Generalized Nonconvex Nonsmooth Low-Rank Minimization

Canyi Lu[1], Jinhui Tang[2], Shuicheng Yan[1], Zhouchen Lin[3,*]
[1] Department of Electrical and Computer Engineering, National University of Singapore
[2] School of Computer Science, Nanjing University of Science and Technology
[3] Key Laboratory of Machine Perception (MOE), School of EECS, Peking University
canyilu@gmail.com, jinhuitang@mail.njust.edu.cn, eleyans@nus.edu.sg, zlin@pku.edu.cn

Abstract

As surrogate functions of L_0-norm, many nonconvex penalty functions have been proposed to enhance the sparse vector recovery. It is easy to extend these nonconvex penalty functions on singular values of a matrix to enhance low-rank matrix recovery. However, different from convex optimization, solving the nonconvex low-rank minimization problem is much more challenging than the nonconvex sparse minimization problem. We observe that all the existing nonconvex penalty functions are concave and monotonically increasing on $[0,\infty)$. Thus their gradients are decreasing functions. Based on this property, we propose an Iteratively Reweighted Nuclear Norm (IRNN) algorithm to solve the nonconvex nonsmooth low-rank minimization problem. IRNN iteratively solves a Weighted Singular Value Thresholding (WSVT) problem. By setting the weight vector as the gradient of the concave penalty function, the WSVT problem has a closed form solution. In theory, we prove that IRNN decreases the objective function value monotonically, and any limit point is a stationary point. Extensive experiments on both synthetic data and real images demonstrate that IRNN enhances the low-rank matrix recovery compared with state-of-the-art convex algorithms.

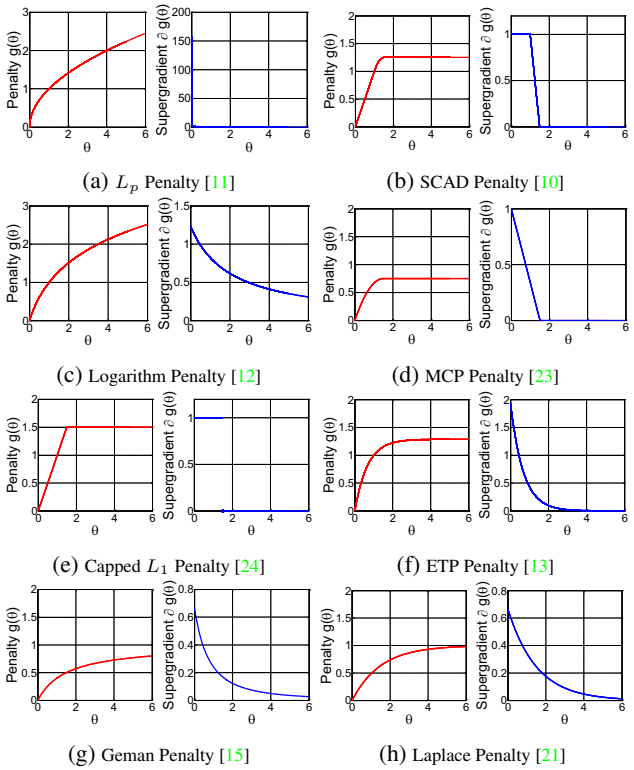
1. Introduction

This paper aims to solve the following general nonconvex nonsmooth low-rank minimization problem

$$\min_{\mathbf{X}\in\mathbb{R}^{m\times n}} F(\mathbf{X}) = \sum_{i=1}^{m} g_\lambda(\sigma_i(\mathbf{X})) + f(\mathbf{X}), \qquad (1)$$

where $\sigma_i(\mathbf{X})$ denotes the i-th singular value of $\mathbf{X}\in\mathbb{R}^{m\times n}$ (we assume $m\le n$ in this work). The penalty function g_λ and loss function f satisfy the following assumptions:

*Corresponding author.



(a) L_p Penalty [11] (b) SCAD Penalty [10]
(c) Logarithm Penalty [12] (d) MCP Penalty [23]
(e) Capped L_1 Penalty [24] (f) ETP Penalty [13]
(g) Geman Penalty [15] (h) Laplace Penalty [21]

Figure 1: Illustration of the popular nonconvex surrogate functions of $||\theta||_0$ (left), and their supergradients (right). All these penalty functions share the common properties: concave and monotonically increasing on $[0,\infty)$. Thus their supergradients (see Section 2.1) are nonnegative and monotonically decreasing. Our proposed general solver is based on this key observation.

A1 $g_\lambda : \mathbb{R}\to\mathbb{R}^+$ is continuous, concave and monotonically increasing on $[0,\infty)$. It is possibly nonsmooth.

A2 f: $\mathbb{R}^{m\times n}\to\mathbb{R}^+$ is a smooth function of type $C^{1,1}$, i.e., the gradient is Lipschitz continuous,

$$||\nabla f(\mathbf{X}) - \nabla f(\mathbf{Y})||_F \le L(f)||\mathbf{X}-\mathbf{Y}||_F, \qquad (2)$$

for any $\mathbf{X},\mathbf{Y}\in\mathbb{R}^{m\times n}$, $L(f)>0$ is called Lipschitz

1063-6919/14 $31.00 © 2014 IEEE
DOI 10.1109/CVPR.2014.526

Table 1: Popular nonconvex surrogate functions of $||\theta||_0$ and their supergradients.

Penalty	Formula $g_\lambda(\theta), \theta \geq 0, \lambda > 0$	Supergradient $\partial g_\lambda(\theta)$
L_p [11]	$\lambda\theta^p$	$\begin{cases}\infty, & \text{if } \theta = 0,\\ \lambda p\theta^{p-1}, & \text{if } \theta > 0.\end{cases}$
SCAD [10]	$\begin{cases}\lambda\theta, & \text{if } \theta \leq \lambda,\\ \frac{-\theta^2+2\gamma\lambda\theta-\lambda^2}{2(\gamma-1)}, & \text{if } \lambda < \theta \leq \gamma\lambda,\\ \frac{\lambda^2(\gamma+1)}{2}, & \text{if } \theta > \gamma\lambda.\end{cases}$	$\begin{cases}\lambda, & \text{if } \theta \leq \lambda,\\ \frac{\gamma\lambda-\theta}{\gamma-1}, & \text{if } \lambda < \theta \leq \gamma\lambda,\\ 0, & \text{if } \theta > \gamma\lambda.\end{cases}$
Logarithm [12]	$\frac{\lambda}{\log(\gamma+1)}\log(\gamma\theta+1)$	$\frac{\gamma\lambda}{(\gamma\theta+1)\log(\gamma+1)}$
MCP [23]	$\begin{cases}\lambda\theta - \frac{\theta^2}{2\gamma}, & \text{if } \theta < \gamma\lambda,\\ \frac{1}{2}\gamma\lambda^2, & \text{if } \theta \geq \gamma\lambda.\end{cases}$	$\begin{cases}\lambda - \frac{\theta}{\gamma}, & \text{if } \theta < \gamma\lambda,\\ 0, & \text{if } \theta \geq \gamma\lambda.\end{cases}$
Capped L_1 [24]	$\begin{cases}\lambda\theta, & \text{if } \theta < \gamma,\\ \lambda\gamma, & \text{if } \theta \geq \gamma.\end{cases}$	$\begin{cases}\lambda, & \text{if } \theta < \gamma,\\ [0,\lambda], & \text{if } \theta = \gamma,\\ 0, & \text{if } \theta > \gamma.\end{cases}$
ETP [13]	$\frac{\lambda}{1-\exp(-\gamma)}(1-\exp(-\gamma\theta))$	$\frac{\lambda\gamma}{1-\exp(-\gamma)}\exp(-\gamma\theta)$
Geman [15]	$\frac{\lambda\theta}{\theta+\gamma}$	$\frac{\lambda\gamma}{(\theta+\gamma)^2}$
Laplace [21]	$\lambda(1-\exp(-\frac{\theta}{\gamma}))$	$\frac{\lambda}{\gamma}\exp(-\frac{\theta}{\gamma})$

constant of ∇f. $f(\mathbf{X})$ is possibly nonconvex.

A3 $F(\mathbf{X}) \to \infty$ iff $||\mathbf{X}||_F \to \infty$.

Many optimization problems in machine learning and computer vision areas fall into the formulation in (1). As for the choice of f, the squared loss $f(\mathbf{X}) = \frac{1}{2}||\mathcal{A}(\mathbf{X}) - \mathbf{b}||_F^2$, with a linear mapping $\mathcal{A}$, is widely used. In this case, the Lipschitz constant of ∇f is then the spectral radius of $\mathcal{A}^*\mathcal{A}$, i.e., $L(f) = \rho(\mathcal{A}^*\mathcal{A})$, where $\mathcal{A}^*$ is the adjoint operator of $\mathcal{A}$. By choosing $g_\lambda(x) = \lambda x$, $\sum_{i=1}^m g_\lambda(\sigma_i(\mathbf{X}))$ is exactly the nuclear norm $\lambda\sum_{i=1}^m \sigma_i(\mathbf{X}) = \lambda||\mathbf{X}||_*$. Problem (1) resorts to the well known nuclear norm regularized problem

$$\min_{\mathbf{X}} \lambda||\mathbf{X}||_* + f(\mathbf{X}). \tag{3}$$

If $f(\mathbf{X})$ is convex, it is the most widely used convex relaxation of the rank minimization problem:

$$\min_{\mathbf{X}} \lambda\text{rank}(\mathbf{X}) + f(\mathbf{X}). \tag{4}$$

The above low-rank minimization problem arises in many machine learning tasks such as multiple category classification [1], matrix completion [20], multi-task learning [2], and low-rank representation with squared loss for subspace segmentation [18]. However, solving problem (4) is usually difficult, or even NP-hard. Most previous works solve the convex problem (3) instead. It has been proved that under certain incoherence assumptions on the singular values of the matrix, solving the convex nuclear norm regularized problem leads to a near optimal low-rank solution [6]. However, such assumptions may be violated in real applications. The obtained solution by using nuclear norm may be suboptimal since it is not a perfect approximation of the rank function. A similar phenomenon has been observed in the convex L_1-norm and nonconvex L_0-norm for sparse vector recovery [7].

In order to achieve a better approximation of the L_0-norm, many nonconvex surrogate functions of L_0-norm have been proposed, including L_p-norm [11], Smoothly Clipped Absolute Deviation (SCAD) [10], Logarithm [12], Minimax Concave Penalty (MCP) [23], Capped L_1 [24], Exponential-Type Penalty (ETP) [13], Geman [15], and Laplace [21]. Table 1 tabulates these penalty functions and Figure 1 visualizes them. One may refer to [14] for more properties of these penalty functions. Some of these nonconvex penalties have been extended to approximate the rank function, e.g. the Schatten-p norm [19]. Another nonconvex surrogate of rank function is the truncated nuclear norm [16].

For nonconvex sparse minimization, several algorithms have been proposed to solve the problem with a nonconvex regularizer. A common method is DC (Difference of Convex functions) programming [14]. It minimizes the nonconvex function $f(\mathbf{x}) - (-g_\lambda(\mathbf{x}))$ based on the assumption that both f and $-g_\lambda$ are convex. In each iteration, DC programming linearizes $-g_\lambda(\mathbf{x})$ at $\mathbf{x} = \mathbf{x}^k$, and minimizes the relaxed function as follows

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} f(\mathbf{x}) - (-g_\lambda(\mathbf{x}^k)) - \left\langle \mathbf{v}^k, \mathbf{x} - \mathbf{x}^k \right\rangle, \tag{5}$$

where $\mathbf{v}^k$ is a subgradient of $-g_\lambda(\mathbf{x})$ at $\mathbf{x} = \mathbf{x}_k$. DC programming may be not very efficient, since it requires some other iterative algorithm to solve (5). Note that the updating rule (5) of DC programming cannot be extended to solve the low-rank problem (1). The reason is that for concave g_λ, $-\sum_{i=1}^m g_\lambda(\sigma_i(\mathbf{X}))$ does not guarantee to be convex w.r.t. $\mathbf{X}$. DC programming also fails when f is nonconvex in problem (1).

Another solver is to use the proximal gradient algorithm which is originally designed for convex problem [3]. It requires computing the proximal operator of g_λ,

$$P_{g_\lambda}(y) = \arg\min_x g_\lambda(x) + \frac{1}{2}(x-y)^2, \tag{6}$$

in each iteration. However, for nonconvex g_λ, there may not exist a general solver for (6). Even if (6) is solvable, differ-

ent from convex optimization, $(P_{g_\lambda}(y_1) - P_{g_\lambda}(y_2))(y_1 - y_2) \geq 0$ does not always hold. Thus we cannot perform $P_{g_\lambda}(\cdot)$ on the singular values of $\mathbf{Y}$ directly for solving

$$P_{g_\lambda}(\mathbf{Y}) = \arg\min_{\mathbf{X}} \sum_{i=1}^{m} g_\lambda(\sigma_i(\mathbf{X})) + ||\,\mathbf{X} - \mathbf{Y}\,||_F^2. \quad (7)$$

The nonconvexity of g_λ makes the nonconvex low-rank minimization problem much more challenging than the nonconvex sparse minimization.

Another related work is the Iteratively Reweighted Least Squares (IRLS) algorihtm. It has been recently extended to handle the nonconvex Schatten-p norm penalty [19]. Actually it solves a relaxed smooth problem which may require many iterations to achieve a low-rank solution. It cannot solve the general nonsmooth problem (1). The alternative updating algorithm in [16] minimizes the truncated nuclear norm by using a special property of this penalty. It contains two loops, both of which require computing SVD. Thus it is not very efficient. It cannot be extended to solve the general problem (1) either.

In this work, all the existing nonconvex surrogate functions of L_0-norm are extended on the singular values of a matrix to enhance low-rank recovery. In problem (1), g_λ can be any existing nonconvex penalty function shown in Table 1 or any other function which satisfies the assumption (**A1**). We observe that all the existing nonconvex surrogate functions are concave and monotonically increasing on $[0, \infty)$. Thus their gradients (or supergradients at the nonsmooth points) are nonnegative and monotonically decreasing. Based on this key fact, we propose an Iteratively Reweighted Nuclear Norm (IRNN) algorithm to solve problem (1). IRNN computes the proximal operator of the weighted nuclear norm, which has a closed form solution due to the nonnegative and monotonically decreasing supergradients. In theory, we prove that IRNN monotonically decreases the objective function value, and any limit point is a stationary point. **To the best of our knowledge, IRNN is the first work which is able to solve the general problem (1) with convergence guarantee. Note that for nonconvex optmization, it is usually very difficult to prove that an algorithm converges to stationary points**. At last, we test our algorithm with several nonconvex penalty functions on both synthetic data and real image data to show the effectiveness of the proposed algorithm.

2. Nonconvex Nonsmooth Low-Rank Minimization

In this section, we present a general algorithm to solve problem (1). To handle the case that g_λ is nonsmooth, e.g., Capped L_1 penalty, we need the concept of supergradient defined on the concave function.

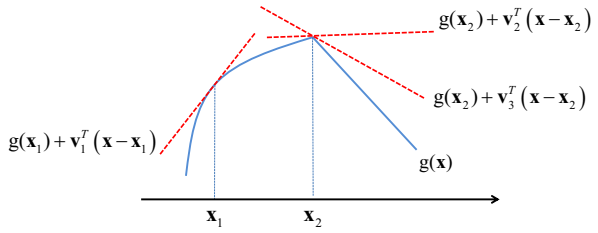


Figure 2: Supergraidients of a concave function. $\mathbf{v}_1$ is a supergradient at $\mathbf{x}_1$, and $\mathbf{v}_2$ and $\mathbf{v}_3$ are supergradients at $\mathbf{x}_2$.

2.1. Supergradient of a Concave Function

The subgradient of the convex function is an extension of gradient at a nonsmooth point. Similarly, the supergradient is an extension of gradient of the concave function at a nonsmooth point. If $g(\mathbf{x})$ is concave and differentiable at $\mathbf{x}$, it is known that

$$g(\mathbf{x}) + \langle \nabla g(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle \geq g(\mathbf{y}). \quad (8)$$

If $g(\mathbf{x})$ is nonsmooth at $\mathbf{x}$, the supergradient extends the gradient at $\mathbf{x}$ inspired by (8) [5].

Definition 1 *Let $g : \mathbb{R}^n \to \mathbb{R}$ be concave. A vector $\mathbf{v}$ is a supergradient of g at the point $\mathbf{x} \in \mathbb{R}^n$ if for every $\mathbf{y} \in \mathbb{R}^n$, the following inequality holds*

$$g(\mathbf{x}) + \langle \mathbf{v}, \mathbf{y} - \mathbf{x} \rangle \geq g(\mathbf{y}). \quad (9)$$

All supergradients of g at $\mathbf{x}$ are called the superdifferential of g at $\mathbf{x}$, and are denoted as $\partial g(\mathbf{x})$. If g is differentiable at $\mathbf{x}$, $\nabla g(\mathbf{x})$ is also a supergradient, i.e., $\partial g(\mathbf{x}) = \{\nabla g(\mathbf{x})\}$. Figure 2 illustrates the supergradients of a concave function at both differentiable and nondifferentiable points.

For concave g, $-g$ is convex, and vice versa. From this fact, we have the following relationship between the supergradient of g and the subgradient of $-g$.

Lemma 1 *Let $g(\mathbf{x})$ be concave and $h(\mathbf{x}) = -g(\mathbf{x})$. For any $\mathbf{v} \in \partial g(\mathbf{x})$, $\mathbf{u} = -\mathbf{v} \in \partial h(\mathbf{x})$, and vice versa.*

The relationship of the supergradient and subgradient shown in Lemma 1 is useful for exploring some properties of the supergradient. It is known that the subdiffierential of a convex function h is a monotone operator, i.e.,

$$\langle \mathbf{u} - \mathbf{v}, \mathbf{x} - \mathbf{y} \rangle \geq 0, \quad (10)$$

for any $\mathbf{u} \in \partial h(\mathbf{x})$, $\mathbf{v} \in \partial h(\mathbf{y})$. The superdifferential of a concave function holds a similar property, which is called antimonotone operator in this work.

Lemma 2 *The superdifferential of a concave function g is an antimonotone operator, i.e.,*

$$\langle \mathbf{u} - \mathbf{v}, \mathbf{x} - \mathbf{y} \rangle \leq 0, \quad (11)$$

for any $\mathbf{u} \in \partial g(\mathbf{x})$, $\mathbf{v} \in \partial g(\mathbf{y})$.

This can be easily proved by Lemma 1 and (10).

Lemma 2 is a key lemma in this work. Supposing that the assumption (**A1**) holds for $g(x)$, (11) indicates that

$$u \geq v, \text{ for any } u \in \partial g(x) \text{ and } v \in \partial g(y), \tag{12}$$

when $x \leq y$. That is to say, the supergradient of g is monotonically decreasing on $[0, \infty)$. Table 1 shows some usual concave functions and their supergradients. We also visualize them in Figure 1. It can be seen that they all satisfy the assumption (**A1**). Note that for the L_p penalty, we further define that $\partial g(0) = \infty$. This will not affect our algorithm and convergence analysis as shown latter. The Capped L_1 penalty is nonsmooth at $\theta = \gamma$, with the superdifferential $\partial g_\lambda(\gamma) = [0, \lambda]$.

2.2. Iteratively Reweighted Nuclear Norm

In this subsection, we show how to solve the general nonconvex and possibly nonsmooth problem (1) based on the assumptions (**A1**)-(**A2**). For simplicity of notation, we denote $\sigma_i = \sigma_i(\mathbf{X})$ and $\sigma_i^k = \sigma_i(\mathbf{X}^k)$.

Since g_λ is concave on $[0, \infty)$, by the definition of the supergradient, we have

$$g_\lambda(\sigma_i) \leq g_\lambda(\sigma_i^k) + w_i^k(\sigma_i - \sigma_i^k), \tag{13}$$

where

$$w_i^k \in \partial g_\lambda(\sigma_i^k). \tag{14}$$

Since $\sigma_1^k \geq \sigma_2^k \geq \cdots \geq \sigma_m^k \geq 0$, by the antimonotone property of supergradient (12), we have

$$0 \leq w_1^k \leq w_2^k \leq \cdots \leq w_m^k. \tag{15}$$

This property is important in our algorithm shown latter. (13) motivates us to minimize its right hand side instead of $g_\lambda(\sigma_i)$. Thus we may solve the following relaxed problem

$$\begin{aligned} \mathbf{X}^{k+1} &= \arg\min_{\mathbf{X}} \sum_{i=1}^m g_\lambda(\sigma_i^k) + w_i^k(\sigma_i - \sigma_i^k) + f(\mathbf{X}) \\ &= \arg\min_{\mathbf{X}} \sum_{i=1}^m w_i^k \sigma_i + f(\mathbf{X}). \end{aligned} \tag{16}$$

It seems that updating $\mathbf{X}^{k+1}$ by solving the above weighted nuclear norm problem (16) is an extension of the weighted L_1-norm problem in IRL1 algorithm [7] (IRL1 is a special DC programming algorithm). However, the weighted nuclear norm is nonconvex in (16) (it is convex if and only if $w_1^k \geq w_2^k \geq \cdots \geq w_m^k \geq 0$ [8]), while the weighted L_1-norm is convex. Solving the nonconvex problem (16) is much more challenging than the convex weighted L_1-norm problem. In fact, it is not easier than solving the original problem (1).

Algorithm 1 Solving problem (1) by IRNN

Input: $\mu > L(f)$ - A Lipschitz constant of $\nabla f(\mathbf{X})$.
Initialize: $k = 0$, $\mathbf{X}^k$, and w_i^k, $i = 1, \cdots, m$.
Output: X^*.
while not converge **do**

1. Update $\mathbf{X}^{k+1}$ by solving problem (18).
2. Update the weights w_i^{k+1}, $i = 1, \cdots, m$, by

$$w_i^{k+1} \in \partial g_\lambda\left(\sigma_i(\mathbf{X}^{k+1})\right). \tag{17}$$

end while

Instead of updating $\mathbf{X}^{k+1}$ by solving (16), we linearize $f(\mathbf{X})$ at $\mathbf{X}^k$ and add a proximal term:

$$f(\mathbf{X}) \approx f(\mathbf{X}^k) + \langle \nabla f(\mathbf{X}^k), \mathbf{X} - \mathbf{X}^k \rangle + \frac{\mu}{2}||\mathbf{X} - \mathbf{X}^k||_F^2,$$

where $\mu > L(f)$. Such a choice of μ guarantees the convergence of our algorithm as shown latter. Then we update $\mathbf{X}^{k+1}$ by solving

$$\begin{aligned} \mathbf{X}^{k+1} &= \arg\min_{\mathbf{X}} \sum_{i=1}^m w_i^k \sigma_i + f(\mathbf{X}^k) \\ &\quad + \langle \nabla f(\mathbf{X}^k), \mathbf{X} - \mathbf{X}^k \rangle + \frac{\mu}{2}||\mathbf{X} - \mathbf{X}^k||_F^2 \\ &= \arg\min_{\mathbf{X}} \sum_{i=1}^m w_i^k \sigma_i + \frac{\mu}{2}\left\|\mathbf{X} - \left(\mathbf{X}^k - \frac{1}{\mu}\nabla f(\mathbf{X}^k)\right)\right\|_F^2. \end{aligned} \tag{18}$$

Problem (18) is still nonconvex. Fortunately, it has a closed form solution due to (15).

Lemma 3 *[8, Theorem 2.3] For any $\lambda > 0$, $\mathbf{Y} \in \mathbb{R}^{m \times n}$ and $0 \leq w_1 \leq w_2 \leq \cdots \leq w_s$ ($s = \min(m, n)$), a globally optimal solution to the following problem*

$$\min \lambda \sum_{i=1}^s w_i \sigma_i(\mathbf{X}) + \frac{1}{2}||\mathbf{X} - \mathbf{Y}||_F^2, \tag{19}$$

is given by the weighted singular value thresholding

$$\mathbf{X}^* = U \mathcal{S}_{\lambda \boldsymbol{w}}(\mathbf{\Sigma}) V^T, \tag{20}$$

where $\mathbf{Y} = U\mathbf{\Sigma}V^T$ is the SVD of $\mathbf{Y}$, and $\mathcal{S}_{\lambda \boldsymbol{w}}(\mathbf{\Sigma}) = \mathrm{Diag}\{(\mathbf{\Sigma}_{ii} - \lambda w_i)_+\}$.

It is worth mentioning that for the L_p penalty, if $\sigma_i^k = 0$, $w_i^k \in \partial g_\lambda(\sigma_i^k) = \{\infty\}$. By the updating rule of $\mathbf{X}^{k+1}$ in (18), we have $\sigma_i^{k+1} = 0$. This guarantees that the rank of the sequence $\{\mathbf{X}^k\}$ is nonincreasing.

Iteratively updating w_i^k, $i = 1, \cdots, m$, by (14) and $\mathbf{X}^{k+1}$ by (18) leads to the proposed Iteratively Reweighted Nuclear Norm (IRNN) algorithm. The whole procedure of IRNN is shown in Algorithm 1. If the Lipschitz constant $L(f)$ is not known or computable, the backtracking rule can be used to estimate μ in each iteration [3].

3. Convergence Analysis

In this section, we give the convergence analysis for the IRNN algorithm. We will show that IRNN decreases the objective function value monotonically, and any limit point is a stationary point of problem (1). We first recall the following well-known and fundamental property for a smooth function in the class $C^{1,1}$.

Lemma 4 *[4, 3] Let $f : \mathbb{R}^{m\times n} \to \mathbb{R}$ be a continuously differentiable function with Lipschitz continuous gradient and Lipschitz constant $L(f)$. Then, for any $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{m\times n}$, and $\mu \geq L(f)$,*

$$f(\mathbf{X}) \leq f(\mathbf{Y}) + \langle \mathbf{X} - \mathbf{Y}, \nabla f(\mathbf{Y})\rangle + \frac{\mu}{2}||\mathbf{X} - \mathbf{Y}||_F^2. \quad (21)$$

Theorem 1 *Assume that g_λ and f in problem (1) satisfy the assumptions (**A1**)-(**A2**). The sequence $\{\mathbf{X}^k\}$ generated in Algorithm 1 satisfies the following properties:*

(1) $F(\mathbf{X}^k)$ is monotonically decreasing. Indeed,

$$F(\mathbf{X}^k) - F(\mathbf{X}^{k+1}) \geq \frac{\mu - L(f)}{2}||\mathbf{X}^k - \mathbf{X}^{k+1}||_F^2 \geq 0;$$

(2) $\lim_{k\to\infty}(\mathbf{X}^k - \mathbf{X}^{k+1}) = \mathbf{0}$;

(3) The sequence $\{\mathbf{X}^k\}$ is bounded.

Proof. First, since $\mathbf{X}^{k+1}$ is a global solution to problem (18), we get

$$\sum_{i=1}^m w_i^k \sigma_i^{k+1} + \langle \nabla f(\mathbf{X}^k), \mathbf{X}^{k+1} - \mathbf{X}^k\rangle + \frac{\mu}{2}||\mathbf{X}^{k+1} - \mathbf{X}^k||_F^2$$
$$\leq \sum_{i=1}^m w_i^k \sigma_i^k + \langle \nabla f(\mathbf{X}^k), \mathbf{X}^k - \mathbf{X}^k\rangle + \frac{\mu}{2}||\mathbf{X}^k - \mathbf{X}^k||_F^2.$$

It can be rewritten as

$$\begin{aligned}&\langle \nabla f(\mathbf{X}^k), \mathbf{X}^k - \mathbf{X}^{k+1}\rangle \\ &\geq -\sum_{i=1}^m w_i^k(\sigma_i^k - \sigma_i^{k+1}) + \frac{\mu}{2}||\mathbf{X}^k - \mathbf{X}^{k+1}||_F^2.\end{aligned} \quad (22)$$

Second, since the gradient of $f(\mathbf{X})$ is Lipschitz continuous, by using Lemma 4, we have

$$\begin{aligned}&f(\mathbf{X}^k) - f(\mathbf{X}^{k+1}) \\ &\geq \langle \nabla f(\mathbf{X}^k), \mathbf{X}^k - \mathbf{X}^{k+1}\rangle - \frac{L(f)}{2}||\mathbf{X}^k - \mathbf{X}^{k+1}||_F^2.\end{aligned} \quad (23)$$

Third, since $w_i^k \in \partial g_\lambda(\sigma_i^k)$, by the definition of the supergradient, we have

$$g_\lambda(\sigma_i^k) - g_\lambda(\sigma_i^{k+1}) \geq w_i^k(\sigma_i^k - \sigma_i^{k+1}). \quad (24)$$

Now, summing (22), (23) and (24) for $i = 1, \cdots, m$, together, we obtain

$$\begin{aligned}&F(\mathbf{X}^k) - F(\mathbf{X}^{k+1}) \\ &= \sum_{i=1}^m \left(g_\lambda(\sigma_i^k) - g_\lambda(\sigma_i^{k+1})\right) + f(\mathbf{X}^k) - f(\mathbf{X}^{k+1}) \\ &\geq \frac{\mu - L(f)}{2}||\mathbf{X}^{k+1} - \mathbf{X}^k||_F^2 \geq 0.\end{aligned} \quad (25)$$

Thus $F(\mathbf{X}^k)$ is monotonically decreasing. Summing all the inequalities in (25) for $k \geq 1$, we get

$$F(\mathbf{X}^1) \geq \frac{\mu - L(f)}{2}\sum_{k=1}^\infty ||\mathbf{X}^{k+1} - \mathbf{X}^k||_F^2, \quad (26)$$

or equivalently,

$$\sum_{k=1}^\infty ||\mathbf{X}^k - \mathbf{X}^{k+1}||_F^2 \leq \frac{2F(\mathbf{X}^1)}{\mu - L(f)}. \quad (27)$$

In particular, it implies that $\lim_{k\to\infty}(\mathbf{X}^k - \mathbf{X}^{k+1}) = \mathbf{0}$. The boundedness of $\{\mathbf{X}^k\}$ is obtained based on the assumption (**A3**). ■

Theorem 2 *Let $\{\mathbf{X}^k\}$ be the sequence generated in Algorithm 1. Then any accumulation point $\mathbf{X}^*$ of $\{\mathbf{X}^k\}$ is a stationary point of (1).*

Proof. The sequence $\{\mathbf{X}^k\}$ generated in Algorithm 1 is bounded as shown in Theorem 1. Thus there exists a matrix $\mathbf{X}^*$ and a subsequence $\{\mathbf{X}^{k_j}\}$ such that $\lim_{j\to\infty}\mathbf{X}^{k_j} = \mathbf{X}^*$. From the fact that $\lim_{k\to\infty}(\mathbf{X}^k - \mathbf{X}^{k+1}) = \mathbf{0}$ in Theorem 1, we have $\lim_{j\to\infty}\mathbf{X}^{k_j+1} = \mathbf{X}^*$. Thus $\sigma_i(\mathbf{X}^{k_j+1}) \to \sigma_i(\mathbf{X}^*)$ for $i = 1, \cdots, m$. By the choice of $w_i^{k_j} \in \partial g_\lambda(\sigma_i(\mathbf{X}^{k_j}))$ and Lemma 1, we have $-w_i^{k_j} \in \partial\left(-g_\lambda(\sigma_i(\mathbf{X}^{k_j}))\right)$. By the upper semi-continuous property of the subdifferential [9, Proposition 2.1.5], there exists $-w_i^* \in \partial\left(-g_\lambda(\sigma_i(\mathbf{X}^*))\right)$ such that $-w_i^{k_j} \to -w_i^*$. Again by Lemma 1, $w_i^* \in \partial g_\lambda(\sigma_i(\mathbf{X}^*))$ and $w_i^{k_j} \to w_i^*$.

Denote $h(\mathbf{X}, \boldsymbol{w}) = \sum_{i=1}^m w_i\sigma_i(\mathbf{X})$. Since $\mathbf{X}^{k_j+1}$ is optimal to problem (18), there exists $\boldsymbol{G}^{k_j+1} \in \partial h(\mathbf{X}^{k_j+1}, \boldsymbol{w}^{k_j})$, such that

$$\boldsymbol{G}^{k_j+1} + \nabla f(\mathbf{X}^{k_j}) + \mu(\mathbf{X}^{k_j+1} - \mathbf{X}^{k_j}) = \mathbf{0}. \quad (28)$$

Let $j \to \infty$ in (28), there exists $\boldsymbol{G}^* \in \partial h(\mathbf{X}^*, \boldsymbol{w}^*)$, such that

$$\mathbf{0} = \boldsymbol{G}^* + \nabla f(\mathbf{X}^*) \in \partial F(\mathbf{X}^*). \quad (29)$$

Thus $\mathbf{X}^*$ is a stationary point of (1). ■

4. Extension to Other Problems

Our proposed IRNN algorithm can solve a more general low-rank minimization problem as follows,

$$\min_{\mathbf{X}} \sum_{i=1}^{m} g_i(\sigma_i(\mathbf{X})) + f(\mathbf{X}), \tag{30}$$

where g_i, $i = 1, \cdots, m$, are concave, and their supergradients satisfy $0 \leq v_1 \leq v_2 \leq \cdots \leq v_m$, for any $v_i \in \partial g_i(\sigma_i(\mathbf{X}))$, $i = 1, \cdots, m$. The truncated nuclear norm $||\mathbf{X}||_r = \sum_{i=r+1}^{m} \sigma_i(\mathbf{X})$ [16] satisfies the above assumption. Indeed, $||\mathbf{X}||_r = \sum_{i=1}^{m} g_i(\sigma_i(\mathbf{X}))$ by letting

$$g_i(x) = \begin{cases} 0, & i = 1, \cdots, r, \\ x, & i = r+1, \cdots, m. \end{cases} \tag{31}$$

Their supergradients are

$$\partial g_i(x) = \begin{cases} 0, & i = 1, \cdots, r, \\ 1, & i = r+1, \cdots, m. \end{cases} \tag{32}$$

The convergence results in Theorem 1 and 2 also hold since (24) holds for each g_i. Compared with the alternating updating algorithms in [16], which require double loops, our IRNN algorithm will be more efficient and with stronger convergence guarantee.

More generally, IRNN can solve the following problem

$$\min_{\mathbf{X}} \sum_{i=1}^{m} g(h(\sigma_i(\mathbf{X}))) + f(\mathbf{X}), \tag{33}$$

when $g(y)$ is concave, and the following problem

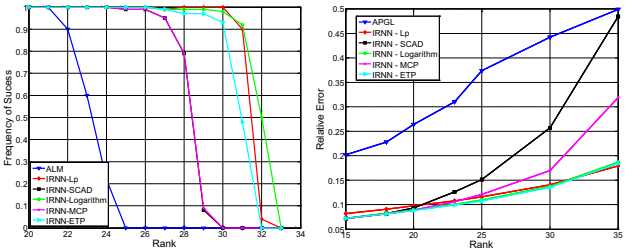
$$\min_{\mathbf{X}} w_i h(\sigma_i(\mathbf{X})) + ||\mathbf{X} - \mathbf{Y}||_F^2, \tag{34}$$

can be cheaply solved. An interesting application of (33) is to extend the group sparsity on the singular values. By dividing the singular values into k groups, i.e., $G_1 = \{1, \cdots, r_1\}$, $G_2 = \{r_1 + 1, \cdots, r_1 + r_2 - 1\}$, $\cdots$, $G_k = \{\sum_i^{k-1} r_i + 1, \cdots, m\}$, where $\sum_i r_i = m$, we can define the group sparsity on the singular values as $||\mathbf{X}||_{2,g} = \sum_{i=1}^{k} g(||\sigma_{G_i}||_2)$. This is exactly the first term in (33) by letting h be the L_2-norm of a vector. g can be nonconvex functions satisfying the assumption (**A1**) or specially the convex absolute function.

5. Experiments

In this section, we present several experiments on both synthetic data and real images to validate the effectiveness of the IRNN algorithm. We test our algorithm on the matrix completion problem

$$\min_{\mathbf{X}} \sum_{i=1}^{m} g_\lambda(\sigma_i(\mathbf{X})) + \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{X} - \mathbf{M})||_F^2, \tag{35}$$



(a) random data without noise (b) random data with noise

Figure 3: Comparison of matrix recovery on (a) random data without noise, and (b) random data with noise.

where Ω is the set of indices of samples, and $\mathcal{P}_\Omega : \mathbb{R}^{m \times n} \to \mathbb{R}^{m \times n}$ is a linear operator that keeps the entries in Ω unchanged and those outside Ω zeros. The gradient of squared loss function in (35) is Lipschitz continuous, with a Lipschitz constant $L(f) = 1$. We set $\mu = 1.1$ in Algorithm 1. For the choice of g_λ, we test all the penalty functions listed in Table 1 except for Capped L_1 and Geman, since we find that their recovery performances are sensitive to the choices of γ and λ in different cases. For the choice of λ in IRNN, we use a continuation technique to enhance the low-rank matrix recovery. The initial value of λ is set to a larger value λ_0, and dynamically decreased by $\lambda = \eta^k \lambda_0$ with $\eta < 1$. It is stopped till reaching a predefined target λ_t. $\mathbf{X}$ is initialized as a zero matrix. For the choice of parameters (e.g., p and γ) in the nonconvex penalty functions, we search it from a candidate set and use the one which obtains good performance in most cases [1].

5.1. Low-Rank Matrix Recovery

We first compare our nonconvex IRNN algorithm with state-of-the-art convex algorithms on synthetic data. We conduct two experiments. One is for the observed matrix $\mathbf{M}$ without noise, and the other one is for $\mathbf{M}$ with noise.

For the noise free case, we generate the rank r matrix $\mathbf{M}$ as $\mathbf{M}_L \mathbf{M}_R$, where $\mathbf{M}_L \in \mathbb{R}^{150 \times r}$, and $\mathbf{M}_R \in \mathbb{R}^{r \times 150}$ are generated by the Matlab command `randn`. 50% elements of $\mathbf{M}$ are missing uniformly at random. We compare our algorithm with Augmented Lagrange Multiplier (ALM) [2] [17] which solves the noise free problem

$$\min_{\mathbf{X}} ||\mathbf{X}||_* \text{ s.t. } \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{M}). \tag{36}$$

For this task, we set $\lambda_0 = ||\mathcal{P}_\Omega(\mathbf{M})||_\infty$, $\lambda_t = 10^{-5}\lambda_0$, and $\eta = 0.7$ in IRNN, and stop the algorithm when $||\mathcal{P}_\Omega(\mathbf{X} - \mathbf{M})||_F \leq 10^{-5}$. For ALM, we use the default parameters in the released codes. We evaluate the recovery performance by the Relative Error defined as $||\hat{\mathbf{X}} -$

[1] Code of IRNN: https://sites.google.com/site/canyilu/.

[2] Code: http://perception.csl.illinois.edu/matrix-rank/sample_code.html.

Figure 4: Comparison of image recovery by using different matrix completion algorithms. (a) Original image. (b) Image with Gaussian noise and text. (c)-(g) Recovered images by APGL, LMaFit, TNNR-ADMM, IRNN-L_p, and IRNN-SCAD, respectively. **Best viewed in $\times 2$ sized color pdf file.**

$\mathbf{M}||_F/||\mathbf{M}||_F$, where $\hat{\mathbf{X}}$ is the recovered solution by a certain algorithm. If the Relative Error is smaller than 10^{-3}, $\hat{\mathbf{X}}$ is regarded as a successful recovery of $\mathbf{M}$. We repeat the experiments 100 times with the underlying rank r varying from 20 to 33 for each algorithm. The frequency of success is plotted in Figure 3a. The legend IRNN-L_p in Figure 3a denotes the L_p penalty function used in problem (1) and solved by our proposed IRNN algorithm. It can be seen that IRNN with all the nonconvex penalty functions achieves much better recovery performance than the convex ALM algorithm. This is because the nonconvex penalty functions approximate the rank function better than the convex nuclear norm.

For the noisy case, the data are generated by $\mathcal{P}_\Omega(\mathbf{M}) = \mathcal{P}_\Omega(\mathbf{M}_L\,\mathbf{M}_R)$+0.1$\times$`randn`. We compare our algorithm with convex Accelerated Proximal Gradient with Line search (APGL) [3] [20] which solves the noisy problem

$$\min_{\mathbf{X}} \lambda||\mathbf{X}||_* + \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{X}) - \mathcal{P}_\Omega(\mathbf{M})||_F^2. \qquad (37)$$

For this task, we set $\lambda_0 = 10||\mathcal{P}_\Omega(\mathbf{M})||_\infty$, and $\lambda_t = 0.1\lambda_0$ in IRNN. All the chosen algorithms are run 100 times with the underlying rank r lying between 15 and 35. The relative errors can be ranging for each test, and the mean errors by different methods are plotted in Figure 3b. It can be seen that IRNN for the nonconvex penalty outperforms the convex APGL for the noisy case. Note that we cannot conclude from Figure 3 that IRNN with L_p, Logarithm and ETP penalty functions always perform better than SCAD and MCP, since the obtained solutions are not globally optimal.

5.2. Application to Image Recovery

In this section, we apply matrix completion for image recovery. As shown in Figure 4, the real image may be corrupted by different types of noises, e.g., Gaussian noise or unrelated text. Usually the real images are not of low-rank, but the top singular values dominate the main information [16]. Thus the corrupted image can be recovered by low-rank approximation. For color images which have three channels, we simply apply matrix completion for each channel independently. The well known Peak Signal-to-Noise Ratio (PSNR) is employed to evaluate the recovery performance. We compare IRNN with some other matrix completion algorithms which have been applied for this task, including APGL, Low-Rank Matrix Fitting (LMaFit) [4]. [22] and Truncated Nuclear Norm Regularization (TNNR) [16]. We use the solver based on ADMM to solve a subproblem of TNNR in the released codes (denoted as TNNR-ADMM) [5]. We try to tune the parameters to be optimal of the chosen algorithms and report the best result.

In our test, we consider two types of noises on the real images. The first one replaces 50% of pixels with random values (sample image (1) in Figure 4 (b)). The other one adds some unrelated texts on the image (sample image (2) in Figure 4 (b)). Figure 4 (c)-(g) show the recovered images by different methods. It can be observed that our IRNN method with different penalty functions achieves much better recovery performance than APGL and LMaFit. Only the results by IRNN-L_p and IRNN-SCAD are plotted due to the limit of space. We further test on more images and plot the results in Figure 5. Figure 6 shows the PSNR values of different methods on all the test images. It can be seen that IRNN with all the evaluated nonconvex functions achieves higher PSNR values, which verifies that the nonconvex penalty functions are effective in this situation. The nonconvex truncated nuclear norm is close to our methods, but its running time is 3$\sim$5 times of that for ours.

6. Conclusions and Future Work

In this work, the nonconvex surrogate functions of L_0-norm are extended on the singular values to approximate

[3] Code: http://www.math.nus.edu.sg/~mattohkc/NNLS.html.

[4] Code: http://lmafit.blogs.rice.edu/.

[5] Code: https://sites.google.com/site/zjuyaohu/.

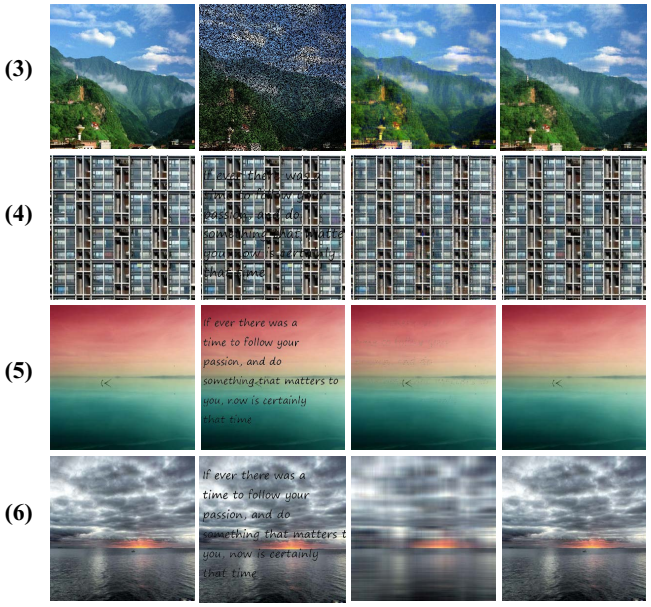


Figure 5: Comparison of image recovery on more images. (a) Original images. (b) Images with noises. Recovered images by (c) APGL, and (d) IRNN-L_p. **Best viewed in $\times 2$ sized color pdf file.**

the rank function. It is observed that all the existing nonconvex surrogate functions are concave and monotonically increasing on $[0, \infty)$. Then a general solver IRNN is proposed to solve problem (1) with such penalties. IRNN is the first algorithm which is able to solve the general nonconvex low-rank minimization problem (1) with convergence guarantee. The nonconvex penalty can be nonsmooth by using the supergradient at the nonsmooth point. In theory, we proved that any limit point is a local minimum. Experiments on both synthetic data and real images demonstrated that IRNN usually outperforms the state-of-the-art convex algorithms. An interesting future work is to solve the nonconvex low-rank minimization problem with affine constraint. A possible way is to combine IRNN with Alternating Direction Method of Multiplier (ADMM).

Acknowledgements

This research is supported by the Singapore National Research Foundation under its International Research Centre @Singapore Funding Initiative and administered by the IDM Programme Office. Z. Lin is supported by NSF of China (Grant nos. 61272341, 61231002, and 61121002) and MSRA.

References

[1] Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *ICML*, 2007.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 2008.

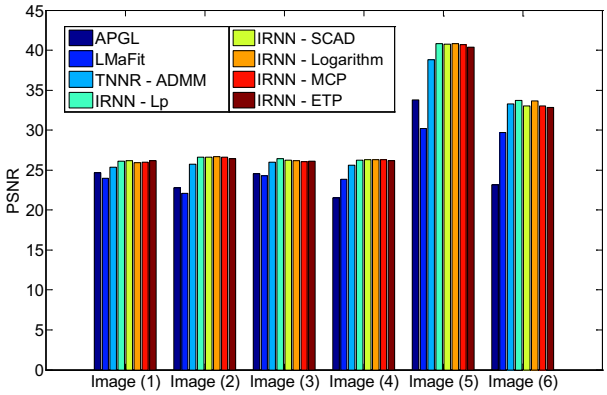


Figure 6: Comparison of the PSNR values by different matrix completion algorithms.

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2009.

[4] D. P. Bertsekas. *Nonlinear programming*. Athena Scientific (Belmont, Mass.), 2nd edition, 1999.

[5] K. Border. The supergradient of a concave function. http://www.hss.caltech.edu/~kcb/Notes/Supergrad.pdf, 2001. [Online].

[6] E. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 2010.

[7] E. Candès, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 2008.

[8] K. Chen, H. Dong, and K. Chan. Reduced rank regression via adaptive nuclear norm penalization. *Biometrika*, 2013.

[9] F. Clarke. Nonsmooth analysis and optimization. In *Proceedings of the International Congress of Mathematicians*, 1983.

[10] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 2001.

[11] L. Frank and J. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 1993.

[12] J. Friedman. Fast sparse regression and classification. *International Journal of Forecasting*, 2012.

[13] C. Gao, N. Wang, Q. Yu, and Z. Zhang. A feasible nonconvex relaxation approach to feature selection. In *AAAI*, 2011.

[14] G. Gasso, A. Rakotomamonjy, and S. Canu. Recovering sparse signals with a certain family of nonconvex penalties and DC programming. *IEEE Transactions on Signal Processing*, 2009.

[15] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *TIP*, 1995.

[16] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He. Fast and accurate matrix completion via truncated nuclear norm regularization. *TPAMI*, 2013.

[17] Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented lagrange multiplier method for exact recovery of a corrupted low-rank matrices. *UIUC Technical Report UILU-ENG-09-2215, Tech. Rep.*, 2009.

[18] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *TPAMI*, 2013.

[19] K. Mohan and M. Fazel. Iterative reweighted algorithms for matrix rank minimization. In *JMLR*, 2012.

[20] K. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 2010.

[21] J. Trzasko and A. Manduca. Highly undersampled magnetic resonance image reconstruction via homotopic ℓ_0-minimization. *TMI*, 2009.

[22] Z. Wen, W. Yin, and Y. Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 2012.

[23] C. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 2010.

[24] T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *JMLR*, 2010.