Journal of Machine Learning Research 18 (2018) 1-52 Submitted 2/17; Revised 11/17; Published 4/18

Efficient Learning with a Family of Nonconvex Regularizers by Redistributing Nonconvexity

Quanming Yao QYAOAA@CSE.UST.HK
James T. Kwok JAMESK@CSE.UST.HK
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Hong Kong

Editor: Mark Schmidt

Abstract

The use of convex regularizers allows for easy optimization, though they often produce biased estimation and inferior prediction performance. Recently, nonconvex regularizers have attracted a lot of attention and outperformed convex ones. However, the resultant optimization problem is much harder. In this paper, a popular subclass of ℓ_1-based nonconvex sparsity-inducing and low-rank regularizers is considered. This includes nonconvex variants of lasso, sparse group lasso, tree-structured lasso, nuclear norm and total variation regularizers. We propose to move the nonconvexity from the regularizer to the loss. The nonconvex regularizer is then transformed to a familiar convex one, while the resultant loss function can still be guaranteed to be smooth. Learning with the convexified regularizer can be performed by existing efficient algorithms originally designed for convex regularizers (such as the proximal algorithm, Frank-Wolfe algorithm, alternating direction method of multipliers and stochastic gradient descent). This is further extended to consider cases where the convexified regularizer does not have a closed-form proximal step, and when the loss function is nonconvex nonsmooth. Extensive experiments on a variety of machine learning application scenarios show that optimizing the transformed problem is much faster than running the state-of-the-art on the original problem.

Keywords: Nonconvex optimization, Nonconvex regularization, Proximal algorithm, Frank-Wolfe algorithm, Matrix completion

1. Introduction

Regularized risk minimization is fundamental to machine learning. It admits a tradeoff between the empirical loss and regularization as:

$$\min_x F(x) \equiv f(x) + g(x), \tag{1}$$

where x is the model parameter, f is the loss and g is the regularizer. The choice of regularizers is important and application-specific, and is often the crux to obtain good prediction performance. Popular examples include the sparsity-inducing regularizers, which have been commonly used in image processing (Beck and Teboulle, 2009; Mairal et al., 2009; Jenatton et al., 2011) and high-dimensional feature selection (Tibshirani et al., 2005; Jacob et al., 2009; Liu and Ye, 2010); and the low-rank regularizers, which have obtained good empirical performance on various matrix and tensor learning tasks such as collaborative filtering (Candès and Recht, 2009; Mazumder et al., 2010) and visual data analysis (Liu et al., 2013; Lu et al., 2014).

©2018 Quanming Yao and James T. Kwok.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/17-078.html.

Most of these regularizers are convex. Well-known examples include the ℓ_1-regularizer for sparse coding (Donoho, 2006), and the nuclear norm regularizer in low-rank matrix learning (Candès and Recht, 2009). Besides having nice theoretical guarantees, convex regularizers also allow easy optimization. Popular optimization algorithms in machine learning include the proximal algorithm (Parikh and Boyd, 2013), Frank-Wolfe (FW) algorithm (Jaggi, 2013), alternating direction method of multipliers (ADMM) (Boyd et al., 2011), stochastic gradient descent (SGD) and its variants (Bottou, 1998; Xiao and Zhang, 2014). Many of these are efficient, scalable, and have sound convergence properties.

However, convex regularizers often lead to biased estimation. For example, in sparse coding, the solution obtained by the ℓ_1-regularizer is often not as sparse and accurate (Zhang, 2010b). In low-rank matrix learning, the estimated rank obtained with the nuclear norm regularizer is often much higher (Mazumder et al., 2010). To alleviate this problem, a number of nonconvex regularizers, which are variants of the convex ℓ_1-norm, have been recently proposed. Examples include the Geman penalty (GP) (Geman and Yang, 1995), log-sum penalty (LSP) (Candès et al., 2008), minimax concave plus (MCP) penalty (Zhang, 2010a), Laplace penalty (Trzasko and Manduca, 2009), and smoothly clipped absolute deviation (SCAD) penalty (Fan and Li, 2001). As can be seen from Figure 1, they are all (i) nonsmooth at zero, which encourage a sparse solution; and (ii) concave, which place a smaller penalty than the ℓ_1-regularizer on features with large magnitudes. Empirically, these nonconvex regularizers usually outperform convex regularizers.

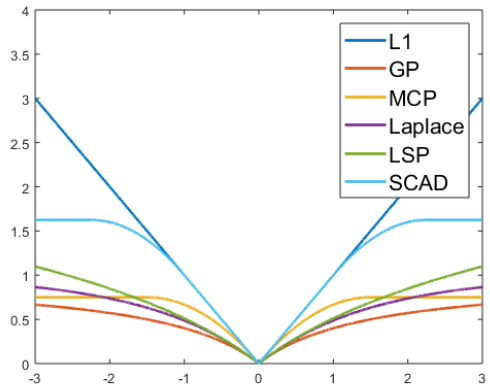


Figure 1: Plot of various nonconvex variants of the convex ℓ_1-regularizer.

In this paper, we consider a popular subclass of ℓ_1-based nonconvex sparsity-inducing and low-rank regularizers such as nonconvex variants of the lasso, sparse group lasso, tree-structured lasso, nuclear norm and total variation regularizers. Even with a convex loss, the resulting nonconvex problem (1) is much harder to optimize. One can use general-purpose nonconvex optimization solvers such as the concave-convex procedure (Yuille and Rangarajan, 2002). Unfortunately, the subproblem in each iteration can be as expensive as the original problem, and the concave-convex procedure is thus often slow in practice (Gong et al., 2013; Zhong and Kwok, 2014).

Recently, the proximal algorithm has been extended for nonconvex problems. Examples include nonconvex inexact proximal splitting (NIPS) (Sra, 2012), inertial proximal algorithm for non-convex optimization (IPiano) (Ochs et al., 2014), unified treatment of accelerated gradient (UAG) (Ghadimi and Lan, 2016), general iterative shrinkage and thresholding (GIST) (Gong et al.,

2013), inertial forward-backward (IFB) algorithm (Bot et al., 2016), and nonmonotone accelerated proximal gradient (nmAPG) (Li and Lin, 2015) algorithm. Specifically, NIPS, IPiano and UAG allow f in (1) to be Lipschitz smooth (and possibly nonconvex) but g has to be convex; while GIST, IFB and nmAPG further allow g to be nonconvex. The current state-of-the-art is nmAPG. Nevertheless, efficient computation of the underlying proximal operator is only possible for simple nonconvex regularizers. When the regularizer is complicated, such as the nonconvex versions of the fused lasso and overlapping group lasso regularizers (Zhong and Kwok, 2014), the corresponding proximal step has to be solved numerically and is again expensive. Another approach is by using the proximal average (Zhong and Kwok, 2014), which computes and averages the proximal step of each underlying regularizer. As the proximal step is only approximate, its convergence is usually slower than typical applications of the proximal algorithm (Yu, 2013; Li and Lin, 2015).

When f is smooth, there are endeavors to extend other algorithms from convex to nonconvex optimization. For the global consensus problem, standard ADMM converges only when g is convex (Hong et al., 2016). When g is nonconvex, convergence of ADMM is only established for problems of the form $\min_{x,y} f(x) + g(y) : y = Ax$, where matrix A has full row rank (Li and Pong, 2015). The convergence of ADMM in more general cases is an open issue. More recently, the stochastic variance reduced gradient (SVRG) algorithm (Xiao and Zhang, 2014), which is a SGD variant with reduced variance in the gradient estimates, has also been extended for problems with nonconvex f, but the regularizer g is still required to be convex (Reddi et al., 2016a; Zhu and Hazan, 2016).

Sometimes, it is desirable to have a nonsmooth loss f. For example, the absolute loss is more robust to outliers than the square loss, and has been popularly used in applications such as image denoising (Yan, 2013), robust dictionary learning (Zhao et al., 2011) and robust PCA (Candès et al., 2011). The resulting optimization problem becomes more challenging. When both f and g are convex, ADMM is often the main optimization tool for problem (1) (He and Yuan, 2012). However, when either f or g is nonconvex, ADMM no longer guarantees convergence. Besides, the loss f may also be nonconvex as this is more robust to outliers and can obtain better performance (e,g., ℓ_0-norm (Yan, 2013) and capped-ℓ_1 norm (Sun et al., 2013)). However, when f is nonsmooth and nonconvex, none of the above-mentioned algorithms (i.e., proximal algorithms, FW algorithms, ADMM, and SVRG) can be used. As a last resort, one can use more general nonconvex optimization approaches such as convex concave programming (CCCP) (Yuille and Rangarajan, 2002), which is slow in general.

In this paper, we first consider the case where the loss function f is smooth (possibly nonconvex) and the regularizer g is nonconvex. We propose to handle nonconvex regularizers by reusing the abundant repository of efficient convex algorithms originally designed for convex regularizers. Motivated by the fact that recent proximal algorithms (Gong et al., 2013; Li and Lin, 2015; Zhong and Kwok, 2014) all rely on the smoothness of f and a simple closed-form proximal step for g, the key is to shift nonconvexity associated with the nonconvex regularizer to the loss function. The nonconvex regularizer is then transformed to a familiar convex regularizer, while the transformed loss function is still smooth. To illustrate the practical usefulness of this convexification scheme, we show how it can be used with popular optimization algorithms in machine learning. For example, for the proximal algorithm, the resultant proximal step can be much easier after transformation. Specifically, for the nonconvex tree-structured lasso and nonconvex sparse group lasso, we show that the corresponding proximal steps have closed-form solutions on the transformed problems, but not on the original ones. For the nonconvex total variation problem, though there is no closed-form solution for the proximal step before and after the transformation, we show that the proximal step

is still cheaper and easier for optimization after the transformation. To allow further speedup, we propose a proximal algorithm variant that allows the use of inexact proximal steps with convex g when it has no closed-form proximal step solution. For the FW algorithm, we consider its application to nonconvex low-rank matrix learning problems, and propose a variant with guaranteed convergence to a critical point of the nonconvex problem. For SVRG in stochastic optimization and ADMM in consensus optimization, we show that these algorithms have convergence guarantees on the transformed problems but not on the original ones.

We further consider the case where f is both nonconvex and nonsmooth (and g is nonconvex). We demonstrate that problem (1) can be similarly transformed to an equivalent problem with a smooth loss and convex regularizer. However, as the proximal step with the transformed regularizer has to be solved numerically and exact proximal step is required, usage with the proximal algorithm may not be efficient. We show that this can be alleviated by the proposed inexact proximal algorithm. Finally, in the experiments, we demonstrate the above-mentioned advantages of optimizing the transformed problems instead of the original ones on various tasks, and show that running algorithms on the transformed problems can be much faster than running the state-of-the-art on the original problems.

The rest of the paper is organized as follows. Section 2 provides a review on the related works. The main idea for problem transformation is presented in Section 3, and its usage with various algorithms are discussed in Section 4. Experimental results are shown in Section 5, and the last section gives some concluding remarks. All the proofs are in Appendix A. Note that this paper extends a shorter version published in the International Conference of Machine Learning (Yao and Kwok, 2016).

Notation: We denote vectors and matrices by lowercase and uppercase boldface letters, respectively. For a vector $x \in \mathbb{R}^d$, $\|x\|_2 = (\sum_{i=1}^d |x_i|^2)^{1/2}$ is its ℓ_2-norm, $\mathrm{Diag}(x)$ returns a diagonal matrix $X \in \mathbb{R}^{d\times d}$ with $X_{ii} = x_i$. For a matrix $X \in \mathbb{R}^{m\times n}$ (where $m \le n$ without loss of generality), its nuclear norm is $\|X\|_* = \sum_{i=1}^m \sigma_i(X)$, where $\sigma_i(X)$'s are the singular values of X, and its Frobenius norm is $\|X\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n X_{ij}^2}$, and $\|X\|_\infty = \max_{i,j} |X_{ij}|$. For a square matrix X, $X \in \mathcal{S}_+$ indicates it is positive semidefinite. For two matrices X and Y, $\langle X, Y\rangle = \sum_{i,j} X_{ij}Y_{ij}$. For a smooth function f, $\nabla f(x)$ is its gradient at x. For a convex but nonsmooth f, $\partial f(x) = \{u : f(y) \ge f(x) + \langle u, y - x\rangle\}$ is its subdifferential at x, and $g \in \partial f(x)$ is a subgradient.

Given a function (e.g., f), a bar on top (e.g., $\bar{f}$) indicates the function is smooth but not necessarily convex, and a breve on top (e.g., $\breve{f}$) means that it is convex but not necessarily smooth.

2. Related Works

In this section, we review some popular algorithms for solving (1). Here, f is assumed to be Lipschitz smooth.

2.1 Convex-Concave Procedure (CCCP)

The convex-concave procedure (CCCP) (Yuille and Rangarajan, 2002; Lu, 2012) is a popular and general solver for (1). It assumes that F can be decomposed as a difference of convex (DC) functions (Hiriart-Urruty, 1985), i.e., $F(x) = \breve{F}(x) + \hat{F}(x)$ where $\breve{F}$ is convex and $\hat{F}$ is concave. In each

CCCP iteration, $\hat{F}$ is linearized at x_t, and x_{t+1} is generated as

$$x_{t+1} = \arg\min_x \breve{F}(x) + \hat{F}(x_t) - (x - x_t)^\top s_t, \tag{2}$$

where $s_t \in \partial(-\hat{F}(x_t))$ is a subgradient. This is a convex problem and can be easier than directly minimizing F.

However, CCCP is expensive as (2) needs to be exactly solved. Sequential convex programming (SCP) (Lu, 2012) improves its efficiency when F is of the form in (1). It assumes that f is L-Lipschitz smooth (possibly nonconvex); while g can be nonconvex, but admits a DC decomposition as $g(x) = \breve{\varsigma}(x) + \hat{\varsigma}(x)$ where $\breve{\varsigma}$ is convex and $\hat{\varsigma}$ is concave. It then generates x_{t+1} as

$$\begin{aligned} x_{t+1} &= \arg\min_x f(x_t) + (x - x_t)^\top \nabla f(x_t) + \frac{L}{2}\|x - x_t\|_2^2 + \breve{\varsigma}(x) + \hat{\varsigma}(x_t) - (x - x_t)^\top s_t \\ &= \arg\min_x \frac{1}{2}\|x - x_t - s_t + \frac{1}{L}\nabla f(x_t)\|_2^2 + \breve{\varsigma}(x), \end{aligned} \tag{3}$$

where $s_t \in \partial\left(-\hat{\varsigma}(x_t)\right)$. When $\breve{\varsigma}$ is a simple function, (3) has a closed-form solution, and SCP can be faster than CCCP. For example, when $\breve{\varsigma}(x) = \|x\|_1$, (3) is the proximal step of the ℓ_1-norm and has a closed-form solution (Tibshirani, 1996). However, convergence of SCP is still slow in general (Gong et al., 2013; Zhong and Kwok, 2014; Li and Lin, 2015).

2.2 Proximal Algorithm

The proximal algorithm (Parikh and Boyd, 2013) has been popularly used for optimization problems of the form in (1). Let f be convex and L-Lipschitz smooth, and g is convex. The proximal algorithm generates iterates $\{x_t\}$ as

$$\begin{aligned} x_{t+1} &= \arg\min_x f(x_t) + (x - x_t)^\top \nabla f(x_t) + \frac{L}{2}\|x - x_t\|_2^2 + g(x) \\ &= \text{prox}_{\frac{1}{L}g}\left(x_t - \frac{1}{L}\nabla f(x_t)\right), \end{aligned}$$

where $\text{prox}_g(z) \equiv \arg\min_x \frac{1}{2}\|x - z\|_2^2 + g(x)$ is the proximal step, The proximal algorithm converges at a rate of $O(1/T)$. This can be further accelerated to $O(1/T^2)$ by modifying the generation of $\{x_t\}$ as (Beck, 2009; Nesterov, 2013):

$$\begin{aligned} y_t &= x_t + \frac{\alpha_{t-1} - 1}{\alpha_t}(x_t - x_{t-1}), \\ x_{t+1} &= \text{prox}_{\frac{1}{L}g}\left(y_t - \frac{1}{L}\nabla f(y_t)\right), \end{aligned}$$

where $\alpha_0 = \alpha_1 = 1$ and $\alpha_{t+1} = \frac{1}{2}(\sqrt{4\alpha_t^2 + 1} + 1)$.

Recently, the proximal algorithm has been extended to nonconvex optimization. In particular, NIPS (Sra, 2012), IPiano (Ochs et al., 2014) and UAG (Ghadimi and Lan, 2016) allow f to be nonconvex, while g is still required to be convex. GIST (Gong et al., 2013), IFB (Bot et al., 2016) and nmAPG (Li and Lin, 2015) further remove this restriction and allow g to be nonconvex. It is desirable that the proximal step has a closed-form solution. This is true for many convex regularizers such as the lasso regularier (Tibshirani, 1996), tree-structured lasso regularizer (Liu and

Ye, 2010; Jenatton et al., 2011) and sparse group lasso regularizer (Jacob et al., 2009). However, when g is nonconvex, a closed-form solution only exists when g is simple (e.g., nonconvex lasso regularizer (Gong et al., 2013)), but not for the more general cases (e.g., nonconvex tree-structured lasso regularizer (Zhong and Kwok, 2014)).

On the other hand, Zhong and Kwok (2014) used the proximal average (Bauschke et al., 2008) to handle complicated g's of the form $g(x) = \sum_{i=1}^{K} \mu_i g_i(x)$, where each g_i has a simple proximal step. The iterates are generated as

$$x_{t+1} = \sum_{i=1}^{K} \mu_i \cdot \text{prox}_{\frac{\mu_i}{L} g_i} \left(x_t - \frac{1}{L} \nabla f(x_t) \right) / \sum_{i=1}^{K} \mu_i.$$

Each of the constituent proximal steps $\text{prox}_{\frac{\mu_i}{L} g_i}(\cdot)$ can be computed inexpensively, and the per-iteration complexity is low. However, it only approximates $\text{prox}_g(z)$ and consequently also the original problem (1). Empirically, its convergence can be slow.

2.3 Frank-Wolfe (FW) Algorithm

The FW algorithm (Frank and Wolfe, 1956) is used for solving optimization problems of the form

$$\min_x f(x) \ : \ x \in \mathcal{C}, \tag{4}$$

where f is Lipschitz-smooth and convex, and $\mathcal{C}$ is a compact convex set. Recently, it has been popularly used in machine learning (Jaggi, 2013). In each iteration, the FW algorithm generates the next iterate x_{t+1} as

$$s_t = \arg\min_{s \in \mathcal{C}} \ s^\top \nabla f(x_t), \tag{5}$$

$$\gamma_t = \arg\min_{\gamma \in [0,1]} f((1-\gamma)x_t + \gamma s_t), \tag{6}$$

$$x_{t+1} = (1-\gamma_t)x_t + \gamma_t s_t. \tag{7}$$

Here, (5) is a linear subproblem which can often be easily solved; (6) performs line search, and the next iterate x_{t+1} is generated from a convex combination of x_t and s_t in (7). The FW algorithm has a convergence rate of $O(1/T)$ (Jaggi, 2013).

In this paper, we will focus on using the FW algorithm to learn a low-rank matrix $X \in \mathbb{R}^{m \times n}$. Without loss of generality, we assume that $m \leq n$. The nuclear norm $\|X\|_*$ of X is the tightest convex envelope of $\text{rank}(X)$, and is often used as a low-rank regularizer (Candès and Recht, 2009). The low-rank matrix learning problem can be written as

$$\min_X f(X) + \mu \|X\|_*, \tag{8}$$

where f is the loss. For example, in matrix completion (Candès and Recht, 2009),

$$f(X) = \frac{1}{2} \|\mathcal{P}_\Omega(X - O)\|_F^2, \tag{9}$$

where O is the observed incomplete matrix, $\Omega \in \{0,1\}^{m \times n}$ contains indices to the observed entries in O, and $[P_\Omega(A)]_{ij} = A_{ij}$ if $\Omega_{ij} = 1$, and 0 otherwise.

The FW algorithm for this nuclear norm regularized problem is shown in Algorithm 1 (Zhang et al., 2012). Let the iterate at the tth iteration be X_t. As in (5), the following linear subproblem has to be solved (Jaggi, 2013):

$$\min_{S:\|S\|_* \leq 1} \langle S, \nabla f(X_t) \rangle. \tag{10}$$

This can be obtained from the rank-one SVD of $\nabla f(X_t)$ (step 3). Similar to (6), line search is performed at step 4. As a rank-one matrix is added into X_t in each iteration, it is convenient to write X_t as

$$\sum_{i=1}^{t} u_i v_i^\top = U_t V_t^\top, \tag{11}$$

where $U_t = [u_1, \ldots, u_t]$ and $V_t = [v_1, \ldots, v_t]$. The FW algorithm has a convergence rate of $O(1/T)$ (Jaggi, 2013). To make it empirically faster, Algorithm 1 also performs optimization at step 6 (Laue, 2012; Zhang et al., 2012). Substituting $\|X\|_* = \min_{X=UV^\top} \frac{1}{2}\left(\|U\|_F^2 + \|V\|_F^2\right)$ (Srebro et al., 2004) into (8), we have the following local optimization problem:

$$\min_{U,V} f(UV^\top) + \frac{\mu}{2}(\|U\|_F^2 + \|V\|_F^2). \tag{12}$$

This can be solved by standard solvers such as L-BFGS (Nocedal and Wright, 2006).

Algorithm 1 Frank-Wolfe algorithm for problem (8) with f convex (Zhang et al., 2012).

1: $U_1 = [\,]$ and $V_1 = [\,]$;
2: **for** $t = 1 \ldots T$ **do**
3: $\quad [u_t, s_t, v_t] = \text{rank1SVD}(\nabla f(X_t))$;
4: $\quad [\alpha_t, \beta_t] = \arg\min_{\alpha \geq 0, \beta \geq 0} f(\alpha X_t + \beta u_t v_t^\top) + \mu(\alpha \|X_t\|_* + \beta)$;
5: $\quad \bar{U}_t = \left[\sqrt{\alpha_t} U_t; \sqrt{\beta_t} u_t\right]$ and $\bar{V}_t = \left[\sqrt{\alpha_t} V_t; \sqrt{\beta_t} v_t\right]$;
6: $\quad$ obtain $[U_{t+1}, V_{t+1}]$ from (12), using $\bar{U}_t$ and $\bar{V}_t$ for warm-start; // $X_{t+1} = U_{t+1} V_{t+1}^\top$
7: **end for**
8: **return** U_{T+1} and V_{T+1}.

2.4 Alternating Direction Method of Multipliers (ADMM)

ADMM is a simple but powerful algorithm first introduced in the 1970s (Glowinski and Marroco, 1975). Recently, it has been popularly used in diverse fields such as machine learning, data mining and image processing (Boyd et al., 2011). It can be used to solve optimization problems of the form

$$\min_{x,y} \; f(x) + g(y) \;:\; Ax + By = c, \tag{13}$$

where f, g are convex functions, and A, B (resp. c) are constant matrices (resp. vector) of appropriate sizes. Consider the augmented Lagrangian $\mathcal{L}(x, y, u) = f(x) + g(y) + u^\top(Ax + By - c) + \frac{\tau}{2}\|Ax + By - c\|_2^2$, where u is the vector of Lagrangian multipliers, and $\tau > 0$ is a penalty parameter. At the tth iteration of ADMM, the values of x, y and u are updated as

$$x_{t+1} = \arg\min_x \mathcal{L}(x, y_t, u_t), \tag{14}$$

$$y_{t+1} = \arg\min_y \mathcal{L}(x_{t+1}, y, u_t), \tag{15}$$

$$u_{t+1} = u_t + \tau(Ax_{t+1} + By_{t+1} - c). \tag{16}$$

By minimizing $L(x, y, u_k)$ w.r.t. x and y in an alternating manner ((14) and (15)), ADMM can more easily decompose the optimization problem when f, g are separable.

In this paper, we will focus a special case of (13), namely, the consensus optimization problem:

$$\min_{y,x^1,\ldots,x^M} \sum_{i=1}^{M} f_i(x^i) + g(y) \quad : \quad x^1 = \cdots = x^M = y. \tag{17}$$

Here, each f_i is Lipschitz-smooth, x^i is the variable in the local objective f_i, and y is the global consensus variable. This type of problems is often encountered in machine learning, signal processing and wireless communication (Bertsekas and Tsitsiklis, 1989; Boyd et al., 2011). For example, in regularized risk minimization, y is the model parameter, f_i is the regularized risk functional defined on the ith data subset, and g is the regularizer. The augmented Lagrangian for (17) is

$$\mathcal{L}(y, x^1, \ldots, x^M, u^1, \ldots, u^M) = g(y) + \sum_{i=1}^{M} f_i\left(x^i\right) + (u^i)^\top (x^i - y) + \frac{\tau}{2}\|x^i - y\|_2^2,$$

where u^i is the dual variable for the constraint $x^i = y$. Substituting into (14)-(16), we have

$$\begin{aligned}
x_{t+1}^i &= \arg\min_{x^i} f_i(x^i) + (u_t^i)^\top (x^i - y_t) + \frac{\tau}{2}\|x^i - y_t\|_2^2, \quad i = 1, \ldots, M, && (18)\\
y_{t+1} &= \arg\min_{y} \frac{1}{2}\left\| y - \sum_{i=1}^{M}\left(x_t^i + \frac{1}{\tau}u_t^i\right)\right\|_2^2 + \frac{1}{\tau}g(y) = \mathrm{prox}_{\frac{1}{\tau}g}\left(\sum_{i=1}^{M} x_t^i + \frac{1}{\tau}u_t^i\right), && (19)\\
u_{t+1} &= u_t + \tau\left(x_{t+1}^i + y_{t+1}\right), \quad i = 1, \ldots, M.
\end{aligned}$$

When f_i is smooth and g is convex, ADMM converges to a critical point of (17) (Hong et al., 2016). However, when g is nonconvex, its convergence is still an open issue.

3. Shifting Nonconvexity from Regularizer to Loss

In recent years, a number of nonconvex regularizers have been proposed. Examples include the Geman penalty (GP) (Geman and Yang, 1995), log-sum penalty (LSP) (Candès et al., 2008) and Laplace penalty (Trzasko and Manduca, 2009). In general, learning with nonconvex regularizers is much more difficult than learning with convex regularizers. In this section, we show how to move the nonconvex component from the nonconvex regularizers to the loss function. Existing algorithms can then be reused to learn with the convexified regularizers.

First, we make the following standard assumptions on (1).

A1. F is bounded from below and $\lim_{\|x\|_2\to\infty} F(x) = \infty$;

A2. f is L-Lipschitz smooth (i.e., $\|\nabla f(x) - \nabla f(y)\|_2 \le L\|x - y\|_2$), but possibly nonconvex.

Let κ be a function that is concave, non-decreasing, ρ-Lipschitz smooth with κ' non-differentiable at finite points, and $\kappa(0) = 0$. All nonconvex regularizers in Table 1 satisfy the requirements on κ.

In this paper, we consider g of the following forms:

	$\kappa(\alpha)$	$\kappa'(\alpha)$	κ_0	ρ
GP (Geman and Yang, 1995)	$\frac{\beta\alpha}{\theta+\alpha}$	$\frac{\beta\theta}{(\theta+\alpha)^2}$	$\frac{\beta}{\theta}$	$\frac{2\beta}{\theta^2}$
LSP (Candès et al., 2008)	$\beta\log(1+\frac{\alpha}{\theta})$	$\frac{\beta}{\theta+\alpha}$	$\frac{\beta}{\theta}$	$\frac{\beta}{\theta^2}$
MCP (Zhang, 2010a)	$\begin{cases}\beta\alpha-\frac{\alpha^2}{2\theta} & \alpha\le\beta\theta\\ \frac{1}{2}\theta\beta^2 & \alpha>\beta\theta\end{cases}$	$\begin{cases}\beta-\frac{\alpha}{\theta} & \alpha\le\beta\theta\\ 0 & \alpha>\beta\theta\end{cases}$	β	$\frac{1}{\theta}$
Laplace (Trzasko and Manduca, 2009)	$\beta(1-\exp(-\frac{\alpha}{\theta}))$	$\frac{\beta}{\theta}\exp\left(-\frac{\alpha}{\theta}\right)$	$\frac{\beta}{\theta}$	$\frac{\beta}{\theta^2}$
SCAD (Fan and Li, 2001)	$\begin{cases}\beta\alpha & \alpha\le\beta\\ \frac{-\alpha^2+2\theta\beta\alpha-\beta^2}{2(\theta-1)} & \beta<\alpha\le\theta\beta\\ \frac{\beta^2(1+\theta)}{2} & \alpha>\theta\beta\end{cases}$	$\begin{cases}\beta & \alpha\le\beta\\ \frac{-\alpha+\theta\beta}{\theta-1} & \beta<\alpha\le\theta\beta\\ 0 & \alpha>\theta\beta\end{cases}$	β	$\frac{1}{\theta-1}$

Table 1: Example nonconvex regularizers. Here, $\kappa_0 \equiv \kappa'(0)$ and $\beta > 0$. For SCAD, $\theta > 2$, whereas for others, $\theta > 0$.

C1. $g(x) = \sum_{i=1}^K \mu_i g_i(x)$, where $\mu_i \ge 0$,

$$g_i(x) = \kappa(\|A_i x\|_2), \tag{20}$$

and A_i is a matrix. When κ is the identity function, $g(x)$ reduces to the convex regularizer $\sum_{i=1}^K \mu_i \|A_i x\|_2$. By using different A_i's, g becomes various structured sparsity regularizers such as the group lasso (Jacob et al., 2009), fused lasso (Tibshirani et al., 2005), and graphical lasso (Jacob et al., 2009).

C2. $g(X) = \mu \sum_{i=1}^m \kappa(\sigma_i(X))$, where X is a matrix and $\mu \ge 0$. When κ is the identity function, g reduces to the nuclear norm.

First, consider g in **C1**. Rewrite each nonconvex g_i in (20) as

$$g_i(x) = \bar{g}_i(x) + \kappa_0 \|A_i x\|_2, \tag{21}$$

where $\kappa_0 = \kappa'(0)$, and $\bar{g}_i(x) = \kappa(\|A_i x\|_2) - \kappa_0 \|A_i x\|_2$. Obviously, $\kappa_0\|A_i x\|_2$ is convex but nonsmooth. The following shows that $\bar{g}_i$, though nonconvex, is concave and Lipschitz smooth.

Proposition 1 *$\kappa(\|z\|_2) - \kappa_0\|z\|_2$, where $z \in \mathbb{R}^d$, is concave and 2ρ-Lipschitz smooth.*

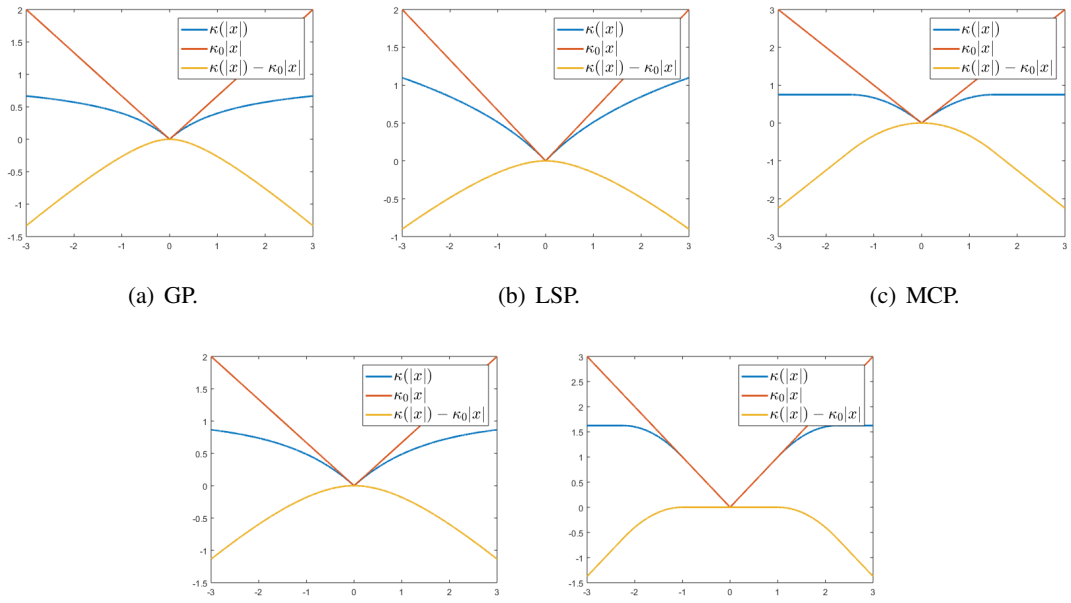
Corollary 2 *$\bar{g}_i$ is concave and Lipschitz smooth with modulus $\bar{L}_i = 2\rho\|A_i\|_F$.*

Corollary 3 *$g(x)$ can be decomposed as $\bar{g}(x) + \breve{g}(x)$, where $\bar{g}(x) \equiv \sum_{i=1}^K \mu_i \bar{g}_i(x)$ is concave and Lipschitz-smooth, while $\breve{g}(x) \equiv \kappa_0 \sum_{i=1}^K \mu_i \|A_i x\|_2$ is convex but nonsmooth.*

Remark 4 *When $A_i = Diag(e_i)$, where e_i is the unit vector for dimension i, $\|A_i x\|_2 = |x_i|$ and*

$$g(x) = \sum_{i=1}^d \mu_i \kappa(\|A_i x\|_2) = \sum_{i=1}^d \mu_i \kappa(|x_i|). \tag{22}$$

Using Corollary 3, g can be decomposed as $\bar{g}(x) + \breve{g}(x)$, where $\bar{g}(x) \equiv \sum_{i=1}^{d} \mu_i(\kappa(|x_i|) - \kappa_0|x_i|)$ is concave and 2ρ-Lipschitz smooth, while $\breve{g}(x) \equiv \kappa_0 \sum_{i=1}^{d} \mu_i|x_i|$ is convex and nonsmooth. When $d = 1$ and $\mu_1 = 1$, an illustration of $g(x) = \kappa(|x|)$, $\bar{g}(x) = \kappa(|x|) - \kappa_0|x|$ and $\breve{g}(x) = \kappa_0|x|$ for the various nonconvex regularizers is shown in Figure 2. When κ is the identity function and $\mu_1 = \cdots = \mu_m = \mu$, g in (22) reduces to the lasso regularizer $\mu\|x\|_1$.



(a) GP. (b) LSP. (c) MCP.

(d) Laplace. (e) SCAD.

Figure 2: Decompositions of the regularizers in Table 1 ($\beta = 1$ for all regularizers; $\theta = 2.25$ for SCAD and 1.5 for others).

Using Corollary 3, problem (1) can then be rewritten as

$$\min_x \bar{f}(x) + \breve{g}(x), \tag{23}$$

where $\bar{f}(x) \equiv f(x) + \bar{g}(x)$. Note that $\bar{f}$ (which can be viewed as an augmented loss) is Lipschitz smooth while $\breve{g}$ (viewed as a convexified regularizer) is convex but possibly nonsmooth. In other words, nonconvexity is shifted from the regularizer g to the loss f, while ensuring that the augmented loss is smooth.

When X is a matrix, similar to Corollary 3, the following Proposition holds for g in **C2**.

Proposition 5 *Any g in* **C2** *can be decomposed as $\bar{g}(X) + \breve{g}(X)$, where*

$$\bar{g}(X) \equiv \mu \sum_{i=1}^{m} \kappa(\sigma_i(X)) - \mu\kappa_0\|X\|_* \tag{24}$$

is concave and 2ρ-Lipschitz smooth, and $\breve{g}(X) \equiv \kappa_0\|X\|_*$ is convex and nonsmooth.

Since $\bar{g}$ is concave and $\breve{g}$ is convex, the nonconvex regularizer $g = \breve{g} - (-\bar{g})$ can be viewed as a difference of convex functions (DC) (Hiriart-Urruty, 1985). Lu (2012); Gong et al. (2013); Zhong and Kwok (2014) also relied on DC decompositions of the nonconvex regularizer. However, they do not utilize this in the computational procedures, while we use the DC decomposition to simplify the regularizers. As will be seen, though the DC decomposition of a nonconvex function is not unique in general, the particular one proposed here is crucial for efficient optimization.

4. Example Use Cases

In this section, we provide concrete examples to show how the proposed convexification scheme can be used with various optimization algorithms. An overview is summarized in Table 2.

	section	advantages
proximal algorithm	4.1, 4.6	cheaper proximal step
FW algorithm	4.2	cheaper linear subproblem
(consensus) ADMM	4.3	cheaper proximal step; provide convergence guarantee
SVRG	4.4	cheaper proximal step; provide convergence guarantee
mOWL-QN	4.5	simpler analysis; capture curvature information

Table 2: Using the proposed convexification scheme with various algorithms.

4.1 Proximal Algorithms

In this section, we provide example applications on using the proximal algorithm for nonconvex structured sparse learning. The proximal algorithm has been commonly used for learning with convex regularizers (Parikh and Boyd, 2013). With a nonconvex regularizer, the underlying proximal step becomes much more challenging. Gong et al. (2013); Li and Lin (2015) and Bot et al. (2016) extended proximal algorithm to simple nonconvex g, but cannot handle more complicated nonconvex regularizers such as the tree-structured lasso regularizer (Liu and Ye, 2010; Schmidt et al., 2011), sparse group lasso regularizer (Jacob et al., 2009) and total variation regularizer (Nikolova, 2004). Using the proximal average (Bauschke et al., 2008), Zhong and Kwok (2014) can handle nonconvex regularizers of the form $g = \sum_{i=1}^{K} \mu_i g_i$, where each g_i is simple. However, the solutions obtained are only approximate. General nonconvex optimization techniques such as the concave-convex procedure (CCCP) (Yuille and Rangarajan, 2002) or its variant sequential convex programming (SCP) (Lu, 2012) can also be used, though they are slow in general (Gong et al., 2013; Zhong and Kwok, 2014).

Using the proposed transformation, one only needs to solve the proximal step of a standard convex regularizer instead of that of a nonconvex regularizer. This allows reuse of existing solutions for the proximal step and is much less expensive. As proximal algorithms have the same convergence guarantee for convex and nonconvex f (Gong et al., 2013; Li and Lin, 2015; Yao et al., 2017), solving the transformed problem can be much faster. The following gives some specific examples.

4.1.1 NONCONVEX SPARSE GROUP LASSO

In sparse group lasso, the feature vector x is divided into groups. Let $\mathcal{G}_j$ be the set containing dimensions of x that are in group j, and $\left[x_{\mathcal{G}_j}\right]_i = x_i$ if $i \in \mathcal{G}_j$ and 0 otherwise. Given training samples $\{(a_1, y_1), \ldots, (a_N, y_N)\}$, (convex) sparse group lasso is formulated as (Jacob et al., 2009):

$$\min_x \sum_{i=1}^N \ell(y_i, a_i^\top x) + \lambda\|x\|_1 + \sum_{j=1}^K \mu_j \|x_{\mathcal{G}_j}\|_2, \tag{25}$$

where ℓ is a smooth loss, and K is the number of (non-overlapping) groups.

For the nonconvex extension, the regularizer becomes

$$g(x) = \lambda \sum_{i=1}^d \kappa(|x_i|) + \sum_{j=1}^K \mu_j \kappa(\|x_{\mathcal{G}_j}\|_2). \tag{26}$$

Using Corollary 3 and Remark 4, the convexified regularizer is $\breve{g}(x) = \kappa_0(\lambda\|x\|_1 + \sum_{j=1}^K \mu_j \|x_{\mathcal{G}_j}\|_2)$. Its proximal step can be easily computed by the algorithm in (Yuan et al., 2011). Specifically, the proximal operator of $\breve{g}$ can be obtained by computing $\text{prox}_{\mu_j\|\cdot\|_2}(\text{prox}_{\lambda\|\cdot\|_1}(x_{\mathcal{G}_j}))$ for each group separately. This can then be used with any proximal algorithm that can handle nonconvex objectives (as $\bar{f}$ is nonconvex). In particular, we will adopt the state-of-the-art nonmontonic APG (nmAPG) algorithm (Li and Lin, 2015) (shown in Algorithm 2). On the other hand, note that nmAPG cannot be directly used with the nonconvex regularizer g in (26), as the corresponding proximal step has no inexpensive closed-form solution.

Algorithm 2 Nonmonotonic APG (nmAPG) (Li and Lin, 2015).

Initialize $z_1 = x_1 = x_0$, $\alpha_0 = 0$, $\alpha_1 = 1$, $\eta \in [0, 1)$, $c_1 = F(x_1)$, $q_1 = 1$, and stepsize $\tau > \bar{L}$, $\delta \in (0, \tau - \bar{L})$;
for $t = 1, \ldots, T$ **do**
  $y_t = x_t + \frac{\alpha_{t-1}}{\alpha_t}(z_t - x_t) + \frac{\alpha_{t-1}-1}{\alpha_t}(x_t - x_{t-1})$;
  $z_{t+1} = \text{prox}_{\frac{1}{\tau}\breve{g}}(y_t - \frac{1}{\tau}\nabla\bar{f}(y_t))$;
  if $F(z_{t+1}) \le c_t - \frac{\delta}{2}\|z_{t+1} - y_t\|_2^2$ **then**
    $x_{t+1} = z_{t+1}$;
  else
    $v_{t+1} = \text{prox}_{\frac{1}{\tau}\breve{g}}(x_t - \frac{1}{\tau}\nabla\bar{f}(x_t))$;
    $x_{t+1} = \begin{cases} z_{t+1} & F(z_{t+1}) \le F(v_{t+1}) \\ v_{t+1} & \text{otherwise} \end{cases}$;
  end if
  $\alpha_{t+1} = \frac{1}{2}(\sqrt{4\alpha_t^2 + 1} + 1)$;
  $q_{t+1} = \eta q_t + 1$;
  $c_{t+1} = \frac{\eta q_t c_t + F(x_{t+1})}{q_{t+1}}$;
end for
return x_{T+1};

4.1.2 NONCONVEX TREE-STRUCTURED GROUP LASSO

In (convex) tree-structured group lasso (Liu and Ye, 2010; Jenatton et al., 2011), the dimensions in x are organized as nodes in a tree, and each group corresponds to a subtree. The regularizer is of the form $\sum_{j=1}^{K} \lambda_j \|x_{\mathcal{G}_j}\|_2$. Interested readers are referred to (Liu and Ye, 2010) for details.

For the nonconvex extension, $g(x)$ becomes $\sum_{j=1}^{K} \lambda_j \kappa(\|x_{\mathcal{G}_j}\|_2)$. Again, there is no closed-form solution of its proximal step. On the other hand, the convexified regularizer is $\breve{g}(x) \equiv \kappa_0 \sum_{j=1}^{K} \lambda_j \|x_{\mathcal{G}_j}\|_2$. As shown in (Liu and Ye, 2010), its proximal step can be computed efficiently by processing all the groups once in some appropriate order.

4.1.3 NONCONVEX TOTAL VARIATION (TV) REGULARIZER

In an image, nearby pixels are usually strongly correlated. The TV regularizer captures such behavior by assuming that changes between nearby pixels are small. Given an image $X \in \mathbb{R}^{m \times n}$, the TV regularizer is defined as $\mathrm{TV}(X) = \|D_v X\|_1 + \|X D_h\|_1$ (Nikolova, 2004), $D_v = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix} \in \mathbb{R}^{(m-1) \times m}$ and $D_h = \begin{bmatrix} -1 & & \\ 1 & \ddots & \\ & \ddots & -1 \\ & & 1 \end{bmatrix} \in \mathbb{R}^{n \times (n-1)}$ are the horizontal and vertical partial derivative operators, respectively. Thus, it is popular on image processing problems, such as image denoising and deconvolution (Nikolova, 2004; Beck and Teboulle, 2009).

As in previous sections, the nonconvex extension of TV regularizer can be defined as

$$\sum_{i=1}^{m-1} \sum_{j=1}^{m} \kappa\left(\left|[D_v X]_{ij}\right|\right) + \sum_{i=1}^{n} \sum_{j=1}^{n-1} \kappa\left(\left|[X D_h]_{ij}\right|\right). \tag{27}$$

Again, it is not clear how its proximal step can be efficiently computed. Instead, with the proposed transformation, the transformed problem is

$$\min_X \bar{f}(X) + \mu \kappa_0 \mathrm{TV}(X),$$

where μ is the regularization parameter, $\bar{f}(X) = f(X) + \mu \sum_{i=1}^{m-1} \sum_{j=1}^{m} (\kappa(|[D_v X]_{ij}|) - \kappa_0 |[D_v X]_{ij}|) + \mu \sum_{i=1}^{n} \sum_{j=1}^{n-1} (\kappa(|[X D_h]_{ij}|) - \kappa_0 |[X D_h]_{ij}|)$ is concave and Lipschitz smooth. One then only needs to compute the proximal step of the standard TV regularizer.

Unlike the proximal steps in Sections 4.1.1 and 4.1.2, the proximal step of the TV regularizer has no closed-form solution and needs to be solved iteratively. In this case, Schmidt et al. (2011) showed that using inexact proximal steps can make proximal algorithms faster, but they only considered the situation where both f and g are convex. In the following, we extend nmAPG (Algorithm 2), which can be used with nonconvex objectives, to allow for inexact proximal steps (steps 5 and 9 of Algorithm 3). However, Lemma 2 of (Li and Lin, 2015), which is key to the convergence of nmAPG, no longer holds because of the inexact proximal step. To fix this problem, in step 6 of Algorithm 3, we use $F(X_t)$ instead of c_t in Algorithm 2. We also drop the comparison of $F(Z_{t+1})$ and $F(V_{t+1})$ (originally in step 9 of Algorithm 2).

Inexactness of the proximal step can be controlled as follows. Let $P = X - \frac{1}{\tau} \nabla \bar{f}(X)$, and $h(X) \equiv \frac{1}{2}\|X - P\|_F^2 + \frac{1}{\tau} \breve{g}(X)$ be the objective in $\mathrm{prox}_{\frac{1}{\tau}\breve{g}}(P)$. As $\breve{g}(X) = \kappa_0 \mathrm{TV}(X)$ is convex,

Algorithm 3 Inexact nmAPG.

1: Initialize $\tilde{Z}_1 = X_1 = X_0$, $\alpha_0 = 0$, $\alpha_1 = 1$ and stepsize $\tau > \bar{L}$, $\delta \in (0, \tau - \bar{L})$;
2: **for** $t = 1, \ldots, T$ **do**
3: choose tolerance ϵ_t;
4: $Y_t = X_t + \frac{\alpha_{t-1}}{\alpha_t}(Z_t - X_t) + \frac{\alpha_{t-1}-1}{\alpha_t}(X_t - X_{t-1})$;
5: $\tilde{Z}_{t+1}$ = approximate $\text{prox}_{\frac{1}{\tau}\breve{g}}(Y_t - \frac{1}{\tau}\nabla \bar{f}(Y_t))$, with inexactness $\vartheta_{t+1} \le \epsilon_t$;
6: **if** $F(\tilde{Z}_{t+1}) \le F(X_t) - \frac{\delta}{2}\|\tilde{Z}_{t+1} - Y_t\|_F^2$ **then**
7: $X_{t+1} = \tilde{Z}_{t+1}$;
8: **else**
9: X_{t+1} = approximate $\text{prox}_{\frac{1}{\tau}\breve{g}}(X_t - \frac{1}{\tau}\nabla \bar{f}(X_t))$, with inexactness $\vartheta_{t+1} \le \epsilon_t$;
10: **end if**
11: $\alpha_{t+1} = \frac{1}{2}(\sqrt{4\alpha_t^2 + 1} + 1)$;
12: **end for**
13: **return** X_{T+1};

h is also convex. Let $\tilde{X}$ be an inexact solution of this proximal step. The inexactness $h(\tilde{X}) - h(\text{prox}_{\frac{1}{\tau}\breve{g}}(P))$ is upper-bounded by the duality gap $\vartheta \equiv h(\tilde{X}) - \mathcal{D}(\tilde{W})$, where $\mathcal{D}$ is the dual of h, and $\tilde{W}$ is the corresponding dual variable. In steps 5 and 9 of Algorithm 3, we solve the proximal step until the duality gap ϑ_{t+1} is smaller than a given threshold ϵ_t. The following Theorem shows convergence of Algorithm 3.

Theorem 6 *Let $\sum_{t=1}^{\infty} \epsilon_t < \infty$. The sequence $\{X_t\}$ generated from Algorithm 3 has at least one limit point, and every limit point is also a critical point of* (1)*.*

If the proximal step is exact, $\|V_t - \text{prox}_{\frac{1}{\tau}\breve{g}}(V_t - \frac{1}{\tau}\nabla \bar{f}(V_t))\|_F^2$ can be used to measure how far V_t is from a critical point (Gong et al., 2013; Ghadimi and Lan, 2016). In Algorithm 3, the proximal step is inexact, and X_{t+1} is an inexact solution to $\text{prox}_{\frac{1}{\tau}\breve{g}}(V_t - \frac{1}{\tau}\nabla \bar{f}(V_t))$, where $V_t = Y_t$ if step 7 is executed, and $V_t = X_t$ if step 9 is executed. As X_{t+1} converges to a critical point of (1), we propose using $d_t \equiv \|X_{t+1} - V_t\|_F^2$ to measure how far X_{t+1} is from a critical point. The following Proposition shows a $O(1/T)$ convergence rate on $\min_{t=1,\ldots,T} d_t$.

Proposition 7 *(i) $\lim_{t\to\infty} d_t = 0$; and (ii) $\min_{t=1,\ldots,T} d_t$ converges to zero at a rate of $O(1/T)$.*

Note that the (exact) nmAPG in Algorithm 2 cannot handle the nonconvex g in (27) efficiently, as the corresponding proximal step has no closed-form solution but has to be solved exactly. Even the proposed inexact nmAPG (Algorithm 3) cannot be directly used with nonconvex g. As the dual of the nonconvex proximal step is difficult to derive and the optimal duality gap is nonzero in general, the proximal step's inexactness cannot be easily controlled.

Remark 8 *As mentioned in Section 3, the proposed decomposition of the nonconvex regularizer g can be regarded as a DC decomposition, which is not unique in general. For example, in Section 4.1.1, we might try to add a quadratic term to convexify the nonconvex sparse group lasso regularizer. Specifically, we can decompose $g(x)$ in (26) as $\breve{\varsigma}(x) + \hat{\varsigma}(x)$, where*

$$\breve{\varsigma}(x) = \lambda \sum_{i=1}^{d} \left(\kappa(|x_i|) + \frac{\rho}{2}x_i^2\right) + \sum_{j=1}^{K} \mu_j \left(\kappa(\|x_{\mathcal{G}_j}\|_2) + \frac{\rho}{2}\|x_{\mathcal{G}_j}\|_2^2\right), \tag{28}$$

and $\hat{\varsigma}(x) = -\frac{\rho}{2}\sum_{j=1}^{K}(\mu_j+\lambda)\|x_{\mathcal{G}_j}\|_2^2$. It can be easily shown that $\hat{\varsigma}$ is concave, and the following Proposition 9 shows that $\breve{\varsigma}$ is convex. Thus, F can be transformed as $F(x) = \bar{f}(x) + \breve{\varsigma}(x)$, where $\bar{f}(x) = f(x) + \hat{\varsigma}(x)$ is Lipschitz-smooth, and $\breve{\varsigma}$ is convex but nonsmooth. However, the proximal step associated with $\breve{\varsigma}$ has no simple closed-form solution.

Proposition 9 *$\kappa(\|\cdot\|) + \frac{\rho}{2}\|\cdot\|^2$, where $\|\cdot\|$ is a norm, is convex.*

Similarly, in Section 4.1.2, we can also add a quadratic term to convexify the nonconvex tree-structured group lasso regularizer as $\sum_{j=1}^{K}\lambda_j(\kappa(\|x_{\mathcal{G}_j}\|_2)+\frac{\rho}{2}\|x_{\mathcal{G}_j}\|_2^2)$. The corresponding proximal step is

$$\min_x \frac{1}{2}\|x-z\|_2^2 + \sum_{j=1}^{K}\lambda_j\left(\kappa\left(\|x_{\mathcal{G}_j}\|_2\right) + \frac{\rho}{2}\|x_{\mathcal{G}_j}\|_2^2\right). \tag{29}$$

However, inexpensive closed-form solution on the proximal step is only known for tree-structured group lasso regularizers of the form $\sum_{j=1}^{K}\lambda_j\|x_{\mathcal{G}_j}\|_2$ (Jenatton et al., 2011; Liu and Ye, 2010). Thus, (29) has to be iteratively solved (e.g., using ADMM), and is slow.

In Section 4.1.3, by adding quadratic terms to the nonconvex TV regularizer, it becomes $\sum_{i=1}^{m-1}\sum_{j=1}^{m}(\kappa\left(|[D_vX]_{ij}|\right)+\frac{\rho}{2}[D_vX]_{ij}^2)+\sum_{i=1}^{n}\sum_{j=1}^{n-1}(\kappa(|[XD_h]_{ij}|)+\frac{\rho}{2}[XD_h]_{ij}^2)$, where $\rho>0$ is a constant. The corresponding proximal step is

$$\min_X \frac{1}{2}\|X-Z\|_F^2+\mu\sum_{i=1}^{m-1}\sum_{j=1}^{m}\left(\kappa\left(|[D_vX]_{ij}|\right)+\frac{\rho}{2}[D_vX]_{ij}^2\right)+\mu\sum_{i=1}^{n}\sum_{j=1}^{n-1}\left(\kappa\left(|[XD_h]_{ij}|\right)+\frac{\rho}{2}[XD_h]_{ij}^2\right),$$

which is difficult to solve. Morevoer, unlike the proposed convexification scheme, the dual of the above is difficult to derive.

4.2 Frank-Wolfe Algorithm

In this section, we use the Frank-Wolfe algorithm to learn a low-rank matrix $X \in \mathbb{R}^{m\times n}$ for matrix completion (Section 2.3). The nuclear norm regularizer in (8) may over-penalize top singular values. Recently, there is growing interest to replace this with nonconvex regularizers (Lu et al., 2014, 2015; Yao et al., 2015). Hence, instead of (8), we consider

$$\min_X f(X) + \mu\sum_{i=1}^{m}\kappa(\sigma_i(X)). \tag{30}$$

When κ is the identity function, (30) reduces to (8). Note that the FW algorithm cannot be directly used on (8), as its linear subproblem in (10) then becomes $\min_{S:\sum_{i=1}^{m}\kappa(\sigma_i(S))\le 1}\langle S, \nabla f(X_t)\rangle$, which is difficult to solve.

Using Proposition 5, problem (30) is transformed into

$$\min_X \bar{f}(X) + \bar{\mu}\|X\|_*, \tag{31}$$

where

$$\bar{f}(X) = f(X) + \bar{g}(X), \quad \bar{g}(X) = \mu\sum_{i=1}^{m}(\kappa(\sigma_i(X)) - \kappa_0\sigma_i(X)), \tag{32}$$

and $\bar{\mu} = \mu\kappa_0$. Although this only involves the standard nuclear norm regularizer, Algorithm 1 still cannot be used as $\bar{f}$ in (32) is no longer convex. A FW variant allowing nonconvex $\bar{f}$ is proposed in (Bredies et al., 2009). However, condition 1 in (Bredies et al., 2009) requires g to satisfy $\lim_{\|X\|_F \to \infty} \frac{g(X)}{\|X\|_F} = \infty$. This condition does not hold with $g(X) = \|X\|_*$ in (31), as

$$\frac{\|X\|_*}{\|X\|_F} = \sqrt{\frac{(\sum_{i=1}^m \sigma_i)^2}{\sum_{i=1}^m \sigma_i^2}} \leq \sqrt{\frac{m\sum_{i=1}^m \sigma_i^2}{\sum_{i=1}^m \sigma_i^2}} = \sqrt{m} < \infty.$$

In the following, we propose a nonconvex FW variant (Algorithm 4) for the transformed problem (31). It is similar to the original FW Algorithm 1, but with three important modifications. First, $\bar{g}(X)$ in (32) depends on the singular values of X, which cannot be directly obtained from the $UV^\top$ factorization in (11). Instead, we use the low-rank factorization

$$X = UBV^\top, \tag{33}$$

where $U \in \mathbb{R}^{m\times k}$, $V \in \mathbb{R}^{n\times k}$ are orthogonal and $B \in \mathcal{S}_+^{k\times k}$ is positive semidefinite.

Algorithm 4 Frank-Wolfe algorithm for solving the nonconvex problem (31).

1: $U_1 = [\,]$, $B_1 = [\,]$ and $V_1 = [\,]$;
2: **for** $t = 1 \dots T$ **do**
3: $[u_t, s_t, v_t] = \text{rank1SVD}(\nabla \bar{f}(X_t))$;
4: obtain α_t and β_t from (36);
5: $[\bar{U}_t, \bar{B}_t, \bar{V}_t] = \text{warmstart}(U_t, u_t, V_t, v_t, B_t, \alpha_t, \beta_t)$;
6: obtain $[U_{t+1}, B_{t+1}, V_{t+1}]$ from (37), using $\bar{U}_t$, $\bar{B}_t$ and $\bar{V}_t$ for warm-start;
// $X_{t+1} = U_{t+1}B_{t+1}V_{t+1}^\top$
7: **end for**
8: **return** U_{T+1}, B_{T+1} and V_{T+1}.

The second problem is that line search in Algorithm 1 is inefficient in general when operated on a nonconvex $\bar{f}$. Specifically, step 4 in Algorithm 1 then becomes

$$[\alpha_t, \beta_t] = \arg\min_{\alpha\geq 0, \beta \geq 0} \bar{f}(\alpha X_t + \beta u_t v_t^\top) + \bar{\mu}(\alpha\|X_t\|_* + \beta). \tag{34}$$

To solve (34), we have to compute $\frac{\partial \bar{f}(S)}{\partial \alpha}$ and $\frac{\partial \bar{f}(S)}{\partial \beta}$, where $S = \alpha X_t + \beta u_t v_t^\top$. As shown in Proposition 10, this requires the SVD of S and can be expensive.

Proposition 10 *Let the SVD of S be $U_S \mathit{Diag}([\sigma_1(S), \dots, \sigma_m(S)])V_S^\top$. Then*

$$\frac{\partial \bar{f}(S)}{\partial \alpha} = \alpha\langle X_t, \nabla \bar{f}(S)\rangle, \quad \textit{and} \quad \frac{\partial \bar{f}(S)}{\partial \beta} = \beta u_t^\top \nabla \bar{f}(S) v_t,$$

where $\nabla \bar{f}(S) = \nabla f(S) + \mu U_S \mathit{Diag}(w) V_S^\top$, and $w = [\kappa'(\sigma_i(S)) - \kappa_0] \in \mathbb{R}^m$.

Alternatively, as S is a rank-one update of X_t, one can perform incremental update on SVD, which takes $O((m+n)t^2)$ time (Golub and Van Loan, 2012). However, every time α, β are changed, this incremental SVD has to be recomputed, and is thus inefficient.

To alleviate this problem, we approximate $\bar{f}(S)$ by an upper bound as

$$\begin{aligned}\bar{f}(S) &= \bar{f}(X_t + (\alpha-1)X_t + \beta u_t v_t^\top) \\ &\leq \bar{f}(X_t) + \langle (\alpha-1)X_t + \beta u_t v_t^\top, \nabla \bar{f}(X_t)\rangle + \frac{\bar{L}}{2}\|(\alpha-1)X_t + \beta u_t v_t^\top\|_F^2. \quad (35)\end{aligned}$$

As (u_t, v_t) is obtained from the rank-1 SVD of $\nabla \bar{f}(X_t)$, we have $\|u_t v_t^\top\|_F = 1$ and $u_t^\top \nabla \bar{f}(X_t) v_t = s_t$. Moreover, $X_t = U_t B_t V_t^\top$, and so $\|X_t\|_F = \|B_t\|_F$ and $\|X_t\|_* = \operatorname{Tr}(B_t)$. Substituting these and the upper bound (35) into (34), we obtain a simple quadratic program:

$$\begin{aligned}\min_{\alpha\geq 0,\beta\geq 0} \quad & \frac{(\alpha-1)^2\bar{L}}{2}\|B_t\|_F^2 + (\alpha-1)\beta\bar{L}(u_t^\top U_t)B_t(V_t^\top v_t) + \frac{\beta^2\bar{L}}{2} + \beta s_t \\ & + \alpha\langle B_t, U_t^\top \nabla \bar{f}(X_t) V_t\rangle + \bar{\mu}(\alpha\|B_t\|_* + \beta). \quad (36)\end{aligned}$$

Note that the objective in (36) is convex, as the RHS in (35) is convex and the last term from (34) is affine. Using the following Corollary 11, $\langle B_t, U_t^\top \nabla \bar{f}(X_t) V_t\rangle$ in (36) can be obtained as

$$\langle B_t, U_t^\top \nabla \bar{f}(X_t) V_t\rangle = \langle B_t, U_t^\top \nabla f(X_t) V_t\rangle + \bar{\mu}\sum_{i=1}^{t} \sigma_i(B_t)(\kappa'(\sigma_i(B_t)) - \kappa_0).$$

Corollary 11 *For X in (33), let the SVD of B be $U_B \mathit{Diag}([\sigma_1(B), \ldots, \sigma_k(B)])V_B^\top$. Then, $\nabla \bar{f}(X) = \nabla f(X) + \bar{\mu}(UU_B)\mathit{Diag}(w)(VV_B)^\top$, where $w = [\kappa'(\sigma_i(B)) - \kappa_0] \in \mathbb{R}^k$.*

Instead of requiring SVD on X_t, it only requires SVD on B_t (which is of size $t \times t$ at the tth iteration of Algorithm 4). As the target matrix is supposed to be low-rank, $t \ll m$. Hence, all the coefficients in (36) can be obtained in $O((m+n)t^2 + \|\Omega\|_1 t)$ time. Besides, (36) is a quadratic program with only two variables, and thus can be very efficiently solved.

The third modification is that with $\bar{f}$ instead of f, (12) can no longer be used for local optimization, as $\bar{g}(X)$ in (32) depends on the singular values of X. On the other hand, with the decomposition of X in (33) and Proposition 12 below, (31) can be rewritten as

$$\begin{aligned}\min_{U,B,V} \quad & f(UBV^\top) + \bar{g}(B) + \bar{\mu}\operatorname{Tr}(B) \quad (37)\\ \text{s.t.} \quad & U^\top U = I, V^\top V = I, B \in \mathcal{S}_+. \quad (38)\end{aligned}$$

This can be efficiently solved using matrix optimization techniques on the Grassmann manifold (Ngo and Saad, 2012).

Proposition 12 *For orthogonal matrices U and V, $\bar{g}(UBV^\top) = \bar{g}(B)$.*

In Algorithm 4, step 5 is used to warm-start (37), and the procedure is shown in Algorithm 5. It expresses $X_t = \alpha_t U_{t-1} B_{t-1} V_{t-1}^\top + \beta_t u_t v_t^\top$ obtained in step 4 to the form $U_t B_t V_t^\top$ so that the orthogonal constraints on U_t, V_t in (38) are satisfied.

Existing analysis for the FW algorithm cannot be used on this nonconvex problem. The following Theorem shows convergence of Algorithm 4 to a critical point of (8).

Theorem 13 *If* (8) *has a rank-r critical point, then Algorithm 4 converges to a critical point of* (8) *after r iterations.*

Algorithm 5 warmstart$(U_t, u_t, V_t, v_t, B_t, \alpha_t, \beta_t)$.

1: $[\bar{U}_t, R_{\bar{U}_t}] = \text{QR}([U_t, u_t])$; // QR denotes the QR factorization
2: $[\bar{V}_t\,, R_{\bar{V}_t}] = \text{QR}([V_t,\, v_t])$;
3: $\bar{B}_t = R_{\bar{U}_t} \begin{bmatrix} \alpha_t B_t & 0 \\ 0 & \beta_t \end{bmatrix} R_{\bar{V}_t}^\top$;
4: **return** $\bar{U}_t$, $\bar{B}_t$ and $\bar{V}_t$;

As in Remark 8, an alternative convexification approach is to decompose the regularizer in (30) as $\breve{\varsigma}(X) + \hat{\varsigma}(X)$, where $\hat{\varsigma}(X) = -\frac{\rho}{2}\sum_{i=1}^m \sigma_i^2(X)$ and $\breve{\varsigma}(X) = \sum_{i=1}^m \kappa(\sigma_i(X)) + \frac{\rho}{2}\sigma_i^2(X)$. The corresponding linear subproblem in (5) becomes $\min_{S:\breve{\varsigma}(S)\le 1}\langle S, \nabla f(X_t)\rangle$, which is difficult to solve. On the other hand, with the proposed procedure, the subproblem associated with the transformed problem (31) can be easily solved via rank-1 SVD (Jaggi, 2013).

4.3 Alternating Direction Method of Multipliers (ADMM)

In this section, we consider using ADMM on the consensus optimization problem (17). When all the f_i's and g are convex, ADMM has a convergence rate of $O(1/T)$ (He and Yuan, 2012). Recently, ADMM has been extended to problems where g is convex but f_i's are nonconvex (Hong et al., 2016). However, when g is nonconvex, such as when a nonconvex regularizer is used in regularized risk minimization, the convergence of ADMM is still an open reseach problem.

Using the proposed transformation, we can decompose a nonconvex g as $\bar{g} + \breve{g}$, where $\bar{g}$ is concave and Lipschitz-smooth, while $\breve{g}$ is convex but possibly nonsmooth. Problem (17) can then be rewritten as

$$\min_{y, x^1, \ldots, x^M} \sum_{i=1}^M \bar{f}_i(x^i) + \breve{g}(y) \quad : \quad x^1 = \cdots = x^M = y, \tag{39}$$

where $\bar{f}_i(x) = f_i(x) + \frac{1}{M}\bar{g}(x)$. Let p^i be the dual variable for the constraint $x^i = y$. The augmented Lagrangian for (39) is

$$\mathcal{L}\left(y, x^1, \ldots, x^M, p^1, \ldots, p^M\right) = \breve{g}(y) + \sum_{i=1}^M \bar{f}_i(x^i) + (p^i)^\top (x^i - y) + \frac{\tau}{2}\|x^i - y\|_2^2.$$

Using (14) and (15), we have the following update equations at iteration t:

$$\begin{aligned} x_{t+1}^i &= \arg\min_{x^i} \bar{f}_i(x^i) + (p_t^i)^\top (x^i - y_t) + \frac{\tau}{2}\|x^i - y_t\|_2^2, \quad i = 1, \ldots, M, \\ y_{t+1} &= \arg\min_y \frac{1}{2}\left\| y - \sum_{i=1}^M \left(x_t^i + \frac{1}{\tau}p_t^i\right)\right\|_2^2 + \frac{1}{\tau}\breve{g}(y) = \text{prox}_{\frac{1}{\tau}\breve{g}}\left(\sum_{i=1}^M x_t^i + \frac{1}{\tau}p_t^i\right). \end{aligned} \tag{40}$$

As in previous sections, the proximal step in (40), which is associated with the convex $\breve{g}$, is usually easier to compute than the proximal step associated with the original nonconvex g. Moreover, since $\breve{g}$ is convex, convergence results in Theorem 2.4 of (Hong et al., 2016) can now be applied. Specifically, the sequence $\{y_t, \{x_t^i\}\}$ generated by the ADMM procedure is bounded and all its limit points are critical points of (39).

As in Remark 8, the alternative convexification approach based on adding a quadratic regularizer (Proposition 9) does not help. For example, when g is the nonconvex tree-structured lasso regularizer, after adding a quadratic regularizer $\frac{\rho}{2}\|y_{\mathcal{G}_j}\|_2^2$, the y_t update in (19) becomes

$$y_{t+1} = \arg\min_y \frac{1}{2}\left\|y - \sum_{i=1}^{M}\left(x_t^i + \frac{1}{\tau}p_t^i\right)\right\|_2^2 + \frac{1}{\tau}\sum_{j=1}^{K}\lambda_j\left(\kappa\left(\|y_{\mathcal{G}_j}\|_2\right) + \frac{\rho}{2}\|y_{\mathcal{G}_j}\|_2^2\right).$$

which is still difficult to solve.

4.4 Stochastic Variance Reduced Gradient

Variance reduction methods have been commonly used to speed up the often slow convergence of SGD. Examples include the stochastic variance reduced gradient (SVRG) and its proximal extension Prox-SVRG (Xiao and Zhang, 2014). They can be used for the following optimization problem

$$\min_x \sum_{i=1}^{N}\ell(y_i, a_i^\top x) + g(x), \tag{41}$$

where $\{(a_1, y_1), \ldots, (a_N, y_N)\}$ are the training samples, ℓ is a smooth convex loss function, and g is a convex regularizer. Recently, Prox-SVRG is also extended for nonconvex objectives. Reddi et al. (2016a) and Zhu and Hazan (2016) considered smooth nonconvex ℓ but without g. This is further extended to the case of smooth ℓ and convex nonsmooth g in (Reddi et al., 2016b). However, convergence is still unknown for the more general case where the regularizer g is also nonconvex.

Using the proposed transformation, (41) can be rewritten as

$$\min_x \sum_{i=1}^{N}\left(\ell(y_i, a_i^\top x) + \frac{1}{N}\bar{g}(x)\right) + \breve{g}(x),$$

where $\ell + \frac{1}{N}\bar{g}$ is smooth and $\breve{g}$ is convex. The convergence results of Theorem 1 of (Reddi et al., 2016b) can now be applied, which shows that SVRG generates a bounded sequence and all its limit points are critical points of (41).

As in Remark 8, adding a quadratic term to convexify the nonconvex regularizer does not make the corresponding proximal step easier, and so does not help.

4.5 With OWL-QN

In this section, we consider OWL-QN (Andrew and Gao, 2007) and its variant mOWL-QN (Gong and Ye, 2015b), which are efficient algorithms for the ℓ_1-regularization problem

$$\min_x f(x) + \mu\|x\|_1. \tag{42}$$

Recently, Gong and Ye (2015a) proposed a nonconvex generalization for (42), in which the standard ℓ_1 regularizer is replaced by the nonconvex $g(x) = \mu\sum_{i=1}^{d}\kappa(|x_i|)$:

$$\min_x f(x) + \mu\sum_{i=1}^{d}\kappa(|x_i|). \tag{43}$$

Gong and Ye (2015a) proposed a sophisticated algorithm (HONOR) which involves a combination of quasi-Newton and gradient descent steps. Though the algorithm is similar to OWL-QN and mOWL-QN, the convergence analysis in (Gong and Ye, 2015b) cannot be directly applied as the regularizer is nonconvex. Instead, a non-trivial extension was developed in (Gong and Ye, 2015a).

Here, by convexifying the nonconvex regularizer, (43) can be rewritten as

$$\min_x \bar{f}(x) + \mu\kappa_0\|x\|_1, \tag{44}$$

where $\bar{f}(x) = f(x) + \bar{g}(x)$, and $\bar{g}(x) = \mu\sum_{i=1}^d(\kappa(|x_i|) - \kappa_0|x_i|)$. It is easy to see that the analysis in (Gong and Ye, 2015b) can be extended to handle smooth but nonconvex $\bar{f}$. As a result, Theorem 1 in (Gong and Ye, 2015b) can still be applied. Thus, mOWL-QN is guaranteed to generate a bounded sequence and its limit points are critical points of (42).

As in previous subsections, adding a quadratic term to convexify the nonconvex regularizer does not help. The mOWL-QN can only work with the ℓ_1-regularizer, but not with the modified regularizer $\breve{\varsigma}(x) = \frac{\rho\mu}{2}\|x\|_2^2 + \mu\sum_{i=1}^d \kappa(|x_i|)$.

Problem (43) can be solved by either (i) directly using HONOR, or (ii) using mOWL-QN on the transformed problem (44). We believe that the latter approach is computationally more efficient. In (43), the Hessian depends on both terms in the objective, as the second-order derivative of κ is not zero in general. However, HONOR constructs the approximate Hessian using only information from f, and thus ignores the curvature information due to $\sum_{i=1}^d \kappa(|x_i|)$. On the other hand, the Hessian in (44) depends only on $\bar{f}$, as the Hessian due to $\|x\|_1$ is zero (Andrew and Gao, 2007), and mOWL-QN now extracts Hessian from $\bar{f}$. Hence, optimizing (44) with mOWL-QN is potentially faster, as all the second-order information is utilized. This will be verified empirically in Section 5.4.

4.6 Nonsmooth and Nonconvex Loss

In many applications, besides having nonconvex regularizers, the loss function may also be nonconvex and nonsmooth. In this section, we consider using the nonconvex functions in Figure 1 as the loss function. Thus, neither f nor g in (1) is convex, smooth. The optimization problem becomes even harder, and many existing algorithms cannot be used. In particular, the proximal algorithm requires f in (1) to be smooth (possibly nonconvex) (Gong et al., 2013; Li and Lin, 2015; Bot et al., 2016). The FW algorithm requires f in (4) to be smooth and convex (Jaggi, 2013). For the ADMM, it allows f in the consensus problem to be smooth, but g has to be convex (Hong et al., 2016). For problems of the form $\min_{x,z} f(y) + g(y) : y = Ax$, ADMM requires A to have full row-rank (Li and Pong, 2015). As will be seen, it is not satisfied for problems considered in this section. CCCP (Yuille and Rangarajan, 2002) and smoothing (Chen, 2012) are more general and can still be used, but are usually very slow.

In this section, we consider two application examples, and show how they can be efficiently solved with the proposed transformation.

4.6.1 TOTAL VARIATION IMAGE DENOISING

Using the (convex) ℓ_1 loss and (convex) TV regularizer introduced in Section 4.1.3, consider the following optimization problem:

$$\min_X \|Y - X\|_1 + \mu\text{TV}(X), \tag{45}$$

where $Y \in \mathbb{R}^{m\times n}$ is a given corrupted image, and X is the target image to be recovered. The use of nonconvex loss and regularizer often produce better performance (Yan, 2013). Thus, we consider the following nonconvex extension:

$$\min_X \sum_{i=1}^{m}\sum_{j=1}^{n} \kappa\left(\left|[Y-X]_{ij}\right|\right) + \mu \sum_{i=1}^{m-1}\sum_{j=1}^{m} \kappa\left(\left|[D_v X]_{ij}\right|\right) + \mu \sum_{i=1}^{n}\sum_{j=1}^{n-1} \kappa\left(\left|[XD_h]_{ij}\right|\right), \quad (46)$$

where both the loss and regularizer are nonconvex and nonsmooth. As discussed above, this can be solved by CCCP and smoothing. However, as will be experimentally demonstrated in Section 5.5, the convergence is slow.

Using the proposed transformation on both the loss and regularizer, problem (46) can be transformed to the following problem:

$$\min_X \bar{f}(X) + \kappa_0\|X-Y\|_1 + \kappa_0\mu\mathrm{TV}(X), \quad (47)$$

where

$$\begin{aligned}\bar{f}(X) =& \sum_{i=1}^{m}\sum_{j=1}^{n} \kappa\left(\left|[Y-X]_{ij}\right|\right) - \kappa_0\|Y-X\|_1 \\ &+ \mu\left[\sum_{i=1}^{m-1}\sum_{j=1}^{m} \kappa\left(\left|[D_v X]_{ij}\right|\right) - \kappa_0\|D_v X\|_1 + \sum_{i=1}^{n}\sum_{j=1}^{n-1} \kappa\left(\left|[XD_h]_{ij}\right|\right) - \kappa_0\|XD_h\|_1\right]\end{aligned}$$

is smooth and nonconvex. As (47) is not a consensus problem, the method in (Hong et al., 2016) cannot be used. To use the ADMM algorithm in (Li and Pong, 2015), extra variables and constraints $Z_v = D_v X$ and $Z_h = XD_h$ have to be imposed. However, the full row-rank condition in (Li and Pong, 2015) does not hold.

In this section, we consider the proximal algorithm. Given some Z, the proximal step in (47) is

$$\arg\min_X \frac{1}{2}\|X-Z\|_F^2 + \frac{1}{\tau}\left(\|X-Y\|_1 + \mu\mathrm{TV}(X)\right), \quad (48)$$

where τ is the stepsize. Though this has no closed-form solution, $\|X-Y\|_1 + \mu\mathrm{TV}(X)$ in (48) is convex and one can monitor inexactness of the proximal step via the duality gap. Thus, we can use the proposed inexact nmAPG algorithm in Algorithm 3 for (47). It can be shown that the dual of (48) is

$$\begin{aligned}\min_{W,P,Q}\quad & \frac{1}{2\tau}\|W + \mu D_v^\top P + \mu Q D_h^\top\|_F^2 - \langle Z, W\rangle - \mu\langle D_v Z, P\rangle - \mu\langle ZD_h, Q\rangle + \langle Y, W\rangle \\ \text{s.t.}\quad & \|W\|_\infty \le 1, \|P\|_\infty \le 1 \text{ and } \|Q\|_\infty \le 1,\end{aligned} \quad (49)$$

and the primal variable can be recovered as $X = Z - \frac{1}{\tau}(W + \mu D_v^\top P + \mu Q D_h^\top)$. By substituting the obtained X into (48) and $\{W, P, Q\}$ into (49), the duality gap can be computed in $O(mn)$ time. As (49) is a smooth and convex problem, both accelerated gradient descent (Nesterov, 2013) and L-BFGS (Nocedal and Wright, 2006) can be applied. Algorithm 3 is then guaranteed to converge to a critical point of (46) (Theorem 6 and Proposition 7).

Note that it is more advantageous to transform both the loss and regularizer in (47). If only the regularizer in (46) is transformed, we obtain

$$\bar{f}_{\text{TV}}(X) + \sum_{i=1}^{m}\sum_{j=1}^{n} \kappa\left(\left|[Y-X]_{ij}\right|\right) + \kappa_0\mu\text{TV}(X), \tag{50}$$

where

$$\bar{f}_{\text{TV}}(X) = \mu\left[\sum_{i=1}^{m-1}\sum_{j=1}^{m} \kappa\left(\left|[D_vX]_{ij}\right|\right) - \kappa_0\|D_vX\|_1 + \sum_{i=1}^{n}\sum_{j=1}^{n-1} \kappa\left(\left|[XD_h]_{ij}\right|\right) - \kappa_0\|XD_h\|_1\right]$$

is nonconvex. The corresponding proximal step for (50) is

$$\arg\min_X \frac{1}{2}\|X-Z\|_F^2 + \frac{1}{\tau}\left(\sum_{i=1}^{m}\sum_{j=1}^{n} \kappa\left(\left|[Y-X]_{ij}\right|\right) + \kappa_0\mu\text{TV}(X)\right). \tag{51}$$

While the proximal steps in both (48) and (51) have no closed-form solution, working with (48) is more efficient. As (48) is convex, its dual can be efficiently solved with methods such as accelerated gradient descent and L-BFGS. In contrast, (51) is nonconvex, its duality gap is nonzero, and so can only be solved in the primal with slower methods like CCCP and smoothing. Besides, one can only use the more expensive nmAPG (Algorithm 2) but not the proposed inexact proximal algorithm.

One may also consider simultaneously transforming both the loss and regularizer using the decomposition discussed in Remark 8. However, it is not helpful here. By adding and subtracting a quadratic term, the objective in (46) can be decomposed as $\breve{\varsigma}(X) + \hat{\varsigma}(X)$, where

$$\begin{aligned}
\hat{\varsigma}(X) =& -\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{\rho}{2}[Y-X]_{ij}^2 - \frac{\rho\mu}{2}\sum_{i=1}^{m-1}\sum_{j=1}^{m}[D_vX]_{ij}^2 - \frac{\rho\mu}{2}\sum_{i=1}^{n}\sum_{j=1}^{n-1}[XD_h]_{ij}^2,\\
\breve{\varsigma}(X) =& \sum_{i=1}^{m}\sum_{j=1}^{n}\left(\kappa(|\,[Y-X]_{ij}\,|) + \frac{\rho}{2}[Y-X]_{ij}^2\right)\\
&+ \mu\sum_{i=1}^{m-1}\sum_{j=1}^{m}\left(\kappa(|[D_vX]_{ij}|) + \frac{\rho}{2}[D_vX]_{ij}^2\right) + \mu\sum_{i=1}^{n}\sum_{j=1}^{n-1}\left(\kappa(|[XD_h]_{ij}|) + \frac{\rho}{2}[XD_h]_{ij}^2\right).
\end{aligned}$$

We need to solve the proximal step associated with $\breve{\varsigma}(X)$, which is difficult.

4.6.2 ROBUST SPARSE CODING

The second application is robust sparse coding, which has been popularly used in face recognition (Yang et al., 2011), image analysis (Lu et al., 2013) and background modeling (Zhao et al., 2011). Given an observed signal $y \in \mathbb{R}^m$, the goal is to seek a robust sparse representation $x \in \mathbb{R}^d$ of y based on the dictionary $D \in \mathbb{R}^{m\times d}$ (which is assumed to be fixed here). Mathematically, it is formulated as the following optimization problem:

$$\min_x \|y - Dx\|_1 + \mu\|x\|_1.$$

Its nonconvex extension is:

$$\min_x \sum_{j=1}^{m} \kappa(|[y - Dx]_j|) + \mu \sum_{i=1}^{d} \kappa(|x_i|). \tag{52}$$

Using the proposed transformation, problem (52) becomes

$$\min_x \bar{f}(x) + \kappa_0 \|y - Dx\|_1 + \mu\kappa_0 \|x\|_1, \tag{53}$$

where

$$\bar{f}(x) = \mu \sum_{j=1}^{d} \kappa(|x_j|) - \kappa_0 \mu \|x\|_1 + \sum_{j=1}^{m} \kappa(|[y - Dx]_j|) - \kappa_0 \|y - Dx\|_1$$

is smooth and nonconvex. Again, we use the inexact nmAPG algorithm in Algorithm 3. The proximal step for (53) is

$$\arg\min_x \frac{1}{2}\|x - z\|_2^2 + \frac{1}{\tau}(\|y - Dx\|_1 + \mu\|x\|_1), \tag{54}$$

where τ is the stepsize and z is given. As in Section 4.6.1, $\|y - Dx\|_1 + \mu\|x\|_1$ in (54) is convex, and one can monitor inexactness of the proximal step by the duality gap. The dual of (54) is

$$\min_{p,q} \frac{1}{2\tau}\|D^\top p + \mu q\|_2^2 - p^\top Dz - \mu q^\top z \ : \ \|p\|_\infty \le 1, \|q\|_\infty \le 1. \tag{55}$$

As in (49), this can be solved with L-BFGS or accelerated gradient descent. The primal variable can be recovered as $x = z - \frac{1}{\tau}(D^\top p + \mu q)$, and the duality gap can be checked in $O(md)$ time.

If only the regularizer is transformed, we obtain

$$\min_x \sum_{j=1}^{m} \kappa(|[y - Dx]_j|) + \bar{f}_{\text{RSC}}(x) + \kappa_0\mu\|x\|_1, \tag{56}$$

where $\bar{f}_{\text{RSC}}(x) = \mu \sum_{j=1}^{d} \kappa(|x_j|) - \kappa_0\mu\|x\|_1$. The corresponding proximal step is

$$\arg\min_x \frac{1}{2}\|x - z\|_2^2 + \sum_{j=1}^{m} \kappa(|[y - Dx]_j|) + \kappa_0\mu\|x\|_1, \tag{57}$$

which still involve the nonconvex function κ. As in Section 4.6.1, (55) is easier to solve than (57).

As in previous sections, adding a quadratic term to convexify the loss and regularizer is not helpful. The objective in (52) will then be decomposed as $\breve{\varsigma}(X) + \hat{\varsigma}(X)$, where

$$\begin{aligned}
\hat{\varsigma}(X) &= -\frac{\rho}{2}\sum_{j=1}^{m}[y - Dx]_j^2 - \frac{\rho\mu}{2}\sum_{i=1}^{d} x_i^2, \\
\breve{\varsigma}(X) &= \sum_{j=1}^{m}\left(\kappa(|[y - Dx]_j|) + \frac{\rho}{2}[y - Dx]_j^2\right) + \mu\sum_{i=1}^{d}\left(\kappa(|x_i|) + \frac{\rho}{2}x_i^2\right),
\end{aligned}$$

and the proximal step associated with $\breve{\varsigma}(X)$ is again difficult to solve.

5. Experiments

In this section, we perform experiments on using the proposed procedure with (i) proximal algorithms (Sections 5.1 and 5.2); (ii) Frank-Wolfe algorithm (Section 5.3); (iii) comparision with HONOR (Section 5.4) and (vi) image denoising (Section 5.5). Experiments are performed on a PC with Intel i7 CPU and 32GB memory. All algorithms are implemented in Matlab.

5.1 Nonconvex Sparse Group Lasso

In this section, we perform experiments on the nonconvex sparse group lasso model in Section 4.1.1. For simplicity, assume that $\mu_1 = \cdots = \mu_K = \mu$. Using the square loss, (25) becomes

$$\min_x \frac{1}{2}\|y - A^\top x\|_2^2 + \lambda \sum_{i=1}^{d} \kappa\left(|x_i|\right) + \mu \sum_{j=1}^{K} \kappa\left(\|x_{\mathcal{G}_j}\|_2\right), \tag{58}$$

where $A = [a_1, \ldots, a_N]$. In this experiment, we use the LSP regularizer in Table 1 (with $\theta = 0.5$) as $\kappa(\cdot)$. The synthetic data set is generated as follows. We set $d = 200,000$. The ground-truth parameter $\bar{x} \in \mathbb{R}^d$ is divided into 200 non-overlapping groups: $\{1, \ldots, 1000\}$, $\{1001, \ldots, 2000\}$, $\ldots$, $\{199001, \ldots, 200000\}$ (Figure 3). We randomly set 87.5% of the groups to zero. In each nonzero group, we randomly set 50% of its features to zero, and generate the nonzero features from the standard normal distribution $\mathcal{N}(0, 1)$. The whole data set has $400,000$ samples, and entries of the input $A \in \mathbb{R}^{200{,}000 \times 400{,}000}$ is a sparse matrix with 0.01% nonzero elements which are generated from $\mathcal{N}(0, 1)$. The ground-truth output is $\bar{y} = A^\top \bar{x}$. This is then corrupted by random Gaussian noise ϵ in $\mathcal{N}(0, 0.05)$ to produce $y = \bar{y} + \epsilon$.

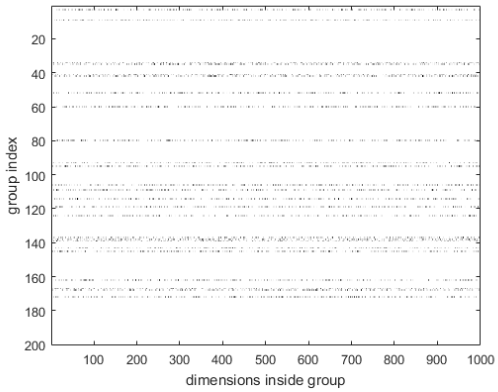


Figure 3: An example ground-truth parameter $\bar{x} \in \mathbb{R}^{200000}$. It is reshaped as a 200×1000 matrix, with each row representing a group.

The proposed algorithm will be called N2C (Nonconvex-to-Convex). The proximal step of the convexified regularizer $\breve{g}(x) = \kappa_0(\lambda\|x\|_1 + \sum_{j=1}^{K} \mu_j \|x_{\mathcal{G}_j}\|_2)$ is obtained using the algorithm in (Yuan et al., 2011). The nmAPG algorithm (Algorithm 2) in (Li and Lin, 2015) is used for optimization. This will be compared with the following state-of-the-art algorithms:

1. SCP: Sequential convex programming (Lu, 2012), in which the LSP regularizer is decomposed as in (28).

2. GIST (Gong et al., 2013): Since the nonconvex regularizer is not separable, the associated proximal operator has no closed-form solution. Instead, we use SCP (with warm-start) to solve it numerically.

3. GD-PAN (Zhong and Kwok, 2014): It performs gradient descent with the proximal average (Bauschke et al., 2008) of the nonconvex regularizers. Closed-form solutions for the proximal operators of each individual regularizer are obtained separately, and then averaged.

4. nmAPG with the original nonconvex regularizer: As in GIST, the proximal step is solved numerically by SCP.

5. As a baseline, we also compare with the FISTA (Beck, 2009) algorithm, which solves the convex sparse group lasso model (with κ removed from (58)).

We do not compare with the concave-convex procedure (Yuille and Rangarajan, 2002), which has been shown to be slow (Gong et al., 2013; Zhong and Kwok, 2014).

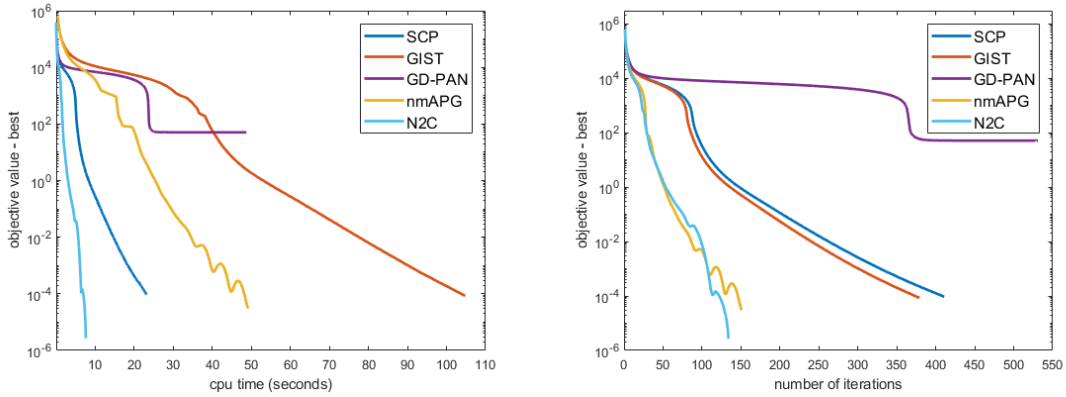
We use 50% of the data for training, another 25% as validation set to tune λ, μ in (58), and the rest for testing. The stepsize is fixed at $\tau = \sigma_1(A^\top A)$. For performance evaluation, we use the (i) testing root-mean-squared error (RMSE) on the predictions; (ii) absolute error between the obtained parameter $\hat{x}$ and the corresponding ground-truth $\bar{x}$: ABS $= \|\hat{x} - \bar{x}\|_1/d$; and (iii) CPU time. To reduce statistical variability, the experimental results are averaged over 5 repetitions.

Results are shown in Table 3. As can be seen, all the nonconvex models obtain better errors (RMSE and ABS) than the convex FISTA. As for the training speed, N2C is the fastest. SCP, GIST, nmAPG and N2C all solve the original problem (1), and have the same recovery performance. GD-PAN solves an approximate problem in each iteration, and its error is slightly worse than the other nonconvex algorithms on this data set.

	non-accelerated			accelerated		convex
	SCP	GIST	GD-PAN	nmAPG	N2C	FISTA
RMSE	**48.9±0.1**	**48.9±0.1**	49.2±0.1	**48.9±0.1**	**48.9±0.1**	64.0±0.1
ABS	**3.1±0.1**	**3.1±0.1**	5.5±0.1	**3.1±0.1**	**3.1±0.1**	13.0±0.1
CPU time(sec)	23.2±7.5	105.6±24.6	33.7±11.7	43.0±3.5	**7.9±1.2**	5.1±0.8

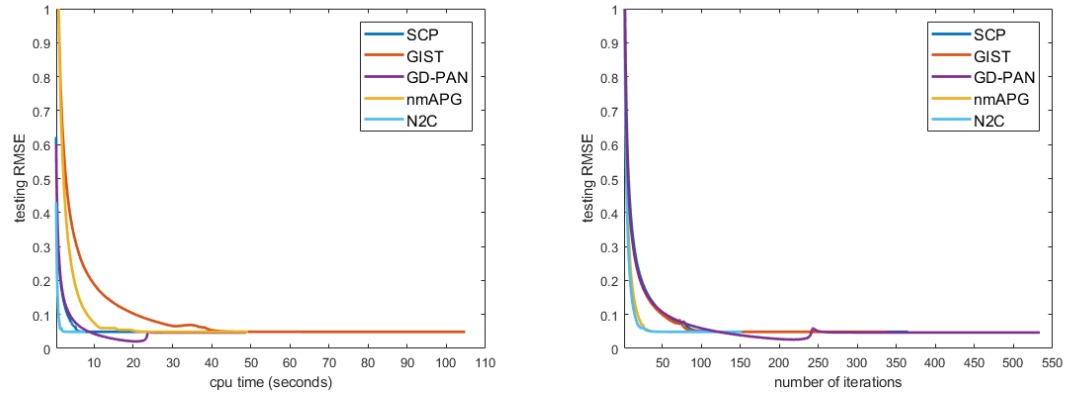
Table 3: Results on nonconvex sparse group lasso. RMSE and ABS are scaled by 10^{-3}, and the CPU time is in seconds. The best and comparable results (according to the paired t-test with 95% confidence) are highlighted.

Figure 4(a) shows convergence of the objective with time for a typical run. SCP, GIST, nmAPG and N2C all converge towards the same objective value. GD-PAN can only approximate the original problem. Thus, it converges to an objective value which is larger than the others. Figure 4(b) shows the convergence with number of iterations. As can be seen, N2C and nmAPG, which are based on the same state-of-the-art proximal algorithm (Algorithm 2), require nearly the same number of iterations for convergence. However, as N2C has an inexpensive closed-form solution for its proximal step, it is much faster when measured in terms of time (Figure 4(a)). Figure 5 shows convergence of the testing RMSE. Its behaviour is similar to those observed in Figure 4. Overall, N2C, which uses acceleration and inexpensive proximal step, is the fastest.



(a) Objective vs CPU time (seconds). (b) Objective vs number of iterations.

Figure 4: Convergence of objective on nonconvex sparse group lasso. In the ordinate, "best" refers to the smallest objective value obtained among the various methods. Note that FISTA is not shown as its (convex) objective is different from the others.



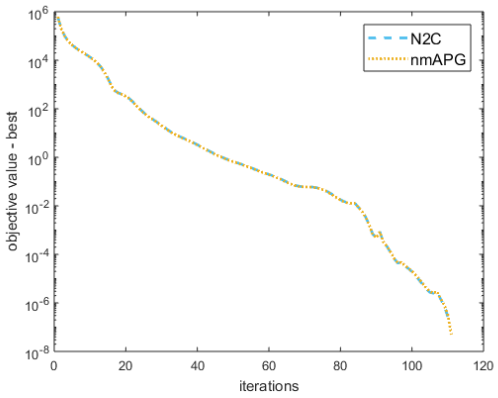
(a) Testing RMSE vs CPU time (seconds). (b) Testing RMSE vs number of iterations.

Figure 5: Convergence of the testing RMSE on nonconvex sparse group lasso.

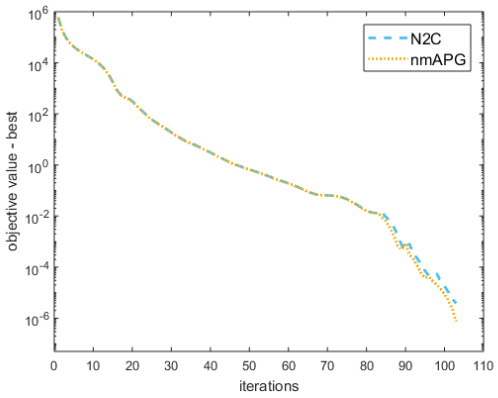
5.1.1 IMPACT OF NONCONVEXITY OF THE LOSS ON NMAPG

Recall that N2C uses nmAPG as the underlying proximal algorithm solver. Thus, in the above experiment, we have compared the performance of using nmAPG on (i) the transformed problem, which has a convex regularizer and a more nonconvex loss; and (ii) the original problem, which has a nonconvex regularizer and a less nonconvex loss). In the following, we will provide more empirical evidence that the increased nonconvexity of the transformed loss does not harm convergence of nmAPG. Instead of tuning the regularization parameter μ in (58) using the validation set, we vary μ in $\{0.01, 0.1, 1\}$. A larger μ makes the regularizer g more nonconvex, and more nonconvexity will be transferred to $\bar{f}$ by the proposed transformation. The other parts of the experimental setup are the same. Figure 6 compares the convergence behavior of N2C and nmAPG w.r.t. the number

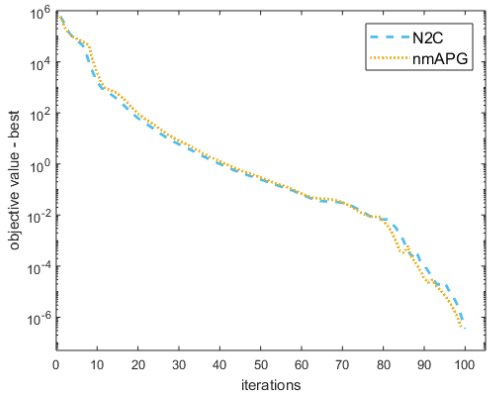
of iterations. As can be seen, they are almost identical, which agrees with Figure 4(b). Hence, nonconvexity of the loss have little effect on the empirical performance of nmAPG.



(a) $\mu = 0.01$.



(b) $\mu = 0.1$.



(c) $\mu = 1$.

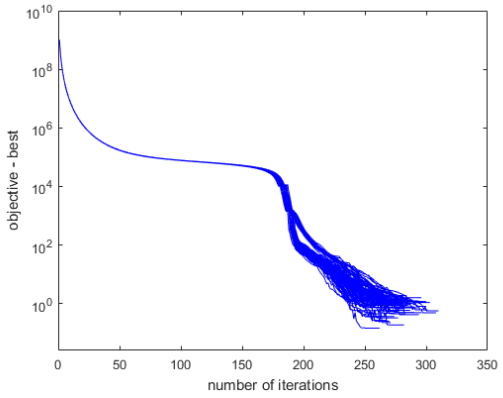
Figure 6: Convergence of N2C and nmAPG on nonconvex sparse group lasso with various μ.

5.1.2 NONCONVEX OPTIMIZATION

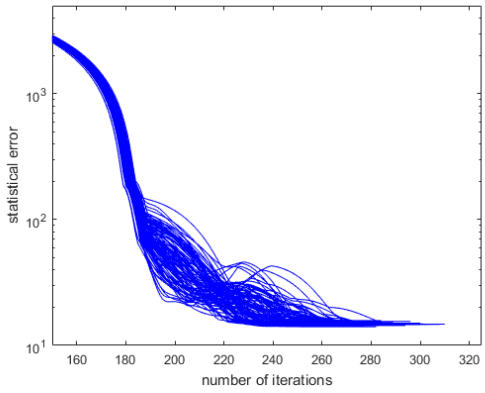
As discussed in (Ge et al., 2015), there are two main theoretical issues in optimizing nonconvex functions. The first issue is that it may be hard to find the global minimum. However, it has been shown that for composite minimization problems with nonconvex regularizer and the loss function satisfying the restricted strong convexity condition (of which nonconvex lasso is such an example), *all stationary points* can have nearly the same statistical error[1] (Loh and Wainwright, 2015). Note that with the proposed transformation, the transformed optimization problem has the same stationary points as the original problem, and so all stationary points still have nearly the same statistical error. To verify this, we experiment on the nonconvex sparse group lasso with 100 different initializations. Figure 7 shows the convergence behaviour and the obtained statistical error from the N2C algorithm. As can be seen from Figure 7(a), the differences in the final objective values obtained from different

1. Let $\bar{x}$ be the ground-truth predictor and x be an arbitrary stationary point. The statistical error is defined as $\|x - \bar{x}\|_2$.

initialization are small. From Figure 7(b), we can also see that the obtained solutions have nearly the same statistical error.



(a) objective vs number of iterations.



(b) statistical error vs number of iterations.

Figure 7: Objective / statistical error on nonconvenx sparse group lasso ($\mu = 1$). Here, N2C algorithm is used, and the "best" in Figure 7(a) is the lowest objective among the 100 random runs.

The second issue is that even finding a local minimum can be hard, as the optimizer may get trapped in saddle points. However, recent papers (Ge et al., 2015, 2017, 2016; Lee et al., 2016, 2017) have shown that batch gradient descent and noisy stochastic gradient descent almost never converge to saddle points, and can converge to a local minimizer. We expect that a similar result also holds for proximal gradient descent, but will leave this as future work.

5.2 Nonconvex Tree-Structured Group Lasso

In this section, we perform experiments on the nonconvex tree-structured group lasso model in Section 4.1.2. Following (Liu and Ye, 2010), we use the face data set *JAFFE*[2], which contains 213 256×256 images with seven facial expressions: anger, disgust, fear, happy, neutral, sadness and surprise. Their tree structure, which is based on pixel neighborhoods, is also used here. The number of groups K is 341.

Since our goal is only to demonstrate usefulness of the proposed convexification scheme, we focus on the binary classification problem "anger vs not-anger" (with 30 anger images and 183 non-anger images). The logistic loss is used, which is more appropriate for classification. Given training samples $\{(a_1, y_1), \ldots, (a_N, y_N)\}$, the optimization problem is then

$$\min_x \sum_{i=1}^{N} w_i \log\left(1 + \exp(-y_i \cdot a_i^\top x)\right) + \mu \sum_{i=1}^{K} \lambda_i \kappa\left(\|x_{\mathcal{G}_i}\|_2\right), \tag{59}$$

where $\kappa(\cdot)$ is the LSP regularizer (with $\theta = 0.5$), w_i's are weights (set to be the reciprocal of the size of sample i's class) used to alleviate class imbalance, and $\lambda_i = 1/\sqrt{\|\mathcal{G}_i\|_1}$ as in (Liu and Ye,

2. `http://www.kasrl.org/jaffe.html`

2010). We use 60% of the data for training, 20% as validation set to tune μ, and the rest for testing. For the proposed N2C algorithm, the proximal step of the convexified regularizer is obtained as in (Liu and Ye, 2010).

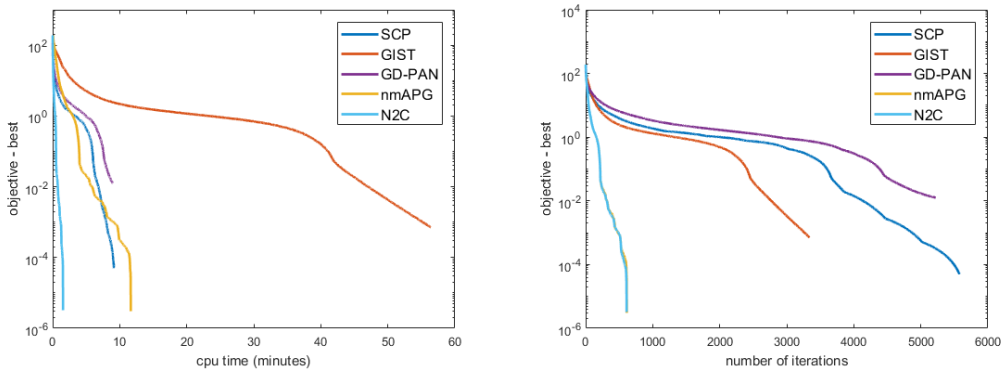
As in Section 5.1, we compare the proposed N2C with SCP, GIST, GD-PAN, nmAPG, and FISTA. The stepsize η is obtained by line search. For performance evaluation, we use (i) the testing accuracy; (ii) solution sparsity (i.e., percentage of nonzero elements); and (iii) CPU time. To reduce statistical variability, the experimental results are averaged over 5 repetitions.

Results are shown in Table 4. As can be seen, all nonconvex models have similar testing accuracies, and they again outperform the convex model. Moreover, solutions from the nonconvex models are sparser. Overall, N2C is the fastest and has the sparsest solution.

	non-accelerated			accelerated		convex
	SCP	GIST	GD-PAN	nmAPG	N2C	FISTA
testing accuracy (%)	**99.5±1.0**	**99.5±1.0**	**99.5±1.0**	**99.5±1.0**	**99.5±1.0**	96.7±1.3
sparsity (%)	**9.8±0.9**	11.6±3.6	**9.7±0.9**	**9.6±0.9**	**9.6±0.8**	24.1±0.8
CPU time (min)	10.3±1.5	68.0±17.2	12.0±2.4	11.7±1.3	**1.7±0.2**	0.5±0.1

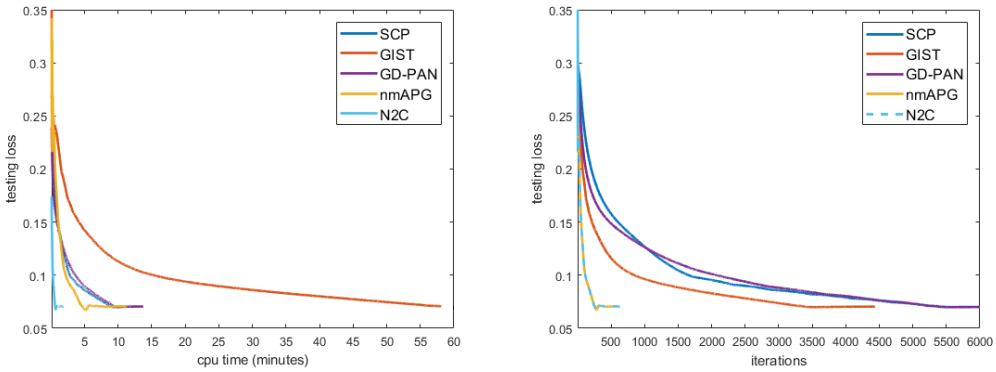
Table 4: Results on tree-structured group lasso. The best and comparable results (according to the paired t-test with 95% confidence) are highlighted.

Figure 8 shows convergence of the algorithms versus CPU time and number of iterations. The observations are similar to those in Figure 8, and N2C is the fastest. GIST is the slowest, as it does not utilize acceleration and its proximal step is solved numerically which is expensive. GD-PAN converges to a less optimal solution due to its use of approximation. Moreover, as in Section 5.1, nmAPG and N2C show similar convergence behavior w.r.t. the number of iterations (Figure 8(b)), but N2C is much faster w.r.t. time (Figure 8(a)). Convergence of the testing loss is shown in Figure 9.



(a) Objective vs CPU time (minutes). (b) Objective vs number of iterations.

Figure 8: Convergence of objective on nonconvex tree-structured group lasso. Note that the curves for nmAPG and N2C overlap in Figure 8(b).



(a) Testing loss vs CPU time (minutes). (b) Testing loss vs number of iterations.

Figure 9: Convergence of testing loss on nonconvex tree-structured group lasso.

As in Section 5.1.1, Figure 10 shows the convergence of N2C and nmAPG w.r.t. the number of iterations when μ is varied in $\{0.01, 0.1, 1\}$. Again, their behavior are almost identical. Hence, the increased nonconvexity of the transformed loss does not harm convergence of nmAPG.

5.3 Nonconvex Low-Rank Matrix Completion

In this section, we perform experiments on nonconvex low-rank matrix completion (Section 4.2), with the square loss in (30). The LSP regularizer is used, with $\theta = \sqrt{\mu}$ as in (Yao et al., 2015). We use the data sets MovieLens, Netflix and Yahoo, which have been commonly used for evaluating matrix completion (Mazumder et al., 2010; Wen et al., 2012; Hsieh and Olsen, 2014). The MovieLens and Netflix data sets contain ratings $\{1, 2, \ldots, 5\}$ assigned by various users on movies, while the Yahoo data set contains ratings $\{10, 20, \ldots, 100\}$ on music. Following (Yao et al., 2015), we normalize the ratings to zero mean and unit variance.

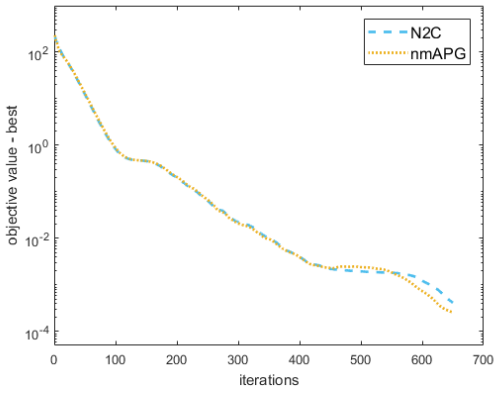
		#users	#movies	#ratings
MovieLens	100K	943	1,682	100,000
	1M	6,040	3,449	999,714
	10M	69,878	10,677	10,000,054
Netflix		480,189	17,770	100,480,507
Yahoo		249,012	296,111	62,551,438

Table 5: Recommendation data sets used in the experiments.

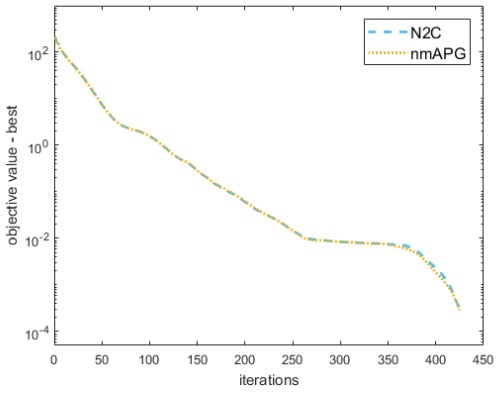
5.3.1 MOVIELENS

The proposed FW procedure (Algorithm 4), denoted N2C-FW, is compared with the following algorithms:

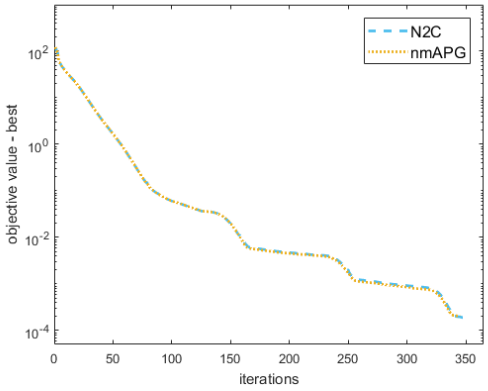
1. FaNCL (Yao et al., 2015): This is a recent nonconvex matrix regularization algorithm. It is based on the proximal algorithm using efficient approximate SVD and automatic thresholding of singular values.



(a) $\mu = 0.01$.



(b) $\mu = 0.1$.



(c) $\mu = 1$.

Figure 10: Convergence of N2C and nmAPG on tree-structured group lasso with various μ.

2. LMaFit (Wen et al., 2012): It factorizes X as a product of low-rank matrices $U \in \mathbb{R}^{m\times k}$ and $V \in \mathbb{R}^{n\times k}$. The nonconvex objective $\frac{1}{2}\|P_\Omega(UV^\top - O)\|_F^2$ is then minimized by alternating minimization on U and V using gradient descent.

3. Active subspace selection (denoted "active") (Hsieh and Olsen, 2014): This solves the (convex) nuclear norm regularized problem (with κ being the identity function in (8)) by using the active row/column subspaces to reduce the optimization problem size.

We do not compare with IRNN (Lu et al., 2014) and GPG (Lu et al., 2015), which have been shown to be much slower than FaNCL (Yao et al., 2015).

Following (Yao et al., 2015), we use 50% of the ratings for training, 25% for validation and the rest for testing. For performance evaluation, we use (i) the testing RMSE; and (ii) the recovered rank. To reduce statistical variability, the experimental results are averaged over 5 repetitions.

Results are shown in Table 6. As can be seen, the nonconvex models (N2C-FW, FaNCL and LMaFit) achieve lower RMSEs than the convex model (active subspace selection), with N2C-FW having the smallest RMSE. Moreover, the convex model needs a much higher rank than the nonconvex models, which agrees with previous observations in (Mazumder et al., 2010; Yao et al.,

2015). Thus, its running time is also much longer than the others. Figure 11 compares convergence of N2C and FaNCL w.r.t. the objective in (30). The objectives of LMaFit and active subspace selection are different from N2C-FW, and thus are not shown. As can be seen, though FaNCL uses singular value thresholding to truncate the SVD, it does not control the rank as directly as N2C-FW and so is still slower. Figures 12 compares convergence of the testing RMSE on all algorithms. As the recovered matrix ranks for the nonconvex models are very low ($2-9$ in Table 6), N2C-FW is much faster than the others as it starts from a rank-one matrix and only increases its rank by one in each iteration.

		RMSE	rank	CPU time(sec)
100K	N2C-FW	**0.855±0.004**	2	**0.8±0.1**
	FaNCL	0.857±0.003	2	2.6±0.5
	LMaFit	0.867±0.004	2	1.8±0.2
	(convex) active	0.875±0.002	52	9.4±0.3
1M	N2C-FW	**0.785±0.001**	5	**19.9±0.5**
	FaNCL	0.786±0.001	5	53.6±6.5
	LMaFit	0.812±0.002	5	45.1±2.7
	(convex) active	0.811±0.001	106	124.6±1.3
10M	N2C-FW	**0.778±0.001**	9	313.0±2.2
	FaNCL	0.779±0.001	9	615.7±6.0
	LMaFit	0.797±0.001	9	**264.9±3.9**
	(convex) active	0.808±0.001	137	904.8±30.2

Table 6: Results on the MovieLens data sets. The best results (according to the paired t-test with 95% confidence) are highlighted.

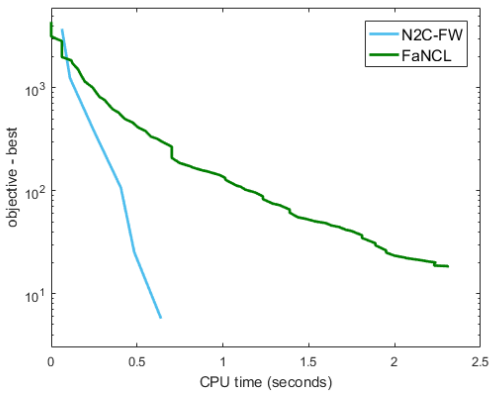
5.3.2 NETFLIX AND YAHOO

Next, we perform experiments on two very large recommendation data sets, Netflix and Yahoo (Table 5). We randomly use 50% of the observed ratings for training, 25% for validation and the rest for testing. As active subspace selection has been shown to be slower and inferior to the others (Table 6), it is not compared here. Each experiment is repeated five times. Results are shown in Table 7, and a more detailed convergence comparison with CPU time is shown in Figures 13 and 14. Again, N2C is much faster than FaNCL, and has the lowest testing RMSE.

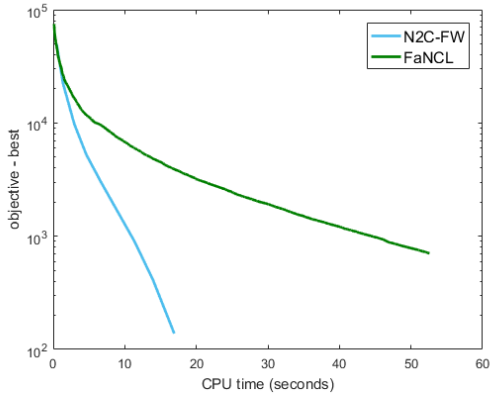
5.4 Comparison with HONOR

In this section, we experimentally compare the proposed method with HONOR (Section 4.5) on the model in (43), using the logistic loss and LSP regularizer. Following (Gong and Ye, 2015a), we fix $\mu = 1$ in (43), and θ in the LSP regularizer to 0.01μ. Experiments are performed on three large data sets,[3] kdd2010a, kdd2010b and url (Table 8). Both kdd2010a and kdd2010b are educational data sets, and the task is to predict students' successful attempts to answer concepts related to algebra.

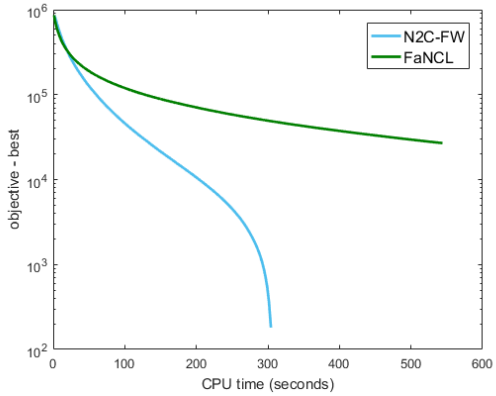
3. https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html



(a) MovieLens-100K.



(b) MovieLens-1M.



(c) MovieLens-10M.

Figure 11: Convergence of objective vs CPU time on the MovieLens data sets for nonconvex low-rank matrix completion.

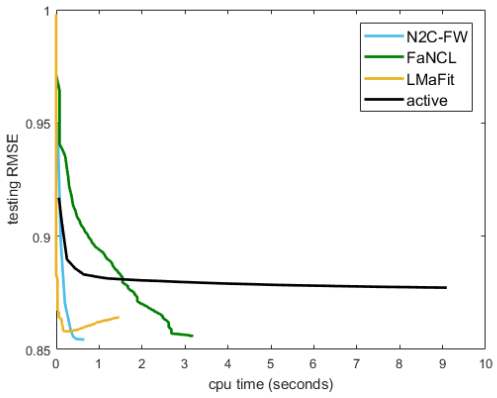
The url data set contains a collection of websites, and the task is to predict whether a particular website is malicious. We compare

1. running HONOR directly on (43). The threshold of the hybrid step in HONOR is set to 10^{-10}, which yields the best empirical performance in (Gong and Ye, 2015a);

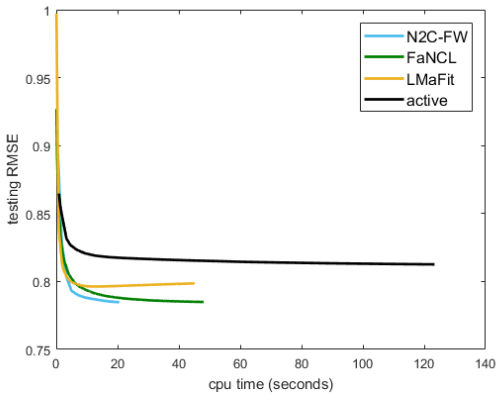
2. running mOWL-QN (Gong and Ye, 2015b) on the transformed problem (44).

To reduce statistical variability, the experimental results are averaged over 5 repetitions.

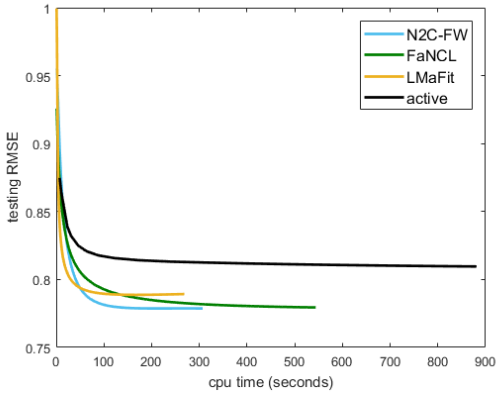
Figure 15 shows convergence of the objective (which is the same in (43) and (44)) with CPU time. As can be seen, mOWL-QN converges faster than HONOR. This validates our claim that the curvature information of the nonconvex regularizer helps.



(a) MovieLens-100K.



(b) MovieLens-1M.



(c) MovieLens-10M.

Figure 12: Convergence of testing RMSE vs CPU time on the MovieLens data sets for nonconvex low-rank matrix completion.

5.5 Image Denoising

In this section, we perform experiments on total variation image denoising with nonconvex loss and nonconvex regularizer (as introduced in Section 4.6.1). The LSP function (with $\theta = 1$) is used as κ in (46) on both the loss and regularizer. Eight popular images[4] are used (Figure 16). They are then corrupted by pepper-and-salt noise, with 10% of the pixels randomly set to 0 or 255 with equal probabilities.

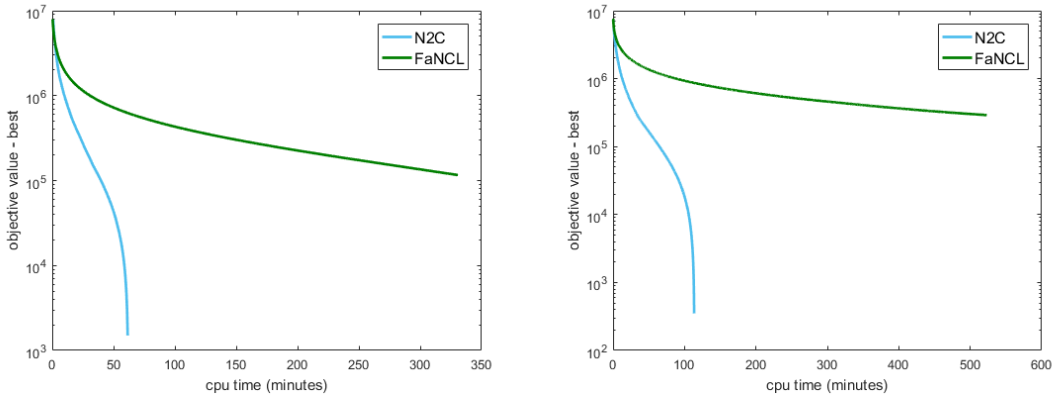
For performance evaluation, we use the RMSE $= \sqrt{\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}(X_{ij} - \bar{X}_{ij})^2}$, where $\bar{X} \in \mathbb{R}^{m\times n}$ is the clean image, and $X \in \mathbb{R}^{m\times n}$ is the recovered image. To tune μ, we pick the value that leads to the smallest RMSE on the first four images (boat, couple, fprint, hill). Denoising performance is then reported on the remaining images (house, lena, man, peppers).

The following algorithms will be compared:

4. http://www.cs.tut.fi/~foi/GCF-BM3D/

		RMSE	rank	CPU time(min)
Netflix	N2C-FW	**0.792±0.001**	13	**64.9±1.8**
	FaNCL	0.793±0.001	13	322.1±7.3
	LMaFit	0.807±0.001	15	98.2±1.8
Yahoo	N2C-FW	**0.643±0.001**	9	**111.3±13.5**
	FaNCL	0.650±0.001	9	444.7±92.4
	LMaFit	0.666±0.001	12	179.3±37.5

Table 7: Results on the Netflix and Yahoo data sets. The best results (according to the paired t-test with 95% confidence) are highlighted.



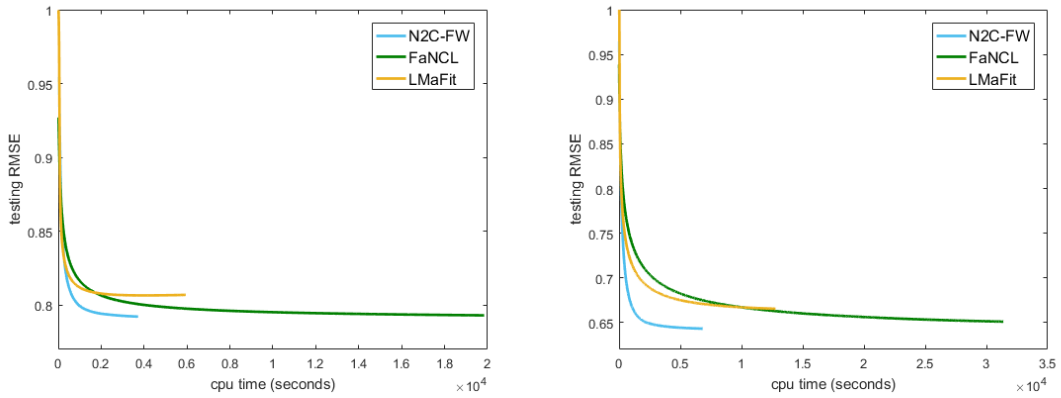
(a) Netflix. (b) Yahoo.

Figure 13: Convergence of objective vs CPU time on the Netflix and Yahoo data sets for nonconvex low-rank matrix completion.

1. CCCP (Yuille and Rangarajan, 2002): Proposition 9 is used to construct the DC decomposition for κ (Details are in Appendix B.1);
2. Smoothing (Chen, 2012): The nonsmooth κ is smoothed, and then gradient descent is used (Details are in Appendix B.2);
3. nmAPG (Li and Lin, 2015): This optimizes (50) with Algorithm 2, and the exact proximal step is solved numerically using CCCP;
4. inexact-nmAPG: This optimizes (47) with Algorithm 3 (with $\epsilon_t = 0.95^t$), and the inexact proximal step is solved numerically using L-BFGS.
5. As a baseline, we also compare with ADMM (Boyd et al., 2011) with the convex formulation.

To reduce statistical variability, the experimental results are averaged over 5 repetitions.

The RMSE results are shown in Table 9. As can be seen, the (convex) ADMM formulation leads to the highest RMSE, while CCCP, smoothing, nmAPG and inexact-nmAPG have the same RMSE which is lower than that of ADMM. This agrees with previous observations that nonconvex formulations can yield better performance than the convex ones. Timing results are shown in



(a) Netflix. (b) Yahoo.

Figure 14: Convergence of testing RMSE vs CPU time on the Netflix and Yahoo data sets for nonconvex low-rank matrix completion.

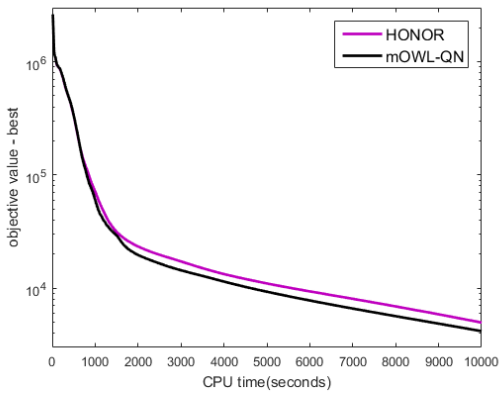
	kdd2010a	kdd2010b	url
number of samples	510,302	748,401	2,396,130
number of features	20,216,830	29,890,095	3,231,961

Table 8: Data sets used in the comparison with HONOR.

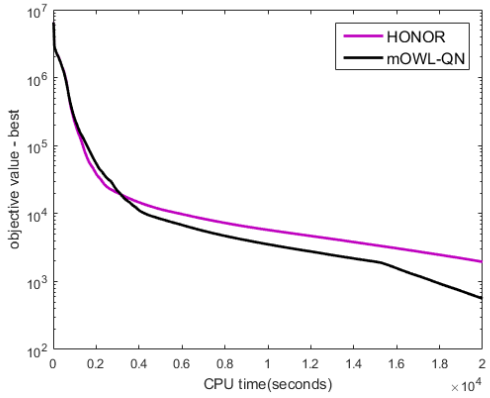
Table 10 and Figure 17. As can be seen, smoothing has low iteration complexity but suffers from slow convergence. CCCP and nmAPG both need to exactly solve a subproblem, and thus are also slow. The inexact-nmAPG algorithm does not guarantee the objective value to be monotonically decreasing as iteration proceeds. As the inexactness is initially large, there is an initial spike in the objective. However, inexact-nmAPG then quickly converges, and is much faster than all the baselines.

	house	lena	man	peppers
CCCP	**0.0205±0.0010**	**0.0174±0.0005**	**0.0223±0.0002**	**0.0207±0.0009**
smoothing	**0.0205±0.0011**	**0.0174±0.0005**	**0.0223±0.0002**	**0.0207±0.0009**
nmAPG	**0.0205±0.0010**	**0.0174±0.0005**	**0.0223±0.0002**	**0.0207±0.0009**
inexact-nmAPG	**0.0205±0.0010**	**0.0174±0.0005**	**0.0223±0.0002**	**0.0207±0.0009**
(convex) ADMM	0.0223±0.0011	0.0193±0.0005	0.0242±0.0002	0.0229±0.0008

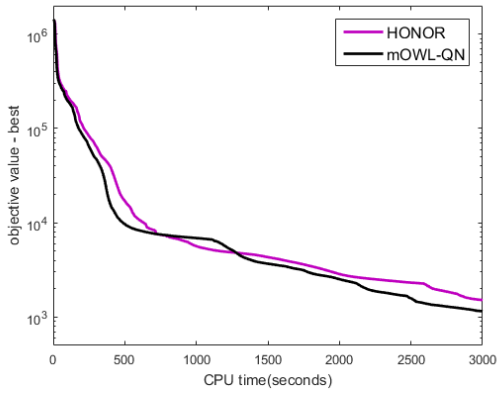
Table 9: RMSE for image denoising. The best RMSE's (according to the paired t-test with 95% confidence) are highlighted.



(a) kdd2010a.



(b) kdd2010b.



(c) url.

Figure 15: Convergence of the objective vs CPU time for HONOR and mOWL-QN.

(a) boat. (b) couple. (c) fprint. (d) hill. (e) house. (f) lena. (g) man. (h) peppers.

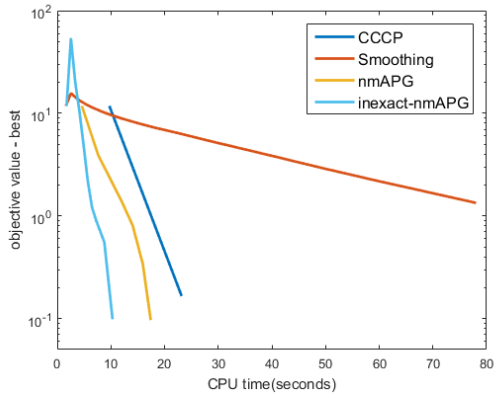
Figure 16: Samples images used in the denoising experiment. Top: Clean images; Bottom: Noisy images.

6. Conclusion

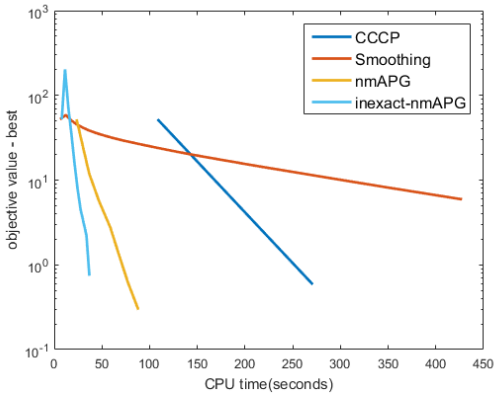
In this paper, we proposed a novel approach to learning with nonconvex regularizers that are variants of the convex ℓ_1-norm. By moving the nonconvexity associated with the nonconvex regularizer

	house	lena	man	peppers
CCCP	21.0±2.3	270.0±13.0	325.3±17.4	14.5±1.2
smoothing	75.5±2.0	433.1±4.8	437.7±6.8	61.9±1.7
nmAPG	19.4±2.3	91.4±7.3	104.4±2.7	16.1±1.8
inexact-nmAPG	**10.3±1.1**	**37.9±5.0**	**43.0±7.6**	**8.1±0.2**
(convex) ADMM	3.0±0.1	42.8±1.1	46.9±1.0	2.2±0.1

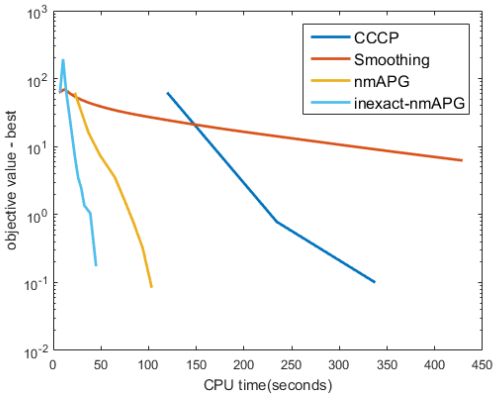
Table 10: CPU time (seconds) for image denoising. The shortest CPU time (according to the paired t-test with 95% confidence) are highlighted.



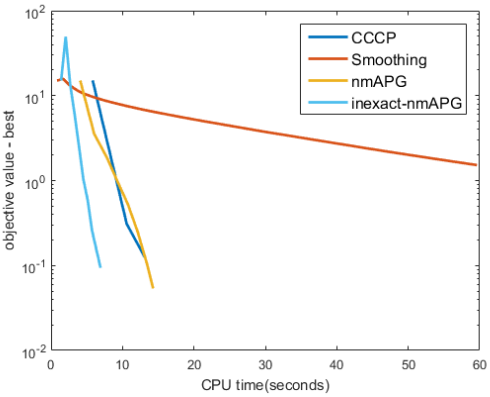
(a) house.



(b) lena.



(c) man.



(d) peppers.

Figure 17: CPU time (seconds) vs objective value on different images.

to the loss, the nonconvex regularizer is convexified to become a familiar convex regularizer while the augmented loss is still Lipschitz smooth. This allows one to reuse efficient algorithms originally designed for convex regularizers on the transformed problem. To illustrate usages with the proposed transformation, we plug it into many popular optimization algorithms. First, we consider the proximal algorithm, and showed that while the proximal step is expensive on the

original problem, it becomes much easier on the transformed problem. We further propose an inexact proximal algorithm, which allows inexact update of proximal step when it does not have a closed-form solution. Second, we combine the proposed convexification scheme with the Frank-Wolfe algorithm on learning low-rank matrices, and showed that its crucial linear programming step becomes cheaper and more easily solvable. As no convergence results exist on this nonconvex problem, we designed a novel Frank-Wolfe algorithm based on the proposed transformation and with convergence guarantee. Third, when using with ADMM and SVRG, we showed that the existing convergence results can be applied on the transformed problem but not on the original one. We further extend the proposed transformation to handle nonconvex and nonsmooth loss functions, and illustrate its benefits on the total variation model and robust sparse coding. Finally, we demonstrate the empirical advantages of working with the transformed problems on various tasks with both synthetic and real-world data sets. Experimental results show that better performance can be obtained with nonconvex regularizers, and algorithms on the transformed problems run much faster than the state-of-the-art on the original problems.

Acknowledgments

This research was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Grant 614513).

Appendix A. Proofs

A.1 Proposition 1

Proof First, we introduce a few Lemmas.

Lemma 14 (Golub and Van Loan (2012)) *For $x \neq 0$, the gradient of the ℓ_2-norm is $\nabla_{x_i}\|x\|_2 = x_i/\|x\|_2$.*

Let $h(z) = \kappa(\|z\|_2) - \kappa_0\|z\|_2$.

Lemma 15

$$\nabla_{z_i} h(z) = \begin{cases} \frac{\kappa'(\|z\|_2)-\kappa_0}{\|z\|_2} z_i & z \neq 0 \\ 0 & \textit{otherwise} \end{cases}. \tag{60}$$

Proof For $z \neq 0$, $\|z\|_2$ is differentiable (Lemma 14), and we obtain the first part of (60). For $z = 0$, let $\bar{h}_i(z) = \frac{\kappa'(\|z\|_2)-\kappa_0}{\|z\|_2} z_i$. Consider any Δ with $\|\Delta\|_2 = 1$.

$$\begin{aligned} \lim_{\alpha\to 0^+} \bar{h}_i(0+\alpha\Delta) &= \lim_{\alpha\to 0^+} \frac{\kappa'(\|\alpha\Delta\|_2) - \kappa_0}{\|\alpha\Delta\|_2}\alpha\Delta_i \\ &= \lim_{\alpha\to 0^+} (\kappa'(\alpha) - \kappa_0)\Delta_i = 0, \end{aligned}$$

as $\lim_{\alpha\to 0^+} \kappa'(\alpha) - \kappa_0 = 0$. Thus, $h(z)$ is smooth at $z = 0$, and we obtain the second part of (60). ■

Lemma 16 (Eriksson et al. (2004)) *Let $f : \mathbb{R} \to \mathbb{R}$ be a differentiable function. If its derivative f' is bounded, then f is Lipschitz-continuous with constant c where c is equal to the maximum value of $|f'|$.*

Lemma 17 (Eriksson et al. (2004)) *If a continuous function $f : \mathbb{R} \to \mathbb{R}$ is L_1-Lipschitz continuous in $[a, b]$ and L_2-Lipschitz continuous in $[b, c]$ (where $-\infty \leq a < b < c \leq \infty$), then it is $\max(L_1, L_2)$-Lipschitz continuous in $[a, c]$.*

Lemma 18 *Let z be an arbitrary vector, and e_i be the unit vector with only its ith dimension equal to 1. Define $\hat{h}_i(\gamma) = \frac{\kappa'(\|z+e_i\gamma\|_2)-\kappa_0}{\|z+e_i\gamma\|_2}(z_i + \gamma)$. Then, $\hat{h}$ is 2ρ-Lipschitz continuous.*

Proof Let the finite non-differentiable points of κ' be $\{\hat{\alpha}_1, \ldots, \hat{\alpha}_k\}$, where $\hat{\alpha}_1 < \cdots < \hat{\alpha}_k$. We partition $(-\infty, \infty)$ into intervals $(-\infty, \hat{\alpha}_1] \cup [\hat{\alpha}_1, \hat{\alpha}_2] \cup \cdots \cup [\hat{\alpha}_k, \infty)$, such that κ'' exists in each interval. Let $w = z + e_i\gamma$. For any interval,

$$\hat{h}_i'(\gamma) = \frac{\kappa''(\|w\|_2)}{\|w\|_2}(z_i + \gamma)^2 + \left(1 - \frac{(z_i + \gamma)^2}{\|w\|_2^2}\right)\frac{\kappa'(\|w\|_2) - \kappa_0}{\|w\|_2}. \tag{61}$$

Let $\phi(\alpha) = \kappa'(\alpha) - \kappa_0$, where $\alpha \geq 0$. Note that $\phi(0) = 0$. Moreover, $\phi(\alpha)$ is ρ-Lipschitz continuous as κ is ρ-Lipschitz smooth. Thus,

$$|\phi(\alpha) - \phi(0)| = |\kappa'(\alpha) - \kappa_0| \leq \rho\alpha,$$

and so

$$\left|\kappa'(\|w\|_2) - \kappa_0\right| \leq \rho\|w\|_2. \tag{62}$$

Note that $(z_i + \gamma)^2 \leq \|w\|_2^2$, (61) can be rewritten as

$$\begin{aligned}\left|\hat{h}_i'(\gamma)\right| &\leq \left|\frac{\kappa''(\|w\|_2)}{\|w\|_2}(z_i + \gamma)^2\right| + \left|\left(1 - \frac{(z_i + \gamma)^2}{\|w\|_2^2}\right)\frac{\kappa'(\|w\|_2) - \kappa_0}{\|w\|_2}\right| \\ &\leq \left|\kappa''(\|w\|_2)\right| + \left|\frac{\kappa'(\|w\|_2) - \kappa_0}{\|w\|_2}\right| \leq 2\rho,\end{aligned}$$

where the last inequality is due to that κ is ρ-Lipschitz smooth and (62). Thus, $|\hat{h}_i'(\gamma)| \leq 2\rho$, and by Lemma 16, we have $\hat{h}_i(\gamma)$ is 2ρ-Lipschitz continuous on any interval. Obviously $\hat{h}_i$ is continuous, and we conclude that $\hat{h}_i$ is also 2ρ-Lipschitz continuous by Lemma 17. ■

From Lemma 18, $\hat{h}_i$ is 2ρ-Lipschitz continuous. Thus, ∇h is 2ρ-Lipschitz continuous in each of its dimensions. For any $x, y \in \mathbb{R}^d$,

$$\begin{aligned}\|\nabla h(x) - \nabla h(y)\|_2^2 &= \sum_{i=1}^{d} \left[\nabla_{x_i} h(x) - \nabla_{y_i} h(y)\right]^2 \\ &\leq 4\rho^2 \sum_{i=1}^{d} (x_i - y_i)^2 = 4\rho^2\|x - y\|_2^2,\end{aligned}$$

and hence h is 2ρ-Lipschitz smooth. Finally, we will show that $h(z)$ is also concave.

Lemma 19 (Boyd and Vandenberghe (2004)) *$\phi(x) = \pi(q(x))$ is concave if π is concave, non-increasing and q is convex.*

Let $\pi(\alpha) = \kappa(\alpha) - \kappa_0\alpha$, where $\alpha \geq 0$. Note that π is concave. Moreover, $\pi(0) = 0$ and $\pi'(\alpha) \leq 0$. Thus, $\pi(\alpha)$ is non-increasing on $\alpha \geq 0$. Next, let $q(z) = \|z\|_2$. Then, $h(z) \equiv \kappa(\|z\|_2) - \kappa_0\|z\|_2 = \pi(q(z))$. As q is convex, $h(z)$ is concave from Lemma 19. ■

A.2 Corollary 2

Proof From Proposition 1 and the definition of $\bar{g}_i$, we can see that $\bar{g}_i$ is concave. Then, for any x, y,

$$\|\nabla h(A_i x) - \nabla h(A_i y)\|_2^2 \leq 4\rho^2\|A_i x - A_i y\|_2^2 \leq 4\rho^2\|A_i\|_F^2\|x - y\|_2^2.$$

Thus, $\bar{g}_i$ is $2\rho\|A_i\|_F$-Lipschitz smooth. ■

A.3 Corollary 3

Proof It is easy to see that $\breve{g}(x) = \kappa_0 \sum_{i=1}^K \mu_i\|A_i x\|_2$ is convex but not smooth. Using Corollary 2, as each $\bar{g}_i$ is concave and Lipschitz-smooth, $\bar{g}$ is also concave and Lipschitz-smooth. ■

A.4 Proposition 5

Proof First, we introduce a few Lemmas.

Definition 20 (Bertsekas (1999)) *A function $f : \mathbb{R}^m \to \mathbb{R}$ is* absolute symmetric *if $f([x_1; \ldots; x_m]) = f\left(\left[|x_{\pi(1)}|; \ldots; |x_{\pi(m)}|\right]\right)$ for any permutation π.*

Lemma 21 (Lewis and Sendov (2005)) *Let $\sigma(X) = [\sigma_1(X); \ldots; \sigma_m(X)]$ be the vector containing singular values of X. For an absolute symmetric function $f : \mathbb{R}^m \to \mathbb{R}$, $\phi(X) \equiv f(\sigma(X))$ is concave on X if and only if f is concave.*

From the definition of $\bar{g}$ in (24),

$$\bar{g}(X) = \bar{\mu}\sum_{i=1}^m(\kappa(\sigma_i(X)) - \kappa_0\|X\|_*) = \bar{\mu}\sum_{i=1}^m(\kappa(\sigma_i(X)) - \kappa_0\sigma_i(X)).$$

Let

$$h(x) = \bar{\mu}\sum_{i=1}^m(\kappa(|x_i|) - \kappa_0|x_i|). \tag{63}$$

Obviously, h is absolute symmetric. From Remark 4, h is concave. Thus, $\bar{g}$ is also concave by Lemma 21.

Lemma 22 (Lewis and Sendov (2005)) *Let the SVD of X be $U\mathrm{Diag}(\sigma(X))V^\top$, where $\sigma(X) = [\sigma_1(X); \ldots; \sigma_m(X)]$, $f : \mathbb{R}^m \to \mathbb{R}$ be smooth and absolute symmetric, and $\phi(X) \equiv f(\sigma(X))$. We have*

1. $\nabla\phi(X) = U\mathit{Diag}(\nabla f(\sigma(X)))V^\top$; *and*

2. *If f is L-Lipschitz smooth, then ϕ is also L-Lipschitz smooth.*

From Remark 4, h in (63) is 2ρ-Lipschitz smooth. Hence, from Lemma 22, $\bar{g}(X)$ is also 2ρ-Lipschitz smooth and $\nabla\bar{g}(X) = U\mathrm{Diag}(\nabla h(\sigma(X)))V^\top$. ■

A.5 Proposition 9

Proof First, we introduce the following Lemma.

Lemma 23 (Boyd and Vandenberghe (2004)) *$\phi(x) = \pi(q(x))$ is convex if π is convex, non-decreasing and q is convex.*

Let $\pi(\alpha) = \kappa(\alpha) + \frac{\rho}{2}\alpha^2$ where $\alpha \geq 0$. As κ is ρ-Lipschitz smooth, $\kappa'(\beta) - \kappa'(\alpha) \leq \rho(\alpha - \beta)$. Thus, $\pi'(\alpha) - \pi'(\beta) = \kappa'(\alpha) + \rho\alpha - \kappa'(\beta) - \rho\beta \geq 0$, i.e., π is convex. Besides, $\pi'(0) = \kappa'(0) \geq 0$. Thus, $\pi'(\alpha) \geq 0$ and π is also non-decreasing. Let $q(x) = \|x\|$ which is obviously convex as $\|\cdot\|$ is a norm, we can express $\phi(x) = \pi(q(x)) = \kappa(\|x\|_x) + \frac{\rho}{2}\|x\|^2$. Finally, $\phi(x)$ is also convex due to Lemma 23. ■

A.6 Theorem 6

Proof First, we introduce a few Lemmas.

Lemma 24 *Let $\tilde{X}$ be an inexact solution of the proximal step $\min_Z h(Z)$, where $h(Z) = \frac{1}{2}\|Z - (X - \frac{1}{\tau}\nabla\bar{f}(X))\|_F^2 + \frac{1}{\tau}\breve{g}(Z)$. Let $\hat{X} = \arg\min_Z h(Z)$. If $h(\tilde{X}) - h(\hat{X}) \leq \epsilon$, then*

$$F(\tilde{X}) \leq F(X) - \frac{\tau - \bar{L}}{2}\|\tilde{X} - X\|_F^2 + \tau\epsilon.$$

Proof Let $\phi(Z) = \langle Z - X, \nabla f(X)\rangle + \frac{\tau}{2}\|Z - X\|_F^2 + \breve{g}(Z)$. We have

$$\hat{X} = \arg\min_Z h(Z) = \arg\min_Z \phi(Z), \tag{64}$$

$$\phi(Z) = \tau h(Z) - \frac{1}{\tau}\|\nabla\bar{f}(X)\|_F^2. \tag{65}$$

From (64), we have

$$\phi(\hat{X}) = \langle \hat{X} - X, \nabla f(X)\rangle + \frac{\tau}{2}\|\hat{X} - X\|_F^2 + \breve{g}(\hat{X}) \leq \breve{g}(X). \tag{66}$$

As $h(\tilde{X}) - h(\hat{X}) \leq \epsilon$, from (65) (note that $\|\nabla\bar{f}(X)\|_F^2$ is a constant), we have

$$\phi(\tilde{X}) - \phi(\hat{X}) = \tau(h(\tilde{X}) - h(\hat{X})) \leq \tau\epsilon.$$

Then with (66), we have $\phi(\tilde{X}) \leq \tau\epsilon + \phi(\hat{X}) \leq \breve{g}(X) + \tau\epsilon$, i.e.,

$$\langle \tilde{X} - X, \nabla f(X)\rangle + \frac{\tau}{2}\|\tilde{X} - X\|_F^2 + \breve{g}(\tilde{X}) \leq \breve{g}(X) + \tau\epsilon. \tag{67}$$

As $\bar{f}$ is $\bar{L}$-Lipschitz smooth,

$$\bar{f}(\tilde{X}) \leq \bar{f}(X) + \langle \tilde{X} - X, \nabla f(X)\rangle + \frac{\bar{L}}{2}\|\tilde{X} - X\|_F^2.$$

Combining with (67), we obtain

$$\bar{f}(\tilde{X}) + \frac{\tau}{2}\|\tilde{X} - X\|_2^2 + \breve{g}(\tilde{X}) \leq \bar{f}(X) + \frac{\bar{L}}{2}\|\tilde{X} - X\|_F^2 + \breve{g}(X) + \tau\epsilon.$$

Thus, $F(\tilde{X}) \leq F(X) - \frac{\tau-\bar{L}}{2}\|\tilde{X} - X\|_F^2 + \tau\epsilon$. ■

If step 6 in Algorithm 3 is satisfied, $X_{t+1} = \tilde{Z}_{t+1}$, and

$$F(X_{t+1}) \leq F(X_t) - \frac{\delta}{2}\|X_{t+1} - Y_t\|_F^2. \tag{68}$$

Otherwise, step 9 is executed, and from Lemma 24, we have

$$F(X_{t+1}) \leq F(X_t) - \frac{\tau - \bar{L}}{2}\|X_{t+1} - X_t\|_F^2 + \tau\epsilon_t. \tag{69}$$

Partition $\Omega(T) = \{1, 2, \ldots, T\}$ into $\Omega_1(T)$ and $\Omega_2(T)$, such that step 7 is performed if $t \in \Omega_1(T)$; and execute step 9 otherwise. Combining (68) and (69), we have

$$\begin{aligned}
&F(X_1) - F(X_{T+1}) \\
&\quad\geq \frac{\delta}{2}\sum_{t\in\Omega_1(T)}\|X_{t+1} - Y_t\|_F^2 + \frac{\tau-\bar{L}}{2}\sum_{t\in\Omega_2(T)}\left(\|X_{t+1} - X_t\|_F^2 - \tau\epsilon_t\right), \\
&\quad\geq \frac{\delta}{2}\sum_{t\in\Omega_1(T)}\|X_{t+1} - Y_t\|_F^2 + \frac{\tau-\bar{L}}{2}\sum_{t\in\Omega_2(T)}\|X_{t+1} - X_t\|_F^2 - \frac{(\tau-\bar{L})\tau}{2}\sum_{t\in\Omega_2(T)}\epsilon_t \\
&\quad\geq \frac{\delta}{2}\sum_{t\in\Omega_1(T)}\|X_{t+1} - Y_t\|_F^2 + \frac{\tau-\bar{L}}{2}\sum_{t\in\Omega_2(T)}\|X_{t+1} - X_t\|_F^2 - \frac{(\tau-\bar{L})\tau}{2}\sum_{t=1}^{\infty}\epsilon_t \\
&\quad\geq \frac{\delta}{2}\sum_{t\in\Omega_1(T)}\|X_{t+1} - Y_t\|_F^2 - c_1 + \frac{\tau-\bar{L}}{2}\sum_{t\in\Omega_2(T)}\|X_{t+1} - X_t\|_F^2,
\end{aligned} \tag{70}$$

where $c_1 = \frac{(\tau-\bar{L})\tau}{2}\sum_{t=1}^{\infty}\epsilon_t < \infty$ and $c_1 \geq 0$. From (70), we have

$$\begin{aligned}
F(X_1) - \inf_X F(X) + c_1 &\geq F(X_1) - \lim_{T\to\infty} F(X_{T+1}) + c_1 \\
&\geq \lim_{T\to\infty}\frac{\delta}{2}\sum_{t\in\Omega_1(T)}\|X_{t+1} - Y_t\|_F^2 + \frac{\tau-\bar{L}}{2}\sum_{t\in\Omega_2(T)}\|X_{t+1} - X_t\|_F^2 \\
&\equiv c_2.
\end{aligned} \tag{71}$$

From Assumption A1, $c_2 \leq F(X_1) - \inf_X F(X) + c_1 < \infty$. Thus, $c_2 \geq 0$ is a finite constant. Let $\Omega_1^\infty = \lim_{T\to\infty}\Omega_1(T)$, and $\Omega_2^\infty = \lim_{T\to\infty}\Omega_2(T)$. Consider the three cases:

1. $\underline{|\Omega_1^\infty| \text{ is finite, and } |\Omega_2^\infty| \text{ is infinite.}}$ As $|\Omega_2^\infty| = \infty$ and $\lim_{\|X\|_F\to\infty} F(X) = \infty$ from Assumption A1 and (71), we have

$$\lim_{t\in\Omega_2^\infty, t\to\infty} \|X_{t+1} - X_t\|_F^2 = 0.$$

Thus, there exists a limit point such that $X_* = \lim_{t_j\in\Omega_2^\infty, t_j\to\infty} X_{t_j}$ for a subsequence $\{X_{t_j}\}$ of $\{X_t\}$. Since $\lim_{t_j\to\infty} \epsilon_{t_j} = 0$, then

$$\lim_{t_j\in\Omega_2^\infty, t_j\to\infty} X_{t_j+1} = \lim_{t_j\in\Omega_2^\infty, t_j\to\infty} \text{prox}_{\frac{1}{\tau}\breve{g}}(X_{t_j} - \frac{1}{\tau}\nabla\bar{f}(X_{t_j})).$$

As a result,

$$0 \in \lim_{t_j\in\Omega_2^\infty, t_j\to\infty} \frac{1}{\tau}\nabla\bar{f}(X_{t_j}) + (X_{t_j+1} - X_{t_j}) + \frac{1}{\tau}\partial\breve{g}(X_{t_j+1}).$$

Since both $\lim_{t_j\in\Omega_2^\infty, t_j\to\infty} X_{t_j} = \lim_{t_j\in\Omega_2^\infty, t_j\to\infty} X_{t_j+1} = X_*$, we then have $\nabla\bar{f}(X_*) + \partial\breve{g}(X_*) \ni 0$, and X_* is a critical point of (1).

2. $\underline{|\Omega_1^\infty| \text{ is infinite, and } |\Omega_2^\infty| \text{ is finite.}}$ As Ω_1^∞ is infinite and $\lim_{\|X\|_F\to\infty} F(X) = \infty$ from Assumption A1 and (71), we have

$$\lim_{t_j\in\Omega_1^\infty, t_j\to\infty} \|X_{t_j+1} - Y_{t_j}\|_F^2 = 0$$

for a subsequence $\{X_{t_j}\}$ of $\{X_t\}$. Thus, there exist a limit point such that

$$X_* = \lim_{t_j\in\Omega_1^\infty, t_j\to\infty} X_{t_j+1} = \lim_{t_j\in\Omega_1^\infty, t_j\to\infty} Y_{t_j}. \tag{72}$$

As $\lim_{t_j\to\infty} \epsilon_{t_j} = 0$, we have

$$0 \in \lim_{t_j\in\Omega_1^\infty, t_j\to\infty} \frac{1}{\tau}\nabla\bar{f}(Y_{t_j}) + (X_{t_j+1} - Y_{t_j}) + \frac{1}{\tau}\partial\breve{g}(X_{t_j+1}).$$

From (72), we have $\nabla\bar{f}(X_*) + \partial\breve{g}(X_*) \ni 0$ and X_* is a critical point of (1).

3. $\underline{\text{Both } |\Omega_1^\infty| \text{ and } |\Omega_2^\infty| \text{ are infinite.}}$ From the above two cases, we can see that once $|\Omega_1^\infty|$ or $|\Omega_2^\infty|$ is infinite, then $\{X_t\}$ is bounded, and any limit point of $\{X_t\}$ is also a critical point. In the third case, both $|\Omega_1^\infty|$ and $|\Omega_2^\infty|$ are infinite. Thus, any limit point of $\{X_t\}$ is also a critical point of (1).

As a result, $\{X_t\}$ is bounded and its limits points are critical points of (1). ■

A.7 Proposition 7

Proof From (71), we have

$$\frac{\delta}{2}\sum_{t_1\in\Omega_1(T)} \|X_{t_1+1} - Y_{t_1}\|_F^2 + \frac{\tau - \bar{L}}{2}\sum_{t_2\in\Omega_2(T)} \|X_{t_2+1} - X_{t_2}\|_F^2 < c_2, \tag{73}$$

where $c_2 \in (0, \infty)$ is a positive constant. Let $c_3 = \min(\frac{\delta}{2}, \frac{\tau - \bar{L}}{2})$. From the definition of V_t, (73) can be rewritten as

$$c_3 \sum_{t=1}^{T} \|X_{t+1} - V_t\|_F^2 \leq \frac{\delta}{2} \sum_{t_1 \in \Omega_1(T)} \|X_{t_1+1} - Y_{t_1}\|_F^2 + \frac{\tau - \bar{L}}{2} \sum_{t_2 \in \Omega_2(T)} \|X_{t_2+1} - X_{t_2}\|_F^2 \leq c_2.$$

Since c_2 is finite, thus $\lim_{t\to\infty} d_t \equiv \|X_{t+1} - V_t\|_F^2 = 0$. Besides, we have

$$\min_{t=1,\ldots,T} \sum_{t=1}^{T} \|X_{t+1} - V_t\|_F^2 \leq \frac{1}{T} \sum_{t=1}^{T} \|X_{t+1} - V_t\|_F^2 \leq \frac{c_2}{c_3 T}.$$

■

A.8 Proposition 10

Proof Note from (32) that $\nabla \bar{f}(S) = \nabla f(S) + \nabla \bar{g}(S)$. Using the matrix chain rule, since $S = \alpha X_t + \beta u_t v_t^\top$ and $\frac{\partial S}{\partial \alpha} = X_t$, then

$$\frac{\partial \bar{f}(S)}{\partial \alpha} = \left\langle \nabla \bar{f}(S), \frac{\partial S}{\partial \alpha} \right\rangle = \alpha \langle X_t, \nabla \bar{f}(S) \rangle.$$

Similarly, since $\frac{\partial S}{\partial \beta} = u_t v_t^\top$,

$$\frac{\partial \bar{f}(S)}{\partial \beta} = \left\langle \nabla \bar{f}(S), \frac{\partial S}{\partial \beta} \right\rangle = \beta \left\langle u_t v_t^\top, \nabla \bar{f}(S) \right\rangle = \beta \left(u_t^\top \nabla \bar{f}(S) v_t \right).$$

As $\bar{g}(S) = \mu \sum_{i=1}^{m} \kappa(\sigma_i(S)) - \mu \kappa_0 \sigma_i(S)$, using Lemma 22, $\nabla \bar{f}(X) = \nabla f(S) + \mu U_S \text{Diag}(w) V_S^\top$ and $w_i = \kappa'(\sigma_i(S)) - \kappa_0$. ■

A.9 Corollary 11

Proof Note that the SVD of X is $(UU_B)\text{Diag}([\sigma_1(B), \ldots, \sigma_k(B)])(VV_B)^\top$. Using Lemma 22,

$$\nabla \bar{f}(X) = \nabla f(X) + \nabla \bar{g}(X) = \nabla f(X) + \mu (UU_B) \text{Diag}(w) (VV_B)^\top,$$

where $w \in \mathbb{R}^k$ with $w_i = \kappa'(\sigma_i(B)) - \kappa_0$. ■

A.10 Proposition 12

Proof As $\bar{g}(X)$ is defined on the singular values of the input matrix X, we only need to show that $UBV^\top$ and B have the same singular values. Let the SVD of B be $U_B \text{Diag}(\sigma(B)) V_B^\top$, where $\sigma(B) = [\sigma_1(B), \ldots, \sigma_m(B)]$. As U and V are orthogonal, it is easy to see that $(UU_B)\,\text{Diag}(\sigma(B))\,(V_B V)^\top$ is the SVD of X. Thus, the Proposition holds. ■

A.11 Theorem 13

Proof We first introduce two Propositions.

Proposition 25 (Mishra et al. (2013)) *For a square matrix X, let sym$(X) = \frac{1}{2}(X + X^\top)$. The first-order optimality conditions for* (37) *are*

$$\begin{aligned}\nabla \bar{f}(X)VB - U\,\mathit{sym}(U^\top \nabla \bar{f}(X)VB) &= 0,\\ (\nabla \bar{f}(X))^\top UB - V\,\mathit{sym}(V^\top \nabla \bar{f}(X)UB) &= 0,\\ \mathit{sym}(U^\top \nabla \bar{f}(X)V) + \bar{\mu} I &= 0.\end{aligned}$$

Proposition 26 *If* (31) *has a critical point with rank r, choose the sizes of matrices U, V and B be $m \times r$, $n \times r$ and $r \times r$, respectively. Then, any critical point of* (37) *is also a critical point of* (31)*.*

Proof The subdifferential of the nuclear norm can be obtained as (Watson, 1992)

$$\partial \|X\|_* = \{UV^\top + W : U^\top W = 0, WV = 0, \|W\|_\infty \le 1\}, \tag{74}$$

where $X = UBV^\top$. Let $\hat{X} = \hat{U}\hat{B}\hat{V}^\top$ be a critical point of (37). We have $\text{sym}(\hat{U}^\top \nabla \bar{f}(\hat{X})\hat{V}) + \bar{\mu} I = 0$ due to Proposition 25. From the property of the matrix norm, we have

$$\lambda = \|\,\text{sym}(\hat{U}^\top \nabla \bar{f}(\hat{X})\hat{V})\|_\infty \le \|\hat{U}^\top \nabla \bar{f}(\hat{X})\hat{V}\|_\infty \le \|\nabla \bar{f}(\hat{X})\|_\infty.$$

The equality holds only when $\nabla \bar{f}(\hat{X}) = -\bar{\mu}\hat{U}\hat{V}^\top - \bar{\mu}\hat{U}_\perp \hat{\Sigma}_\perp \hat{V}_\perp^\top$, where $\hat{U}_\perp$ and $\hat{V}_\perp$ are orthogonal matrices with $\hat{U}^\top \hat{U}_\perp = 0$ and $\hat{V}^\top \hat{V}_\perp = 0$, and $\hat{\Sigma}_\perp$ is a diagonal matrix with positive elements $[\Sigma_\perp]_{ii} \le 1$. Combining this with (74), we have

$$\nabla \bar{f}(\hat{X}) \in -\bar{\mu}\partial \|\hat{X}\|_*. \tag{75}$$

Then, for (31), if X_* is a critical point, we have

$$\nabla \bar{f}(X_*) \in -\bar{\mu}\partial \|X_*\|_*. \tag{76}$$

Comparing (75) and (76), the difference is on the ranks of $\hat{X}$ and X_*. As (31) has a critical point with rank-r, $\hat{X}$ is also a critical point of (31). ■

In Algorithm 4, the sizes of U, V and B are selected as $m \times t$, $n \times t$, and $t \times t$, respectively. If (31) has a critical point with rank r, then as iteration goes and $t = r$, from Proposition 26, Algorithm 4 will return a critical point of (31). ■

Appendix B. Details in Section 5.5

B.1 CCCP

Using Proposition 9, we can decompose $\kappa(|x|) = \hat{\varsigma}(x) + \breve{\varsigma}(x)$, where $\hat{\varsigma}(x) = -\frac{\rho}{2}x^2$ is convex and $\breve{\varsigma}(x) = \kappa(|x|) + \frac{\rho}{2}x^2$ is concave. Apply the above decomposition on κ in (46), and we have the

following DC decomposition:

$$\breve{F}(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} \breve{\varsigma}\left([Y-X]_{ij}\right) + \mu \sum_{i=1}^{m-1}\sum_{j=1}^{m} \breve{\varsigma}\left([D_v X]_{ij}\right) + \mu \sum_{i=1}^{n}\sum_{j=1}^{n-1} \breve{\varsigma}\left([X D_h]_{ij}\right),$$
$$\hat{F}(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} \hat{\varsigma}\left([Y-X]_{ij}\right) + \mu \sum_{i=1}^{m-1}\sum_{j=1}^{m} \hat{\varsigma}\left([D_v X]_{ij}\right) + \mu \sum_{i=1}^{n}\sum_{j=1}^{n-1} \hat{\varsigma}\left([X D_h]_{ij}\right).$$

The CCCP procedures at Section 2.1 can then be applied.

B.2 Smoothing

As the LSP is used as κ, a smoothed version of it can be obtained as $\tilde{\kappa}_\lambda(x) = \beta \log\left(1 + \frac{h_\lambda(x)}{\theta}\right)$, where $h_\lambda(x) = \begin{cases} |x| & \text{if } |x| \geq \lambda \\ \frac{x^2}{2\lambda} + \frac{\lambda}{2} & \text{otherwise} \end{cases}$. Thus, (46) is smoothed as

$$\breve{F}_\lambda(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} \tilde{\kappa}_\lambda\left([Y-X]_{ij}\right) + \mu \sum_{i=1}^{m-1}\sum_{j=1}^{m} \tilde{\kappa}_\lambda\left([D_v X]_{ij}\right) + \mu \sum_{i=1}^{n}\sum_{j=1}^{n-1} \tilde{\kappa}_\lambda\left([X D_h]_{ij}\right).$$

Gradient descent is then used for optimization (Chen, 2012). Specifically, we need to minimize a sequence of subproblems $\{\breve{F}_{\lambda_1}(X), \breve{F}_{\lambda_2}(X), \dots\}$ with $\lambda_i = \lambda_0 \nu^i$, and using X from $\breve{F}_{\lambda_{i-1}}(X)$ to warm start $\breve{F}_{\lambda_i}(X)$. In the experiment, we set $\lambda_0 = 0.1$ and $\nu = 0.95$.

References

G. Andrew and J. Gao. Scalable training of ℓ_1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine learning*, pages 33–40, 2007.

H. Bauschke, R. Goebel, Y. Lucet, and X. Wang. The proximal average: Basic theory. *SIAM Journal on Optimization*, 19(2):766–785, 2008.

A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE Transactions on Image Processing*, 18(11):2419–2434, 2009.

M. Beck, A.and Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, 1989.

R.I. Bot, E. Robert Csetnek, and S.C. László. An inertial forward-backward algorithm for the minimization of the sum of two nonconvex functions. *EURO Journal on Computational Optimization*, 4(1):3–25, 2016.

L. Bottou. Online learning and stochastic approximations. *On-line Learning in Neural Networks*, 17(9):142–174, 1998.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

K. Bredies, D.A. Lorenz, and P. Maass. A generalized conditional gradient method and its connection to an iterative shrinkage method. *Computational Optimization and Applications*, 42 (2):173–193, 2009.

E.J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

E.J. Candès, M.B. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905, 2008.

E.J. Candès, X. Li, Yi Ma, and J. Wright. Robust principal component analysis. *Journal of the ACM*, 58(3):1–37, 2011.

X. Chen. Smoothing methods for nonsmooth, nonconvex minimization. *Mathematical Programming*, 134(1):71–99, 2012.

D.L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

K. Eriksson, F. Estep, and C. Johnson. *Applied Mathematics: Body and Soul: Volume 1: Derivatives and Geometry in IR3*. Springer-Verlag, 2004.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics*, 3(1-2): 95–110, 1956.

R. Ge, F. Huang, C. Jin, and Y. Yuan. Escaping from saddle points - online stochastic gradient for tensor decomposition. In *International Conference on Learning Theory*, pages 797–842, 2015.

R. Ge, J. Lee, and T. Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016.

R. Ge, C. Jin, and Y. Zheng. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995.

S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2):59–99, 2016.

R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, 9(2):41–76, 1975.

G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 2012.

P. Gong and J. Ye. HONOR: Hybrid Optimization for NOn-convex Regularized problems. In *Advances in Neural Information Processing Systems*, pages 415–423, 2015a.

P. Gong and J. Ye. A modified orthant-wise limited memory quasi-Newton method with convergence analysis. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 276–284, 2015b.

P. Gong, C. Zhang, Z. Lu, J. Huang, and J. Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *Proceedings of the 30th International Conference on Machine Learning*, pages 37–45, 2013.

B. He and X. Yuan. On the $o(1/n)$ convergence rate of the douglas-rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012.

J.B. Hiriart-Urruty. Generalized differentiability, duality and optimization for problems dealing with differences of convex functions. *Convexity and Duality in Optimization*, pages 37–70, 1985.

M. Hong, Z.-Q. Luo, and M. Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016.

C.-J. Hsieh and P. Olsen. Nuclear norm minimization via active subspace selection. In *Proceedings of the 31st International Conference on Machine Learning*, pages 575–583, 2014.

L. Jacob, G. Obozinski, and J.-P. Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th International Conference on Machine Learning*, pages 433–440, 2009.

M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning*, pages 427–435, 2013.

R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12:2297–2334, 2011.

S. Laue. A hybrid algorithm for convex semidefinite optimization. In *Proceedings of the 29th International Conference on Machine Learning*, pages 177–184, 2012.

J. Lee, M. Simchowitz, M. Jordan, and B. Recht. Gradient descent only converges to minimizers. In *Proceedings of the 29th Conference on Learning Theory*, pages 1246–1257, 2016.

J. Lee, I. Panageas, G. Piliouras, M. Simchowitz, M.I. Jordan, and B. Recht. First-order methods almost always avoid saddle points. arxiv preprint, University of Southern California, 2017.

A.S. Lewis and H.S. Sendov. Nonsmooth analysis of singular values. *Set-Valued Analysis*, 13(3): 243–264, 2005.

G. Li and T.K. Pong. Global convergence of splitting methods for nonconvex composite optimization. *SIAM Journal on Optimization*, 25(4):2434–2460, 2015.

H. Li and Z. Lin. Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems*, pages 379–387, 2015.

J. Liu and J. Ye. Moreau-Yosida regularization for grouped tree structure learning. In *Advances in Neural Information Processing Systems*, pages 1459–1467, 2010.

J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2013.

P. Loh and M. Wainwright. Regularized M-estimators with nonconvexity: Statistical and algorithmic theory for local optima. *Journal of Machine Learning Research*, 16:559–616, 2015.

C. Lu, J. Shi, and J. Jia. Online robust dictionary learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 415–422, 2013.

C. Lu, J. Tang, S. Yan, and Z. Lin. Generalized nonconvex nonsmooth low-rank minimization. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 4130–4137, 2014.

C. Lu, C. Zhu, C. Xu, S. Yan, and Z. Lin. Generalized singular value thresholding. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 1805–1811, 2015.

Z. Lu. Sequential convex programming methods for a class of structured nonlinear programming. Preprint arXiv:1210.3039, 2012.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th International Conference on Machine Learning*, pages 689–696, 2009.

R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 11:2287–2322, 2010.

B. Mishra, G. Meyer, F. Bach, and R. Sepulchre. Low-rank optimization with trace norm penalty. *SIAM Journal on Optimization*, 23(4):2124–2149, 2013.

Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

T. Ngo and Y. Saad. Scaled gradients on Grassmann manifolds for matrix completion. In *Advances in Neural Information Processing Systems*, pages 1412–1420, 2012.

M. Nikolova. A variational approach to remove outliers and impulse noise. *Journal of Mathematical Imaging and Vision*, 20(1-2):99–120, 2004.

J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2006.

P. Ochs, Y. Chen, T. Brox, and T. Pock. iPiano: Inertial proximal algorithm for nonconvex optimization. *SIAM Journal on Imaging Sciences*, 7(2):1388–1419, 2014.

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013.

S.J. Reddi, A. Hefny, S. Sra, B. Póczos, and A.J. Smola. Stochastic variance reduction for nonconvex optimization. In *Proceedings of the 33nd International Conference on Machine Learning*, pages 314–323, 2016a.

S.J. Reddi, S. Sra, B. Poczos, and A. Smola. Fast stochastic methods for nonsmooth nonconvex optimization. In *Advances in Neural Information Processing Systems*, pages 1145–1153, 2016b.

M. Schmidt, N.L. Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in Neural Information Processing Systems*, pages 1458–1466, 2011.

S. Sra. Scalable nonconvex inexact proximal splitting. In *Advances in Neural Information Processing Systems*, pages 530–538, 2012.

N. Srebro, J. Rennie, and T.S. Jaakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems*, pages 1329–1336, 2004.

Q. Sun, S. Xiang, and J. Ye. Robust principal component analysis via capped norms. In *Proceedings of the 19th International Conference on Knowledge Discovery and Data Mining*, pages 311–319, 2013.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B*, 73(3):273–282, 1996.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B*, 67(1):91–108, 2005.

J. Trzasko and A. Manduca. Highly undersampled magnetic resonance image reconstruction via homotopic-minimization. *IEEE Transactions on Medical Imaging*, 28(1):106–121, 2009.

G.A. Watson. Characterization of the subdifferential of some matrix norms. *Linear Algebra and its Applications*, 170:33–45, 1992.

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*, 4(4): 333–361, 2012.

L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

M. Yan. Restoration of images corrupted by impulse noise and mixed gaussian impulse noise using blind inpainting. *SIAM Journal on Imaging Sciences*, 6(3):1227–1245, 2013.

M. Yang, L. Zhang, J. Yang, and D. Zhang. Robust sparse coding for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 625–632, 2011.

Q. Yao and J.T. Kwok. Efficient learning with a family of nonconvex regularizers by redistributing nonconvexity. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2645–2654, 2016.

Q. Yao, J.T. Kwok, and W. Zhong. Fast low-rank matrix learning with nonconvex regularization. In *Proceedings of IEEE International Conference on Data Mining*, pages 539–548, 2015.

Q. Yao, J. Kwok, F. Gao, W. Chen, and T.-Y. Liu. Efficient inexact proximal gradient algorithm for nonconvex problems. In *International Joint Conferences on Artifical Intelligence*, pages 3308–3314, 2017.

Y. Yu. Better approximation and faster algorithm using the proximal average. In *Advances in Neural Information Processing Systems*, pages 458–466, 2013.

L. Yuan, J. Liu, and J. Ye. Efficient methods for overlapping group lasso. In *Advances in Neural Information Processing Systems*, pages 352–360, 2011.

A.L. Yuille and A. Rangarajan. The concave-convex procedure (CCCP). In *Advances in Neural Information Processing Systems*, pages 1033–1040, 2002.

C.H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics*, 38(2):894–942, 2010a.

T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1081–1107, 2010b.

X. Zhang, D. Schuurmans, and Y.-L. Yu. Accelerated training for matrix-norm regularization: A boosting approach. In *Advances in Neural Information Processing Systems*, pages 2906–2914, 2012.

C. Zhao, X. Wang, and W.-K. Cham. Background subtraction via robust dictionary learning. *EURASIP Journal on Image and Video Processing*, 2011(1):1–12, 2011.

W. Zhong and J.T. Kwok. Gradient descent with proximal average for nonconvex and composite regularization. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 2206–2212, 2014.

Z.A. Zhu and E. Hazan. Variance reduction for faster non-convex optimization. In *Proceedings of the 33nd International Conference on Machine Learning*, pages 699–707, 2016.