A Scalable, Adaptive and Sound Nonconvex Regularizer for Low-rank Matrix Learning

Yaqing Wang  
Business Intelligence Lab, Baidu Research  
Beijing, China  
wangyaqing01@baidu.com

Quanming Yao  
4Paradigm Inc.  
EE, Tsinghua University  
Beijing, China  
qyaoaa@connect.ust.hk

James T. Kwok  
Department of Computer Science, Hong Kong University of Science and Technology  
Hong Kong, China  
jamesk@cse.ust.hk

ABSTRACT

Matrix learning is at the core of many machine learning problems. A number of real-world applications such as collaborative filtering and text mining can be formulated as a low-rank matrix completion problem, which recovers incomplete matrix using low-rank assumptions. To ensure that the matrix solution has a low rank, a recent trend is to use nonconvex regularizers that adaptively penalize singular values. They offer good recovery performance and have nice theoretical properties, but are computationally expensive due to repeated access to individual singular values. In this paper, based on the key insight that adaptive shrinkage on singular values improve empirical performance, we propose a new nonconvex low-rank regularizer called “nuclear norm minus Frobenius norm” regularizer, which is scalable, adaptive and sound. We first show it provably holds the adaptive shrinkage property. Further, we discover its factored form which bypasses the computation of singular values and allows fast optimization by general optimization algorithms. Stable recovery and convergence are guaranteed. Extensive low-rank matrix completion experiments on a number of synthetic and real-world data sets show that the proposed method obtains state-of-the-art recovery performance while being the fastest in comparison to existing low-rank matrix learning methods. ¹

CCS CONCEPTS
• Computing methodologies → Machine learning: Regularization; Factorization methods; • Information systems → Collaborative filtering, Recommender systems; • Human-centered computing → Collaborative filtering.

KEYWORDS
Low-rank Matrix Learning, Matrix Completion, Nonconvex Regularization, Collaborative Filtering, Recommender Systems

ACM Reference Format:

1 INTRODUCTION

In many real-world scenarios, the data can be naturally represented as matrices. Examples include the rating matrices in recommender systems [7, 22, 23, 37, 39], the term-document matrices in natural language processing [33, 38], images in computer vision [14, 20], and climate observations in spatial-temporal analysis [1]. Thus, matrix learning is an important and fundamental tool in machine learning [9, 10, 39], data mining [1, 22], and computer vision [14, 44].

In this paper, we focus on an important class of matrix learning problems, namely matrix completion, which tries to predict the missing entries of a partially observed matrix [9]. For example, in collaborative filtering [22], the rating matrix is often incomplete and one wants to predict the missing user ratings for all items. In climate analysis [1], observation records from only a few meteorological stations are available, and one wants to predict climate information for the other locations. In image inpainting [14, 20], the image has pixels missing and one wants to fill in these missing values. To avoid the problem to be ill-posed, the target matrix is often assumed to have a low rank [7]. To obtain such a solution, a direct approach is to add a rank-minimizing term to the optimization objective. However, rank minimization is NP-hard [7]. Thus, computationally, a more feasible approach is to use a regularizer that encourages the target matrix to have a small rank.

There exist various low-rank regularizers. Nuclear norm regularizer is the tightest convex surrogate for matrix rank [7], which has good recovery and convergence guarantees. Defined as the sum of singular values, nuclear norm requires repeatedly computing the singular value decomposition (SVD), which is expensive. To be more efficient, a series of works instead turn to matrix factorization which factorizes the recovered matrix into factor matrices. Some of them work towards theoretical justification [40, 42], while the other targets at designing better algorithms [3, 16, 41]. However, the performance of matrix factorization is not satisfactory [11, 44]. To this end, factored low-rank regularizers are invented to balance efficiency and effectiveness, such as factored nuclear norm [39] and factored group-sparse regularizer (GSR) [11]. It is proved that factored nuclear norm can obtain comparable result as nuclear norm under mild condition [39].

Recently, nonconvex low-rank regularizers (Table 1) which penalize less on the more informative large singular values are proposed,
such as Schatten-p norm [30], truncated ℓ₁,₂ norm [27], capped-ℓ₁ penalty [46], log-sum penalty (LSP) [9], and minimax concave penalty (MCP) [45]. These nonconvex regularizers can outperform nuclear norm both theoretically [15, 28] and empirically [25, 26, 44]. However, as shown in Table 1, none of the above-mentioned regularizers obtain (A) scalability; (B) good performance and (C-D) sound theoretical guarantee simultaneously.

To fill in this blank, we propose a scalable, adaptive and sound nonconvex regularizer based on the key insight that adaptive shrinkage property of common nonconvex regularizers can improve empirical performance. Specifically, Our contribution can be summarized as follows:

- We propose a new nonconvex regularizer called “nuclear norm minus Frobenius norm” (NNFN) regularizer for low-rank matrix learning, which is scalable, adaptive and theoretically guaranteed.
- We show that NNFN regularizer can be factorized to sidestep the expensive SVD. This problem can be optimized by general algorithms such as gradient descent.
- We provide sound theoretical analysis on statistical and convergence properties of both NNFN and factored NNFN regularizers.
- We conduct extensive experiments on both synthetic and a number of real-world data sets including recommendation data and climate record data. In comparison to existing methods, results consistently show that the proposed algorithm obtains state-of-the-art recovery performance while being the fastest.

Notations: Vectors are denoted by lowercase boldface, matrices by uppercase boldface. (·)⊤ denotes transpose operator and A∗ = [max(Aij, 0)]. For a vector x = [xij] ∈ ℝm, Diag(x) constructs a m × m diagonal matrix with the ith diagonal element being x. l denotes the identity matrix. For a square matrix X, tr(X) is its trace. For matrix X ∈ ℝm×n (without loss of generality, we assume that m ≥ n), ∥X∥F = √tr(X⊤X) is its Frobenius norm. Let the singular value decomposition (SVD) of a rank-k X be UD(σ(X))V⊤, where U ∈ ℝm×k, V ∈ ℝn×k, σ(X) = |σi(X)| ∈ ℝk with σi(X) being the ith singular value of X and σ1(X) ≥ σ2(X) ≥ · · · ≥ σk(X) ≥ 0.

2. BACKGROUND: LOW-RANK MATRIX LEARNING

As minimizing the rank is NP-hard [7], low-rank matrix learning is often formulated as the following optimization problem:

\[
\min_X f(X) + \lambda r(X),
\]

where f is a smooth function (usually the loss), r(X) is a regularizer that encourages X to be low-rank, and λ ≥ 0 is a tradeoff hyperparameter. Let Ω ∈ {0, 1}m×n record positions of the observed entries (with Ωij = 1 if Oij is observed, and 0 otherwise), and PΩ() is a projection operator such that [PΩ(A)]ij = Aij if Ωij = 1 and 0 otherwise. Low-rank matrix completion [7] tries to recover the underlying low-rank matrix X ∈ ℝm×n from an incomplete matrix 0 ∈ ℝm×n with only a few observed entries. It usually sets f(X) as

\[
f(X) = \frac{1}{2} ∥PΩ(X - O)∥_F^2,
\]

which measures the recovery error.

2.1 Convex Nuclear Norm Regularizer

The convex nuclear norm ∥X∥∗ = ∥σ(X)∥₁ [7], is the tightest convex surrogate of the matrix rank [12]. Problem (1) is usually solved by the proximal algorithm [32]. At the tth iteration, it generates the next iterate by computing the proximal step Xt+1 = proxτ,λ,X( Xt − τ∇V(Xt)), where τ > 0 is the stepsize, and

\[
\text{prox}_τ,λ,X(Z) = \arg\min_X \frac{1}{2} ∥X - Z∥_F^2 + λτ(X)
\]

is the proximal operator. In general, the proximal operator should be easily computed. For the nuclear norm, its proximal operator is computed as [6]:

\[
\text{prox}_τ,λ,X(Z) = U(\text{Diag}(σ(Z)) - λI)_+ V^T,
\]

where UDiag(σ(Z))V⊤ is the SVD of Z.

2.2 Nonconvex Regularizers

Recently, various nonconvex regularizers appear (Table 1). Common examples include the capped-ℓ₁ penalty [46], log-sum penalty (LSP) [9], and minimax concave penalty (MCP) [45]. As shown in Table 2, they can be written in the general form of

\[
r(X) = \sum_{i=1}^n \hat{r}(σ_i(X)),
\]

where r(α) is nonlinear, concave and non-decreasing for α ≥ 0 with r(0) = 0. In contrast to the proximal operator in (4) which penalizes all singular values of Z by the same amount λ, these nonconvex regularizers penalize less on the larger singular values which are more informative. Additionally, the nonconvex Schatten-p norm [30] can better approximate rank than nuclear norm. Truncated ℓ₁,₂ regularizer [27] can obtain unbiased approximation for rank [27]. These nonconvex regularizers outperform nuclear norm on many applications empirically [14, 26, 44], and can obtain lower recovery errors [15]. However, learning with nonconvex regularizers is very difficult. It usually requires dedicated solvers to leverage special structures (such as the low-rank-plus-sparse structure in [18, 44]) or involves several iterative algorithms (such as the difference of convex functions algorithm (DCA) [19] with subproblems solved by the alternating direction method of multipliers (ADMM) [5] for truncated ℓ₁,₂ regularized problem). This computation bottleneck limits their applications in practice.

2.3 Factored Regularizers

Note that aforementioned regularizers require access to individual singular values. As computing the singular values of a m × n matrix (with m ≥ n) via SVD takes O(mn²) time, this can be costly for a large matrix. Even when rank-k truncated SVD is used, the computation cost is still O(mnk). To relieve the computational burden, factored low-rank regularizers are invented. (1) can then be rewritten into a factored form as

\[
\min_{W, H} f(WH^T) + μg(W, H),
\]

where X is factored into W ∈ ℝm×k and H ∈ ℝn×k, and μ ≥ 0 is a hyperparameter. When μ = 0, this reduces to matrix factorization [3, 16, 40–42]. Not all regularizers r(X) have equivalent factored form g(W, H). For a matrix X with rank k ≤ k, it is already discovered that nuclear norm can be rewritten in a factored form [39]
Table 1: Comparisons among nonconvex low-rank regularizers on (A): Scalability (can be optimized in factored form); (B): Performance (can adaptively penalize singular values); (C): Statistical guarantee; (D): Convergence guarantee.

<table>
<thead>
<tr>
<th>nonconvex low-rank regularizer</th>
<th>expression</th>
<th>(A)</th>
<th>(B)</th>
<th>(C)</th>
<th>(D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>factored nuclear norm [39]</td>
<td>( \min_{X=WH^*} \frac{4}{2} (|W|_F^2 + |H|_F^2) )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Schatten-p [30]</td>
<td>( \lambda (\sum_{i=1}^n \sigma_i^2 (X))^{1/p} )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>factored GSR [11]</td>
<td>( \min_{X=WH^*} \frac{4}{2} (|W|_2^1 + |H|_2^1) )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>capped-( \ell_1 ), LSP, and MCP [26, 44]</td>
<td>( \sum_{i=1}^m r(\sigma_i(X)) ) (see ( r ) in Table 2)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>truncated ( \ell_{1,2} ) [27]</td>
<td>( \sum_{i=t+1}^n \hat{\sigma}<em>i(X) - (\sum</em>{i=t+1}^n \sigma_i^2 (X))^{1/2} )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>NNFN</td>
<td>( |X|_* - |X|_F )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>factored NNFN</td>
<td>( \min_{X=WH^*} \frac{4}{2} (|W|_F^2 + |H|_F^2) - \lambda |WH^T|_F )</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2: Nonconvex low-rank regularizers in the form of \( f \).

<table>
<thead>
<tr>
<th>regularizer</th>
<th>( f(\sigma_i(X)) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>capped-( \ell_1 )</td>
<td>( \min(\sigma_i(X), \theta) )</td>
</tr>
<tr>
<td>LSP</td>
<td>( \log(\frac{1}{\theta \sigma_i(X) + 1}) )</td>
</tr>
</tbody>
</table>
| MCP            | \( \sigma_i(X) - \frac{\sigma_i^2(X)}{2\theta \lambda} \) if \( \sigma_i(X) \leq \theta \lambda \) \<br> \( \frac{\theta \lambda}{2} \) otherwise |}

As \( \|X\|_* = \min_{X=WH^*} \|X\|_* - \|X\|_F \). As for nonconvex low-rank regularizers, only Schatten-p norm can be approximated by factored forms [11, 36]. Other nonconvex regularizers, which need to penalize individual singular values, cannot be written in factored form.

3 NUCLEAR NORM MINUS FROBENIUS NORM (NNFN) REGULARIZER

Based on the insight that adaptive shrinkage on singular values can improve empirical performance, we present a new nonconvex regularizer

\[
r_{\text{NNFN}}(X) = \|X\|_* - \|X\|_F, \tag{6}
\]

which will be called the "nuclear norm minus Frobenius norm" (NNFN) regularizer. Next, we will show that NNFN regularizer applies adaptive shrinkage for singular values provably, has factored form which allows fast optimization by general algorithms, and has sound theoretically guarantee.

3.1 Adaptive Shrinkage Property

Recall from (3) that the proximal operator of the nuclear norm equally penalizes each singular value by \( \lambda \) until it reaches zero. In contrast, we find that common nonconvex regularizers \( r(X) \) of the general form (4) all hold the adaptive shrinkage property in Proposition 1.

**Proposition 1 (Adaptive Shrinkage Property).** Let \( r(X) \) be a nonconvex low-rank regularizer of the form (4), and \( \hat{\sigma} = [\hat{\sigma}_1] = \text{prox}_{\lambda r(\cdot)}(\sigma(Z)) \) in (7). Then, (i) \( \sigma_i(Z) \geq \hat{\sigma}_i \) (shrinkage); and (ii) \( \sigma_i(Z) - \hat{\sigma}_i \leq \sigma_{i+1}(Z) - \sigma_{i+1} \) (adaptivity), where strict inequality holds at least for one \( i \).

\( ^2 \)All the proofs are in Appendix A.

It shows that \( \text{prox}_{z} f(\cdot) \) adaptively shrinks the singular values of its matrix argument, in that larger singular values are penalized less. This property is important for obtaining good empirical performance [14, 20, 25, 26, 44]. Other nonconvex regularizers such as truncated \( \ell_{1,2} \) and Schatten-p norm, do not have this property due to the lack of analytic proximal operators. Figure 1 shows the shrinkage performed by adaptive nonconvex regularizers versus the convex nuclear norm regularizer. As can be seen, the convex nuclear norm regularizer shrinks all singular values by the same amount; whereas the adaptive nonconvex regularizers enforce different amounts of shrinkage depending on the magnitude of \( \sigma_i(Z) \).

![Figure 1: Shrinkage performed by different regularizers. The hyperparameters are tuned such that \( \sigma_i = \) zero for \( \sigma_i(Z) \leq 5 \).](image.png)

Here, we show the proposed NNFN regularizer in (6) also provably satisfies adaptive shrinkage of the singular values when used with a proximal algorithm. We first present the proximal operator of \( r_{\text{NNFN}}(\cdot) \) in Proposition 2. As \( \text{prox}_{\lambda \|\cdot\|_{1,2}}(\sigma(Z)) \) returns a sparse vector [24], the resultant \( \text{prox}_{\lambda r_{\text{NNFN}}(\cdot)}(Z) \) is low-rank.

**Proposition 2.** Given a matrix \( Z \), let its SVD be \( \hat{U}\text{Diag}(\sigma(Z))\hat{V}^T \), and \( \lambda \leq \|\sigma(Z)\|_{1,2} \).

\[
\text{prox}_{\lambda r_{\text{NNFN}}}(Z) = \hat{U}\text{Diag}(\text{prox}_{\lambda \|\cdot\|_{1,2}}(\sigma(Z)))\hat{V}^T, \tag{7}
\]

where \( \text{prox}_{\lambda \|\cdot\|_{1,2}}(z) \) has closed-form solution [24].

Now, we are ready to prove the following Corollary that NNFN regularizer also shares the adaptive shrinkage property. This can lead to better empirical performance as discussed earlier.

**Corollary 3.** The two properties in Proposition 1 also hold for the proximal operators of the NNFN regularizer.
4 ALGORITHMS FOR (1) WITH NNFN REGULARIZER

With the proximal operator obtained in Proposition 2, learning with the NNFN regularizer can be readily solved with the proximal algorithm. However, it still relies on computing the SVD in each iteration. To tackle this problem, we then present a simple and scalable algorithm that avoids SVD computations by using the factored NNFN regularizer.

4.1 A Proximal Algorithm

We first present a direct application of the proximal algorithm to problem (1) with the NNFN regularizer. At the tth iteration, we obtain \( Z^t = X^{t-1} - \eta \nabla f(X^{t-1}) \), and then perform the proximal step in Proposition 2. The complete procedure is shown in Algorithm 1.

Algorithm 1 A proximal algorithm for (1) with NNFN.

**Input:** Randomly initialized \( X^0 \), stepsize \( \eta \)

1. for \( t = 1, \ldots, T \) do
2. obtain \( Z^t = X^{t-1} - \eta \nabla f(X^{t-1}) \);
3. update \( X^t \) as \( \text{prox}_{\lambda \text{NNFN}}(Z^t) \);
4. end for

4.1.1 Complexity. The iteration time complexity of Algorithm 1 is dominated by SVD. Let \( r_t \) (\( n \geq r_t \geq k \)) be the rank estimated at the tth iteration. We can perform rank-k truncated SVD, which takes \( O(mk^2) \). The space complexity is \( O(mn) \) to keep full matrices.

4.2 A General Solver for Factored Form

In this section, we propose a more efficient solver which removes the SVD bottleneck. Let \( r_t \) (\( n \geq r_t \geq k \)) be the rank estimated at each iteration. Without the operation to truncate singular values, the Frobenius norm of a matrix can be computed without using its singular values, and the nuclear norm can be replaced by the factored nuclear norm. With this factored NNFN regularizer, the matrix learning problem then becomes:

\[
\min_{W,H} f(W,H) = f(W^T) + \frac{1}{2} \left( \|W\|_F^2 + \|H\|_F^2 \right) - \lambda \|WH^T\|_F. \tag{8}
\]

Thus, SVD can be completely avoided. In contrast, the other nonconvex low-rank regularizers (including the very related truncated \( \ell_2 \)-regularizer with \( t \geq 0 \)) need to penalize individual singular values, and so do not have factored form.

Unlike other regularizers which requires dedicated solvers, the reformulated problem (8) can be simply solved by general solvers such as gradient descent. In particular, gradients of \( f(W,H) \) can be easily obtained. Let \( Q \equiv WH^T \neq 0 \), and \( c = \lambda \|WH^T\|_F \). Then, we obtain

\[
\nabla_W f(W,H) = [\nabla_Q f(Q)]H + \lambda W - c(WH^T H), \tag{9}
\]

\[
\nabla_H f(W,H) = [\nabla_Q f(Q)]^T W + \lambda H - c(H^T W). \tag{10}
\]

These only involve simple matrix multiplications, without any SVD computation. Moreover, we can easily replace the simple gradient descent by recent solvers with improved performance. The complete procedure is shown in Algorithm 2.

Algorithm 2 A general solver for (1) with factored NNFN.

**Input:** Randomly initialized \( W^0, H^0 \), stepsize \( \eta \)

1. for \( t = 1, \ldots, T \) do
2. update \( W^t = W^{t-1} - \eta \nabla_W f(W^{t-1}, H^{t-1}) \) using (9);
3. update \( H^t = H^{t-1} - \eta \nabla_H f(W^{t-1}, H^{t-1}) \) using (10);
4. end for

4.2.1 Complexity. Learning with factored NNFN does not need the expensive SVD, thus it has a much lower time complexity. Specifically, multiplication of the sparse matrix \( \nabla_Q f(Q) = \mathcal{P}_Q(Q - O) \) and \( H \) in \( \nabla_W f(W,H) \) (and similarly multiplication of \( [\nabla_Q f(Q)]^T W \) and \( H \) in \( \nabla_H f(W,H) \)) takes \( O(\|Q\|k) \) time, computation of \( WH^T \) takes \( O(\|Q\|k + mk^2) \) time. As for space, using the factored form reduces the parameter size from \( O(mn + \|Q\|k) \) to \( O(mk + \|Q\|_0) \), where \( \|Q\|_0 \) is the space for keeping a sparse \( Q \).

4.3 Comparison with Optimizing Other Regularizers

We compare the proposed solvers with state-of-the-art solvers for other regularizers in Table 3. Among nonconvex regularizers, only factored NNFN can be solved by general solvers such as gradient descent, which makes it simple and efficient. In contrast, other nonconvex regularizers are difficult to optimize and require dedicated solvers. Although the time complexity is comparable in big \( O \), we observe in experiments that learning with factored NNFN is much more scalable. Additionally, for space, only solvers for truncated \( \ell_1 \)-[27] and NNFN require keeping the complete matrix which takes \( O(mn) \) space, while the other methods have comparable and much smaller space requirements.

**Remark 1.** Truncated \( \ell_1 \)-regularizer [27] is a related existing nonconvex regularizer. When \( t = 0 \), it reduces to NNFN regularizer. However, without the operation to truncate singular values, NNFN regularizer (1) is proved to enforce adaptive shrinkage while truncated \( \ell_1 \)-regularizer does not (2); allows cheap closed-form proximal operator while truncated \( \ell_1 \)-regularizer requires a combined use of DCA and ADMM; (3) can be efficiently optimized in factored form without taking SVD while truncated \( \ell_1 \)-regularizer cannot; (4) has recovery bound for both itself and its factored form while the analysis in [27] does not apply for factored form. Therefore, the discovery of NNFN regularizer is new and important.

5 THEORETICAL ANALYSIS

Here, we analyze the statistical and convergence properties for the proposed algorithms.

5.1 Recovery Guarantee

We establish statistical guarantee based on Restricted Isometry Property (RIP) [8] introduced below.

**Definition 1 (Restricted Isometry Property (RIP) [8]).** An affine transformation \( A \) satisfies RIP if for all \( X \in \mathbb{R}^{m \times n} \) of rank at
Table 3: State-of-the-art solvers for various matrix completion methods. Here, \( r_t \) (usually \(\geq k \)) is an estimated rank at the \(t\)th iteration, \( r_t = r_t + r_{t-1} \), and \( q \) is number of inner ADMM iterations used in [27].

<table>
<thead>
<tr>
<th>regularizer</th>
<th>state-of-the-art solver</th>
<th>time complexity</th>
<th>space complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>nuclear norm [7]</td>
<td>softimpute algorithm with alternating least squares[18]</td>
<td>( O(|\Omega|s + mr^2_t) )</td>
<td>( O((m + n)r + |\Omega|) )</td>
</tr>
<tr>
<td>factored nuclear norm[39]</td>
<td>alternating gradient descent [13]</td>
<td>( O(|\Omega|k + mk) )</td>
<td>( O((m + n)k + |\Omega|) )</td>
</tr>
<tr>
<td>probabilistic matrix factorization [29]</td>
<td>Bayesian probabilistic matrix factorization solver using Markov Chain Monte Carlo [35]</td>
<td>( O(|\Omega|k + 2 + mk^3) )</td>
<td>( O((m + n)k + |\Omega|) )</td>
</tr>
<tr>
<td>factored GSR [11]</td>
<td>proximal alternating linearized algorithm coupled with iteratively reweighted minimization [11]</td>
<td>( O(mnk) )</td>
<td>( O((m + n)k + |\Omega|) )</td>
</tr>
<tr>
<td>truncated ( \ell_{1,2} ) [27]</td>
<td>DCA algorithm with sub-problems solved by ADMM algorithm[27]</td>
<td>( O(qmn^2) )</td>
<td>( O(mn) )</td>
</tr>
<tr>
<td>capped-( \ell_1 ), LSP, and MCP [26, 44]</td>
<td>a solver leveraging power method and &quot;low-rank plus sparse&quot; structure [44]</td>
<td>( O(|\Omega|r + m^2r_t^2) )</td>
<td>( O((m + n)r + |\Omega|) )</td>
</tr>
<tr>
<td>NNFN</td>
<td>proximal algorithm</td>
<td>( O(mnr_t) )</td>
<td>( O(mn) )</td>
</tr>
<tr>
<td>factored NNFN</td>
<td>general solvers such as gradient descent</td>
<td>( O(|\Omega|k + mk^2) )</td>
<td>( O((m + n)k + |\Omega|) )</td>
</tr>
</tbody>
</table>

most \( k \), there exists a constant \( \delta_k \in (0, 1) \) such that:

\[
(1 - \delta_k)\|X\|_F^2 \leq \|\hat{A}(X) - b\|_2^2 \leq (1 + \delta_k)\|X\|_F^2.
\]

Under the RIP condition, we prove in the following that stable recovery is guaranteed where the estimation error depends linearly on \( \|e\|_2^2 \).

THEOREM 4 (STABLE RECOVERY). Consider \( f(X) = \frac{1}{2}\|\hat{A}(X) - b\|_2^2 \), where \( \hat{A} \) is an affine transform satisfying the RIP with \( \delta_{2k} \leq 1/3 \), and \( b = \hat{A}(X^*) + e \) is a measurement vector corresponding to a rank-\( k \) matrix \( X^* \) and error vector \( e \). Assume sequence \( \{X^t\} \) with \( f(X^{t+1}) < f(X^t) \) and each \( X^t \) is the iterate obtained by optimizing the following two equivalent constrained formulations of (8): (i) \( X^t \) is the iterate of optimizing \( \min_X f(X) \) s.t. \( \text{NNFN}(X) \leq \beta' \), where \( \beta' \geq 0 \) is a hyperparameter, or (ii) \( X^t = W^t(H^t)^\top \) is the iterate of optimizing \( \min_{W,H} f(WH^\top) \) s.t. \( \frac{1}{2}\|W\|_F^2 + \|H\|_F^2 - \|WH^\top\|_F \leq \beta' \), where \( \beta' \geq 0 \) is a hyperparameter. Then, the recovery error is bounded as \( \|X^* - X^t\|_F^2 \leq c\|e\|_2^2 \) for some constant \( c \) and sufficiently large \( t \).

Existing theoretical analysis [15] applies for adaptive nonconvex regularizer with separable penalty on individual singular values. Hen it does not apply for NNFN regularizer which is not separable.

6 EXPERIMENTS

Here, we perform matrix completion experiments on both synthetic and real-world data sets, using a PC with Intel i7 3.6GHz CPU and 48GB memory. Experiments are repeated five times, and the averaged performance are reported.

6.1 Experimental Settings

6.1.1 Baselines. The proposed NNFN regularizer solved by proximal algorithm, and its scalable variant factored NNFN solved by gradient descent, are compared with the representative regularizers optimized by their respective state-of-the-art solvers as listed in Table 3. For all methods that we compare in the experiments, we use public codes unless they are not available.

- Low-rank regularizers include: (i) nuclear norm [18] [7]; (ii) truncated \( \ell_{1,2} \) regularizer [27]; (iii) adaptive nonconvex low-rank regularizers of the form (4); including the capped-\( \ell_1 \) penalty [46], LSP [9]; and MCP [45].
- Factored regularizers include (i) factored nuclear norm [39]; (ii) BMF [29]; and (iii) factored GSR [11].

Note that learning with factored regularizers solves (5), which reduces to matrix factorization when \( \mu = 0 \). Therefore, we do not additionally compare with matrix factorization methods [39, 40, 43].

All the algorithms are implemented in MATLAB (with sparse operations written in C as MEX functions). Each algorithm is stopped when the relative difference between objective values in consecutive iterations is smaller than \( 10^{-4} \). All hyperparameters including stepsize, \( \lambda \), \( k \), \( r_t \) and hyperparameters of baseline methods are tuned by grid search using the validation set. Specifically, \( \lambda \) in (1) is chosen from \([10^{-3}, 10^2]\), \( r_t \) and \( k \) is an integer chosen from \([1, \min(m,n)]\).

3https://cran.r-project.org/src/contrib/softImpute_1.4.tar.gz, we rewrite it in MATLAB
4https://sites.google.com/site/loysufe/TL12-webcode.zip?attredirects=0&d=1
5https://github.com/quanmingyao/FaNCL
6We implement it on our own.
7https://www.cs.toronto.edu/~rsalakhu/BPMF.html
8https://github.com/udellgroup/Codes-of-FGSR-for-efficient-low-rank-matrix-recovery

5.2 Convergence Guarantee

The proximal algorithm for NNFN regularizer is guaranteed to converge to critical points [2]. As for the non-smooth factored NNFN regularizer, the following guarantee convergence to a critical point of (8), which can be used to form a critical point of the original low-rank matrix completion problem in (1).

THEOREM 5 (CONVERGENCE GUARANTEE). Assume that \( k \) is sufficiently large and \( W(W^\top)^\top \neq 0 \) during the iterations, gradient descent on (8) can converge to a critical point \( (W,H) \). Moreover, the obtained \( X = WH^\top \) is also a critical point of (1), with \( r \) being the NNFN regularizer.
and stepsize is chosen from $[10^{-5}, 1]$. For the other baselines, we use the hyperparameter ranges as mentioned in the respective papers.

6.1.2 Evaluation Metrics. Given an incomplete matrix $Ω$, let $Ω^+$ record positions of the unobserved elements (i.e., $Ω^+_{ij} = 0$ if $Ω_{ij}$ is observed, and 1 otherwise), and $X$ be the matrix recovered. Following [34, 44], performance on the synthetic data is measured by the normalized mean squared error (NMSE) on $Ω^+$: $\text{NMSE} = \|P_{Ω^+}(X - G)∥_F/∥P_{Ω^+}(G)∥_F$, where $G$ is the ground-truth matrix. On the real-world data sets, we use the root mean squared error (RMSE) on $Ω^+$: $\text{RMSE} = (∥P_{Ω^+}(X - O)∥_F^2/∥Ω^+∥_F)$. Besides the error, we also report the training time in seconds.

6.2 Synthetic Data
First, $W, H \in \mathbb{R}^{m \times k}$ are generated with elements sampled i.i.d. from the standard normal distribution $N(0, 1)$. We set $k^* = 5$, and vary $m$ in {500, 1000, 2000}. The $m \times m$ ground-truth matrix (with rank $k^*$) is then constructed as $G = WH^T$. The observed matrix $O$ is generated as $O = G + E$, where the elements of $E$ are sampled from $N(0, 1)$. A set of $∥Ω∥_0 = 2mk^* \log(m)$ random elements in $O$ are observed, where 50% of them are randomly sampled for training, and the rest is taken as validation set for hyperparameter tuning.

We define the sparsity ratio $s$ of the observed matrix as its fraction of observed elements (i.e., $s = ∥Ω∥_0/m^2$).

6.2.1 Performance. Table 4 shows the results. As can be seen, non-convex regularizers (including the proposed $r_2\text{NNFN}$) consistently yield better recovery performance. Among the nonconvex regularizers, all of them yield comparable errors. Additionally, we calculate the rank of recovered matrices and find that all methods (except the nuclear norm regularizer) can recover the true rank. As for speed, factored NNFN allows significantly faster optimization than NNFN, which validates the efficiency of using the factored form. Only factored nuclear norm regularizer is comparable to factored NNFN in speed (but it is much worse in terms of recovery performance), and both are orders of magnitudes faster than the others. Optimization with the truncated $ℓ_{1,2}$ is exceptionally slow, which is due to the need of having two levels of DCA and ADMM iterations. The convergence of testing NMSE is put in Figure 2, which also shows factored NNFN always has the fastest convergence to the lowest NMSE.

6.2.2 Effects of Noise, Rank and Sparsity Ratio. In this section, we vary (i) the variance of the Gaussian noise matrix $E$ in the range $\{0.01, 0.1, 1\}$; (ii) the true rank $k^*$ of the data in $\{5, 10, 20\}$;
Among them, factored NNFN is again the fastest. Figure 4 shows the testing NMSE results and the timing results. As expected, a larger noise, smaller true rank, or sparser matrix lead to a harder matrix completion problem and subsequently higher NMSE's. However, the relative performance ranking of the various methods remain the same, and nonconvex regularization always obtain a smaller NMSE. For time, although the exact timing results vary across different settings, consistent observation can be made: factored NNFN is consistently faster than the others.

6.4 Climate Data

Additionally, we evaluated the proposed method on climate record data sets. The GAS\textsuperscript{11} and USHCN\textsuperscript{12} data sets from [1] are used. GAS contains monthly observations for the green gas components from January 1990 to December 2001, of which we use CO\textsubscript{2} and H\textsubscript{2}. USHCN contains monthly temperature and precipitation readings from January 1919 to November 2019. For these two data sets, some rows (which correspond to locations) of the observed matrix are completely missing. The task is to predict climate observations for locations that do not have any records. Following [1], we normalize the data to zero mean and unit variance, then randomly sample 10% of the locations for training, another 10% for validation, and the rest for testing. To allow generalization to these completely unknown locations, we follow [1] and add a graph Laplacian regularizer to (1). Specifically, the \( m \) locations are represented as nodes on a graph. The affinity matrix \( A = [A_{ij}] \in \mathbb{R}^{m \times m} \), which contains pairwise node similarities, is computed as \( A_{ij} = \exp(-2b(i,j)) \), where \( b(i,j) \) is the Haversine distance between locations \( i \) and \( j \). The graph Laplacian regularizer is then defined as \( a(X) = \text{tr}(X^\top (D - A)X) \), where \( D_{ii} = \sum_j A_{ij} \). For the factored models (factored nuclear norm, BPMF and factored NNFN), we write \( a(X) \) as \( a(WW^\top) \). Additionally, we compare with graph regularized alternating least squares (GRALS) [34], which optimizes for factored nuclear norm with \( a(W) \).

---

\textsuperscript{11}https://viterbi-web.usc.edu/~liu32/data/NA-1990-2002-Monthly.csv
\textsuperscript{12}http://www.ncdc.noaa.gov/oa/climate/research/ushcn

---

Figure 3: Testing NMSE (first row) and clock time (second row) with different settings on the synthetic data (\( m = 1000 \)). The default setting is \( k^* = 5 \), \( E \sim N(0,0.1) \) and \( s = 6.91\% \). For each figure, we only vary one variable while keeping the others as the default setting.

(a) different noise variances. (b) different ranks. (c) different sparsity ratios.

and (iii) the sparsity ratio \( s \) in \( \{0.5, 1, 2\} \times (2mk^* \log(m))/m^2 \). The experiment is performed on the synthetic data set, with \( m = 1000 \). In each trial, we only vary one variable while keeping the others at default, i.e., \( k^* = 5 \), \( E \sim N(0,0.1) \) and \( s = 6.91\% \). Figure 3 shows the testing NMSE results and the timing results. As expected, a larger noise, smaller true rank, or sparser matrix lead to a harder matrix completion problem and subsequently higher NMSE’s. However, the relative performance ranking of the various methods remain the same, and nonconvex regularization always obtain a smaller NMSE. For time, although the exact timing results vary across different settings, consistent observation can be made: factored NNFN is consistently faster than the others.

6.3 Recommendation Data

In this section, experiments are performed on the popularly used benchmark recommendation data sets: MovieLens-1M data set \([17]\) (of size 6,040\times3,449), MovieLens-10M data set\([17]\) (of size 69,878\times10,677) and Yahoo\textsuperscript{10} \([22]\) data set (of size 249,012 \times 296,111). We uniformly sample 50% of the ratings as observed for training, 25% for validation (hyperparameter tuning) and the rest for testing. Optimization with the truncated \( L_{1:2} \) cannot converge in three hours on MovieLens-10M and Yahoo, while Factored GSR runs out of memory on MovieLens-10M and Yahoo as it requires full matrices. Thus, their results are not reported.

Table 5 shows the results. As can be seen, nonconvex regularizers obtain the best recovery performance than the other methods. Among them, factored NNFN is again the fastest. Figure 4 shows convergence of the testing RMSE. Consistent observation can be made, factored NNFN always obtains the best performance while being the fastest.
Table 5: Performance on the recommendation data sets. Entries marked as "-" mean that the corresponding methods cannot complete in three hours. The best and comparable results (according to the pairwise t-test with 95% confidence) are highlighted in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>MovieLens-1M testing RMSE</th>
<th>MovieLens-10M testing RMSE</th>
<th>Yahoo testing RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>time (s)</td>
<td>time (s)</td>
<td>time (s)</td>
</tr>
<tr>
<td>Nuclear</td>
<td>0.820±0.002</td>
<td>0.807±0.001</td>
<td>0.721±0.001</td>
</tr>
<tr>
<td>Factored nuclear</td>
<td>0.810±0.001</td>
<td>0.795±0.001</td>
<td>0.710±0.008</td>
</tr>
<tr>
<td>BPMF</td>
<td>0.807±0.001</td>
<td>0.791±0.001</td>
<td>0.707±0.003</td>
</tr>
<tr>
<td>Factored GSR</td>
<td>0.805±0.001</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Truncated ℓ_1-2</td>
<td>0.797±0.001</td>
<td>6068.4±172.0</td>
<td>-</td>
</tr>
<tr>
<td>Capped-ℓ_1</td>
<td>0.800±0.001</td>
<td>0.787±0.001</td>
<td>0.658±0.001</td>
</tr>
<tr>
<td>LSP</td>
<td>0.799±0.001</td>
<td>0.787±0.001</td>
<td>0.656±0.001</td>
</tr>
<tr>
<td>MCP</td>
<td>0.801±0.001</td>
<td>0.787±0.001</td>
<td>0.678±0.001</td>
</tr>
<tr>
<td>NNFN</td>
<td>0.797±0.001</td>
<td>0.782±0.001</td>
<td>0.652±0.001</td>
</tr>
<tr>
<td>Factored NNFN</td>
<td>0.797±0.001</td>
<td>0.782±0.001</td>
<td>0.652±0.001</td>
</tr>
</tbody>
</table>

Figure 4: Testing RMSE versus clock time on recommendation data.

Table 6: Performance on the climate data sets. The best and comparable results (according to the pairwise t-test with 95% confidence) are highlighted in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>CO2 testing RMSE</th>
<th>CO2 time (s)</th>
<th>H2 testing RMSE</th>
<th>H2 time (s)</th>
<th>Temperature testing RMSE</th>
<th>Temperature time (s)</th>
<th>Precipitation testing RMSE</th>
<th>Precipitation time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nuclear</td>
<td>0.584±0.005</td>
<td>1.0±0.1</td>
<td>0.593±0.006</td>
<td>0.8±0.1</td>
<td>0.480±0.014</td>
<td>108.5±4.1</td>
<td>0.828±0.020</td>
<td>84.9±11.2</td>
</tr>
<tr>
<td>Factored nuclear</td>
<td>0.565±0.006</td>
<td>0.05±0.02</td>
<td>0.574±0.005</td>
<td>0.06±0.03</td>
<td>0.483±0.016</td>
<td>6.0±1.6</td>
<td>0.823±0.018</td>
<td>13.7±1.7</td>
</tr>
<tr>
<td>BPMF</td>
<td>0.552±0.005</td>
<td>3.2±0.3</td>
<td>0.554±0.005</td>
<td>3.4±0.4</td>
<td>0.464±0.012</td>
<td>148.1±7.7</td>
<td>0.819±0.015</td>
<td>125.1±8.1</td>
</tr>
<tr>
<td>GRALS</td>
<td>0.565±0.006</td>
<td>0.7±0.1</td>
<td>0.578±0.005</td>
<td>0.4±0.1</td>
<td>0.498±0.015</td>
<td>37.2±1.9</td>
<td>0.818±0.016</td>
<td>49.6±2.8</td>
</tr>
<tr>
<td>Truncated ℓ_1-2</td>
<td>0.530±0.007</td>
<td>11.0±1.2</td>
<td>0.531±0.005</td>
<td>7.3±1.8</td>
<td>0.444±0.013</td>
<td>573.9±18.1</td>
<td>0.806±0.014</td>
<td>318.5±9.7</td>
</tr>
<tr>
<td>Capped-ℓ_1</td>
<td>0.533±0.003</td>
<td>0.6±0.1</td>
<td>0.531±0.005</td>
<td>0.7±0.2</td>
<td>0.450±0.014</td>
<td>108.5±10.5</td>
<td>0.806±0.014</td>
<td>87.2±6.2</td>
</tr>
<tr>
<td>LSP</td>
<td>0.537±0.008</td>
<td>1.2±0.1</td>
<td>0.540±0.007</td>
<td>1.3±0.2</td>
<td>0.448±0.010</td>
<td>133.3±7.7</td>
<td>0.806±0.014</td>
<td>105.8±7.4</td>
</tr>
<tr>
<td>MCP</td>
<td>0.530±0.008</td>
<td>1.0±0.1</td>
<td>0.534±0.006</td>
<td>0.5±0.1</td>
<td>0.444±0.012</td>
<td>92.5±6.1</td>
<td>0.806±0.014</td>
<td>85.2±7.3</td>
</tr>
<tr>
<td>NNFN</td>
<td>0.530±0.008</td>
<td>0.4±0.1</td>
<td>0.531±0.005</td>
<td>0.5±0.1</td>
<td>0.444±0.012</td>
<td>57.1±3.3</td>
<td>0.806±0.014</td>
<td>66.4±5.1</td>
</tr>
<tr>
<td>Factored NNFN</td>
<td>0.530±0.006</td>
<td>0.05±0.01</td>
<td>0.531±0.005</td>
<td>0.05±0.02</td>
<td>0.444±0.012</td>
<td>5.9±1.4</td>
<td>0.806±0.015</td>
<td>13.5±1.9</td>
</tr>
</tbody>
</table>
Results are shown in Table 6. They are consistent with the observations in the previous experiments. In terms of recovery performance, all the nonconvex regularizers (including NNFN and factored NNFN) have comparable performance and obtain the lowest testing RMSE. In terms of speed, factored NNFN and factored nuclear norm are again the fastest, and this speed advantage is particularly apparent on the larger USHCN data set. Figure 5 shows the convergence of RMSE. As shown, nonconvex regularizers generally obtain better testing RMSEs. Among them, factored NNFN is the fastest in convergence. This again validates the efficiency and effectiveness of factored NNFN.

7 CONCLUSION

We propose a scalable, adaptive and sound nonconvex regularizer for low-rank matrix learning. This regularizer can adaptively penalize singular values as common nonconvex regularizers. Further, we discover that learning with its factored form can be optimized by general solvers such as gradient-based method. We provide theoretical analysis for recovery and convergence guarantee. Extensive experiments on matrix completion problem show that the proposed algorithm achieves state-of-the-art recovery performance, while being the fastest among existing low-rank convex / nonconvex regularization and factored regularization methods. In sum, the proposed method can be useful to solve many large-scale matrix learning problems in the real world.

REFERENCES


The second point: The optimal of (12) p satisfies
\[ p - z + \lambda \hat{r}(p) = 0. \]  
(13)
As \( \hat{r} \) is concave, for \( z_1 \geq z_2 \), we have
\[ \hat{r}(p_1) \leq \hat{r}(p_2). \]  
(14)
Combining (14) and (13), we get \( z_1 - p_1 \geq z_2 - p_2 \), and the second point is proved. \( \Box \)

Now, we can prove Proposition 1.

**Proof.** Let the SVD of \( \mathbf{Z} \) be \( \mathbf{U} \text{Diag}(\sigma(\mathbf{Z})) \mathbf{V}^T \). From Theorem 1 in [26,2], we have \( X \equiv \text{proj}_J(\mathbf{Z}) = \mathbf{U} \text{Diag}(\sigma(\mathbf{Z})) \mathbf{V}^T \), where \( \hat{\sigma} = \text{proj}_J(\sigma(\mathbf{Z})) \). Then, as every singular value \( \sigma_i(\mathbf{Z}) \) is nonnegative, by Lemma 6, we get the conclusion. Note that since \( \hat{r} \) is not linear, thus the strict inequality in \( \sigma_i(\mathbf{Z}) - \hat{\sigma}_i \leq \sigma_{i+1}(\mathbf{Z}) - \hat{\sigma}_{i+1} \) holds at least for one \( i \). \( \Box \)

### A.2 Proposition 2

**Proof.** Let \( X = \mathbf{U} \text{Diag}(\sigma) \mathbf{V}^T \) and \( Z = \mathbf{U} \text{Diag}(\sigma(\mathbf{Z})) \mathbf{V}^T \) be the SVD decomposition of \( X \) and \( Z \). By simple expansion, we have
\[ \frac{1}{2} \| X - \mathbf{P} \|_F^2 + \lambda \nu_{\text{NN}}(X) \]
\[ = \frac{1}{2} \text{tr}(X^T X + Z^T Z - 2X^T Z) + \lambda \| \| \mathbf{X} \|_p - 0 \|_{F} \]
\[ = \frac{1}{2} \| \| \sigma \|_2^2 + |\sigma(Z)|_2^2 \| - \text{tr}(X^T Z) + \lambda \| \| \sigma \|_1 - \lambda \| \sigma \|_2 \| \]
Recall that \( \text{tr}(X^T Z) \leq \sigma^*_T \sigma(Z) \) achieves its equality at \( \mathbf{U} = \bar{\mathbf{U}}, \mathbf{V} = \mathbf{V} \) [21]. Then solving (7) can be instead computed by solving \( \hat{\sigma} \) as
\[ \arg \min_{\hat{\sigma}} \frac{1}{2} \| \| \sigma - \sigma(Z) \|_2^2 + \lambda \| \| \sigma \|_1 - 0 \| \sigma \|_2 \| \]
\[ \text{s.t. } \hat{\sigma}_1 \geq \hat{\sigma}_2 \geq \cdots \geq \hat{\sigma}_m \geq 0. \]  
(15)
It can be solved by proximal operator as \( \hat{\sigma} = \text{proj}_{\lambda \| \cdot \|_1} (\sigma(Z)) \). The constraint in (15) is naturally satisfied. As \( \sigma_i(\mathbf{Z}) \geq 0 \) and \( \sigma_i(\mathbf{Z}) \geq \sigma_{i+1}(\mathbf{Z}) \forall i \), we must have \( \sigma_i \geq \sigma_0 \) and \( \sigma_i \geq \sigma_{i+1} \forall i \). Otherwise, we can always swap the sign or value of \( \sigma_i \) and \( \sigma_{i+1} \) and obtain a smaller objective of (15). \( \Box \)

### A.3 Corollary 3

**Proof.** Let \( X = \mathbf{U} \text{Diag}(\sigma) \mathbf{V}^T \) where \( \hat{\sigma} = \text{proj}_{\lambda \| \cdot \|_2} (\sigma(Z)) \). Now we prove that
\begin{itemize}
  \item shrinkage: \( \sigma_i(\mathbf{Z}) \geq \hat{\sigma}_i \),
  \item adaptivity: \( \sigma_i(Z) - \hat{\sigma}_i \leq \sigma_{i+1}(Z) - \hat{\sigma}_{i+1} \), where the strict inequality holds at least for one \( i \).
\end{itemize}
The first point: From Proposition 2, we can see the optimization problem on matrix (7) can be transformed to an optimization problem on singular values (15). As shown in Proposition 2, for \( \sigma = \text{proj}_{\lambda \| \cdot \|_2}(\sigma(Z)) \), we have \( \sigma_i(Z) \geq \hat{\sigma}_i \geq 0 \).
The second point: The optimal of (15) satisfies
\[ \hat{\sigma} - \sigma(Z) + \lambda - \frac{\lambda}{\| \sigma \|_2} = 0. \]
As \( \sigma_i(Z) \geq \hat{\sigma}_i \geq 0 \), we have
\[ \sigma_i(Z) - \hat{\sigma}_i = \lambda - \frac{\lambda}{\| \sigma \|_2} \geq 0. \]
Then as $\delta_i \geq \delta_{i+1}$, we have
\[
\lambda - \lambda \frac{\delta_i}{\|e\|^2_2} \leq \lambda - \lambda \frac{\delta_{i+1}}{\|e\|^2_2},
\]
and correspondingly $\sigma_i(Z) - \delta_i \leq \sigma_{i+1}(Z) - \delta_{i+1}$. The inequality holds only when $\sigma_i(Z) \neq \sigma_{i+1}(Z)$. □

### A.4 Theorem 4

Proof. The regularized and constrained low-rank matrix completion problem obtain equivalent solutions [4]. Here, we prove for the constrained problem, but the conclusion applies for both forms.

Assume sequence $(X^t)$ with $f(X^{t+1}) < f(X^t)$ and each $X^t$ is the iterate obtained by optimizing the following two equivalent constrained formulations of (8): (i) $X^t$ is the iterate of optimizing $\min_{X} f(X)$ s.t. $\|X\|_F \leq \beta^t$, where $\beta^t \geq 0$ is a hyperparameter, or (ii) $X^t = W^t (H^t)^\top$ is the iterate of optimizing $\min_{W,H} f(WH^\top)$ s.t. $\frac{1}{2} (\|W\|_F^2 + \|H\|_F^2) - \|WH^\top\| \leq \beta^t$, where $\beta^t \geq 0$ is a hyperparameter. These sequences can be obtained by optimizing the two constrained problems via projected gradient descent which guarantees sufficient decrease in $f$ [4, 31].

Obviously, the optimal $X^*$ satisfies $b - A(X^*) = 0$ and hence $f(X^*) = \frac{1}{2} \|A(X^*) - b\|^2_2 = \frac{1}{2} \|e\|^2_2$. Thus $X^t$ obtained at the $t$th iteration satisfies $f(X^t) \geq \frac{c_1 \|e\|^2_2}{2} \geq \frac{\|e\|^2_2}{2}$ for constant $c_1$ whose absolute value is larger than 1. By choosing dimension $t$ of $\mathbf{W} \in \mathbb{R}^{m \times k}$, one can let $X^t \leq k^*$, where $k^*$ is the true rank of the optimal matrix $X^*$.

We can derive
\[
\|A(X^* - X^t)\|^2_2 \leq \|b - A(X^*)\|_2^2 - \frac{1}{2} \|A(X^* - X^t)\|_2^2 \leq \frac{1}{2} \left( f(X^t) - e^\top (b - A(X^*)) + \frac{\|e\|^2_2}{2} \right),
\]
\[
\leq \frac{1}{2} \left( f(X^t) + \frac{2}{c_1} f(X^t) + \frac{1}{c_1} f(X^t) \right),
\]
\[
\leq \frac{1}{2} \left( 1 + \frac{1}{c_1} \right)^2 f(X^t)
\]
Now, we are ready to bound the difference between this $X^t$ and the optimal $X^*$.
\[
\|X^t - X^*\|^2_F \leq \frac{1}{2} \frac{1}{1 - \delta_{2k^*}} \|A(X^* - X^t)\|^2_2 \geq \frac{2}{2 - \delta_{2k^*}} \left( 1 + \frac{1}{c_1} \right)^2 f(X^t) \geq \frac{1}{2 - \delta_{2k^*}} \left( 1 + \frac{1}{c_1} \right)^2 (c_1^2 + e) \|e\|^2_2
\]
\[
= \frac{(c_1 + 1)^2 (c_1^2 + e) \|e\|^2_2}{c_1^2 (1 - \delta_{2k^*})},
\]
where the isometry constant is $\delta_{2k^*}$ as $X^t - X^*$ is a matrix of rank at most $2k^*$, (17) is derived from RIP, (18) comes from (16), and (19) is obtained as one can choose a small constant $\epsilon$ such that $\frac{(c_1^2 + \epsilon) \|e\|^2_2}{2} \geq f(X^t) \geq \frac{c_1^2 \|e\|^2_2}{2}$.

### A.5 Theorem 5

Proof. For smooth functions, gradient descent can obtain sufficient decrease as shown in the following Proposition.

**Proposition 7** ([31]). A differentiable function $h$ with $L$-Lipschitz continuous gradient, i.e., $\|\nabla_x h(x^t) - \nabla_x h(x^{t+1})\|_2 \leq L \|x^t - x^{t+1}\|_2$, satisfies the following inequality,
\[
h(x^t) - h(x^{t+1}) \geq \frac{1}{2L} \|\nabla_x h(x^t)\|^2_F.
\]
Moreover, when $h$ is bounded from below, i.e., $\inf_{x} h(x) > -\infty$ and $\lim_{\|x\|_2 \to \infty} h(x) = \infty$, optimizing $h$ by gradient descent is guaranteed to converge.

Since $WH^\top$ is also $F(W, H)$ is smooth. As gradient descent is used, we then have
\[
F(W^t, H^t) - F(W^{t+1}, H^{t+1}) \geq \frac{\eta}{2} \|\nabla_W F(W^t, H^t)\|_F^2 + \frac{\eta}{2} \|\nabla_H F(W^t, H^t)\|_F^2.
\]
At the $(T + 1)$th iteration, the difference between $F(W^1, H^1)$ and $F(W^{T+1}, H^{T+1})$ is calculated as
\[
F(W^1, H^1) - F(W^{T+1}, H^{T+1}) \geq \sum_{t=1}^T \left( \frac{\eta}{2} \|\nabla_W F(W^t, H^t)\|_F^2 + \frac{\eta}{2} \|\nabla_H F(W^t, H^t)\|_F^2 \right).
\]
As assumed, $\lim_{\|W\|_F \to \infty} F(W, \cdot) = \infty$, $\lim_{\|H\|_F \to \infty} F(\cdot, H) = \infty$. Thus $\infty > F(W^1, H^1) - F(W^{T+1}, H^{T+1}) \geq e$, where $e$ is a finite constant. Combining this with (20), when $T \to \infty$, we see that a sum of infinite sequence is smaller than a finite constant. This means the sequence \{$W^t, H^t$\} has limit points. Let $(\bar{W}, \bar{H})$ be a limit point, we must have
\[
\nabla_W F(\bar{W}, H) = 0 \text{ and } \nabla_H F(W, \bar{H}) = 0.
\]
By definition, this shows $\{\bar{W}, \bar{H}\}$ is a critical point of (8).

Next, we proceed to prove that $X = \bar{W} \bar{H}^\top$ is the critical point of (1) with $r(X) = r_{\text{NNMF}}(X)$.

As shown in [39], the nuclear norm can be reformulated in terms of factorized matrices. Then we have
\[
\min_{W,H} f(WH^\top) - \lambda \|WH^\top\|_F + \frac{\lambda}{2} (\|W\|_F^2 + \|H\|_F^2)
\]
\[
\geq \min_X f(X) - \lambda \|X\|_F + \min_{X=WH} \frac{\lambda}{2} (\|W\|_F^2 + \|H\|_F^2)
\]
\[
\geq \min_X f(X) - \lambda \|X\|_F + \lambda \|X\|_*.
\]
Thus, if $(\bar{W}, \bar{H})$ is a critical point of (8), then $X = \bar{W} \bar{H}^\top$ is also critical point of (1) with $r(X) = r_{\text{NNMF}}(X)$. □