on the nuclear norm heuristic, such as singular value thresholding (SVT [17]), nuclear norm regularized least squares (NNLS [18]), and robust principal component analysis (Robust PCA [19]), have strong theoretical guarantees. However, they may obtain suboptimal performance in real applications since the nuclear norm may not be a good approximation to the rank function. Specifically, compared to the rank function in which all the nonzero singular values have equal contributions, the nuclear norm treats the singular values differently by adding them together. Moreover, the theoretical requirements (e.g., incoherence property) of the nuclear norm heuristic are usually very hard to satisfy in practice [1], [2].

- *Y. Hu, D. Zhang, and X. He are with the State Key Lab of CAD&CG, College of Computer Science, Zhejiang University, 388 Yu Hang Tang Road, Hangzhou, Zhejiang 310058, China. E-mail: {huyao001, debingzhangchina, xiaofeihe}@gmail.com.*
- *J. Ye is with the Computer Science and Engineering Department and Center for Evolutionary Medicine and Informatics, the Biodesign Institute, Arizona State University, Tempe, AZ 85287. E-mail: jieping.ye@asu.edu.*
- *X. Li is with the Center for OPTical IMagery Analysis and Learning (OPTIMAL), State Key Laboratory of Transicent Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, Shaanxi, P.R.China. E-mail: xuelong_li@opt.ac.cn.*

Manuscript received 7 July 2012; revised 10 Dec. 2012; accepted 17 Dec. 2012; published online 19 Dec. 2012.
Recommended for acceptance by S. Sarkar.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2012-07-0513.
Digital Object Identifier no. 10.1109/TPAMI.2012.271.

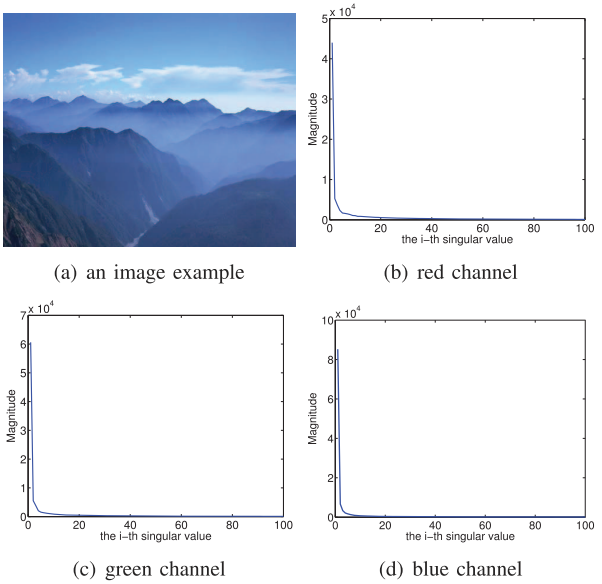


Fig. 1. (a) A 400×500 image example. (b) The singular values of the red channel. (c) The singular values of the green channel. (d) The singular values of the blue channel. As can be seen, the information is dominated by the top 20 singular values.

In this paper, we propose a novel matrix completion method called *Truncated Nuclear Norm Regularization* (TNNR) to recover the low rank matrices with missing values. Different from the nuclear norm based approaches which minimize the summation of all the singular values, our approach only minimizes the smallest $\min(m,n)-r$ singular values since the rank of a matrix only corresponds to the first r nonzero singular values. In this way, we can get a more accurate and robust approximation to the rank function. We further propose an efficient iterative optimization scheme for the matrix completion problem by minimizing the truncated nuclear norm. Differently from traditional nuclear norm minimization problem, the truncated nuclear norm minimization is not a convex problem. Thus, the standard methods for convex optimization problems cannot be directly used to solve this problem. Instead, we propose a simple but efficient two-step iterative scheme. In the first step, we fix some variables to get a closed form solution of the rest variables. In the second step, we fix the rest of the variables and solve a convex subproblem to update the fixed variables in the first step. By updating the two steps alternately, we can achieve a reasonably good recovery of the original matrix with missing values. To solve the convex subproblem in the second step, we develop three algorithms: TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP. The TNNR-ADMM algorithm applies the alternating direction method of multipliers (ADMM) to solve the convex subproblem as a separable convex programming problem with multiple constraints. The TNNR-APGL algorithm employs the accelerated proximal gradient line search method (APGL) to seek an optimal solution of a soft constrained version of the convex subproblem [20]. Furthermore, as the multiple constraints may slow down the convergence of TNNR-ADMM, we reformulate the original convex subproblem as a general separable convex programming problem with just one new constraint. This new problem can be solved by utilizing ADMM efficiently. To achieve fast convergence, we allow the penalty to change adaptively according to a novel update rule. We call this algorithm *Truncated Nuclear Norm Regularization by Alternating Direction Method of Multipliers with Adaptive Penalty* (TNNR-ADMMAP).

The remainder of the paper is organized as follows: In the next section, we provide a brief description of the related work. In Section 3, we present the proposed TNNR algorithm and design an efficient two-step optimization scheme. We then introduce three optimization methods, TNNR-ADMM, TNNR-APGL and TNNR-ADMMAP, in Sections 4, 5, and 6, respectively. Experimental results are presented in Section 7. Finally, we provide some concluding remarks in Section 8.

Notations. *Let $X=(\mathbf{x}_1,\ldots,\mathbf{x}_n)$ be an $m\times n$ matrix, $\Omega\subset\{1,\ldots,m\}\times\{1,\ldots,n\}$ denote the indices of the observed entries of X, and let Ω^c denote the indices of the missing entries. The Frobenius norm of X is defined as $\|X\|_F^2=\sqrt{\Sigma_{(i,j)}X_{ij}^2}$. Let $X=U\Sigma V^T$ be the singular value decomposition for X, where $\Sigma=diag(\sigma_i)$, $1\le i\le\min\{m,n\}$, and σ_i is the ith largest singular value of X. The nuclear norm of X is denoted as $\|X\|_*=\sum_{i=1}^{\min\{m,n\}}\sigma_i$. Let $\mathcal{P}_\Omega$ be the orthogonal projection operator onto the span of matrices vanishing outside of Ω so that*

$$(\mathcal{P}_\Omega(X))_{ij}=\begin{cases}X_{ij} & \text{if}(i,j)\in\Omega\\ 0 & \text{if}(i,j)\in\Omega^c.\end{cases}$$

The inner product of the matrix space is defined as $\langle X,Y\rangle=\sum_{i,j}X_{ij}Y_{ij}$.

2 Related Work

The matrix completion techniques have been frequently applied in many areas, including machine learning [21], [22], [23], computer vision [24], collaborative filtering [25], control [26], and signal processing [27]. As the nuclear norm is the convex surrogate of the rank function of matrices, recent theoretical progress shows that under some general settings, a perfect recovery can be achieved by solving the nuclear norm-based convex optimization if the underlying matrix is indeed of low rank. Fazel [28] first solves the rank minimization problem (1) in control system and design by approximating the rank function using the nuclear norm:

$$\begin{aligned}\min_X\quad & \|X\|_*\\ s.t.\quad & \mathcal{P}_\Omega(X)=\mathcal{P}_\Omega(M),\end{aligned}\tag{2}$$

where $\|X\|_*=\sum_{i=1}^{\min(m,n)}\sigma_i(X)$ is the nuclear norm and $\sigma_i(X)$ is the ith largest singular value of X. It is suggested in [28] to reformulate the optimization problem (2) as a semi-definite programming (SDP) problem which can be solved by the interior-point method. In many cases, however, the matrices are of very high dimensions, which makes the SDP problem intractable. Existing SDP solvers such as SDPT3 [29] and SeDuMi [30] can only handle matrices whose size is less than 100×100 efficiently. This limits the usage of matrix completion techniques in the applications of computer vision and image processing.

Instead of solving the nuclear norm minimization problem (2) by considering it as an SDP problem which is computationally infeasible, several recent works proposed solving its approximations involving strongly convex

objective functions. In particular, Cai et al. propose the SVT algorithm [17]:

$$\begin{aligned} \min_{X} \quad & \|X\|_* + \alpha\|X\|_F^2 \\ s.t. \quad & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M). \end{aligned} \tag{3}$$

The Uzawa method can be applied to efficiently solve the above optimization problem [17]. As the Lagrange dual function of the strongly convex programming problem (3) is differentiable, the SVT algorithm is actually the gradient descent method applied to the Lagrange dual function of (3). So, SVT has a global convergence rate of $O(\frac{1}{N})$, which is still too slow for dealing with large scale matrices.

To further improve the convergence rate, Toh and Yun [18] and Ji and Ye [31] independently apply an accelerated proximal gradient optimization technique for solving the nuclear norm regularized least squares problem:

$$\min_X \frac{1}{2}\|\mathcal{P}_\Omega(X) - \mathcal{P}_\Omega(M)\|_F^2 + \mu\|X\|_*, \tag{4}$$

where μ is a given parameter. Both Toh and Yun [18] and Ji and Ye [31] show that the primal error of their algorithms is smaller than ϵ after $O(\frac{1}{\sqrt{\epsilon}})$ iterations.

Another class of techniques used for matrix completion problems is based on matrix factorization. Srebro et al. [32] propose a maximum margin factorization method by using a factor model for the original matrix and consider the following problem:

$$\min_{U,V} \sum_{(i,j)\in\Omega} (X_{ij} - (UV^T)_{ij})^2 + \beta(\|U\|_F^2 + \|V\|_F^2). \tag{5}$$

This formulation and its extensions have been explored in [32], [33], and [34]. It has been observed empirically and theoretically [35], [32] that biconvex methods used in the optimization of (5) get stuck in suboptimal local minimum if the rank r is small. If r and the dimensions m, n are large, the computational cost may be very high. Keshavan et al. [36] consider the matrix completion problem in a very similar way. They also give a theoretical guarantee that one could reconstruct a low-rank matrix by observing a small fraction of its entries. Some other work related to the matrix completion problem includes the singular value projection (SVP) [37], which solves the rank minimization problem directly.

This journal paper is an extension of our own previous work [20], where we proposed a better approximation of the rank function of matrices called *truncated nuclear norm*. We use the accelerated proximal gradient method to solve the soft constrained problem and ADMM for hard constrained problems with multiple constraints, respectively.

The ADMM method has attracted a lot of attention recently for its great advantage in solving the separable convex optimization problems. However, when applying ADMM for problems with many variables and constraints, the memory requirement becomes very high and the convergence rate becomes slow. Moreover, it is hard to choose an optimal penalty parameter for ADMM [38]. Lin et al. [39] propose a linearized ADMM method to avoid the difficulty of multiple constraints. Instead of using a fixed penalty parameter in the ADMM, Lin et al. also show that ADMM with an adaptive penalty parameter could speed up the convergence of ADMM [39].

3 Truncated Nuclear Norm Regularization

3.1 Our Approach

The key to the proposed approach is the use of the *truncated nuclear norm*, which achieves a better approximation of the rank function than the nuclear norm.

Definition 3.1. *Given a matrix $X \in \mathbb{R}^{m\times n}$, the truncated nuclear norm $\|X\|_r$ is defined as the sum of $\min(m,n) - r$ minimum singular values, i.e., $\|X\|_r = \sum_{i=r+1}^{\min(m,n)} \sigma_i(X)$.*

Thus, the proposed approach can be formulated as follows:

$$\begin{aligned} \min_{X} \quad & \|X\|_r \\ s.t. \quad & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M). \end{aligned} \tag{6}$$

Since the values of the largest r nonzero singular values will not affect the rank of the matrix, we leave them free in our newly designed truncated nuclear norm and focus on minimizing the sum of the smallest $\min(m,n) - r$ singular values. Since $\|X\|_r$ is nonconvex, it is not easy to solve (6) directly. We have the following theorem.

Theorem 3.1. *For any given matrix $X \in \mathbb{R}^{m\times n}$, any matrices $A \in \mathbb{R}^{r\times m}$, $B \in \mathbb{R}^{r\times n}$ that $AA^T = I_{r\times r}$, $BB^T = I_{r\times r}$. For any nonnegative integer r ($r <= \min(m,n)$), we have*

$$\mathrm{Tr}(AXB^T) \le \sum_{i=1}^{r} \sigma_i(X). \tag{7}$$

Proof. By Von Neumann's trace inequality, we get

$$\begin{aligned} \mathrm{Tr}(AXB^T) &= \mathrm{Tr}(XB^TA) \\ &\le \sum_{i=1}^{\min(m,n)} \sigma_i(X)\sigma_i(B^TA), \end{aligned} \tag{8}$$

where $\sigma_1(X) \ge \cdots \ge \sigma_{\min(m,n)}(X) \ge 0$. As $rank(A) = r$ and $rank(B) = r$, so $rank(B^TA) = s \le r$. For $i \le s$, $\sigma_i(B^TA) \ge 0$ and $\sigma_i^2(B^TA)$ is the ith eigenvalue of $B^TAA^TB = B^TB$, which is also an eigenvalue of $BB^T = I$. Therefore, $\sigma_i(B^TA) = 1$, for $i = 1, 2, \ldots, s$, and the rest are all 0s. It follows that:

$$\begin{aligned} &\sum_{i=1}^{\min(m,n)} \sigma_i(X)\sigma_i(B^TA) \\ &= \sum_{i=1}^{s} \sigma_i(X)\sigma_i(B^TA) + \sum_{i=s+1}^{\min(m,n)} \sigma_i(X)\sigma_i(B^TA) \\ &= \sum_{i=1}^{s} \sigma_i(X)\cdot 1 + \sum_{i=s+1}^{\min(m,n)} \sigma_i(X)\cdot 0 \\ &= \sum_{i=1}^{s} \sigma_i(X). \end{aligned} \tag{9}$$

Since $s \le r$ and $\sigma_i(X) \ge 0$:

$$\sum_{i=1}^{s} \sigma_i(X) \le \sum_{i=1}^{r} \sigma_i(X).$$

Combining inequalities (8) and (9), we have

$$\mathrm{Tr}(AXB^T) \le \sum_{i=1}^{s} \sigma_i(X) \le \sum_{i=1}^{r} \sigma_i(X). \tag{10}$$

□

Suppose, $U\Sigma V^T$ is the singular value decomposition of X, where $U = (\mathbf{u}_1, \ldots, \mathbf{u}_m) \in \mathbb{R}^{m\times m}$, $\Sigma \in \mathbb{R}^{m\times n}$, and $V = (\mathbf{v}_1, \ldots, \mathbf{v}_n) \in \mathbb{R}^{n\times n}$. The equality of (7) holds when

$$A = (\mathbf{u}_1, \ldots, \mathbf{u}_r)^T, B = (\mathbf{v}_1, \ldots, \mathbf{v}_r)^T. \tag{11}$$

This is because

$$\begin{aligned}
&\mathrm{Tr}((\mathbf{u}_1, \ldots, \mathbf{u}_r)^T X(\mathbf{v}_1, \ldots, \mathbf{v}_r)) \\
&= \mathrm{Tr}((\mathbf{u}_1, \ldots, \mathbf{u}_r)^T U\Sigma V^T(\mathbf{v}_1, \ldots, \mathbf{v}_r)) \\
&= \mathrm{Tr}(((\mathbf{u}_1, \ldots, \mathbf{u}_r)^T U)\Sigma(V^T(\mathbf{v}_1, \ldots, \mathbf{v}_r))) \\
&= \mathrm{Tr}\left(\begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}\Sigma\begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}\right) \\
&= \mathrm{Tr}(diag(\sigma_1(X), \ldots, \sigma_r(X), 0, \ldots, 0)) \\
&= \sum_{i=1}^{r} \sigma_i(X).
\end{aligned} \tag{12}$$

Combining (10) and (12), we get

$$\max_{AA^T=I, BB^T=I} \mathrm{Tr}(AXB^T) = \sum_{i=1}^{r} \sigma_i(X). \tag{13}$$

Then we have

$$\begin{aligned}
&\|X\|_* - \max_{AA^T=I, BB^T=I} \mathrm{Tr}(AXB^T) \\
&= \sum_{i=1}^{\min(m,n)} \sigma_i(X) - \sum_{i=1}^{r} \sigma_i(X) \\
&= \sum_{i=r+1}^{\min(m,n)} \sigma_i(X) \\
&= \|X\|_r.
\end{aligned} \tag{14}$$

Thus, the optimization problem (6) can be rewritten as follows:

$$\begin{aligned}
\min_X \quad & \|X\|_* - \max_{AA^T=I, BB^T=I} \mathrm{Tr}(AXB^T) \\
s.t. \quad & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M),
\end{aligned} \tag{15}$$

where $A \in \mathbb{R}^{r\times m}, B \in \mathbb{R}^{r\times n}$.

Based on (15), we design a simple but efficient iterative scheme. We set $X_1 = M_\Omega$. In the lth iteration, we first fix X_l and compute A_l and B_l from (11) based on the singular value decomposition (SVD) of X_l. And then we fix A_l and B_l to update X_{l+1} by solving the following problem:

$$\begin{aligned}
\min_X \quad & \|X\|_* - \mathrm{Tr}(A_l X B_l^T) \\
s.t. \quad & \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M).
\end{aligned} \tag{16}$$

This procedure is summarized in Algorithm 1. By updating the matrices alternately based on the two steps, the iterative scheme will converge to a local minimum of (15).

Algorithm 1. The Proposed Two-Step Approach for Solving (6)

Input: original incomplete data matrix M_Ω, where Ω is the position of the observed entries, tolerance ϵ_0.

Initialize: $X_1 = \mathcal{P}_\Omega(M)$.

repeat

STEP 1. Given X_l

$$[U_l, \Sigma_l, V_l] = svd(X_l),$$

where

$$U_l = (\mathbf{u}_1, \ldots, \mathbf{u}_m) \in \mathbb{R}^{m\times m},$$
$$V_l = (\mathbf{v}_1, \ldots, \mathbf{v}_n) \in \mathbb{R}^{n\times n}.$$

Compute A_l and B_l as

$$A_l = (\mathbf{u}_1, \ldots, \mathbf{u}_r)^T, B_l = (\mathbf{v}_1, \ldots, \mathbf{v}_r)^T.$$

STEP 2. Solve

$$X_{l+1} = \arg\min_X \|X\|_* - \mathrm{Tr}(A_l X B_l^T) \quad s.t \quad \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M).$$

until $\|X_{l+1} - X_l\|_F \le \epsilon_0$.

Return the recovered matrix.

How to solve problem (16) efficiently is critical in our algorithm. Because both $\|X\|_*$ and $-\mathrm{Tr}(A_l X B_l^T)$ are convex, the objective function in (16) is also convex. In the following we introduce three optimization schemes for solving (16) by using the ADMM, the APGL method, and the ADMM with adaptive penalty (ADMMAP), respectively.

In the following, we first introduce a very useful tool, known as the singular value shrinkage operator [17].

Definition 3.2. *Consider the SVD of a matrix $X \in \mathbb{R}^{m\times n}$:*

$$X = U\Sigma V^T, \Sigma = diag(\{\sigma_i\}_{1\le i\le \min(m,n)}). \tag{17}$$

Define the singular value shrinkage operator $\mathcal{D}_\tau$ as follows:

$$\mathcal{D}_\tau(X) = U\mathcal{D}_\tau(\Sigma)V^T, \mathcal{D}_\tau(\Sigma) = diag(\max\{\sigma_i - \tau, 0\}). \tag{18}$$

We have the following useful theorem:

Theorem 3.2 ([17]). *For each $\tau \ge 0$ and $Y \in \mathbb{R}^{m\times n}$, we have*

$$\mathcal{D}_\tau(Y) = \arg\min_X \frac{1}{2}\|X - Y\|_F^2 + \tau\|X\|_*. \tag{19}$$

4 The Optimization Using TNNR-ADMM

ADMM is an algorithm that is intended to blend the decomposability of dual ascent with the superior convergence properties of the method of multipliers. In this section, we propose a new algorithm to solve (16) called TNNR-ADMM. First, we rewrite (16) as follows:

$$\begin{aligned}
\min_{X,W} \quad & \|X\|_* - \mathrm{Tr}(A_l W B_l^T) \\
s.t. \quad & X = W, \mathcal{P}_\Omega(W) = \mathcal{P}_\Omega(M).
\end{aligned} \tag{20}$$

The augmented lagrange function of (20) is

$$\begin{aligned}
L(X, Y, W, \beta) =& \|X\|_* - \mathrm{Tr}(A_l W B_l^T) + \frac{\beta}{2}\|X - W\|_F^2 \\
&+ \mathrm{Tr}(Y^T(X - W)),
\end{aligned} \tag{21}$$

where $\beta > 0$ is the penalty parameter. Given the initial setting $X_1 = \mathcal{P}_\Omega(M)$, $W_1 = X_1$, and $Y_1 = X_1$, TNNR-ADMM

updates variables alternately by minimizing the augmented Lagrange function $L(X, Y, W, \beta)$ with respect to the variables in a Gauss-Seidel manner. Specifically, the optimization problem (20) can be solved via the following three steps.

Computing X_{k+1}. Fix W_k and Y_k, and minimize $L(X, Y_k, W_k, \beta)$ for X_{k+1} as follows:

$$\begin{aligned} X_{k+1} &= \arg\min_X L(X, Y_k, W_k, \beta) \\ &= \arg\min_X \|X\|_* - \mathrm{Tr}(A_l W_k B_l^T) + \frac{\beta}{2}\|X - W_k\|_F^2 \\ &\quad + \mathrm{Tr}(Y_k^T(X - W_k)). \end{aligned} \tag{22}$$

Ignoring constant terms, this can be rewritten as

$$X_{k+1} = \arg\min_X \ \|X\|_* + \frac{\beta}{2}\|X - \left(W_k - \frac{1}{\beta}Y_k\right)\|_F^2.$$

By Theorem 3.2, we can obtain the closed form solution of the above problem (22) as follows:

$$X_{k+1} = \mathcal{D}_{\frac{1}{\beta}}\left(W_k - \frac{1}{\beta}Y_k\right). \tag{23}$$

Computing W_{k+1}. Fix X_{k+1} and Y_k to calculate W_{k+1} as follows:

$$\begin{aligned} W_{k+1} &= \arg\min_{\mathcal{P}_\Omega(W)=\mathcal{P}_\Omega(M)} L(X_{k+1}, Y_k, W, \beta) \\ &= \arg\min_{\mathcal{P}_\Omega(W)=\mathcal{P}_\Omega(M)} \|X_{k+1}\|_* - \mathrm{Tr}(A_l W B_l^T) \\ &\quad + \frac{\beta}{2}\|X_{k+1} - W\|_F^2 + \mathrm{Tr}(Y_k^T(X_{k+1} - W)). \end{aligned} \tag{24}$$

Discarding the constant terms, we can rewrite the above problem as follows:

$$W_{k+1} = \arg\min_{\mathcal{P}_\Omega(W)=\mathcal{P}_\Omega(M)} \frac{\beta}{2}\left\|W - \left(X_{k+1} + \frac{1}{\beta}\left(A_l^T B_l + Y_k\right)\right)\right\|_F^2,$$

which has a closed form solution. First, we set

$$W_{k+1} = X_{k+1} + \frac{1}{\beta}(A_l^T B_l + Y_k). \tag{25}$$

Then we fix the values at the observed entries and obtain

$$W_{k+1} = \mathcal{P}_{\Omega^c}(W_{k+1}) + \mathcal{P}_\Omega(M). \tag{26}$$

It is easy to verify that W_{k+1} given above is the unique optimal solution of problem (24).

Computing Y_{k+1}. Fix X_{k+1} and W_{k+1}, calculate Y_{k+1} as follows:

$$Y_{k+1} = Y_k + \beta(X_{k+1} - W_{k+1}). \tag{27}$$

The whole procedure of TNNR-ADMM is summarized in Algorithm 2. The main computational cost of TNNR-ADMM in each iteration is the computation of SVD in STEP 1. The additional cost in STEP 2 and STEP 3 is much smaller. For large-size problems, we can adopt some existing techniques [40] to accelerate the computation of SVD and make Algorithm 2 more efficient. Furthermore, the convergence of Algorithm 2 for problem (20) is guaranteed by the ADMM [41].

Algorithm 2. The Optimization using TNNR-ADMM
Input: A_l, B_l, M_Ω and tolerance ϵ are given.
Initialize: $X_1 = M_\Omega$, $W_1 = X_1$, $Y_1 = X_1$, and $\beta = 1$.
repeat
STEP 1. $X_{k+1} = \mathcal{D}_{\frac{1}{\beta}}(W_k - \frac{1}{\beta}Y_k)$.
STEP 2. $W_{k+1} = X_{k+1} + \frac{1}{\beta}(A_l^T B_l + Y_k)$.
Fix values at observed entries
$W_{k+1} = \mathcal{P}_{\Omega^c}(W_{k+1}) + \mathcal{P}_\Omega(M)$.
STEP 3. $Y_{k+1} = Y_k + \beta(X_{k+1} - W_{k+1})$.
until $\|X_{k+1} - X_k\|_F \le \epsilon$.

5 The Optimization Using TNNR-APGL

Although TNNR-ADMM can solve problem (16) directly, it may be less effective for applications with noisy data. To this end, we relax the constrained problem (16) into

$$\min_X \|X\|_* - \mathrm{Tr}(A_l X B_l^T) + \frac{\lambda}{2}\|\mathcal{P}_\Omega(X) - \mathcal{P}_\Omega(M)\|_F^2, \tag{28}$$

for some $\lambda > 0$, which is an unconstrained nonsmooth convex optimization problem. This type of problem has attracted a lot of attention recently in machine learning and computer vision. Considering the fact that classical gradient algorithms fail to solve nonsmooth problems, many accelerated gradient techniques [42], [43] are proposed with a convergence rate of $O(\frac{1}{k^2})$, where k is the number of iterations. These methods are built upon early work of Nesterov [44] and have outstanding ability to deal with large-scale, possibly nonsmooth problems. In this section, we adopt an accelerated proximal gradient line (APGL) search method proposed by Beck and Teboulle [43] to solve problems of the following form:

$$\min_X F(X) = g(X) + f(X), \tag{29}$$

where g is a closed, convex, possibly nondifferentiable function and f is a convex and differentiable function. First, for any $t > 0$, the APGL method constructs an approximation of $F(X)$ at a given point Y as

$$Q(X, Y) = f(Y) + \langle X - Y, \nabla f(Y)\rangle + \frac{1}{2t}\|X - Y\|_F^2 + g(X). \tag{30}$$

Then APGL solves the optimization problem (29) by iteratively updating X, Y, and t. In the kth iteration, we update X_{k+1} as the unique minimizer of $Q(X, Y_k)$:

$$\begin{aligned} X_{k+1} &= \arg\min_X Q(X, Y_k) \\ &= \arg\min_X g(X) + \frac{1}{2t_k}\|X - (Y_k - t_k\nabla f(Y_k))\|_F^2. \end{aligned} \tag{31}$$

In problem (28), we choose

$$g(X) = \|X\|_*,$$

and

$$f(X) = -\mathrm{Tr}(A_l X B_l^T) + \frac{\lambda}{2}\|\mathcal{P}_\Omega(X) - \mathcal{P}_\Omega(M)\|_F^2.$$

So according to Theorem 3.2, we get

$$\begin{aligned} X_{k+1} &= \arg\min_X \|X\|_* + \frac{1}{2t_k}\|X - (Y_k - t_k\nabla f(Y_k))\|_F^2 \\ &= \mathcal{D}_{t_k}(Y_k - t_k\nabla f(Y_k)) \\ &= \mathcal{D}_{t_k}(Y_k + t_k(A_l^T B_l - \lambda(\mathcal{P}_\Omega(Y_k) - \mathcal{P}_\Omega(M)))). \end{aligned} \tag{32}$$

Finally, t_{k+1} and Y_{k+1} are updated in the same way as [31]:

$$\begin{aligned} t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2}, \\ Y_{k+1} &= X_{k+1} + \frac{t_k - 1}{t_{k+1}}(X_{k+1} - X_k). \end{aligned} \tag{33}$$

We summarize the main procedure for solving problem (28) by APGL in Algorithm 3, which is called TNNR-APGL. Compared with TNNR-ADMM, Algorithm 3 is more suitable for dealing with noisy data, especially for real computer vision problems. Moreover, Algorithm 3 has a convergence rate of $O(\frac{1}{k^2})$, which is guaranteed by the convergence property of the APGL method [43].

Algorithm 3. The Optimization using APGL

Input: A_l, B_l, M_Ω and tolerance ϵ.

Initialize: $t_1 = 1, X_1 = M_\Omega, Y_1 = X_1$.

repeat

STEP 1. Update X_{k+1} as $X_{k+1} = \mathcal{D}_{t_k}(Y_k + t_k(A_l^T B_l - \lambda(\mathcal{P}_\Omega(Y_k) - \mathcal{P}_\Omega(M))))$.

STEP 2. $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.

STEP 3. $Y_{k+1} = X_k + \frac{t_k - 1}{t_{k+1}}(X_k - X_{k-1})$.

until $\|X_{k+1} - X_k\|_F \le \epsilon$.

6 The Optimization Using TNNR-ADMMAP

In Algorithm 2, we solve the problem (20) iteratively by considering the two constraints separately. Specifically, We consider the constraint $X = W$ in the augmented lagrange function, while the condition $\mathcal{P}_\Omega(W) = \mathcal{P}_\Omega(M)$ is enforced during the computation of W_{k+1}. The inconsistency of the two constraints may slow down the convergence. Moreover, as we know, the convergence of ADMM becomes slower with more constraints [45].

In this section, we propose a new approach to solve the problem (16) by using the ADMM with adaptive penalty (TNNR-ADMMAP). First, we show that the two constraints in problem (20) can be merged into a new constraint in a special way and we can deal with all the constraints together to accelerate the convergence. Moreover, as it is hard to choose an optimal penalty parameter in advance, we also allow the penalty parameter to change adaptively and adopt a simple and efficient rule to update it. In this way, we can further speed up the convergence.

6.1 The Reformulation of Problem (20)

Note that $X = W$ and $\mathcal{P}_\Omega(W) = \mathcal{P}_\Omega(M)$ are two linear constraints in the problem (20). To deal with the two constraints simultaneously, in this section, we reformulate the problem (20) as follows

$$\begin{aligned} \min_{X,W} \quad & \| X \|_* - \mathrm{Tr}(A_l W B_l^T) \\ s.t. \quad & \mathcal{A}(X) + \mathcal{B}(W) = C, \end{aligned} \tag{34}$$

where $\mathcal{A}$ and $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^{2m\times 2n}$ are linear operators defined as follows:

$$\mathcal{A}(X) = \begin{pmatrix} X & 0 \\ 0 & 0 \end{pmatrix},\ \mathcal{B}(W) = \begin{pmatrix} -W & 0 \\ 0 & \mathcal{P}_\Omega(W) \end{pmatrix},$$

and

$$C = \begin{pmatrix} 0 & 0 \\ 0 & \mathcal{P}_\Omega(M) \end{pmatrix}.$$

Obviously, (34) is a typical linearly constrained separable convex problem and ADMM can be used to solve it efficiently. Before presenting the iterative scheme of ADMM for problem (34), we first discuss the properties of adjoint operators $\mathcal{A}$ and $\mathcal{B}$.

Suppose

$$Y = \begin{pmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{pmatrix},$$

where $Y_{ij} \in \mathbb{R}^{m\times n}$, $i = 1, 2$ and $j = 1, 2$. Denote $\mathcal{A}^*, \mathcal{B}^* : \mathbb{R}^{2m\times 2n} \to \mathbb{R}^{m\times n}$ as the adjoint operators of $\mathcal{A}$ and $\mathcal{B}$ separately satisfying

$$\langle \mathcal{A}(X), Y\rangle = \langle X, \mathcal{A}^*(Y)\rangle, \langle \mathcal{B}(X), Y\rangle = \langle X, \mathcal{B}^*(Y)\rangle. \tag{35}$$

By the definition of operators $\mathcal{A}$ and $\mathcal{B}$, it is easy to verify that

$$\begin{aligned} \langle \mathcal{A}(X), Y\rangle &= \mathrm{Tr}\begin{pmatrix} X & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{pmatrix}^T \\ &= \mathrm{Tr}(XY_{11}^T) \\ &= \langle X, Y_{11}\rangle \end{aligned}$$

and

$$\begin{aligned} \langle \mathcal{B}(W), Y\rangle &= \mathrm{Tr}\begin{pmatrix} -W & 0 \\ 0 & \mathcal{P}_\Omega(W) \end{pmatrix}\begin{pmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{pmatrix}^T \\ &= \mathrm{Tr}\begin{pmatrix} -WY_{11}^T & -WY_{21}^T \\ \mathcal{P}_\Omega(W)Y_{12}^T & \mathcal{P}_\Omega(W)Y_{22}^T \end{pmatrix} \\ &= \mathrm{Tr}(-WY_{11}^T) + \mathrm{Tr}(\mathcal{P}_\Omega(W)Y_{22}^T) \\ &= \langle W, -Y_{11}\rangle + \langle \mathcal{P}_\Omega(W), Y_{22}\rangle \\ &= \langle W, -Y_{11}\rangle + \langle W, \mathcal{P}_\Omega(Y_{22})\rangle \\ &= \langle W, -Y_{11} + \mathcal{P}_\Omega(Y_{22})\rangle, \end{aligned}$$

where the fifth equality holds since the orthogonal projection operator $\mathcal{P}_\Omega$ is self-adjoint.

Thus, by the definition of the adjoint operator (35), the adjoint operators $\mathcal{A}^*$ and $\mathcal{B}^*$ can be computed as

$$\begin{aligned} \mathcal{A}^*(Y) &= Y_{11}, \\ \mathcal{B}^*(Y) &= -Y_{11} + \mathcal{P}_\Omega(Y_{22}). \end{aligned} \tag{36}$$

Let us rewrite the augmented Lagrange function for the optimization problem (34) as

$$\begin{aligned} \mathcal{L}_{AP}(X, W, Y, \beta) = &- \mathrm{Tr}(A_l W B_l^T) + \langle Y, \mathcal{A}(X) + \mathcal{B}(W) - C\rangle \\ &+ \|X\|_* + \frac{\beta}{2}\|\mathcal{A}(X) + \mathcal{B}(W) - C\|_F^2, \end{aligned} \tag{37}$$

where Y is the Lagrange multiplier matrix and $\beta > 0$ is the penalty parameter. The iterative scheme of ADMM for problem (34) is given as follows:

$$\begin{aligned} X_{k+1} &= \arg\min_X \mathcal{L}_{AP}(X, W_k, Y_k, \beta) \\ &= \arg\min_X \frac{\beta}{2}\|\mathcal{A}(X) + \mathcal{B}(W_k) - C + \frac{1}{\beta}Y_k\|_F^2 \\ &\quad + \|X\|_*, \end{aligned} \tag{38}$$

$$\begin{aligned} W_{k+1} &= \arg\min_W \mathcal{L}_{AP}(X_{k+1}, W, Y_k, \beta) \\ &= \arg\min_W \frac{\beta}{2}\|\mathcal{A}(X_{k+1}) + \mathcal{B}(W) - C + \frac{1}{\beta}Y_k\|_F^2 \\ &\quad - \mathrm{Tr}(A_l W B_l^T), \end{aligned} \tag{39}$$

$$Y_{k+1} = Y_k + \beta[\mathcal{A}(X_{k+1}) + \mathcal{B}(W_{k+1}) - C]. \tag{40}$$

6.2 The Iterative Scheme of TNNR-ADMMAP

In this section, we introduce our TNNR-ADMMAP algorithm based on the iterative scheme (38)-(40) via the following three steps.

Computing X_{k+1}. With the special structure of $\mathcal{A}$ and $\mathcal{B}$, we can solve the subproblem (38) as follows:

$$\begin{aligned} X_{k+1} &= \arg\min_X \frac{\beta}{2}\|\mathcal{A}(X) + \mathcal{B}(W_k) - C + \frac{1}{\beta}Y_k\|_F^2 + \|X\|_* \\ &= \arg\min_X \frac{\beta}{2}\|P\|_F^2 + \|X\|_*, \end{aligned} \tag{41}$$

where

$$P = \begin{pmatrix} X - W_k + \frac{1}{\beta}(Y_k)_{11} & \frac{1}{\beta}(Y_k)_{12} \\ \frac{1}{\beta}(Y_k)_{21} & \mathcal{P}_\Omega(W_k - M) + \frac{1}{\beta}(Y_k)_{22} \end{pmatrix}.$$

Ignoring the constant terms, we can calculate X_{k+1} as follows:

$$X_{k+1} = \arg\min_X \frac{\beta}{2}\|X - W_k + \frac{1}{\beta}(Y_k)_{11}\|_F^2 + \|X\|_*.$$

By Theorem 3.2, we can solve problem (38) as

$$X_{k+1} = \mathcal{D}_{\frac{1}{\beta}}\left(W_k - \frac{1}{\beta}(Y_k)_{11}\right). \tag{42}$$

Computing W_{k+1}. It is obvious that the objective function of (39) is a quadratic function. So, we can solve subproblem (39) directly. By setting the first derivative of $\mathcal{L}_{AP}(X_{k+1}, W, Y_k, \beta)$ to zero, we get

$$\beta\mathcal{B}^*\left[\mathcal{B}(W) + \mathcal{A}(X_{k+1}) - C + \frac{1}{\beta}Y_k\right] - A_l^T B_l = 0,$$

which can be rewritten as

$$\mathcal{B}^*\mathcal{B}(W) = \frac{1}{\beta}A_l^T B_l - \mathcal{B}^*\left[\mathcal{A}(X_{k+1}) - C + \frac{1}{\beta}Y_k\right]. \tag{43}$$

From the property of adjoint operator $\mathcal{B}^*$, the left-hand side of the above equation can be rewritten as

$$\mathcal{B}^*\mathcal{B}(W) = \mathcal{B}^*\begin{pmatrix} -W & 0 \\ 0 & \mathcal{P}_\Omega(W) \end{pmatrix} = W + \mathcal{P}_\Omega(W). \tag{44}$$

Then we apply the orthogonal projection operator $\mathcal{P}_\Omega$ on both sides of (44), and finally we get

$$\mathcal{P}_\Omega(W) = \frac{1}{2}\mathcal{P}_\Omega(\mathcal{B}^*\mathcal{B}(W)). \tag{45}$$

From (44) and (45), we obtain

$$W_{k+1} = \mathcal{B}^*\mathcal{B}(W) - \mathcal{P}_\Omega(W) = \mathcal{B}^*\mathcal{B}(W) - \frac{1}{2}\mathcal{P}_\Omega(\mathcal{B}^*\mathcal{B}(W)).$$

The computation of $\mathcal{B}^*\mathcal{B}(W)$ is critical for the updating of W. Based on equality (43), we can compute $\mathcal{B}^*\mathcal{B}(W)$ directly as follows:

$$\begin{aligned} \mathcal{B}^*\mathcal{B}(W) &= \frac{1}{\beta}A_l^T B_l - \mathcal{B}^*[\mathcal{A}(X_{k+1}) - C + Y_k/\beta] \\ &= \frac{1}{\beta}A_l^T B_l - \mathcal{B}^*\left[\begin{pmatrix} X_{k+1} & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & \mathcal{P}_\Omega(M) \end{pmatrix} \right. \\ &\quad \left. + \frac{1}{\beta}\begin{pmatrix} (Y_k)_{11} & (Y_k)_{12} \\ (Y_k)_{21} & (Y_k)_{22} \end{pmatrix}\right] \\ &= -\mathcal{B}^*\begin{pmatrix} \frac{1}{\beta}(Y_k)_{11} + X_{k+1} & \frac{1}{\beta}(Y_k)_{12} \\ \frac{1}{\beta}(Y_k)_{21} & \frac{1}{\beta}(Y_k)_{22} - \mathcal{P}_\Omega(M) \end{pmatrix} \\ &\quad + \frac{1}{\beta}A_l^T B_l. \end{aligned}$$

From property (36) of adjoint operator $\mathcal{B}^*$, we can get

$$\begin{aligned} &\mathcal{B}^*\begin{pmatrix} \frac{1}{\beta}(Y_k)_{11} + X_{k+1} & \frac{1}{\beta}(Y_k)_{12} \\ \frac{1}{\beta}(Y_k)_{21} & \frac{1}{\beta}(Y_k)_{22} - \mathcal{P}_\Omega(M) \end{pmatrix} \\ &= -\left(\frac{1}{\beta}(Y_k)_{11} + X_{k+1}\right) + \mathcal{P}_\Omega\left(\frac{1}{\beta}(Y_k)_{22} - M\right). \end{aligned}$$

Thus,

$$\begin{aligned} &\mathcal{B}^*\mathcal{B}(W) \\ &= -\left[-\left(\frac{1}{\beta}(Y_k)_{11} + X_{k+1}\right) + \mathcal{P}_\Omega\left(\frac{1}{\beta}(Y_k)_{22} - M\right)\right] \\ &\quad + \frac{1}{\beta}A_l^T B_l. \end{aligned} \tag{46}$$

Based on (45) and (46), we can compute W_{k+1} as follows:

$$\begin{aligned} W_{k+1} &= \mathcal{B}^*\mathcal{B}(W) - \mathcal{P}_\Omega(W) \\ &= \mathcal{B}^*\mathcal{B}(W) - \frac{1}{2}\mathcal{P}_\Omega(\mathcal{B}^*\mathcal{B}(W)) \\ &= \frac{1}{2\beta}\mathcal{P}_\Omega[\beta(M - X_{k+1}) - (A_l^T B_l + (Y_k)_{11} + (Y_k)_{22})] \\ &\quad + X_{k+1} + \frac{1}{\beta}(A_l^T B_l + (Y_k)_{11}). \end{aligned} \tag{47}$$

Computing Y_{k+1}. The update of Lagrange multipliers is the same as (40).

From the above three steps, we can see that the updating of X and W of TNNR-ADMMAP is quite different from that of TNNR-ADMM. The main difference is that the iterative

TABLE 1
Running Time of Randomly Masked Image Recovery

Method	Observed Ratios					
	0.4	0.5	0.6	0.7	0.8	0.9
TNNR-ADMM	32.57	30.23	28.26	25.32	22.35	18.42
TNNR-APGL	34.05	30.21	26.41	25.18	21.69	21.83
TNNR-ADMMAP	**19.01**	**19.03**	**17.95**	**16.79**	**16.52**	**15.41**

scheme of TNNR-ADMMAP deals with the two constraints, $X = W$ and $\mathcal{P}_\Omega(W) = \mathcal{P}_\Omega(M)$, simultaneously.

6.3 Adaptive Penalty

In the traditional ADMM [46] and linearized ADMM approaches [38], the penalty parameter β is fixed. Previous studies have shown that if the fixed penalty parameter β is chosen too small or too large, the computational cost can increase significantly. On the other hand, it is not easy to choose an optimal fixed β. Thus, a dynamical β may be preferred in real applications. Based on the theoretical results in [39], we adopt the following adaptive update strategy for the penalty parameter β:

$$\beta_{k+1} = \min(\beta_{\max}, \rho\beta_k), \tag{48}$$

where $\beta_{\max}$ is an upper bound of β_k. The value of ρ is defined as

$$\rho = \begin{cases} \rho_0, & \text{if } \frac{\beta_k \max\{\|X_{k+1}-X_k\|_F, \|W_{k+1}-W_k\|_F\}}{\|C\|_F} < \kappa \\ 1, & \text{otherwise,} \end{cases} \tag{49}$$

where $\rho_0 > 1$ is a constant and $\kappa > 0$ is a threshold chosen in advance. When the difference between (X_{k+1}, W_{k+1}) and (X_k, W_k) is small enough, β_{k+1} increases to $\rho_0\beta_k$ and the convergence rate is improved.

We summarize the complete procedure of TNNR-ADMMAP in Algorithm 4. As the choice of $\{\beta_k\}$ is nondecreasing, the convergence of TNNR-ADMMAP is guaranteed by the theoretical result in [47].

Algorithm 4. Inner optimization by ADMMAP
Input: A_l, B_l, M_Ω and tolerance ϵ.
Initialize: $X_1 = M_\Omega$, $W_1 = X_1$, $\kappa = 10^{-3}$, ρ_0, $Y_1 = X_1$ and β_0.
repeat
STEP 1. Set Y_k and W_k fixed
$$X_{k+1} = \mathcal{D}_{\frac{1}{\beta_k}}(W_k - \tfrac{1}{\beta_k}(Y_k)_{11}).$$
STEP 2. Set Y_k and X_{k+1} fixed
$$\begin{aligned} & W_{k+1} \\ = \; & \tfrac{1}{2\beta_k}\mathcal{P}_\Omega[\beta_k(M - X_{k+1}) - (A_l^T B_l + (Y_k)_{11} + (Y_k)_{22})] \\ & + X_{k+1} + \tfrac{1}{\beta_k}(A_l^T B_l + (Y_k)_{11}). \end{aligned}$$
STEP 3. Set X_{k+1} and W_{k+1} fixed
$$Y_{k+1} = Y_k + \beta_k(\mathcal{A}(X_{k+1}) + \mathcal{B}(W_{k+1}) - C)).$$
STEP 4. Update β_k by (48) and (49)
until $\|X_{k+1} - X_k\|_F \le \epsilon$.

Overall, TNNR-ADMM and TNNR-ADMMAP both solve the optimization problem (16) with hard constraints, while TNNR-APGL solves the same problem with soft constraints. The efficiency comparison (see Table 1) indicates that TNNR-ADMMAP is the most efficient one among our three proposed procedures. In practice, TNNR-ADMMAP is preferred to solve noiseless problems, while TNNR-APGL is preferred for problems with noisy matrices.

7 Experimental Results

In this section, we conduct several experiments on both synthetic data and real visual data to show the effectiveness of our proposed approaches (TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP) for matrix completion. We compare the following six matrix completion approaches:

- Our proposed **TNNR-ADMM** algorithm: The ADMM method is used in the optimization of problem (20) under hard constraint.
- Our proposed **TNNR-APGL** algorithm: The APGL method is used to solve unconstrained problem (28).
- Our proposed **TNNR-ADMMAP** algorithm: The ADMM method with adaptive penalty is used in the optimization of problem (34) under hard constraint.
- **SVT** algorithm [17], which aims to recover the missing values by seeking the optimal solution of the nuclear norm minimizaiton problem.
- **SVP** algorithm [37], which seeks the minimum rank solution for affine constraints that satisfy the restricted isometry property.
- **OptSpace** algorithm [48], which considers the matrix completion problem in a matrix factorization view based on the traditional SVD heuristic.

The experiments are conducted by using Matlab on a thinkpad W520 laptop of i7-2720 CPU and 16G memory.

7.1 Synthetic Data

We generate $m \times n$ matrices of rank r_0 by sampling two matrices, i.e., $M_L \in \mathbb{R}^{m\times r_0}$ and $M_R \in \mathbb{R}^{r_0\times n}$, each having i.i.d. Gaussian entries, and setting $M = M_L M_R$. The locations of observed indices Ω are sampled uniformly at random. Let p be the percentage of observed entries over $m * n$. We generate synthetic data as follows:

$$B = M + \sigma Z,\ B_{ij} = M_{ij} + \sigma Z_{ij},\ (i,j) \in \Omega,$$

where Z is Gaussian white noise with standard deviation 1. Denote the full noiseless matrix as X_{full} and the solution given by an algorithm as X_{sol}. We define the total reconstruction error as $\|\mathcal{P}_{\Omega^c}(X_{sol} - X_{full})\|_F$, which is a commonly used criterion in matrix completion.

In this section, we compare the reconstruction error of the six methods under different problem settings such as different matrix ranks (r_0), different noise levels (σ), and different observed ratios ($\#|\Omega|/(m*n)$).

First, we fix the matrix size to be $m = 100, n = 200$, and set the rank of the matrix as $r_0 = 10$. We run the six algorithms under different noise levels and different observed ratios. The results are shown in Fig. 2.

Then, we fix the matrix size to be $m = 100, n = 200$, and set the noise level as $\sigma = 0.5$. We run the six algorithms under different observed ratios on matrices with different ranks. The results are shown in Fig. 3. As can be seen, as the matrix rank (r_0) increases, the total reconstruction error becomes larger. This is reasonable since the structure of the matrix becomes more complex as the rank (r_0) increases, and consequently the recovery of the incomplete matrix becomes more difficult.

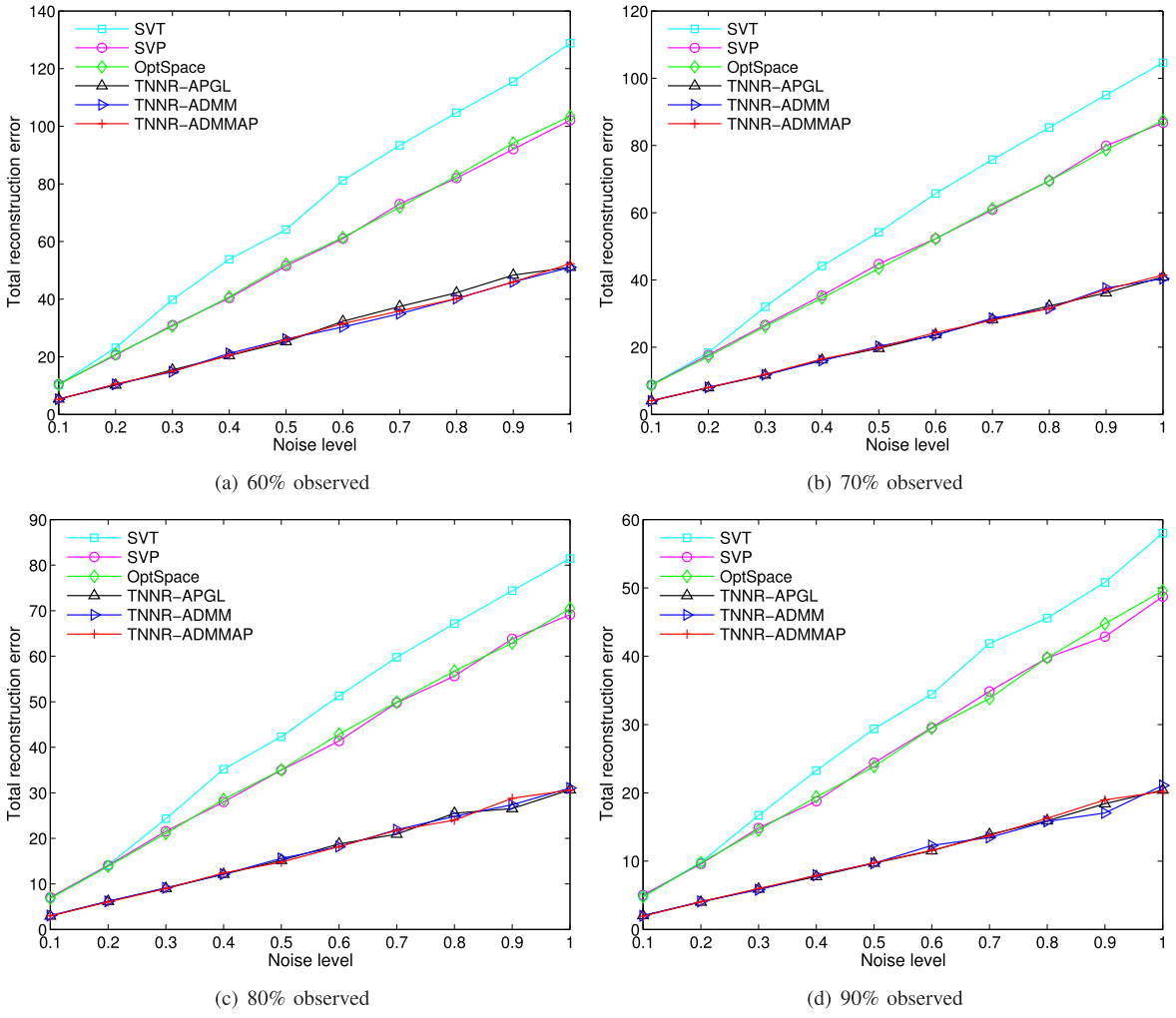


Fig. 2. The reconstruction error versus the noise level using the synthetic dataset. The observed ratio ranges from 60 to 90 percent. These results show that our proposed TNNR-APGL, TNNR-ADMM, and TNNR-ADMMAP algorithms can achieve much lower reconstruction error compared with the three state-of-the-art matrix completion algorithms (SVT, SVP, and OptSpace).

Figs. 2 and 3 show that our methods based on the TNNR can recover the missing values with much smaller reconstruction error in all the cases. The results show that TNNR can better capture the structure of low rank matrices, and is more robust to noise.

7.2 Real Visual Data

In many cases, the images can be viewed as approximately low rank matrices, as shown in Fig. 1. Sometimes the images may be partially damaged due to coding and transmission issues or may be partially covered by some text or logos. Naturally, the matrix completion algorithms can be applied to recover the missing information for these images. As color images have three channels (red, green, and blue), we simply deal with different channels separately and combine the results to get the final result.

The Peak Signal-to-Noise Ratio (PSNR) value is a widely used criterion to evaluate the quality of an image. We use the PSNR value of the recovered image to evaluate the performance of different methods. Suppose the total number of missing pixels is T, then the total squared error $\text{SE} = error_r^2 + error_g^2 + error_b^2$, the total mean squared error $\text{MSE} = \frac{SE}{3T}$, and PSNR can be calculated as $10 \times \log_{10}(\frac{255^2}{MSE})$.

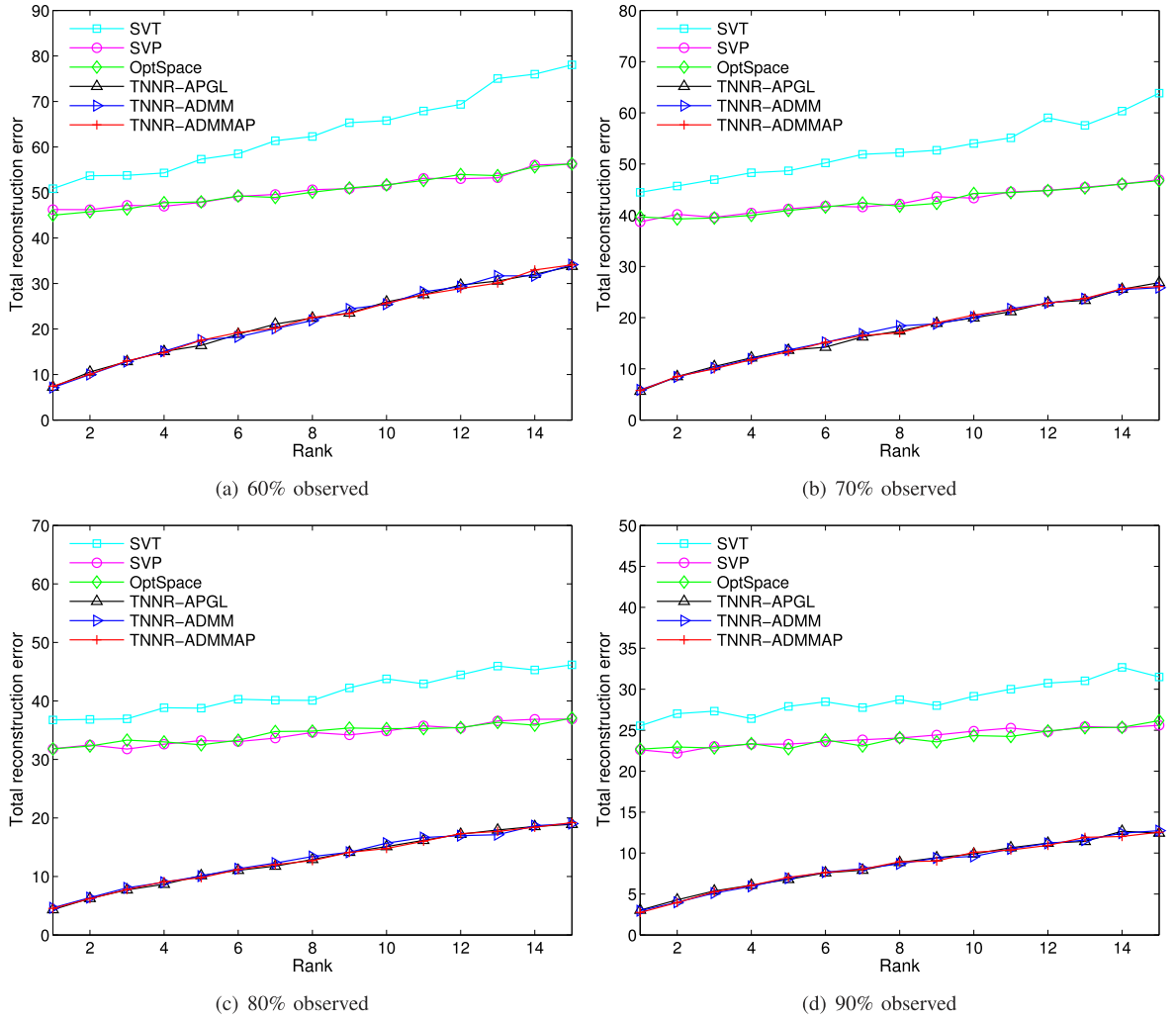
7.2.1 Parameter Setting

In our experiments, the parameters of SVP, SVT, and OptSpace are tuned to achieve the best performance. For TNNR-ADMM and TNNR-ADMMAP, we set both $\beta = 0.001$. For TNNR-ADMMAP, we empirically set $\beta_{max} = 10^{10}$, $\rho_0 = 1.9$, and $\kappa = 10^{-3}$. For TNNR-APGL, the parameter λ is empirically set to be 0.06.

For the stop conditions of TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP, we set the tolerance ϵ_0 in Algorithm 1 to be 10^{-3} and the tolerance ϵ in Algorithms 2-4 to be 10^{-4}. Given an incomplete image, for each one of the six algorithms we try all the possible values of r (the parameter in $\|X\|_r$) and choose the best result as the final recovered image. In most cases, it is sufficient to test r from 1 to 20.

7.2.2 Random Mask

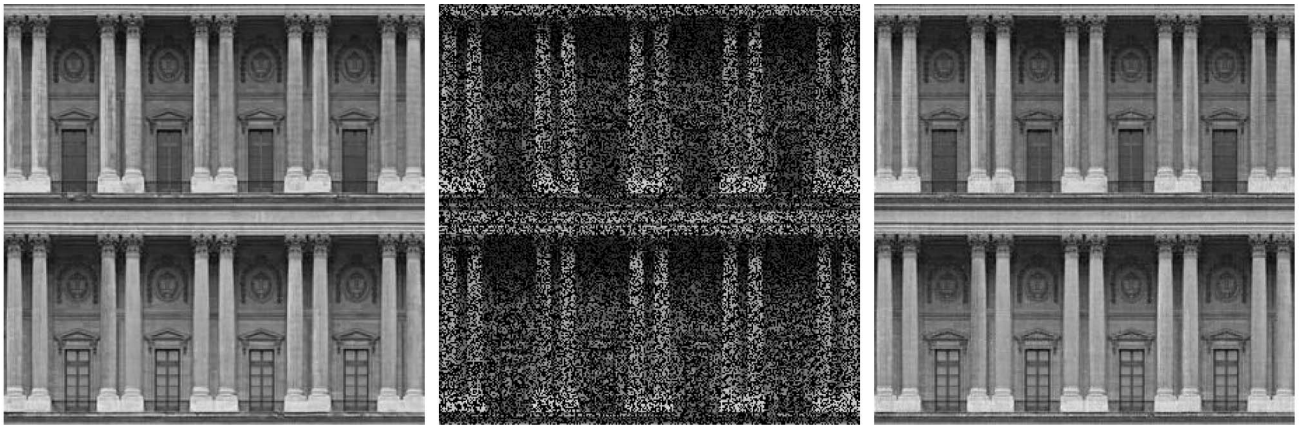
We first conduct experiments to deal with a relatively easy matrix completion problem where the missing entries are randomly distributed on the matrices. So, we randomly cover some pixels of an image (300×300), and then compare the recovery performance of the six algorithms. The results are shown in Figs. 4 and 5. We can see that the



(a) 60% observed (b) 70% observed

(c) 80% observed (d) 90% observed

Fig. 3. The reconstruction error versus the matrix rank (r_0) using the synthetic dataset. The observed ratio ranges from 60 to 90 percent. These results show that our proposed TNNR-APGL, TNNR-ADMM, and TNNR-ADMMAP algorithms can achieve much lower reconstruction error compared with the three state-of-the-art matrix completion algorithms (SVT, SVP, and OptSpace).



(a) Original image (b) Masked image (c) Recovered image by TNNR-ADMMAP

Fig. 4. Recovery of an incomplete image with random mask. We cover 50 percent pixels of the original image to get a masked image. And we use TNNR-ADMMAP to recover the missing values.The comparison results of all methods are shown in Fig. 5.

larger the observed ratio is the better recovery of the image that can be achieved. From Fig. 5, we can see that the TNNR-based algorithms can achieve much higher PSNR values compared with the other three state-of-the-art matrix completion methods.

7.2.3 Text Mask

Text removal is a hard task since the pixels covered by the text are not randomly distributed in the image and the text may cover some important texture information. We first check the position of the text and regard the corresponding

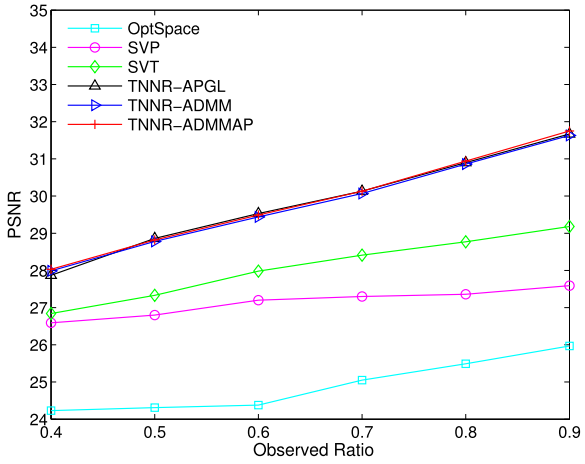


Fig. 5. The PSNR values of the recovered image under different observed ratios (random mask) by all six methods. Experimental results show that our three TNNR-based algorithms (TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP) outperform the competing methods for the recovery of the image with random mask.

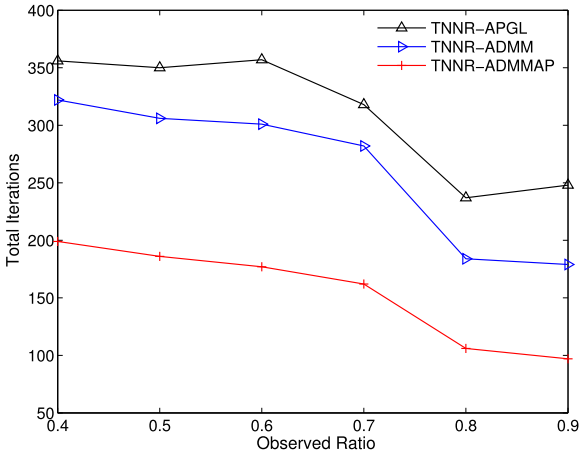


Fig. 6. Iterations needed for our three TNNR-based algorithms (TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP) to recover the given randomly masked image under different observed ratios. Experimental results show that TNNR-ADMMAP is much faster than TNNR-ADMM and TNNR-APGL.

entries as missing values. So, the text removal problem can be considered as a general matrix completion problem. Figs. 7 and 8 show the experimental results of the six matrix completion algorithms. Specifically, for the example image in Fig. 7, the PSNR values for SVT, SVP, OptSpace, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP are 26.95, 24.41, 23.50, 28.04, 28.20, and 28.38, respectively. And for the example image in Fig. 8, the PSNR values for SVT, SVP, OptSpace, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP are 23.40, 21.57, 20.20, 24.29, 24.25, and 24.30, respectively. From these results we can see that our proposed TNNR-based algorithms achieve better performance than the other three matrix completion algorithms.

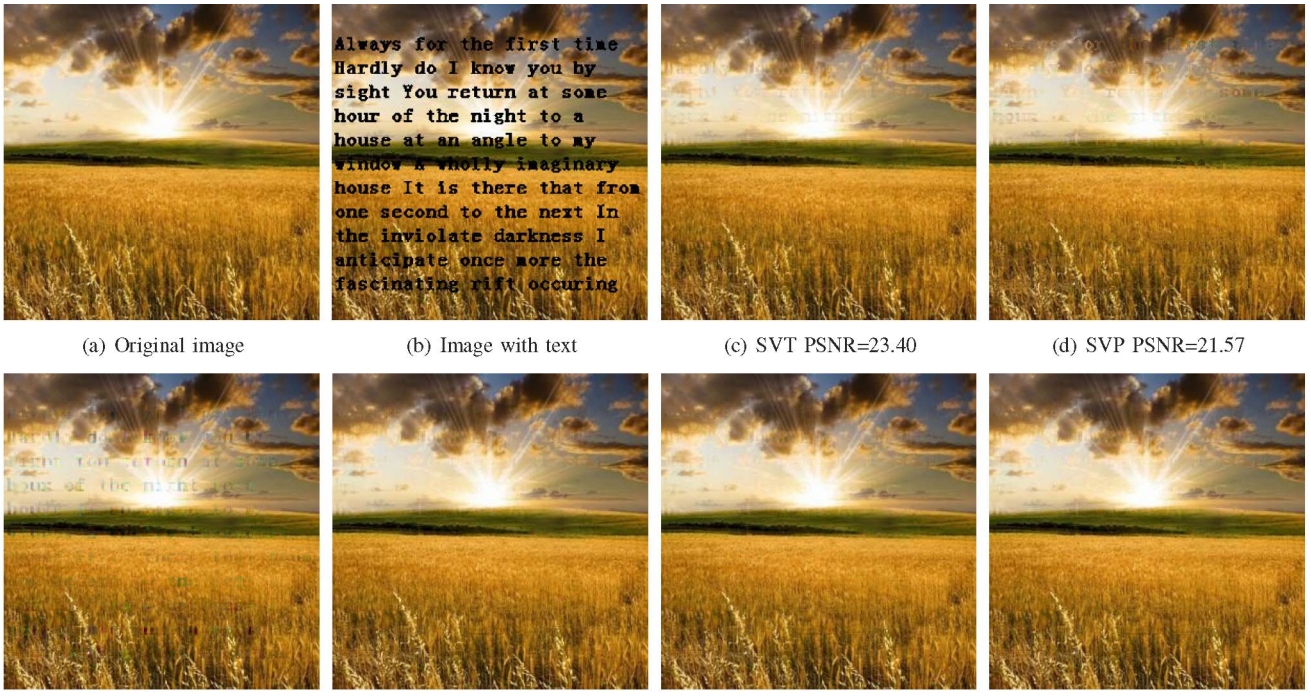
7.2.4 Block Mask

In some situations, there may be large missing blocks in an image, which makes the recovery of missing information more difficult. As shown in Fig. 9a, some windows in the original image are broken. Some pixels are damaged and there are some annoying texts and logos in the bottom of the image. We first mask all these kinds of pixels in the image, as shown in Fig. 9b. And then we recover these large missing blocks by using six matrix completion algorithms.

(a) Original image (b) Image with text (c) SVT PSNR=26.95 (d) SVP PSNR=24.41

(e) OptSpace PSNR=23.50 (f) TNNR-ADMM PSNR=28.04 (g) TNNR-APGL PSNR=28.20 (h) TNNR-ADMMAP PSNR=28.38

Fig. 7. Comparison of matrix completion algorithms for text removal problem. (a) Original image. (b) Image with text. (c)-(h) Recovered images by SVT, SVP, OptSpace, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP, respectively. These results show that our proposed TNNR-based algorithms outperform the competing methods.



(a) Original image (b) Image with text (c) SVT PSNR=23.40 (d) SVP PSNR=21.57

(e) OptSpace PSNR=20.20 (f) TNNR-ADMM PSNR=24.29 (g) TNNR-APGL PSNR=24.25 (h) TNNR-ADMMAP PSNR=24.30

Fig. 8. Comparison of matrix completion algorithms for text removal problem. (a) Original image. (b) Image with text. (c)-(h) Recovered images by SVT, SVP, OptSpace, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP, respectively. These results show that our proposed TNNR-based algorithms outperform the competing methods.

Figs. 9c, 9d, 9e, 9f, 9g, 9h show the recovered images by using the six different matrix completion algorithms. Comparing the recovered blocks by different matrix completion algorithms with the corresponding parts in the original image, we can see that our proposed TNNR-based algorithms (TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP) can satisfactorily recover the missing pixels.

7.3 Convergence Rate and Computational Cost

As we can see, in all the experiments, our TNNR-based algorithms (TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP) are more effective than the other three matrix completion algorithms. Since all three TNNR-based algorithms are used to solve the same convex optimization problem (16), their results are very similar. However, their computational costs are quite different. From Algorithms 1-4, we can see that the main computational cost of each TNNR-based algorithm is the calculation of SVD. And in each iteration all three solvers involve only one SVD computation. Note that the TNNR-based algorithms have both inner and outer iterations. Empirical experimental results show that usually 10 outer iterations are sufficient for convergence. And 20-30 inner iterations are needed for each outer iteration's convergence. We evaluate our three TNNR-based algorithms using the total number of iterations and the running time needed to recover an given image.

We conduct an experiment for recovering the randomly masked color image in Fig. 4b (the image size is 300×300) under different observed ratios to test the convergence rate and computational cost of our proposed three algorithms. First, from Fig. 5, we can see that the recovered results of the image Fig. 4b under different ratios by TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP have almost the same PSNR value. Fig. 6 shows the total number of iterations versus the observed ratios. As can be seen, TNNR-ADMMAP converges much faster than TNNR-APGL and TNNR-ADMM. The total number of iterations needed for TNNR-ADMMAP to converge is roughly half of that for TNNR-APGL and TNNR-ADMM, while they obtain almost the same result. Furthermore, considering the slightly different time cost in each iteration of our proposed three algorithms, we also give the detailed running time information of our proposed three algorithms versus the observed ratios in Table 1. We note that TNNR-ADMM and TNNR-APGL have nearly the same time cost to recover the missing information of the incomplete image under different observed ratios. And TNNR-ADMMAP only needs 15-19 seconds (almost 60 percent of the time cost of TNNR-ADMM and TNNR-APGL) to obtain the final recovered matrix. This experiment shows that TNNR-ADMMAP performs much faster than TNNR-APGL and TNNR-ADMM without sacrificing the recovery accuracy. Moreover, we can see that as the observed ratio increases, the computational time cost needed for each algorithm to converge gets smaller.

8 Conclusion

We have presented a novel method called truncated nuclear norm regularization for estimating missing values in a matrix. Unlike traditional nuclear norm heuristics, which take into account all the singular values, our approach tries to minimize the summation of the smallest $\min(m,n)-r$ singular values, where r is the matrix rank. This is due to the fact that the smallest $\min(m,n)-r$ singular values have little effect on the approximation of the matrix rank when the original matrix has a low rank structure. In this way, our proposed approaches can give better approximation to the

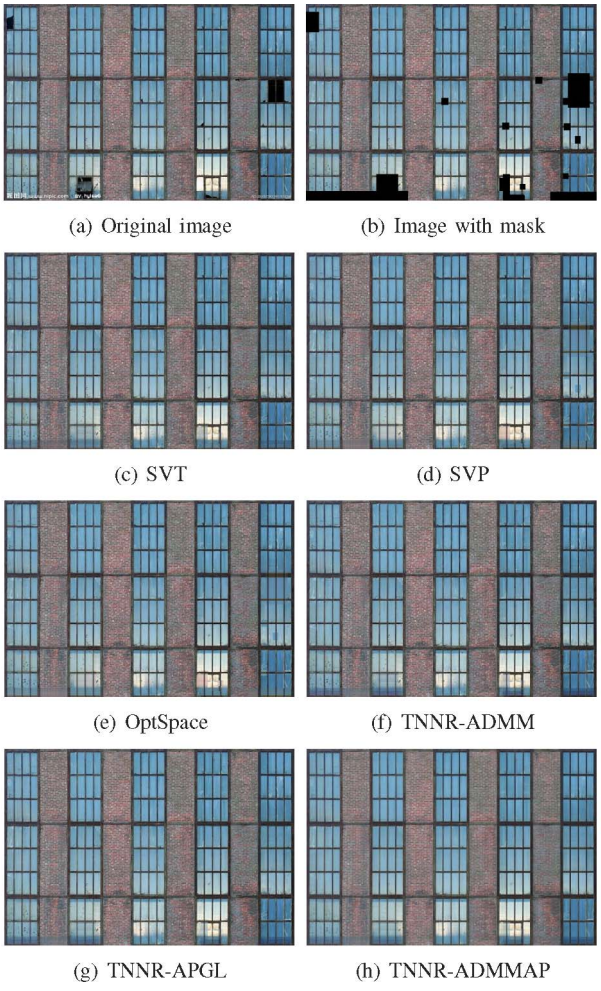


Fig. 9. Comparison of different methods for recovering missing blocks of a natural image. (a) Original image. (b) Image with mask. (c)-(h) Recovered images by SVT, SVP, OptSpace, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP, respectively. Experimental results show that our TNNR-based algorithms achieve better performance to recover the details of the original image.

rank function than the nuclear norm based approaches. Although the new objective function is no longer convex, we introduce three simple but efficient iterative schemes, i.e., TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP, to solve the optimization problem by using the ADMM, APGL, and ADMMAP methods, respectively. We conduct several experiments on both synthetic and real visual datasets. The experimental results demonstrate the advantages of the TNNR-based algorithms in comparison with three state-of-the-art matrix completion methods based on the nuclear norm or matrix factorization. We can also observe from our experiments that among the three TNNR-based algorithms, TNNR-ADMMAP is much more efficient than TNNR-ADMM and TNNR-APGL due to the combination of linear constraints and the use of an adaptive penalty. We plan to apply the proposed algorithms to other matrix completion problems, for example, recommender systems.

Acknowledgments

This work is supported by the National Basic Research Program of China (973 Program) (Grant No: 2012CB316400), National Natural Science Foundation of China (Grant No: 61125203, 61125106, 61233011, 91120302, 61072093), US National Science Foundation (NSF) (IIS-0953662, CCF-1025177).

References

[1] E.J. Candès and B. Recht, "Exact Matrix Completion via Convex Optimization," *Foundations on Computational Math.*, vol. 9, pp. 717-772, 2009.
[2] E.J. Candès and T. Tao, "The Power of Convex Relaxation: Near-Optimal Matrix Completion," *IEEE Trans. Information Theory*, vol. 56, no. 5, pp. 2053-2080, May 2009.
[3] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor Completion for Estimating Missing Values in Visual Data," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208-220, Jan. 2013.
[4] A. Eriksson and A. van den Hengel, "Efficient Computation of Robust Low-Rank Matrix Approximations in the Presence of Missing Data Using the l1 Norm," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.
[5] T. Okatani, T. Yoshida, and K. Deguchi, "Efficient Algorithm for Low-Rank Matrix Factorization with Missing Components and Performance Comparison of Latest Algorithms," *Proc. IEEE Int'l Conf. Computer Vision*, 2011.
[6] H.-Y. Shum, K. Ikeuchi, and R. Reddy, "Principal Component Analysis with Missing Data and Its Application to Polyhedral Object Modeling," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 9, pp. 854-867, Sept. 1995.
[7] S.J. Ting Kei Tong, P. Tseng, and J. Ye, "Trace Norm Regularization: Reformulation, Algorithms, and Multi-Task Learning," *SIAM J. Optimization*, vol. 20, no. 6, pp. 3465-3489, 2010.
[8] N. Komodakis and G. Tziritas, "Image Completion Using Global Optimization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006.
[9] C. Rasmussen and T. Korah, "Spatiotemporal Inpainting for Recovering Texture Maps of Partially Occluded Building Facades," *Proc. IEEE Int'l Conf. Image Processing*, 2005.
[10] H. Ji, C. Liu, Z. Shen, and Y. Xu, "Robust Video Denoising Using Low Rank Matrix Completion," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.
[11] Y. Koren, "Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model," *Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, 2008.
[12] H. Steck, "Training and Testing of Recommender Systems on Data Missing Not at Random," *Proc. 16th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, 2010.
[13] B. Recht, M. Fazel, and P.A. Parrilo, "Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization," *SIAM Rev.*, vol. 52, no. 3, pp. 471-501, 2010.
[14] S. Ma, D. Goldfarb, and L. Chen, "Fixed Point and Bregman Iterative Methods for Matrix Rank Minimization," *Math. Programming*, vol. 128, nos. 1-2, pp. 321-353, 2011.
[15] B. Recht, "A Simpler Approach to Matrix Completion," *J. Machine Learning Research*, vol. 12, pp. 413-3430, 2011.
[16] E.J. Candès and Y. Plan, "Matrix Completion with Noise," *Proc. IEEE*, vol. 98, no. 6, pp. 925-936, 2009.
[17] J.F. Cai, E.J. Candès, and Z. Shen, "A Singular Value Thresholding Algorithm for Matrix Completion," *SIAM J. Optimization*, vol. 20, pp. 1956-1982, 2010.
[18] K.-C. Toh and S. Yun, "An Accelerated Proximal Gradient Algorithm for Nuclear Norm Regularized Least Squares Problems," *Pacific J. Optimization*, pp. 615-640, 2010.
[19] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma, "Robust Principal Component Analysis: Exact Recovery of Corrupted Low-Rank Matrices via Convex Optimization," *Proc. Advances in Neural Information Processing Systems*, 2009.
[20] D. Zhang, Y. Hu, J. Ye, X. Li, and X. He, "Matrix Completion by Truncated Nuclear Norm Regularization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012.
[21] R.S. Cabral, F. De la Torre, J.P. Costeira, and A. Bernardino, "Matrix Completion for Multi-Label Image Classification," *Proc. Advances in Neural Information Processing Systems*, 2011.
[22] R. Foygel, R.R. Salakhutdinov, O. Shamir, and N. Srebro, "Learning with the Weighted Trace-Norm under Arbitrary Sampling Distributions," *Proc. Advances in Neural Information Processing Systems*, 2011.

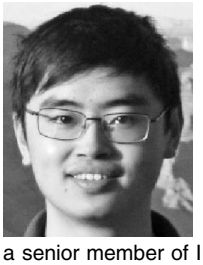
[23] A. Goldberg, X.J. Zhu, B. Recht, J. Xu, and R. Nowak, "Transduction with Matrix Completion: Three Birds with One Stone," *Proc. Advances in Neural Information Processing Systems,* 2010.
[24] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "RASL: Robust Alignment by Sparse and Low-Rank Decomposition for Linearly Correlated Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 34, no. 11, pp. 2233-2246, Nov. 2012.
[25] R. Salakhutdinov and N. Srebro, "Collaborative Filtering in a Non-Uniform World: Learning with the Weighted Trace Norm," *Proc. Advances in Neural Information Processing Systems,* 2010.
[26] M. Fazel, H. Hindi, and S.P. Boyd, "A Rank Minimization Heuristic with Application to Minimum Order System Approximation," *Proc. Am. Control Conf.,* pp. 4734-4739. 2001,
[27] W. Dai, O. Milenkovic, and E. Kerman, "Subspace Evolution and Transfer (Set) for Low-Rank Matrix Completion," *IEEE Trans. Signal Processing,* vol. 59, no. 7, pp. 3120-3132, July 2011.
[28] M. Fazel, "Matrix Rank Minimization with Applications," PhD thesis, Stanford Univ., 2002.
[29] R.H. Tutuncu, K.C. Toh, and M.J. Todd, *Sdpt3—a Matlab Software Package for Semidefinite Quadratic Linear Programming, Version 3.0,* 2001.
[30] J.F. Sturm, *Using Sedumi 1.02, a Matlab Toolbox for Optimization over Symmetric Cones,* 1998.
[31] S. Ji and J. Ye, "An Accelerated Gradient Method for Trace Norm Minimization," *Proc. 26th Ann. Int'l Conf. Machine Learning,* 2009.
[32] N. Srebro, J.D.M. Rennie, and T.S. Jaakkola, "Maximum-Margin Matrix Factorization," *Proc. Advances in Neural Information Processing Systems,* 2005.
[33] J.D.M. Rennie and N. Srebro, "Fast Maximum Margin Matrix Factorization for Collaborative Prediction," *Proc. 22nd Int'l Conf. Machine Learning,* 2005.
[34] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable Collaborative Filtering Approaches for Large Recommender Systems," *J. Machine Learning Research,* 2009.
[35] S. Burer and R.D.C. Monteiro, "Local Minima and Convergence in Low-Rank Semidefinite Programming," *Math. Programming,* vol. 103, no. 3, pp. 427-444, 2005.
[36] R.H. Keshavan, A. Montanari, and S. Oh, "Matrix Completion from Noisy Entries," *J. Machine Learning Research,* vol. 11, pp. 2057-2078, 2010.
[37] R. Meka, P. Jain, and I.S. Dhillon, "Guaranteed Rank Minimization via Singular Value Projection," *Proc. Advances in Neural Information Processing Systems,* 2010.
[38] J. Yang and X. Yuan, "Linearized Augmented Lagrangian and Alternating Direction Methods for Nuclear Norm Minimization," submitted, 2011.
[39] Z. Lin, R. Liu, and Z. Su, "Linearized Alternating Direction Method with Adaptive Penalty for Low-Rank Representation," *Proc. Advances in Neural Information Processing Systems,* 2011.
[40] M. Li, J.T. Kwok, and B.-L. Lu, "Making Large-Scale Nyström Approximation Possible," *Proc. 27th Ann. Int'l Conf. Machine Learning,* 2010.
[41] S. Boyd, N. Parikh, E. Chu, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Information Systems J.,* vol. 3, no. 1, pp. 1-122, 2010.
[42] Y. Nesterov, "Gradient Methods for Minimizing Composite Objective Function," technical report, 2007.
[43] A. Beck and M. Teboulle, "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems," *SIAM J. Imaging Sciences,* vol. 2, no. 1, pp. 183-202, 2009.
[44] Y. Nesterov, "A Method of Solving a Convex Programming Problem with Convergence Rate o(1/sqr(k))," *Soviet Math. Doklady,* vol. 27, pp. 372-376, 1983.
[45] B. He, M. Tao, and X. Yuan, "Alternating Direction Method with Gaussian Back Substitution for Separable Convex Programming," *SIAM J. Optimization,* vol. 22, no. 2, pp. 313-340, 2012.
[46] M. Tao and X. Yuan, "Recovering Low-Rank and Sparse Components of Matrices from Incomplete and Noisy Observations," *SIAM J. Optimizaiton,* vol. 21, no. 1, pp. 57-81, 2011.
[47] B.S. He, H. Yang, and S.L. Wang, "Alternating Direction Method with Self-Adaptive Penalty Parameters for Monotone Variational Inequalities," *J. Optimization Theory and Applications,* vol. 106, no. 2, pp. 337-356, 2000.
[48] R.H. Keshavan, A. Montanari, and S. Oh, "Matrix Completion from a Few Entries," *IEEE Trans. Information Theory,* vol. 56, pp. 2980-2998, 2010.

Yao Hu received the BS degree in math and applied mathematics from Zhejiang University, China, in 2010. He is currently working toward the PhD degree in computer science at Zhejiang University. His research interests include machine learning, computer vision, and data mining.

Debing Zhang received the BS degree in math and applied mathematics from Zhejiang University, China, in 2010. He is currently working toward the PhD degree in computer science at Zhejiang University. His research interests include machine learning, computer vision, and data mining.

Jieping Ye received the PhD degree in computer science and engineering from the University of Minnesota, Twin Cities, in 2005. He is an associate professor of the computer science and engineering program at Arizona State University (ASU). He is also affiliated with the Center for Evolutionary Medicine and Informatics of the Biodesign Institute at ASU. His research interests include machine learning, data mining, and biomedical informatics. He won the outstanding student paper award at ICML in 2004, the SCI Young Investigator of the Year Award at ASU in 2007, the SCI Researcher of the Year Award at ASU in 2009, the NSF CAREER Award in 2010, and the KDD best research paper award nomination in 2010, 2011, and 2012. He is a senior member of the IEEE.

Xuelong Li is a full professor with the Center for OPTical IMagery Analysis and Learning (OPTIMAL), State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, Shaanxi, P.R. China. He is a fellow of the IEEE.



Xiaofei He received the BS degree in computer science from Zhejiang University, China, in 2000 and the PhD degree in computer science from the University of Chicago in 2005. He is a professor in the State Key Lab of CAD&CG at Zhejiang University, China. Prior to joining Zhejiang University, he was a research scientist at Yahoo! Research Labs, Burbank, California. His research interests include machine learning, information retrieval, and computer vision. He is a senior member of IEEE.