Reproduction of "Fast and Accurate Matrix Completion via Truncated Nuclear Norm Regularization"

Zhao-Yang Liu
Sun Yat-sen University
liuzhy86@mail2.sysu.edu.cn

Xin-Hua Zheng
Sun Yat-sen University
zhengxh56@mail2.sysu.edu.cn

Abstract

Recovering a large matrix from a small subset of its entries is a challenging problem arising in many real applications, such as computer vision. This can be considered as a low rank matrix approximation problem. If the nuclear norm is used as a convex relaxation of the rank operator, all singular values are minimized simultaneously by minimizing the nuclear norm, so the rank is not well approximated in practice. Hu proposed 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. Furthermore, they developed a novel matrix completion algorithm and further developed three efficient iterative procedures TNNR-ADMM, TNNR-APGL and TNNR-ADMMAP to solve the optimization problem. In this report, we reproduce the algorithm of Hu et al. and related work, and perform experiments similar to their work. Our experiments show encouraging results of the algorithm proposed by Hu et al. in comparison to the related work on both synthetic and real visual datasets.

1 Introduction

For the problem of estimating missing values in a matrix, or matrix completion, a commonly adoped assumption is the underlying matrix has a low rank or approximately low rank structure. The visual data, such as images, is probably of low rank structure, as shown in Fig. 1. Specifically, given the incomplete data matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ of low rank, the matrix completion problem can be formulated as follows:

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

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

Unfortunately, the rank minimization problem is NP-hard in general and is also NP-hard to approximation. A widely used approach is to apply the nuclear norm (i.e., the sum of singular values) as a convex surrogate for the rank function of a non-convex matrix. It has been shown that under some general constraints, nuclear norm minimization can recover incomplete matrices from enough observed entries.

In this report, we reproduce a matrix completion algorithm called truncated nuclear norm regularization (TNNR) and three optimization schemes proposed by Hu et al[2]: the alternating direction method of multipliers (**A**DMM), the accelerated proximal gradient line search method (**A**PGL) and the Alternating Direction Method of Multipliers with Adaptive Penalty (**A**DMMAP). Unlike other

nuclear norm methods, TNNR only minimize the smallest $\min(m,n)-r$ singular values since the rank of a matrix only corresponds to the r non-zero singular values.

In Section 2, we briefly introduce three kinds of work related to matrix completion. In Section 3, we introduce the TNNR algorithm and three optimization schemes. In Section 4, we conduct experiments similar to the original paper and give the experimental results. In Section 5, we summarize what we have accomplished.

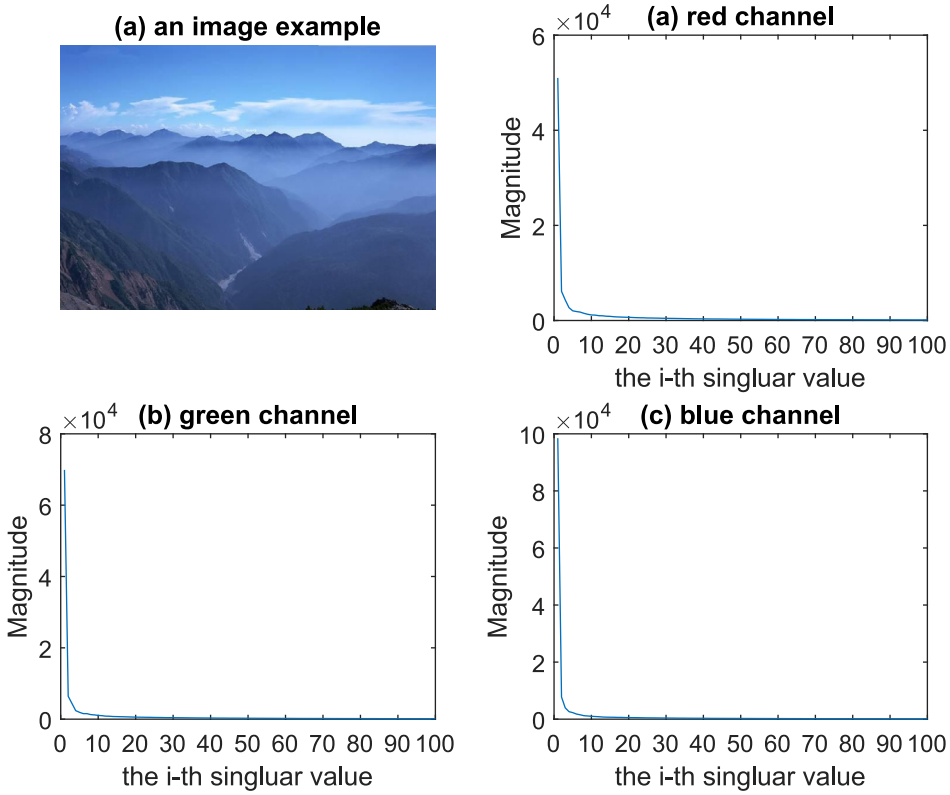


Figure 1: A 400×500 image example

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

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

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

Definition 1.1 *Consider the SVD of a matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$:*

$$\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^T, \ \Sigma = diag(\{\sigma_i\}_{1 \leq \min(m,n)}). \tag{2}$$

Define the singular value shrinkage operator D_τ as follows:

$$D_\tau(\mathbf{X}) = \mathbf{U}D_\tau(\Sigma)\mathbf{V}^T, \ D_\tau(\Sigma) = diag(\max\{\sigma_i - \tau, 0\}). \tag{3}$$

We have the following useful theorem:

Theorem 1.1 *For each $\tau \geq 0$ and $\mathbf{Y} \in \mathbb{R}^{m\times n}$, we have*

$$D_\tau(\mathbf{X}) = \arg\min_{\mathbf{X}} \frac{1}{2}\|\mathbf{X}-\mathbf{Y}\|_F^2 + \tau\|\mathbf{X}\|_*. \tag{4}$$

2 Related work

In this section, we introduce three methods for matrix completion: Optspace[4], singular value thresholding (SVT)[1] and singular value projection (SVP)[3]. These methods are also used to compare with the TNNR algorithm in subsequent experiments. In addition, ADMM and APDL is is briefly explained for better algorithm derivation in Section 3.

By approximating the rank function using the nuclear norm to solve the rank minimization problem (1), the matrix completion problem can be formulated as follows:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \|\mathbf{X}\|_* \\ \text{s.t.} \quad & \mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}), \end{aligned} \tag{5}$$

For SVT, Cai et al. propose the SVT algorithm :

$$\begin{aligned} \min_{\mathbf{X}} \quad & \|\mathbf{X}\|_* + \alpha\|\mathbf{X}\|_F^2 \\ \text{s.t.} \quad & \mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}), \end{aligned} \tag{6}$$

Construct the Lagrangian function,

$$\mathcal{L}(\mathbf{X},\mathbf{Y}) = \|\mathbf{X}\|_* + \alpha\|\mathbf{X}\|_F^2 + \langle \mathbf{Y}, P_{\mathbf{\Omega}}(\mathbf{M}-\mathbf{X})\rangle, \tag{7}$$

where Lagrangian multipliers $\mathbf{Y} \in \mathbb{R}^{m\times n}$. SVT solves problem (5) using alternating iterative methods,

$$\begin{aligned} \mathbf{X}_k &= D_\tau(\mathbf{Y}_{k-1}) \\ \mathbf{Y}_k &= \mathbf{Y}_{k+1} + \sigma_k \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}-\mathbf{X}_k). \end{aligned} \tag{8}$$

For SVP, rewrite (5) as:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \psi(\mathbf{X}) = \frac{1}{2}\|\mathcal{A}(\mathbf{X}) - b\|_2^2 \\ \text{s.t.} \quad & \mathbf{X} \in \mathcal{C}(k) = \{X\colon \text{rank}(\mathbf{X}) \leq k\}, \end{aligned} \tag{9}$$

where $\mathcal{A}$ is affine transformations that satisfy a restricted isometry property. Construct the Lagrangian function,

$$\mathcal{L}(\mathbf{X},\mathbf{Y}) = \min\|\mathcal{A}(\mathbf{X}) - \mathbf{b}\|_F^2. \tag{10}$$

The iterative equation is

$$\begin{aligned} \mathbf{Y}_{k+1} &= \mathbf{X}_k - \gamma_k \mathcal{A}^*(\mathcal{A}(\mathbf{X}_k) - \mathbf{y}) \\ \mathbf{X}_{k+1} &= \text{Trancated SVD}_r(\mathbf{Y}_{k+1}). \end{aligned} \tag{11}$$

For OptSpace, the Lagrangian function of problem (5) is

$$L(\mathbf{X},\mathbf{Y}) = \min_{\mathbf{S}\in\mathbb{R}^{r\times r}} L(\mathbf{X},\mathbf{Y},\mathbf{S}) = \frac{1}{2}\|\mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}-\mathbf{X}\mathbf{S}\mathbf{Y}^T)\|_F^2 + \frac{\lambda}{2}\|\mathcal{P}_{\mathbf{\Omega}^c}(\mathbf{M}-\mathbf{X}\mathbf{S}\mathbf{Y}^T)\|_F^2 \tag{12}$$

ADMM algorithm is an important method for solving separable convex optimization problems, with fast processing speed and good convergence performance.The classic ADMM algorithm is suitable for solving the following 2-block (or N-block) convex optimization problems.

$$\begin{aligned} \min_{x,w} \quad & f(\mathbf{x}) + g(\mathbf{w}) \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{w} = \mathbf{b}, \end{aligned} \tag{13}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{w} \in \mathbb{R}^m$. And $\mathbf{A} \in \mathbb{R}^{p\times n}$, $\mathbf{B} \in \mathbb{R}^{p\times m}$, $\mathbf{b} \in \mathbb{R}^p$ is convex set.f, g is convex function. Using an augmented Lagrangian with a square regularization term with coefficient $\frac{\beta}{2}$:

$$\mathcal{L}(\mathbf{x},\mathbf{z},\mathbf{w}) = f(\mathbf{x}) + g(\mathbf{w}) + \mathbf{z}^T(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{w}-\mathbf{b}) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{w}-\mathbf{b}\|_2^2. \tag{14}$$

By dual ascent,

$$\begin{aligned}(\mathbf{x}_{k+1}, \mathbf{w}_{k+1}) &:= \min_{\mathbf{x},\mathbf{w}} \mathcal{L}(\mathbf{x}, \mathbf{z}_k, \mathbf{w}) \\ z_{k+1} &:= \mathbf{z}_k + \beta(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{w}_{k+1} - \mathbf{b}).\end{aligned} \tag{15}$$

Change $(\mathbf{x}, \mathbf{w})$ joint optimization to separate alternate iterations and let $\mathbf{Y}_k = \frac{z_k}{\beta}$,

$$\begin{aligned}\mathbf{x}_{k+1} &= \arg\min_{\mathbf{x}} \mathcal{L}_\beta(\mathbf{x}, \mathbf{y}_k, \mathbf{w}_k) \\ \mathbf{w}_{k+1} &= \arg\min_{\mathbf{w}} \mathcal{L}_\beta(\mathbf{x}_{k+1}, \mathbf{y}_k, \mathbf{w}) \\ \mathbf{y}_{k+1} &= \mathbf{y}_k + (\mathbf{A}\mathbf{x}_{k+1} + \mathbf{B}\mathbf{y}_{k+1} - \mathbf{b}).\end{aligned} \tag{16}$$

APGL, also called ast iterative shrinkage-thresholding algorithm (FISTA), solves problems like

$$\min_{\mathbf{X}} g(\mathbf{X}) + f(\mathbf{X}), \tag{17}$$

where g is closed, convex, possibly, nondifferentiable and f is a convex and differentiable function.

$$Q(\mathbf{X}, \mathbf{Y}) = f(\mathbf{Y}) + \langle \mathbf{X} - \mathbf{Y}, \nabla f(\mathbf{Y}) \rangle + \frac{1}{2t}\|\mathbf{X} - \mathbf{Y}\|_F^2 + g(\mathbf{X}). \tag{18}$$

Then APGL method solves optimization problem (17) by iteratively updating $\mathbf{X}$, $\mathbf{Y}$ and t. In the $k\text{-}th$ iteration, we update $\mathbf{X}_{k+1}$ as the unique minimizer of $Q(\mathbf{X}, \mathbf{Y}_k)$:

$$\mathbf{X}_{k+1} = \arg\min_{\mathbf{X}} Q(\mathbf{X}, \mathbf{Y}_k) = \arg\min_{\mathbf{X}} g(\mathbf{X}) + \frac{1}{2t}\|\mathbf{X} - (\mathbf{Y}_k - t_k \nabla f(\mathbf{Y}_k))\|_F^2. \tag{19}$$

3 Truncated nuclear norm regularization

3.1 The approach proposed by Hu et al.

By using Truncated nuclear norm, this approach achieves a better approximation of the rank function than the nuclear norm.

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

Thus, the objective function of this approach can be formulated as follows

$$\begin{aligned}\min_{\mathbf{X}} \quad & \|\mathbf{X}\|_r \\ \text{s.t.} \quad & \mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}).\end{aligned} \tag{20}$$

Since $\|\mathbf{X}\|_r$ is non-convex, they propose the follow theorem:

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

$$\text{Tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) \leq \sum_{i=1}^{r} \sigma_i(\mathbf{X}). \tag{21}$$

The proof is given in the Appendix.

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

$$\mathbf{A} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m)^T, \ \mathbf{B} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)^T. \tag{22}$$

This is because

$$\begin{aligned}\text{Tr}((\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m)^T X(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)) &= \text{Tr}((\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m)^T U\Sigma\mathbf{V}^T(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)) \\ &= \text{Tr}(((\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m)^T\mathbf{U})\Sigma(\mathbf{V}^T(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m))) \\ &= \text{Tr}\left(\begin{bmatrix}\mathbf{I}_r & 0 \\ 0 & 0\end{bmatrix}\Sigma\begin{bmatrix}\mathbf{I}_r & 0 \\ 0 & 0\end{bmatrix}\right) \\ &= \text{Tr}(diag(\sigma_1(\mathbf{X}), \ldots, \sigma_r(\mathbf{X}), 0, \ldots, 0)) \\ &= \sum_{i=1}^{r} \sigma_i(\mathbf{X})\end{aligned} \tag{23}$$

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

Input: original incomplete data matrix $\mathbf{M}_{\mathbf{\Omega}}$, where $\mathbf{\Omega}$ is the indices of the observed entries, tolerance ϵ_0

Initialization: $\mathbf{X}_1 = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M})$.

repeat

step 1. Given $\mathbf{X}_l$, compute $\mathbf{U}_l, \Sigma_l, \mathbf{V}_l = svd(\mathbf{X}_l)$,
where $\mathbf{U}_l = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m) \in \mathbb{R}^{m\times m}$, $\mathbf{V}_l = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) \in \mathbb{R}^{n\times n}$.
Compute $\mathbf{A}_l$ and $\mathbf{B}_l$, as $\mathbf{A}_l = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m)^T$, $\mathbf{B}_l = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)^T$

step 2. Solve $\mathbf{X}_{l+1} = \arg\min_{\mathbf{X}} \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) \quad \text{s.t.} \quad \mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M})$

until $\|\mathbf{X}_{l+1} - \mathbf{X}_l\|_F \leq \epsilon_0$.

return the recovered matrix.

Combining (21) and (23), we get

$$\max_{\mathbf{A}\mathbf{A}^T=\mathbf{I},\mathbf{B}\mathbf{B}^T=\mathbf{I}} \text{Tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) \leq \sum_{i=1}^{r} \sigma_i(\mathbf{X}). \tag{24}$$

Then we have

$$\|\mathbf{X}\|_* - \max_{\mathbf{A}\mathbf{A}^T=\mathbf{I},\mathbf{B}\mathbf{B}^T=\mathbf{I}} \text{Tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) = \sum_{i=1}^{\min(m,n)} \sigma_i(\mathbf{X}) - \sum_{i=1}^{r} \sigma_i(\mathbf{X}) = \|\mathbf{X}\|_r. \tag{25}$$

Thus, the optimization problem (20) can be rewritten as follows:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \|\mathbf{X}\|_* - \max_{\mathbf{A}\mathbf{A}^T=\mathbf{I},\mathbf{B}\mathbf{B}^T=\mathbf{I}} \text{Tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) \\ \text{s.t.} \quad & \mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}), \end{aligned} \tag{26}$$

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

Based on (26), they design a simple but efficient iterative scheme, which is summarized in Algorithm 1. In the lth iteration, we first fix $\mathbf{X}_l$ and compute $\mathbf{A}_l$ and $\mathbf{B}_l$. And then we fix $\mathbf{A}_l$ and $\mathbf{B}_l$ ro update $\mathbf{X}_{l+1}$ by solving the following problem:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) \\ \text{s.t.} \quad & \mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}). \end{aligned} \tag{27}$$

3.2 The optimization using TNNR-ADMM

ADMM is a method for solving a decomposable convex optimization problem. It can equivalently decompose the objective function of the original problem into several solvable sub-problems, then solve each sub-problem in parallel, and finally coordinate the solutions of the sub-problems to obtain the global solution of the original problem.

By using ADMM to solve (27), an optimization algorithm TNNR-ADMM is proposed. First, rewrite (27) as follows:

$$\begin{aligned} \min_{\mathbf{X},\mathbf{W}} \quad & \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}_l\mathbf{X}\mathbf{B}_l^T) \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{W},\ \mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}). \end{aligned} \tag{28}$$

By (13), we have $f(\mathbf{X}) = \|\mathbf{X}\|_*$ and $g(\mathbf{W}) = \text{Tr}(\mathbf{A}_l\mathbf{X}\mathbf{B}_l^T)$.

The augmented lagrange function of is

$$L(\mathbf{X}, \mathbf{Y}, \mathbf{W}, \beta) = \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}_l\mathbf{X}\mathbf{B}_l^T) + \frac{\beta}{2}\|\mathbf{X} - \mathbf{W}\|_F^2 + \text{Tr}(\mathbf{Y}^T(\mathbf{X} - \mathbf{W})), \tag{29}$$

where $\beta > 0$ is the penalty parameter. Given the initial setting $\mathbf{X}_1 = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M})$, $\mathbf{W}_1 = \mathbf{X}_1$, and $\mathbf{Y}_1 = \mathbf{X}_1$, the optimization problem (28) can be solved via the following three steps.

Algorithm 2 The Optimization using TNNR-ADMM

Input: $\mathbf{A}_l$, $\mathbf{B}_l$, $\mathbf{M}_{\mathbf{\Omega}}$, and tolerance ϵ are given.
Initialization: $\mathbf{X}_1 = \mathbf{M}_{\mathbf{\Omega}}$, $\mathbf{W}_1 = X_1$, $\mathbf{Y}_1 = \mathbf{X}_1$, and $\beta = 1$.
repeat
step 1. $\mathbf{X}_{k+1} = D_{\frac{1}{\rho}}(\mathbf{W}_k - \frac{1}{\rho}\mathbf{Y}_k)$.
step 2. $\mathbf{W}_{k+1} = \mathbf{X}_{k+1} + \frac{1}{\beta}(\mathbf{A}_l^T\mathbf{B}_l + \mathbf{Y}_k)$.
Fix values at observed entries, $\mathbf{W}_{k+1} = \mathbf{X}_{k+1} + \frac{1}{\beta}(\mathbf{A}_l^T\mathbf{B}_l + \mathbf{Y}_k)$.
step 3. $\mathbf{Y}_{k+1} = \mathbf{Y}_k + \beta(\mathbf{X}_{k+1} - \mathbf{W}_{k+1})$.
until $\|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F \leq \epsilon$.

Computing $\mathbf{X}_{k+1}$. Fix $\mathbf{W}_k$ and $\mathbf{Y}_k$, and minimize $L(\mathbf{X}, \mathbf{Y}_k, \mathbf{W}_k, \beta)$ for $\mathbf{X}_{k+1}$ as follows:

$$\begin{aligned}\mathbf{X}_{k+1} &= \arg\min_{\mathbf{X}} L(\mathbf{X}, \mathbf{Y}_k, \mathbf{W}_k, \beta) \\ &= \arg\min_{\mathbf{X}} \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}_l\mathbf{W}_k\mathbf{B}_l^T) + \frac{\beta}{2}\|\mathbf{X} - \mathbf{W}_k\|_F^2 + \text{Tr}(\mathbf{Y}_k^T(\mathbf{X} - \mathbf{W}_k))\end{aligned} \tag{30}$$

Ignoring constant terms, this can be rewritten as

$$\mathbf{X}_{k+1} = \arg\min_{\mathbf{X}} \|\mathbf{X}\|_* + \frac{\beta}{2}\|\mathbf{X} - \left(\mathbf{W}_k - \frac{1}{\beta}\mathbf{Y}_k\right)\|_F^2 \tag{31}$$

By Theorem 1.1 ,we can solve the above problem as

$$\mathbf{X}_{k+1} = D_{\frac{1}{\rho}}(\mathbf{W}_k - \frac{1}{\rho}\mathbf{Y}_k). \tag{32}$$

Computing $\mathbf{W}_{k+1}$. Fix $\mathbf{X}_{k+1}$ and $\mathbf{Y}_k$ to calculate $\mathbf{W}_{k+1}$ as follows:

$$\begin{aligned}\mathbf{W}_{k+1} &= \arg\min_{\mathcal{P}_{\mathbf{\Omega}}(\mathbf{W}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M})} L(\mathbf{X}_{k+1}, \mathbf{Y}_k, \mathbf{W}, \beta) \\ &= \arg\min_{\mathcal{P}_{\mathbf{\Omega}}(\mathbf{W}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M})} \|\mathbf{X}_{k+1}\|_* - \text{Tr}(\mathbf{A}_l\mathbf{W}\mathbf{B}_l^T) + \frac{\beta}{2}\|\mathbf{X}_{k+1} - \mathbf{W}\|_F^2 + \text{Tr}(\mathbf{Y}_k^T(\mathbf{X}_{k+1} - \mathbf{W})).\end{aligned} \tag{33}$$

then we get

$$\mathbf{W}_{k+1} = \mathbf{X}_{k+1} + \frac{1}{\beta}(\mathbf{A}_l^T\mathbf{B}_l + \mathbf{Y}_k). \tag{34}$$

fix the values at the observed entries

$$\mathbf{W}_{k+1} = \mathcal{P}_{\Omega^c}(\mathbf{X}_{k+1}) + \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}). \tag{35}$$

Computing $\mathbf{Y}_{k+1}$. Fix $\mathbf{X}_{k+1}$ and $\mathbf{W}_{k+1}$ to calculate $\mathbf{Y}_{k+1}$ as follows:

$$\mathbf{Y}_{k+1} = \mathbf{Y}_k + \beta(\mathbf{X}_{k+1} - \mathbf{W}_{k+1}). \tag{36}$$

The whole procedure of TNNR-ADMM is summarized in Algorithm 2.

3.3 The optimization using TNNR-APGL

Considering the noisy data in the real applications, it is beneficial to relax the constrained problem (27) into

$$\min \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}_l\mathbf{X}\mathbf{B}_l^T) + \frac{\lambda}{2}\|\mathbf{X}_{\mathbf{\Omega}} - \mathbf{M}_{\mathbf{\Omega}}\|_F^2, \tag{37}$$

for some $\lambda > 0$.

According to (17), in problem (37), we choose

$$g(\mathbf{X}) = \|\mathbf{X}\|_*,$$

Algorithm 3 The Optimization using TNNR-APGL

Input: $\mathbf{A}_l$, $\mathbf{B}_l$, $\mathbf{M}_{\mathbf{\Omega}}$, and tolerance ϵ are given.
Initialization: $t_1 = 1$, $\mathbf{X}_1 = \mathbf{M}_{\mathbf{\Omega}}$, $\mathbf{Y}_1 = \mathbf{X}_1$.
repeat
step 1. $\mathbf{X}_{k+1} = \mathcal{D}_{t_k}(\mathbf{Y}_k - t_k(\mathbf{A}_l^T\mathbf{B}_l - \lambda(\mathcal{P}_{\mathbf{\Omega}}(\mathbf{Y}_k) - \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}))))$.
step 2. $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.
step 3. $\mathbf{Y}_{k+1} = \mathbf{X}_{k+1} + \frac{t_k-1}{t_{k+1}}(\mathbf{X}_{k+1} - \mathbf{X}_k)$.
until $\|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F \leq \epsilon$.

and

$$f(\mathbf{X}) = -\text{Tr}(\mathbf{A}_l\mathbf{X}\mathbf{B}_l^T) + \frac{\lambda}{2}\|\mathbf{X}_{\mathbf{\Omega}} - \mathbf{M}_{\mathbf{\Omega}}\|_F^2.$$

By Theorem 1.1, we get

$$\begin{aligned}\mathbf{X}_{k+1} &= \arg\min_{\mathbf{X}} \|\mathbf{X}\|_* + \frac{1}{2t}\|\mathbf{X} - (\mathbf{Y}_k - t_k\nabla f(\mathbf{Y}_k))\|_F^2 \\ &= \mathcal{D}_{t_k}(\mathbf{Y}_k - t_k\nabla f(\mathbf{Y}_k)) \\ &= \mathcal{D}_{t_k}(\mathbf{Y}_k - t_k(\mathbf{A}_l^T\mathbf{B}_l - \lambda(\mathcal{P}_{\mathbf{\Omega}}(\mathbf{Y}_k) - \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M})))).\end{aligned} \tag{38}$$

Finally, t_{k+1} and $\mathbf{Y}_{k+1}$ are updated as:

$$\begin{aligned}t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2}, \\ \mathbf{Y}_{k+1} &= \mathbf{X}_{k+1} + \frac{t_k-1}{t_{k+1}}(\mathbf{X}_{k+1} - \mathbf{X}_k).\end{aligned} \tag{39}$$

The procedures of solving (26) are summarized in Algorithm 3.

3.4 The optimization using TNNR-ADMMAP

In Algorithm 2, the two constraints are considered separately, but the inconsistency of the two constraints may slow down the convergence. By combining the two constraints into a new constraint in a special way and adding an adaptive penalty parameter to speed up the convergence, a new approach by using the ADMM with adaptive penalty (TNNR-ADMMAP) is proposed.

3.4.1 The reformulation of problem

To deal with the two constraints simultaneously, in this section, we reformulate the problem 28 as follows:

$$\begin{aligned}\min_{\mathbf{X}} \quad & \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}_l\mathbf{X}\mathbf{B}_l^T) \\ \text{s.t.} \quad & \mathcal{A}(\mathbf{X}) + \mathcal{B}(\mathbf{M}) = \mathcal{C},\end{aligned} \tag{40}$$

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

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

Then discuss the properties of adjoint operators $\mathcal{A}$ and $\mathcal{B}$.

Suppose

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

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

$$\langle\mathcal{A}(\mathbf{X}), \mathbf{Y}\rangle = \langle\mathbf{X}, \mathcal{A}^*(\mathbf{Y})\rangle,\ \langle\mathcal{B}(\mathbf{X}), \mathbf{Y}\rangle = \langle\mathbf{X}, \mathcal{B}^*(\mathbf{Y}) \tag{41}$$

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

$$\langle \mathcal{A}(\mathbf{X}), \mathbf{Y} \rangle = \text{Tr} \begin{pmatrix} \mathbf{X} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{21} & \mathbf{Y}_{22} \end{pmatrix}^T = \text{Tr}(\mathbf{X}\mathbf{Y}_{11}^T) = \langle \mathbf{X}, \mathbf{Y}_{11} \rangle$$

and

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

By the definition of the adjoint operator (41), the adjoint operators $\mathcal{A}$ and $\mathcal{B}$ can be computed as

$$\begin{aligned} &\mathcal{A}^*(\mathbf{Y}) = \mathbf{Y}_{11}, \\ &\mathcal{B}^*(\mathbf{Y}) = -\mathbf{Y}_{11} + \mathcal{P}_{\mathbf{\Omega}}(\mathbf{Y}_{22}). \end{aligned} \tag{42}$$

Let us rewrite the augmented Lagrange function for the optimization problem (40) as

$$\mathcal{L}_{AP}(\mathbf{X}, \mathbf{W}, \mathbf{Y}, \beta) = -\text{Tr}(\mathbf{A}_l \mathbf{W} \mathbf{B}_l^T) + \langle \mathbf{Y}, \mathcal{A}(\mathbf{X}) + \mathcal{B}(\mathbf{W}) - \mathcal{C} \rangle + \|\mathbf{X}\|_* + \frac{\beta}{2} \|\mathcal{A}(\mathbf{X}) + \mathcal{B}(\mathbf{W}) - \mathcal{C}\|_F^2, \tag{43}$$

where $\mathbf{Y}$ is the Lagrange multiplier matrix and $\beta > 0$ is the penalty parameter. The iterative scheme of ADMM for problem (40) is given as follows:

$$\begin{aligned} \mathbf{X}_{k+1} &= \arg\min_{\mathbf{X}} \mathcal{L}_{AP}(\mathbf{X}, \mathbf{W}_k, \mathbf{Y}_k, \beta) \\ &= \arg\min_{\mathbf{X}} \frac{\beta}{2} \|\mathcal{A}(\mathbf{X}) + \mathcal{B}(\mathbf{W}_k) - \mathcal{C} + \frac{1}{\beta} \mathbf{Y}_k\|_F^2 + \|\mathbf{X}\|_*, \end{aligned} \tag{44}$$

$$\begin{aligned} \mathbf{W}_{k+1} &= \arg\min_{W} \mathcal{L}_{AP}(\mathbf{X}_{k+1}, \mathbf{W}, \mathbf{Y}_k, \beta) \\ &= \arg\min_{W} \frac{\beta}{2} \|\mathcal{A}(\mathbf{X}_{k+1}) + \mathcal{B}(\mathbf{W}) - \mathcal{C} + \frac{1}{\beta} \mathbf{Y}_k\|_F^2 - \text{Tr}(\mathbf{A}_l \mathbf{W} \mathbf{B}_l^T), \end{aligned} \tag{45}$$

$$\mathbf{Y}_{k+1} = \mathbf{Y}_k + \beta[\mathcal{A}(\mathbf{X}_{k+1}) + \mathcal{B}(\mathbf{W}) - \mathcal{C}]. \tag{46}$$

3.4.2 The iterative scheme of TNNR-ADMMAP

Based on the iterative scheme (44)-(46), TNNR-ADMMAP algorithm is introduced.

Computing $\mathbf{X}_{k+1}$.

$$\begin{aligned} \mathbf{X}_{k+1} &= \arg\min_{\mathbf{X}} \frac{\beta}{2} \|\mathcal{A}(\mathbf{X}) + \mathcal{B}(\mathbf{W}_k) - \mathcal{C} + \frac{1}{\beta} \mathbf{Y}_k\|_F^2 + \|\mathbf{X}\|_* \\ &= \arg\min_{\mathbf{X}} \frac{\beta}{2} \|\mathbf{P}\|_F^2 + \|\mathbf{X}\|_*, \end{aligned} \tag{47}$$

where

$$\mathbf{P} = \begin{pmatrix} \mathbf{X} - \mathbf{W}_k + \frac{1}{\beta} & \frac{1}{\beta}(\mathbf{Y}_k)_{12} \\ \frac{1}{\beta}(\mathbf{Y}_k)_{21} & \mathcal{P}_{\mathbf{\Omega}} + \frac{1}{\beta}(\mathbf{Y}_k)_{22} \end{pmatrix}.$$

By Theorem (1.1), we get

$$\mathbf{X}_{k+1} = \mathcal{D}\left(\mathbf{W}_k - \frac{1}{\beta}(\mathbf{Y}_k)_{11} \right). \tag{48}$$

Computing $\mathbf{W}_{k+1}$. Setting the first derivative of $\mathcal{L}_{AP}(\mathbf{X}_{k+1}, \mathbf{W}, \mathbf{Y}_k, \beta)$ to zero, we get

$$\beta \mathcal{B}^* \left[\mathcal{B}(\mathbf{W}) + \mathcal{A}(\mathbf{X}_{k+1}) - \mathcal{C} + \frac{1}{\beta} \mathbf{Y}_k \right] - \mathbf{A}_l^T \mathbf{B}_l = 0,$$

which can be rewritten as

$$\mathcal{B}^*\mathcal{B}(\mathbf{W}) = \frac{1}{\beta}\mathbf{A}_l^T\mathbf{B}_l - \mathcal{B}^*\left[\mathcal{A}(\mathbf{X}_{k+1}) - \mathcal{C} + \frac{1}{\beta}\mathbf{Y}_k\right]. \tag{49}$$

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

$$\mathcal{B}^*\mathcal{B}(\mathbf{W}) = \mathcal{B}^*\begin{pmatrix} -\mathbf{W} & 0 \\ 0 & \mathcal{P}_{\mathbf{\Omega}}(\mathbf{W}) \end{pmatrix} = \mathbf{W} + \mathcal{P}_{\mathbf{\Omega}}(\mathbf{W}). \tag{50}$$

Then we apply the orthogonal projection operator $\mathcal{P}_{\mathbf{\Omega}}$ on both sides of (50), and finally we get

$$\mathcal{P}_{\mathbf{\Omega}}(\mathbf{W}) = \frac{1}{2}\mathcal{P}_{\mathbf{\Omega}}(\mathcal{B}^*\mathcal{B}(\mathbf{W})). \tag{51}$$

From (50) and (51), we obtain

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

Then compute $\mathcal{B}^*\mathcal{B}(\mathbf{W})$ directly as follows:

$$\begin{aligned}
\mathcal{B}^*\mathcal{B}(\mathbf{W}) &= \frac{1}{\beta}\mathbf{A}_l^T\mathbf{B}_l - \mathcal{B}^*\left[\mathcal{A}(\mathbf{X}_{k+1}) - \mathcal{C} + \frac{1}{\beta}\mathbf{Y}_k\right] \\
&= \frac{1}{\beta}\mathbf{A}_l^T\mathbf{B}_l - \mathcal{B}^*\left[\begin{pmatrix} \mathbf{X}_{k+1} & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & \mathcal{P}_{\mathbf{\Omega}} \end{pmatrix} + \frac{1}{\beta}\begin{pmatrix} (\mathbf{Y}_k)_{11} & (\mathbf{Y}_k)_{12} \\ (\mathbf{Y}_k)_{21} & (\mathbf{Y}_k)_{22} \end{pmatrix}\right] \\
&= -\mathcal{B}^*\begin{pmatrix} \frac{1}{\beta}(\mathbf{Y}_k)_{11} + \mathbf{X}_{k+1} & \frac{1}{\beta}(\mathbf{Y}_k)_{12} \\ \frac{1}{\beta}(\mathbf{Y}_k)_{21} & \frac{1}{\beta}(\mathbf{Y}_k)_{22} - \mathcal{P}_{\mathbf{\Omega}}(\mathbf{M}) \end{pmatrix} + \frac{1}{\beta}\mathbf{A}_l^T\mathbf{B}_l
\end{aligned}$$

From property (42) of the adjoint operator $\mathcal{B}^*$, we can get

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

Thus,

$$\mathcal{B}^*\mathcal{B}(\mathbf{W}) = -\left[-\left(\frac{1}{\beta}(\mathbf{Y}_k)_{11} + \mathbf{X}_{k+1}\right) + \mathcal{P}_{\mathbf{\Omega}}(\frac{1}{\beta}(\mathbf{Y}_k)_{22} - \mathbf{M})\right] + \frac{1}{\beta}\mathbf{A}_l^T\mathbf{B}_l. \tag{52}$$

Based on (51) and (52), we obtain

$$\mathbf{W}_{k+1} = \frac{1}{2\beta}\mathcal{P}_{\mathbf{\Omega}}[\beta(\mathbf{M}-\mathbf{X}_{k+1}) - (\mathbf{A}_l^T\mathbf{B}_l + (\mathbf{Y}_k)_{11} + (\mathbf{Y}_k)_{22})] + \mathbf{X}_{k+1} + \frac{1}{\beta}(\mathbf{A}_l^T\mathbf{B}_l + (\mathbf{Y}_k)_{11}). \tag{53}$$

Computing $\mathbf{Y}_{k+1}$. The update of Lagrange multipliers is the same as (46).

3.4.3 Adapive penalty

For a fixed penalty parameter, if it is chosen to be too small or too large, the computational cost will increase significantly. At the same time, finding the optimal penalty parameter is difficult. Therefore, dynamic adjustment of penalty parameters may be better in practical applications. The adaptive update strategy adopted is as follows:

$$\beta_{k+1} = \min(\beta_{\max}, \rho\beta_k), \tag{54}$$

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

$$\rho = \begin{cases} \rho_0, & \text{if } \dfrac{\beta_k \max\{\|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F\}}{\|\mathcal{C}\|_F} < \kappa \\ 1, & \text{otherwise,} \end{cases} \tag{55}$$

where $\rho > 1$ is a constant and $\kappa > 0$ is a threshold chosen in advance. When the difference between $(\mathbf{X}_{k+1}, \mathbf{W}_{k+1})$ and $(\mathbf{X}_k, \mathbf{W}_k)$ is small enough, β_{k+1} increases to $\rho_0\beta_k$ and the convergence rate is improved.

The procedures of TNNR-ADMMAP are summarized in Algorithm 4.

Algorithm 4 Inner Optimization by ADMMAP

Input: $\mathbf{A}_l$, $\mathbf{B}_l$, $\mathbf{M}_{\mathbf{\Omega}}$, and tolerance ϵ are given.
Initialization: $t_1 = 1$, $\mathbf{X}_1 = \mathbf{M}_{\mathbf{\Omega}}$, $\mathbf{Y}_1 = \mathbf{X}_1$.
repeat
step 1. $\mathbf{X}_{k+1} = \mathcal{D}\left(\mathbf{W}_k - \frac{1}{\beta}(\mathbf{Y}_k)_{11}\right)$.
step 2. $\mathbf{W}_{k+1} = \frac{1}{2\beta}\mathcal{P}_{\mathbf{\Omega}}[\beta(\mathbf{M} - \mathbf{X}_{k+1}) - (\mathbf{A}_l^T\mathbf{B}_l + (\mathbf{Y}_k)_{11} + (\mathbf{Y}_k)_{22})]$
$+\mathbf{X}_{k+1} + \frac{1}{\beta}(\mathbf{A}_l^T\mathbf{B}_l + (\mathbf{Y}_k)_{11})$.
step 3. $\mathbf{Y}_{k+1} = \mathbf{Y}_k + \beta[\mathcal{A}(\mathbf{X}_{k+1}) + \mathcal{B}(\mathbf{W}) - \mathcal{C}]$.
step 4. Update β_k by (54) and (55)
until $\|\mathbf{X}_{k+1} - \mathbf{X}_k\|_F \leq \epsilon$.

4 Experimental results

In this section, we conduct experiments on synthetic data and real visual data, and compare six matrix completion algorithms: SVT, SVP, OptSapce, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP.

4.1 Synthetic data

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

$$\mathbf{B} = \mathbf{M} + \sigma\mathbf{Z},\ \mathbf{B}_{ij} = \mathbf{M}_{ij} + \sigma\mathbf{Z}_{ij},\ (i,j) \in \mathbf{\Omega},$$

where $\mathbf{Z} \sim \mathcal{N}(0,1)$. Denote the full noiseless matrix as $\mathbf{X}_{full}$ and the solution given by an algorithm as $\mathbf{X}_{sol}$, define a commonly used criterion in matrix completion:

$$\text{total reconstruction error} = \|\mathcal{P}_{\Omega^c}(\mathbf{X}_{sol} - \mathbf{X}_{full})\|_F.$$

Denote different matrix ranks as r_0 and differnet nosie levels as σ. We compare the reconstruction error of the six methods under different setting:

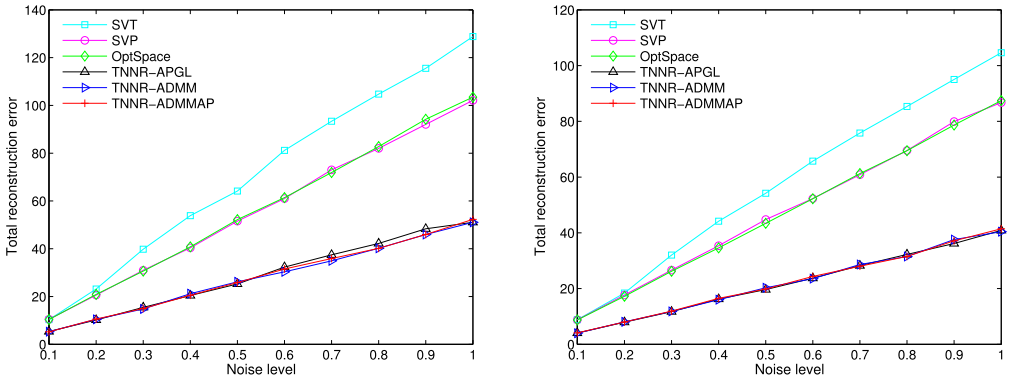
- the matrix size $m = 100$, $n = 200$, $r_0 = 10$. Fig. 3 shows the results under different noise levels and different observed ratios.
- the matrix size $m = 100$, $n = 200$, $\sigma = 0.5$. Fig. 5 shows the results under different ranks and different observed ratios.

As shown in picture 3, under different noise levels and different observed ratios, compared with the original image:

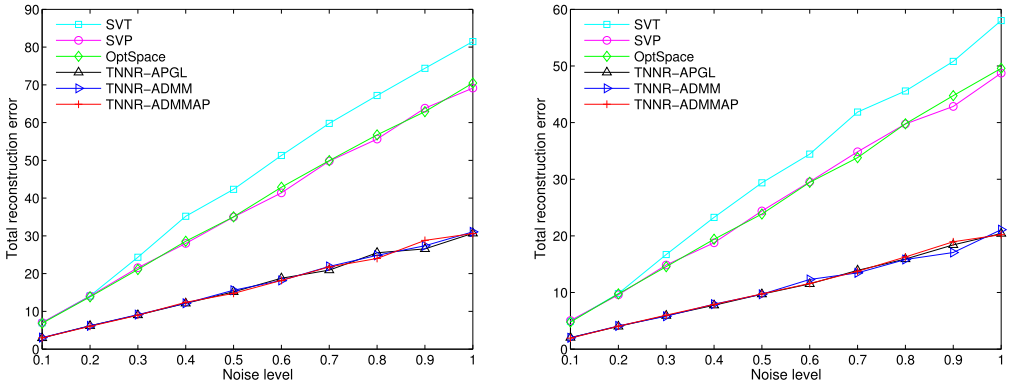
- Our reproduced (abbreviated as our)SVT algorithm has better overall reconstruction error performance.
- Our SVP and OptSpace performs better and has similar performance to TNNR-APGL.
- Our TNNR-ADMM performs similarly to the original image only at low noise levels. However, as the noise level increases, the performance deteriorates faster.
- Our TNNR-APGL has a similar performance to the original image.

As shown in picture 5, under different observed ratios, as the matrix rank increases, the total reconstruction error becomes larger. Compared with the original image:

- Our SVT algorithm has better overall reconstruction error performance at low ranks, and our implementation of the SVT algorithm deteriorates more rapidly as the matrix rank increases.
- Our SVP and OptSpace performs better and has similar performance to TNNR-APGL.



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



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

Figure 2: The reconstruction error versus the noise level using the synthetic dataset. (The original result of paper)

- The total reconstruction error of our TNNR-ADMM and TNNR-ADMMAP is relatively large, and the results that are not close to the original work have been reached.
- Our TNNR-APGL has a similar performance to the original image.

4.2 Real visual data

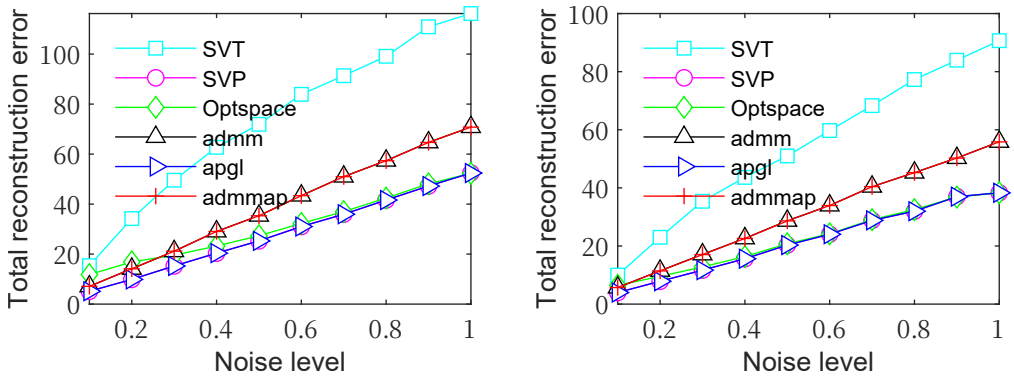
In practice, images may be covered with text etc. For color images, the three channels need to be processed separately and combined to get the final result. We use the peak signal-to-noise ratio (PSNR) to evaluate the performance of different methods, assuming the total number of missing pixels is T, then

$$\text{SE} = error_r^2 + error_g^2 + error_b^2$$
$$\text{MSE} = \frac{\text{SE}}{3T}$$
$$\text{PSNR} = 10\log_{10}(\frac{255^2}{\text{MSE}}).$$

where SE denotes the total squared error, MSE denotes the total mean squared error.

4.2.1 Parameter setting

In our experiments, the parameters of SVP, SVT, and OptSpace are tuned to achieve the best performance. For TNNR-ADMM, TNNR-APGL and TNNR-ADMMAP, We did not get good results using the original parameters. For TNNR-ADMM, we set $\beta = 10^{-3}$. For TNNR-ADMMAP, we set $\beta_{\max} = 10^{10}$, $\kappa = 10^{-3}$, $\rho_0 = 1.1$ in section 4.2.2, and $\rho_0 = 1.01$ in other sections. For



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

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

Figure 3: The reconstruction error versus the noise level using the synthetic dataset. (The results we reproduced)

TNNR-APGL, the parameter λ is empirically set to be 10^{-2} in section 4.2.3 and 10^{-3} in other sections.

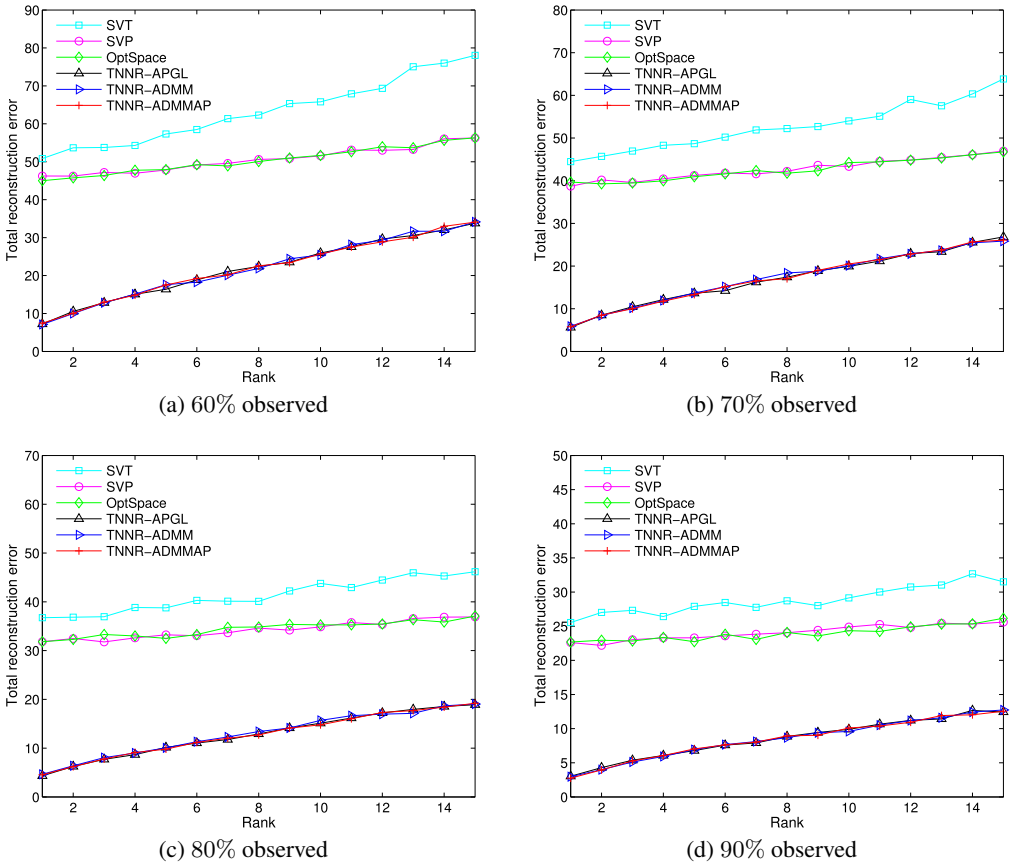
For the stop conditions of TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP, we set the tolerance ϵ_0 in Algorithm 1 to be 10^{-3} and the tolerance ϵ in Algorithms 2-4 to be 10^{-4}. Given an incomplete image, for each one of the six algorithms we try all the possible values of r to choose the best results.

4.2.2 Random mask

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 Fig, 7 and 9. Naturally, the larger the observed ratio is the better recovery of the image that can be achieved. Form Fig. 9, we can see that the TNNR-based algorithms can achieve higher PSNR values compared with the other three methods. Compare with the results in the original paper, our SVT recovers images better. And the results of TNNR-APGL are not as expected in the high observed ratio.

4.2.3 Text mask

Since the text is not random, it may cover important information in the picture, making the text more difficult to remove than pixels. Fig. 11 and Fig. 13 show the experimental results of the six matrix completion algorithms. Specifically, for the example image in Fig. 11, the PSNR values for SVT, SVP, OptSpace, TNNR- ADMM, TNNR-APGL, and TNNR-ADMMAP are 23.47, 23.47, 22.43, 25.26, 25.26, and 25.27, respectively. And for the example image in Fig. 8, the PSNR values for SVT, SVP, OptSpace, TNNR-ADMM, TNNR-APGL, and TNNR-ADMMAP are 21.61, 21.61,



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

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

Figure 4: Recovery of an incomplete image with random mask. (The original result of paper)

Table 1: Running time of text masked image recovery

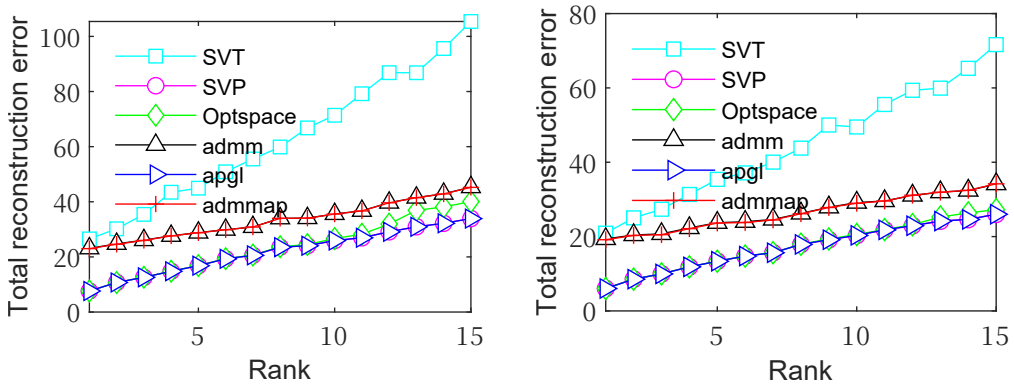
Method	Observed Ratios					
	0.4	0.5	0.6	0.7	0.8	0.9
TNNR-ADMM						
TNNR-APGL						
TNNR-ADMMAP						

21.95, 23.94, 23.16, and 23.95, respectively. From these results we can see that TNNR-ADMM and TNNR-ADMMAP algorithms achieve better performance than the other three matrix completion algorithms. Our TNNR-APGL cannot obtain the same image restoration effect as the original paper.

4.2.4 Block mask

In some situations, Some pixels in the image have been broken. We first mask all these pixels, then we use six matrix completion algorithms to recover these large missing blocks.

4.3 Convergence rate and computational cost



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

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

Figure 5: Recovery of an incomplete image with random mask. (The results we reproduced)

5 Conclusion

References

[1] Jian-Feng Cai, Emmanuel J. Candes, and Zuowei Shen. A Singular Value Thresholding Algorithm for Matrix Completion.

[2] Yao Hu, Debing Zhang, Jieping Ye, Xuelong Li, and Xiaofei He. Fast and Accurate Matrix Completion via Truncated Nuclear Norm Regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(9):2117–2130, 2013.

[3] Prateek Jain, Raghu Meka, and Inderjit Dhillon. Guaranteed Rank Minimization via Singular Value Projection. In *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010.

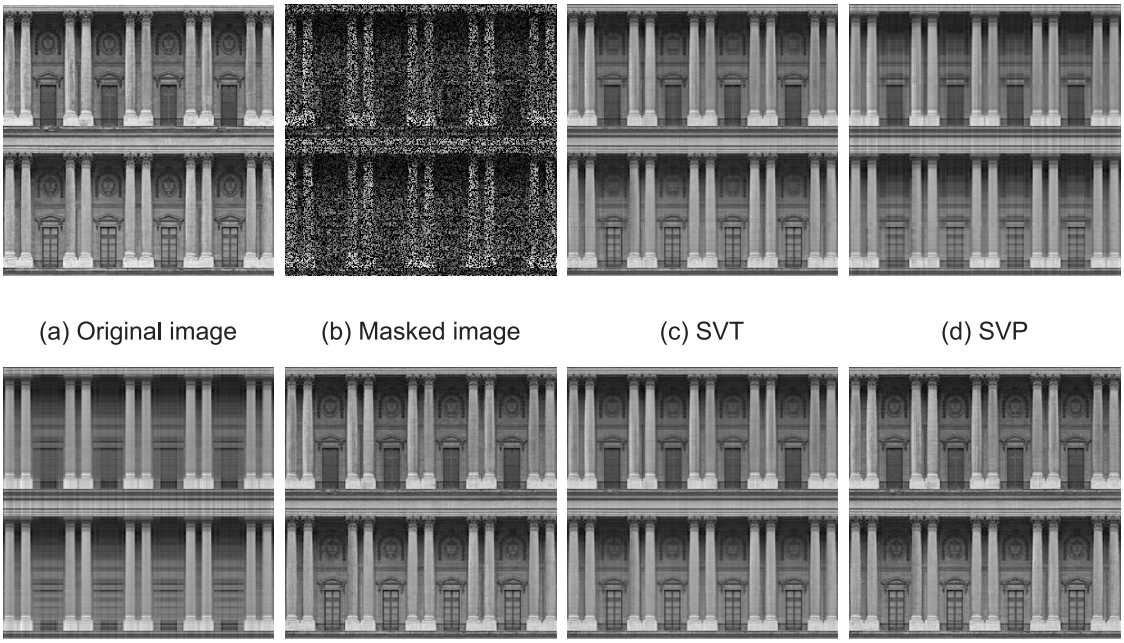
[4] Raghunandan H. Keshavan, Sewoong Oh, and Andrea Montanari. Matrix completion from a few entries. In *2009 IEEE International Symposium on Information Theory*, pages 324–328, 2009.

A Appendix

proof Theorem 3.1

By Von Neumann's trace inequality, we get

$$\begin{aligned} \mathrm{Tr}(\mathbf{AXB}^T) &= \mathrm{Tr}(\mathbf{XB}^T\mathbf{A}) \\ &\leq \sum_{i=1}^{\min(m,n)} \sigma_i(\mathbf{X})\sigma_i(\mathbf{B}^T\mathbf{A}), \end{aligned} \tag{56}$$



(a) Original image (b) Masked image (c) SVT (d) SVP

(e) Opt (f) tnnr-admm (g) tnnr-apgl (h) tnnr-admmap

Figure 6: The reconstruction error versus the matrix rank (r_0) using the synthetic dataset. (The original result of paper)

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

$$\begin{aligned}\sum_{i=1}^{\min(m,n)} \sigma_i(\mathbf{X})\sigma_i(\mathbf{B}^T\mathbf{A}) &= \sum_{i=1}^{s} \sigma_i(\mathbf{X})\sigma_i(\mathbf{B}^T\mathbf{A}) + \sum_{i=s+1}^{\min(m,n)} \sigma_i(\mathbf{X})\sigma_i(\mathbf{B}^T\mathbf{A}) \\ &= \sum_{i=1}^{s} \sigma_i(\mathbf{X}) \cdot 1 + \sum_{i=s+1}^{\min(m,n)} \sigma_i(\mathbf{X}) \cdot 0 \\ &= \sum_{i=1}^{s} \sigma_i(\mathbf{X}).\end{aligned} \tag{57}$$

Since $s \leq r$ and $\sigma_i(\mathbf{X}) \geq 0$:

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

Combining inequalities and ,we have

$$\text{Tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) \leq \sum_{i=1}^{s} \sigma_i(\mathbf{X}) \leq \sum_{i=1}^{r} \sigma_i(\mathbf{X}). \tag{58}$$

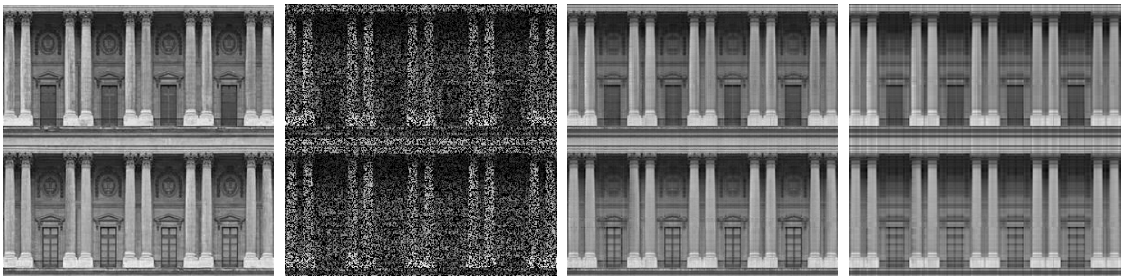
proof Equation 32

Fix $\mathbf{W}_k$ and $\mathbf{Y}_k$,

$$\mathbf{X}_{k+1} = \arg\min_{\mathbf{X}} \|\mathbf{X}\|_* - \text{Tr}(\mathbf{A}_l\mathbf{W}_k\mathbf{B}_l^T) + \frac{\beta}{2}\|\mathbf{X} - \mathbf{W}_k\|_F^2 + \text{Tr}(\mathbf{Y}_k(\mathbf{X} - \mathbf{W}_k)). \tag{59}$$

For the last two terms of (59),

$$\frac{\beta}{2}\|\mathbf{X} - \mathbf{W}_k\|_F^2 + \text{Tr}(\mathbf{Y}_k^T(\mathbf{X} - \mathbf{W}_k)) = \frac{\beta}{2}\left(\|\mathbf{X} - \mathbf{W}_k\|_F^2 + \frac{2}{\beta}\text{Tr}(\mathbf{Y}_k^T(\mathbf{X} - \mathbf{W}_k))\right). \tag{60}$$



(a) Original image (b) Masked image (c) SVT (d) SVP

(e) Opt (f) tnnr-admm (g) tnnr-apgl (h) tnnr-admmap

Figure 7: The reconstruction error versus the matrix rank (r_0) using the synthetic dataset. (The result we reproduced)

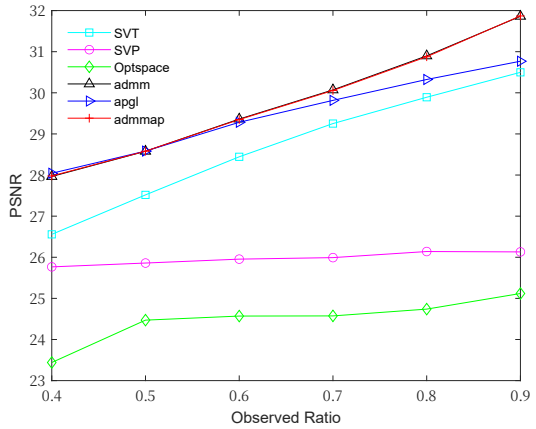


Figure 8: The PSNR values of the recovered image under different observed ratios (random mask) by all six methods. (The original result of paper)

Because

$$\begin{aligned}\|\mathbf{X}-\mathbf{W}_k+\frac{1}{\beta}\mathbf{Y}_k\|_F^2 &= \mathrm{Tr}\,(\mathbf{X}-\mathbf{W}_k+\frac{1}{\beta}\mathbf{Y}_k)(\mathbf{X}-\mathbf{W}_k+\frac{1}{\beta}\mathbf{Y}_k)^T\\ &= \mathrm{Tr}(\mathbf{X}-\mathbf{W}_k)(\mathbf{X}-\mathbf{W}_k)^T+2\,\mathrm{Tr}(\mathbf{X}-\mathbf{W}_k)\frac{1}{\beta}\mathbf{Y}_k^T+\frac{1}{\beta^2}\mathrm{Tr}(\mathbf{Y}_k\mathbf{Y}_k^T)\\ &= \|\mathbf{X}-\mathbf{W}_k\|_F^2+\frac{1}{\beta^2}\|\mathbf{Y}_k\|_F^2+\frac{2}{\beta}\mathrm{Tr}(\mathbf{Y}_k^T(\mathbf{X}-\mathbf{W}_k)).\end{aligned}$$

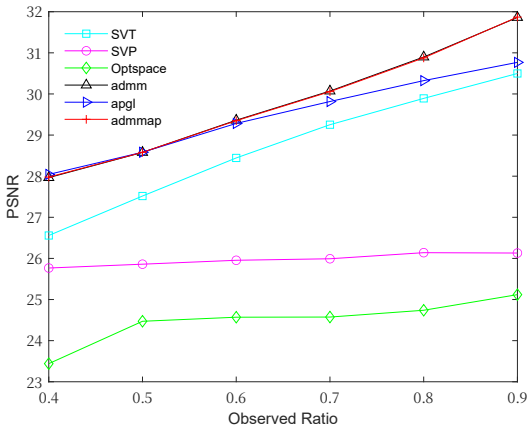


Figure 9: The PSNR values of the recovered image under different observed ratios (random mask) by all six methods. (The result we reproduced)

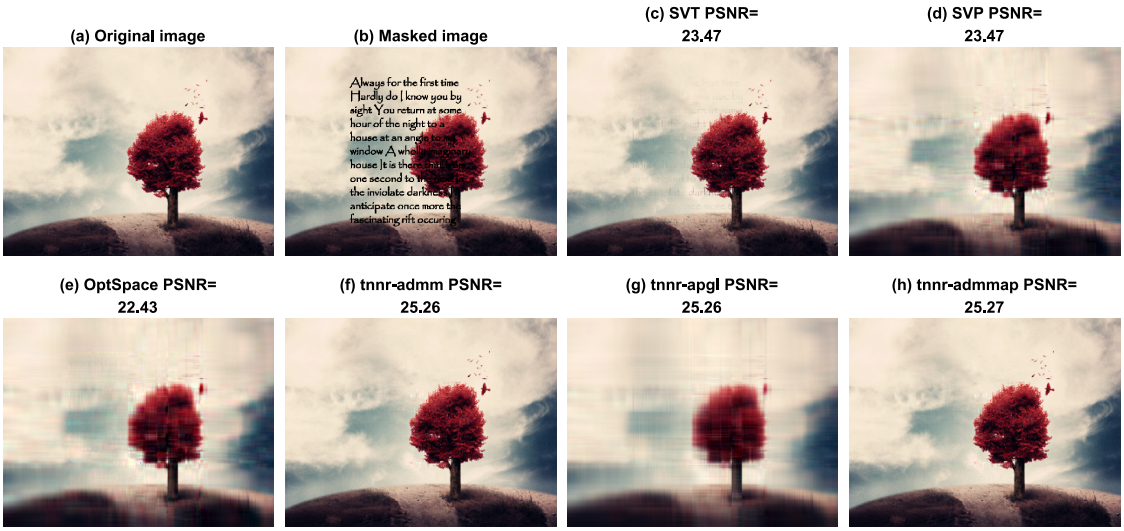


Figure 10: Comparison of matrix completion algorithms for text removal problem. (The original result of paper)

For the latter term in parentheses of (60),

$$\begin{aligned}\frac{2}{\beta}\mathrm{Tr}(\mathbf{Y}_k^T(\mathbf{X}-\mathbf{W}_k)) &= 2\,\mathrm{Tr}(\frac{1}{\beta}\mathbf{Y}_k^T(\mathbf{X}-\mathbf{W}_k)) \\ &= \|\mathbf{X}-\mathbf{W}_k+\frac{1}{\beta}\mathbf{Y}_k\|_F^2-\frac{1}{\beta^2}\|\mathbf{Y}_k\|_F^2-\|\mathbf{X}-\mathbf{W}_k\|_F^2.\end{aligned} \tag{61}$$

Ignoring all constant terms,

$$\begin{aligned}\mathbf{X}_{k+1} &= \arg\min_{\mathbf{X}} L(\mathbf{X},\mathbf{Y}_k,\mathbf{W}_k,\beta) \\ &= \arg\min_{\mathbf{X}} \|\mathbf{X}\|_*-\mathrm{Tr}(\mathbf{A}_l\mathbf{W}_k\mathbf{B}_l^T)+\frac{\beta}{2}\|\mathbf{X}-\mathbf{W}_k\|_F^2+\mathrm{Tr}(\mathbf{Y}_k^T(\mathbf{X}-\mathbf{W}_k)) \\ &= \arg\min_{\mathbf{X}} \|\mathbf{X}\|_*+\frac{\beta}{2}\|\mathbf{X}-\left(\mathbf{W}_k-\frac{1}{\beta}\mathbf{Y}_k\right)\|_F^2-\frac{1}{2\beta}\|\mathbf{Y}_k\|_F^2 \\ &= \arg\min_{\mathbf{X}} \|\mathbf{X}\|_*+\frac{\beta}{2}\|\mathbf{X}-\left(\mathbf{W}_k-\frac{1}{\beta}\mathbf{Y}_k\right)\|_F^2.\end{aligned} \tag{62}$$

That is

$$\mathbf{X}_{k+1} = D_{\frac{1}{\rho}}(\mathbf{W}_k-\frac{1}{\rho}\mathbf{Y}_k). \tag{63}$$

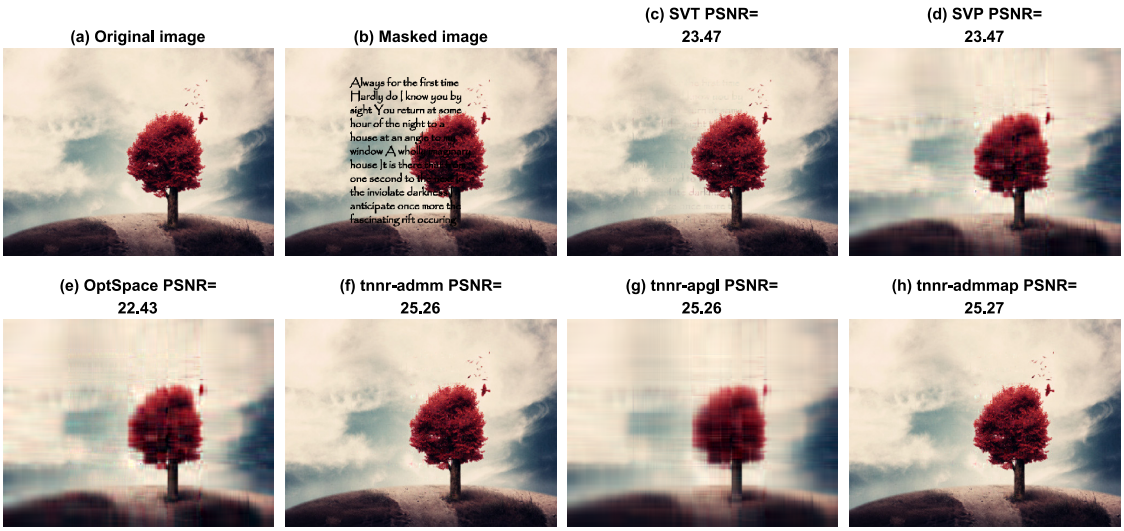


Figure 11: Comparison of matrix completion algorithms for text removal problem. (The result we reproduced)

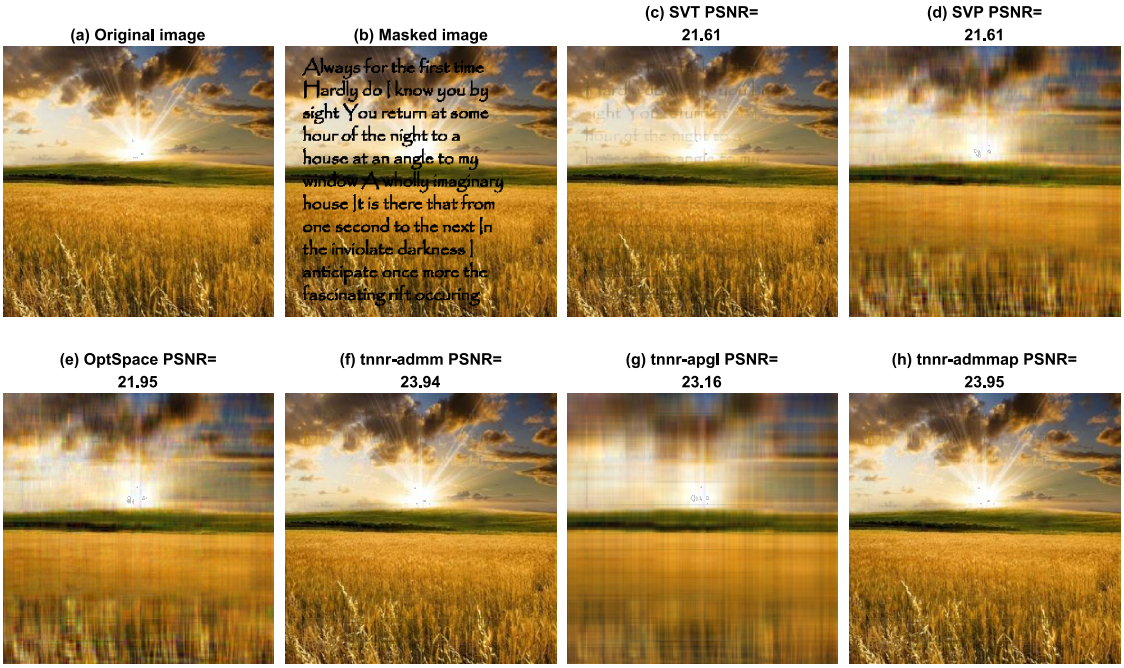


Figure 12: Comparison of matrix completion algorithms for text removal problem. (The original result of paper)

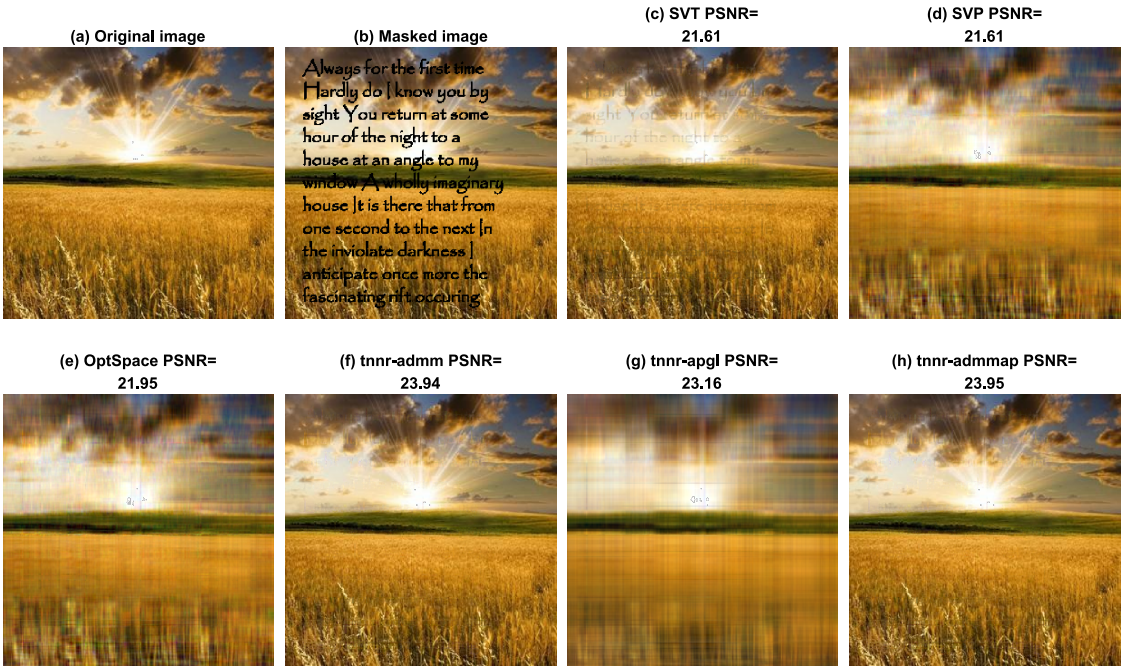


Figure 13: Comparison of matrix completion algorithms for text removal problem. (The result we reproduced)

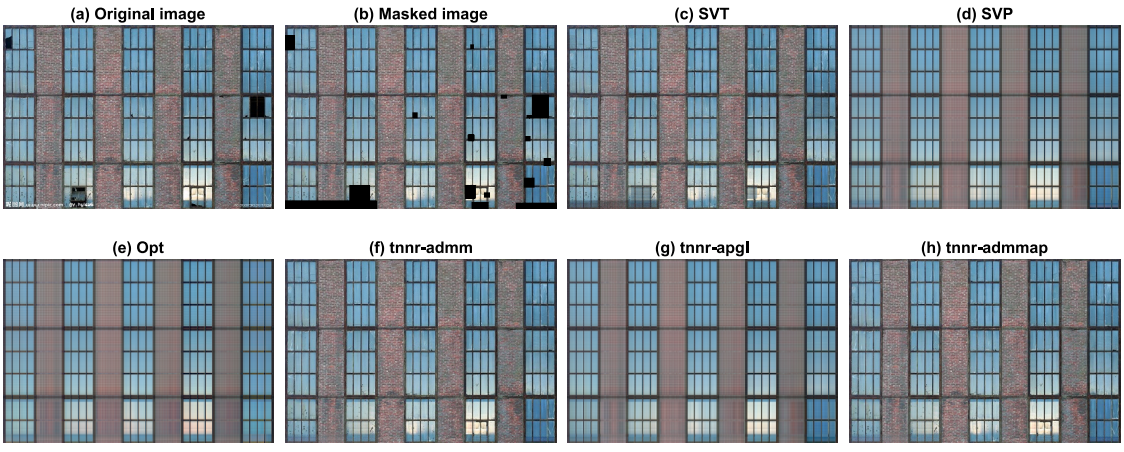


Figure 14: Comparison of different methods for recovering missing blocks of a natural image. (The original result of paper)

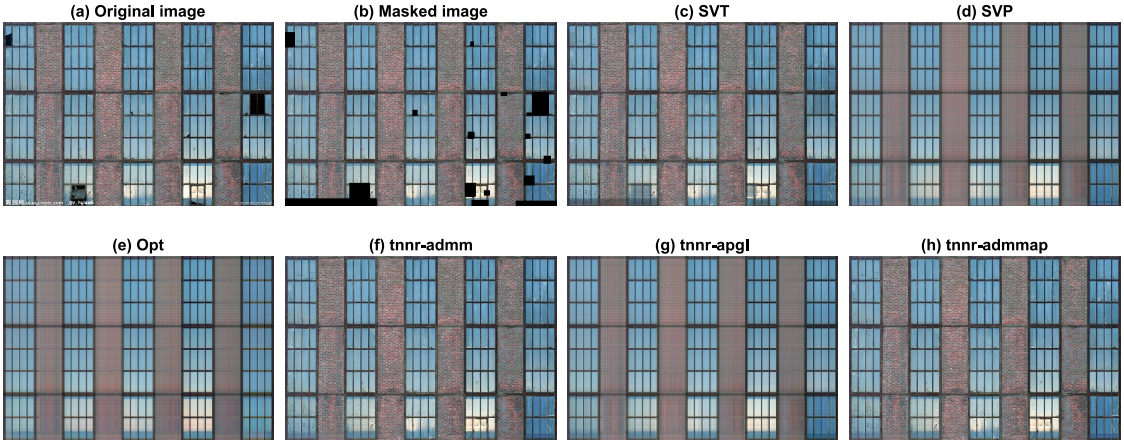


Figure 15: Comparison of different methods for recovering missing blocks of a natural image. (The result we reproduced)