Robust Principal Component Analysis via Capped Norms

Qian Sun
Arizona State University
Tempe, AZ 85287, USA
qsun21@asu.edu

Shuo Xiang
Arizona State University
Tempe, AZ 85287, USA
shuo.xiang@asu.edu

Jieping Ye
Arizona State University
Tempe, AZ 85287, USA
jieping.ye@asu.edu

ABSTRACT

In many applications such as image and video processing, the data matrix often possesses simultaneously a low-rank structure capturing the global information and a sparse component capturing the local information. How to accurately extract the low-rank and sparse components is a major challenge. Robust Principal Component Analysis (RPCA) is a general framework to extract such structures. It is well studied that under certain assumptions, convex optimization using the trace norm and ℓ_1-norm can be an effective computation surrogate of the difficult RPCA problem. However, such convex formulation is based on a strong assumption which may not hold in real-world applications, and the approximation error in these convex relaxations often cannot be neglected. In this paper, we present a novel non-convex formulation for the RPCA problem using the capped trace norm and the capped ℓ_1-norm. In addition, we present two algorithms to solve the non-convex optimization: one is based on the Difference of Convex functions (DC) framework and the other attempts to solve the sub-problems via a greedy approach. Our empirical evaluations on synthetic and real-world data show that both of the proposed algorithms achieve higher accuracy than existing convex formulations. Furthermore, between the two proposed algorithms, the greedy algorithm is more efficient than the DC programming, while they achieve comparable accuracy.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications-Data Mining

General Terms

Algorithm

Keywords

Non-convex optimization, DC programming, ADMM, low-rank, sparsity, trace norm, image processing

KDD'13, August 11–14, 2013, Chicago, Illinois, USA.

1. INTRODUCTION

In many applications, we encounter very high-dimensional data such as images, texts, and genomic data. Analysis of such data is challenging due to the curse of dimensionality. One promising approach is to exploit the special structures of the data and it has recently achieved great success in many applications [2, 8, 14, 19, 24]. Two particularly interesting structures are the low-rank structure and the sparse structure. For example, the images of the same scene may be taken from different illusions, thus the shadows represent the sparse component and the scene relates to the low-rank part [1]. For a collection of text documents, the low-rank component could capture common words used in all the documents while the sparse component may capture the few key words that best distinguish each document from others [6]. Robust Principal Component Analysis (RPCA) [6], or Stable Principal Component Pursuit (SPCP) [1] is an efficient tool for such analysis and has received increasing attentions in many areas [5, 6, 17, 18, 20, 22].

The most general form of the RPCA problem can be formulated as follows:

$$\begin{aligned} \underset{X,Y}{\text{minimize}} \quad & \text{rank}(X) + \lambda\|Y\|_0, \\ \text{subject to} \quad & A = X + Y, \end{aligned} \tag{1}$$

where we assume that the data matrix $A \in \mathbb{R}^{m\times n}$ is the summation of a low-rank matrix X and a sparse component Y. We minimize the rank of X as well as the number of non-zero entries in Y. Due to the discrete nature of the rank function, such a problem has been proven to be NP-hard. Therefore computing a global optimal solution of (1) is a challenge. Recently, one proper relaxation with theoretical guarantees has been proposed in [6]. In particular, the authors approximate the original problem by minimizing the sum of trace norm of X and ℓ_1-norm of Y with an equality constraint:

$$\begin{aligned} \underset{X,Y}{\text{minimize}} \quad & \|X\|_* + \lambda\|Y\|_1, \\ \text{subject to} \quad & A = X + Y, \end{aligned} \tag{2}$$

where the trace norm $\|X\|_*$ is defined as the sum of all singular values of X, and $\|Y\|_1 = \sum_{ij} |Y_{ij}|$ denotes the sum of absolute values of all entries in Y. Notice that (2) is a convex optimization problem, therefore a global optimal can be computed. Moreover, the authors of [6] also provide an efficient algorithm for (2). It is well known that the trace norm and the ℓ_1-norm are capable of inducing low-rank and sparse structures [20], achieving our desired goal. Furthermore, it has been shown in [6] that with the balance

parameter λ equal to $\frac{1}{\sqrt{\max(m,n)}}$, (2) will provide the correct answer.

Problem (2) can recover a low-rank matrix only with sparse corruptions. In practice, as discussed in [5], it is necessary to consider the observed data matrix under more realistic conditions. Particularly, the given data may not only be corrupted by impulse noise which is sparse and large, but also by Gaussian noise which is small but dense. To deal with this more realistic scenario, a modified model with tolerance of both Gaussian noise and impulse noise has been proposed in [20], by changing the constraint from equality to inequality:

$$\begin{aligned} \underset{X,Y}{\text{minimize}} \quad & \|X\|_* + \lambda\|Y\|_1, \\ \text{subject to} \quad & \|A - X - Y\|_F \leq \sigma, \end{aligned} \tag{3}$$

where σ denotes the level of the Gaussian noise. The authors proposed to apply the augmented lagrangian approaches to efficiently solve problem (3). Based on the same model, an Augmented Lagrange Method of Multipliers (ADMM) is proposed in [13] to solve the optimization problem, which iteratively solves X and Y using soft-thresholding. In [1], the authors developed a non-smooth augmented lagrange method to efficiently solve problem (3). There are also many other approaches which aim to solve a similar problem. For example, in [7], the authors develop a notion of rank-sparsity incoherence and uses it to characterize both of the fundamental identifiability and the sufficient conditions for exact recovery. In [23], the authors presented an approach to recover the correct column space of the uncorrupted matrix, rather than the matrix itself.

To our best knowledge, most of the recent research focuses on solving RPCA via solving a convex problem with certain constraints [5, 17, 18, 22]. By contrast, this paper tackles this interesting problem from a different point of view: non-convex optimization using a mixture of capped trace norm and capped ℓ_1-norm. We propose two algorithms to solve the non-convex formulation. First, we apply the Difference of Convex functions (DC programming) framework to iteratively solve the non-convex formulation, and apply ADMM to solve the sub-problem. In our second approach, instead of using ADMM to solve the sub-problem, we present a greedy algorithm to solve the sub-problem, based on which we propose a fast alternating optimization algorithm. Both low-rank part X and sparse component Y can be recovered by our algorithms with higher accuracy than most of the existed work.

The RPCA formulation has applications in many areas [6]. In this paper, we evaluate the proposed algorithms on synthetic data and two real-world applications: background detection in surveillance video and shadow removal from illuminated portraits. In both synthetic experiments and real-world applications, our proposed algorithms achieve better recovery of X and Y than existing convex formulations. In particular, our proposed algorithms can capture the sparse locations with higher accuracy than existing methods.

The rest of the paper is organized as follows. In Section 2, we present our non-convex formulation for the RPCA problem. In Section 3, we present a DC framework to solve our formulation. In Section 4, we develop a fast alternating optimization algorithm to solve the formulation. In Section 5, we evaluate our algorithms on both synthetic data and real-world applications. We conclude the paper in Section 6.

2. PROBLEM FORMULATION

In this section, we formulate the RPCA problem as a non-convex minimization problem via capped norms.

Given a data matrix A, the goal of RPCA is to extract a low rank X and a sparse component Y from A. Following (1), we consider a non-convex formulation of RPCA with an inequality constraint:

$$\begin{aligned} \underset{X,Y}{\text{minimize}} \quad & \text{rank}(X) + \lambda\|Y\|_0, \\ \text{subject to} \quad & \|A - X - Y\|_F^2 \leq \sigma^2, \end{aligned} \tag{4}$$

where $A \in \mathbb{R}^{m\times n}$ is the observed matrix, σ^2 is the level of Gaussian noise and $\lambda > 0$ is a trade-off parameter between the low rank and the sparse component. Here, the $\|\cdot\|_0$ norm is the number of non-zero entries of a matrix. In [1, 20], the trace norm and the ℓ_1-norm were used to approximate the rank function and ℓ_0-norm to convert the non-convex problem into a convex one. It is noteworthy that the convex relaxation may not be a good approximation of (4) in real-world applications. The main motivation of the reformulation to be presented in the next subsection is to reduce the approximation error using non-convex formulations.

2.1 Capped norms

We first introduce the capped norms for matrices and vectors, which are the surrogates of the rank function and the ℓ_0-norm. Let $p = \min(m, n)$. We can approximate the rank function and the ℓ_0-norm by:

$$\begin{aligned} \text{rank}(X) &\approx \sum_{i=1}^{p} \min(1, \frac{\sigma_i(X)}{\theta_1}) \\ &= \frac{1}{\theta_1}\left[\|X\|_* - \sum_{i=1}^{p}\max(\sigma_i(X) - \theta_1, 0)\right], \\ \|Y\|_0 &\approx \sum_{ij} \min(1, \frac{|Y_{ij}|}{\theta_2}) \\ &= \frac{1}{\theta_2}\left[\|Y\|_1 - \sum_{i,j}\max(|Y_{ij}| - \theta_2, 0)\right], \end{aligned}$$

for some small parameters $\theta_1, \theta_2 > 0$. We can observe that if all the singular values of X are greater than θ_1 and all the absolute values of elements in Y are greater than θ_2, then the approximation will become equality.

The smaller θ_1 and θ_2 are, the more accurate the capped norm approximation would be. We can control the recovery precision via making use of θ_1 and θ_2. By carefully choosing θ_1 and θ_2, we can recovery X and Y more accurately than the trace norm and the ℓ_1-norm approximation.

2.2 Proposed non-convex formulation

With the aforementioned capped norms, we propose to solve the following non-convex RPCA formulation:

$$\begin{aligned} \underset{X,Y}{\text{minimize}} \quad & \frac{1}{\theta_1}\|X\|_* + \frac{\lambda}{\theta_2}\|Y\|_1 - \left[\frac{1}{\theta_1}P_1(X) + \frac{\lambda}{\theta_2}P_2(Y)\right], \\ \text{subject to} \quad & \|A - X - Y\|_F^2 \leq \sigma^2, \end{aligned} \tag{5}$$

where $P_1(X) = \sum_{i=1}^{p}\max(\sigma_i(X) - \theta_1, 0)$, and $P_2(Y) = \sum_{i,j}\max(|Y_{ij}| - \theta_2, 0)$.

Clearly, the new objective function in problem (5) is not convex due to the last two terms being concave functions.

However, the intrinsic structure of the formulation naturally leads to use of Difference of Convex (DC) Programming [21]. DC programming treats a non-convex function as the difference of two convex functions, and then iteratively solves it on the basis of the combination of the first convex part and the linear approximation of the second convex part. Obviously, the trace norm and the ℓ_1-norm of matrices are convex, and the summation of maximum is also convex. Thus problem (5) exhibits the DC structure. The details of DC programming to solve the RPCA problem are presented in the next section.

3. A DC PROGRAMMING-BASED ALGORITHM

In this section, we detail the DC programming framework for solving problem (5). In each iteration of the framework, the first-order approximation is used to substitute the non-convex part. To generate the first-order approximation of $P_1(X)$ and $P_2(Y)$ in our formulation, we need to compute the subdifferential of a capped trace norm, as summarized in the following lemma:

LEMMA 1. *The subdifferential of*

$$P_1(X) = \sum_{i=1}^{p} \max(\sigma_i(X) - \theta_1, 0)$$

is given by:

$$\partial P_1(X) = \left\{ U Diag(z) V^T : z \in Z^* \right\}, \tag{6}$$

where U and V are the left and right singular vectors of X, respectively, and

$$Z^* = \left\{ z \in \mathbb{R}^p \middle| z_i \begin{cases} = 1 & \text{if } \sigma_i(X) > \theta_1, \\ = 0 & \text{if } \sigma_i(X) < \theta_1, \\ \in [0,1] & \text{otherwise.} \end{cases} \right\}$$

and p is the rank of X.

PROOF. First, we can denote $X = U\Sigma_X V^T$ as the SVD of matrix X where U and V are unitary matrices. Then we define auxiliary matrices $B = U\Sigma_B V^T$ and $C = U\Sigma_\theta V^T$, where $\Sigma_B = \text{Diag}(b)$, $b_i \in \{0,1\}$ and $\Sigma_\theta = \text{Diag}(\theta)$. For simplicity, we denote

$$\sigma_X = (\sigma_1(X), \sigma_2(X), \ldots, \sigma_p(X))^T,$$

$$\sigma_B = (\sigma_1(B), \sigma_2(B), \ldots, \sigma_p(B))^T,$$

and

$$\sigma_\theta = (\theta_1, \theta_1, \ldots, \theta_1)^T.$$

Using the notations above, $P_1(X)$ can be written as

$$\begin{aligned} P_1(X) &= \sum_{i=1}^{p} \max(\sigma_i(X) - \theta_1, 0) \\ &= \max_{\sigma_B \in E} < \sigma_B, \sigma_X - \sigma_\theta >, \end{aligned}$$

where $E = \{s \in \mathbb{R}^p : s_i \in \{0,1\}\}$. We can see that the maximum can be achieved if and only if:

$$\sigma_i(B) \in \begin{cases} 1 & \text{if } \sigma_i(X) - \theta_1 > 0, \\ 0 & \text{if } \sigma_i(X) - \theta_1 < 0, \\ \{0,1\} & \text{otherwise.} \end{cases}$$

The subdifferential should be the convex hull of B [16]:

$$\begin{aligned} \partial P_1(X) &= \text{conv}\{B : B = U\Sigma_B V^T\} \\ &= U\Sigma_B^* V^T, \end{aligned}$$

where $\Sigma_B^* = \text{Diag}(\sigma_B^*)$ and

$$\sigma_B^*(i) \in \begin{cases} 1 & \text{if } \sigma_i(X) - \theta_1 > 0, \\ 0 & \text{if } \sigma_i(X) - \theta_1 < 0, \\ [0,1] & \text{otherwise.} \end{cases}$$

This completes the proof of the lemma. □

In addition, it can be easily shown that:

$$\partial P_2(Y) = \left\{ V \in \mathbb{R}^{m \times n} \middle| V_{ij} \in \begin{cases} \{1\} & \text{if } Y_{ij} > \theta_2, \\ [0,1] & \text{if } Y_{ij} = \theta_2, \\ \{0\} & \text{if } |Y_{ij}| < \theta_2, \\ [-1,0] & \text{if } Y_{ij} = -\theta_2, \\ \{-1\} & \text{if } Y_{ij} < -\theta_2. \end{cases} \right\} \tag{7}$$

By denoting $U = \frac{1}{\theta_1}\partial P_1(X)$ and $V = \frac{1}{\theta_2}\partial P_2(Y)$, the formulation (5) can be rewritten as:

$$\begin{aligned} \underset{X,Y}{\text{minimize}} \quad & \frac{1}{\theta_1}\|X\|_* + \frac{\lambda}{\theta_2}\|Y\|_1 - \langle U, X\rangle - \langle \lambda V, Y\rangle, \\ \text{subject to} \quad & \|A - X - Y\|_F^2 \le \sigma^2, \end{aligned} \tag{8}$$

where $\langle U, X\rangle = \sum_{i=1}^{m}\sum_{j=1}^{n} U_{ij}X_{ij}$. Thus, we solve the original non-convex problem by solving a series of convex problems. In each iteration, we approximate problem (5) by the sub-problem (8) at the current X^k and Y^k. The key subproblem is to solve the convex problem (8).

3.1 Solving the sub-problem

Here, we apply the Augmented Lagrange Method of Multipliers (ADMM) [3] to solve the sub-problem (8). ADMM has been applied successfully to solve many sparse learning problems. We introduce an auxiliary variable $S = X$ and rewrite the problem (8) as:

$$\begin{aligned} \underset{X,Y}{\text{minimize}} \quad & \frac{1}{\theta_1}\|S\|_* + \frac{\lambda}{\theta_2}\|Y\|_1 - \langle U, X\rangle - \langle \lambda V, Y\rangle, \\ \text{subject to} \quad & \|A - X - Y\|_F^2 \le \sigma^2, \\ & X - S = 0. \end{aligned} \tag{9}$$

The augmented lagrangian function of (9) is:

$$\begin{aligned} L_\rho(S, X, Y, \Lambda) = \quad & \frac{1}{\theta_1}\|S\|_* + \frac{\lambda}{\theta_2}\|Y\|_1 - \langle U, X\rangle - \langle \lambda V, Y\rangle \\ & + \langle \Lambda, X - S\rangle + \frac{\rho}{2}\|X - S\|_F^2, \\ \text{subject to} \quad & \|A - X - Y\|_F^2 \le \sigma^2, \end{aligned}$$

where Λ is the lagrangian multiplier and ρ is the step size of dual update.

The general approach of ADMM consists of the following iterations:

$$\begin{aligned} S^{k+1} &= \arg\min_S L_\rho(S, X^k, Y^k, \Lambda^k), \\ \{X^{k+1}, Y^{k+1}\} &= \arg\min_{X,Y} L_\rho(S^{k+1}, X, Y, \Lambda^k), \\ \Lambda^{k+1} &= \Lambda^k + \rho(X^{k+1} - S^{k+1}). \end{aligned} \tag{10}$$

Next, we present the details for updating each variable in (10).

3.1.1 Updating S

The update of S involves the following problem:

$$S^{k+1} = \arg\min_S \frac{1}{2}\left\|S - X^k - \frac{\Lambda^k}{\rho}\right\|_F^2 + \frac{1}{\theta_1\rho}\|S\|_*, \quad (11)$$

which is the proximal operator of the trace norm. It has an analytical solution as summarized in the following lemma [4]:

LEMMA 2. *The proximal operator associated with the trace norm, i.e., the minimizer of the following problem:*

$$\underset{X\in\mathbb{R}^{m\times n}}{\text{minimize}} \quad \frac{1}{2}\|X - A\|_F^2 + \lambda\|X\|_*,$$

is given by:

$$prox_{\lambda\|\cdot\|_*}(X) = USV^T,$$
$$S = Diag(\max(\sigma - \lambda, 0)),$$

where $\sigma = (\sigma_1, \ldots, \sigma_p)$ are the singular values of A and $A = UDiag(\sigma)V^T$ is the SVD of A.

From Lemma 2, if we denote the SVD of matrix $X^k + \frac{\Lambda^k}{\rho}$ as $U_s\Sigma_s V_s^T$, where $\Sigma_s = \text{Diag}(\sigma)$, and $p_i = \max(\sigma_i - \frac{1}{\theta_1\rho}, 0)$, then S^{k+1} can be obtained by $S^{k+1} = U_s\text{Diag}(p)V_s^T$.

3.1.2 Updating X and Y

The update of X and Y amounts to solving:

$$\begin{aligned}\{X^{k+1}, Y^{k+1}\} = &\arg\min_{X,Y} \frac{\lambda}{\theta_2}\|Y\|_1 - \langle U^k, X\rangle - \langle \lambda V^k, Y\rangle \\ &+ \langle \Lambda^k, X - S^{k+1}\rangle + \frac{\rho}{2}\|X - S^{k+1}\|_F^2 \\ &\text{s.t.} \quad \|A - X - Y\|_F^2 \le \sigma^2.\end{aligned} \quad (12)$$

By introducing a lagrangian multiplier ν for the inequality constraint, the lagrangian function is given by:

$$\begin{aligned}L = &\frac{\lambda}{\theta_2}\|Y\|_1 - \langle U^k, X\rangle - \langle \lambda V^k, Y\rangle + \langle \Lambda^k, X - S^{k+1}\rangle \\ &+ \frac{\rho}{2}\|X - S^{k+1}\|_F^2 + \nu(\|A - X - Y\|_F^2 - \sigma^2).\end{aligned}$$

Taking partial derivatives of L with respect to X and Y results in:

$$\begin{cases} \frac{\partial L}{\partial X} &= -U^k + \Lambda^k + \rho(X - S^{k+1}) + 2\nu(X + Y - A), \\ \frac{\partial L}{\partial Y} &= \frac{\lambda}{\theta_2}D_Y - \lambda V^k + 2\nu(X + Y - A), \end{cases}$$

where D_Y is the subdifferential of $\|Y\|_1$. Setting both partial derivatives to 0, we have

$$\rho Y + \frac{\lambda(2\nu + \rho)}{2\nu\theta_2}D_Y + C = 0, \quad (13)$$

where C is a constant. It is easy to verify that (13) is precisely the first-order optimality condition for the following problem:

$$\underset{Y}{\text{minimize}} \quad \frac{1}{2}\|Y + \frac{C}{\rho}\|_F^2 + \frac{\lambda(2\nu + \rho)}{2\nu\theta_2\rho}\|Y\|_1, \quad (14)$$

which has a closed-form solution summarized below [9]:

LEMMA 3. *The proximal operator associated with the ℓ_1-norm, i.e., the minimizer of the following problem:*

$$\underset{X\in\mathbb{R}^{m\times n}}{\text{minimize}} \quad \frac{1}{2}\|X - V\|_F^2 + \lambda\|X\|_1,$$

is given by:

$$prox_{\lambda\|\cdot\|_1}(V_{ij}) = sign(V_{ij}) \otimes \max(|V_{ij}| - \lambda, 0),$$

where the ℓ_1-norm of a matrix is given by the summation of the absolute value of all elements.

From Lemma 3, the solution of problem (13) can be obtained by $Y^{k+1} = \text{sign}(-\frac{C}{\rho}) \otimes \max\{|\frac{C}{\rho}| - \frac{\lambda(2\nu+\rho)}{2\nu\theta_2\rho}, 0\}$ based on ν. And from the relation between X and Y, we can get X^{k+1}. Since there is no direct way to calculate ν, we use the binary search to find the proper ν.

Thus, we obtain the solution of sub-problem (9), which is one iteration of DC framework. In the DC framework, a series of sub-problem (9) are solved iteratively. The details are summarized in Algorithm 1.

Algorithm 1 Robust PCA via DC programming

Require: X_0, Y_0, θ_1, θ_2, σ
Ensure: an optimal solution X and Y

```
while not converge do
    Calculate P_1(X) = sum_{i=1}^{p} max(sigma_i(X) - theta_1, 0) and
    P_2(Y) = sum_{i,j} max(|Y_ij| - theta_2, 0)
    Compute dP_1 and dP_2 according to (6) and (7)
    Apply ADMM to solve problem (9):
    for j = 1 to MaxIter do
        update S using (11)
        update X and Y using (14)
        update Lambda using (10)
    end for
end while
```

4. A FAST ALTERNATING ALGORITHM

The DC programming is known to have a slow convergence rate. In this section, we propose an algorithm based on alternating optimization which is practically much faster than DC programming. Notice that problem (5) is equivalent to:

$$\begin{aligned}&\underset{X,Y}{\text{minimize}} \quad \frac{1}{\theta_1}\sum_i \min\{\sigma_i(X), \theta_1\} + \frac{1}{\theta_2}\sum_{i,j}\min\{|Y_{ij}|, \theta_2\}, \\ &\text{subject to} \quad \|A - X - Y\|_F^2 \le \sigma^2.\end{aligned} \quad (15)$$

In the proposed algorithm, we iteratively fix one variable and compute the other one.

4.1 Computing the optimal Y

When X is fixed, computing the optimal Y amounts to solving the following sub-problem after dropping constants and changing variables:

$$\begin{aligned}&\underset{Y}{\text{minimize}} \quad \frac{1}{\theta_2}\sum_{i,j}\min\{|Y_{ij}|, \theta_2\}, \\ &\text{subject to} \quad \|Y - Z\|_F^2 \le \sigma^2,\end{aligned} \quad (16)$$

where $Z = A - X$. It is easy to see that we only need to consider the situation that $Z \ge 0$. Moreover, notice that for any feasible solution Y, if $Y_i > Z_i$ for some i, by setting $Y_i = Z_i$ we can always reach another feasible solution with equal or less objective value. Therefore we can assume that the optimal Y^* satisfies $Y \le Z$. In addition, we may further

assume Y and Z are both represented in a vector which is formed by stacking all columns of the matrix.

It is not hard to see that the objective function of problem (16) is non-convex and it is usually very difficult to find a globally optimal solution. We first present our method in Algorithm 2. Then we will show that, given any feasible solution of (17), we can always improve it to get a better local solution through our Algorithm 2.

Algorithm 2 An approximate algorithm for solving (16)

Require: Y, Z, θ
Ensure: an optimal solution Y
1: **if** $\|Z\|_F^2 \leq \sigma^2$ **then**
2: **return** 0
3: **end if**
4: Initialize Y by Z.
5: Sort Y in increasing order
6: $i = 1$
7: **while** $\sigma^2 > 0$ **do**
8: **if** $\sigma > Y_i$ **then**
9: $\sigma = \sqrt{\sigma^2 - Y_i^2}, \quad Y_i = 0.$
10: **else**
11: $\sigma = 0, \quad Y_i = Y_i - \sigma.$
12: **end if**
13: $i = i + 1$
14: **end while**
15: **return** Y

An intuitive example is presented in Figure 1. We first initialize Y by Z and sort Y such that the elements of Y form a non-decreasing sequence. The key idea behind our algorithm is as follows: among all the solutions of (16), there must be one Y^* such that the elements of Y^* preserves the order of Z.

LEMMA 4. *Let Y^* be one feasible solution of (16) such that there exist indices i and j satisfying $Z_i < Z_j$ and $Y_i^* > Y_j^*$. There always exists another local solution $\hat{Y}$ such that $\hat{Y}_i \leq \hat{Y}_j$ and $\hat{Y}_k = Y_k^*$ for all $k \neq i, j$.*

PROOF. If there exist indices i and j such that $Z_i < Z_j$ and $Y_i^* > Y_j^*$. Let $\hat{Y}_i = Y_j^*$, $\hat{Y}_j = Y_i^*$ and $\hat{Y}_k = Y_k^*$ for all $k \neq i, j$, then we have:

$$
\begin{aligned}
&\|Y^* - Z\|_F^2 - \|\hat{Y} - Z\|_F^2 \\
=&\|Y^* - Z\|_F^2 - \Big(\sum_{k\neq i,j} (\hat{Y}_k - Z_k)^2 + (\hat{Y}_i - Z_i)^2 + (\hat{Y}_j - Z_j)^2\Big) \\
=&\|Y^* - Z\|_F^2 - \Big(\sum_{k\neq i,j} (Y_k^* - Z_k)^2 + (Y_j^* - Z_i)^2 + (Y_i^* - Z_j)^2\Big) \\
=&(Y_i^* - Z_i)^2 + (Y_j^* - Z_j)^2 - (Y_j^* - Z_i)^2 - (Y_i^* - Z_j)^2 \\
=&2Y_j^* Z_i + 2Y_i^* Z_j - 2Y_j^* Z_j - 2Y_i^* Z_i \\
=&2(Y_i^* - Y_j^*)(Z_j - Z_i) \geq 0.
\end{aligned}
$$

The above result essentially shows that exchanging Y_i^* and Y_j^* will not violate the constraint and clearly the objective remains unchanged. Therefore we find an alternative feasible solution that preserves the order of elements in Z. □

LEMMA 5. *Let Y^* be one feasible solution of (16) such that there exists an index i satisfying $0 < Y_i^* < Y_{i+1}^* < Z_{i+1}$. There always exists another feasible solution $\hat{Y}$ such that $\hat{Y}_k = Y_k^*$ for all $k \neq i, i+1$ and either $\hat{Y}_i = 0$ or $\hat{Y}_{i+1} = Z_{i+1}$ holds.*

Lemma 4 essentially states that the optimal solution preserves the order in Z and Lemma 5 shows that we can obtain a solution such that there exists an index i such that $Y_j^* = 0$ for all $j < i$ and $Y_j^* = Z_j$ for all $j > i$. Then it is straight-forward to show that our algorithm provides a better local solution.

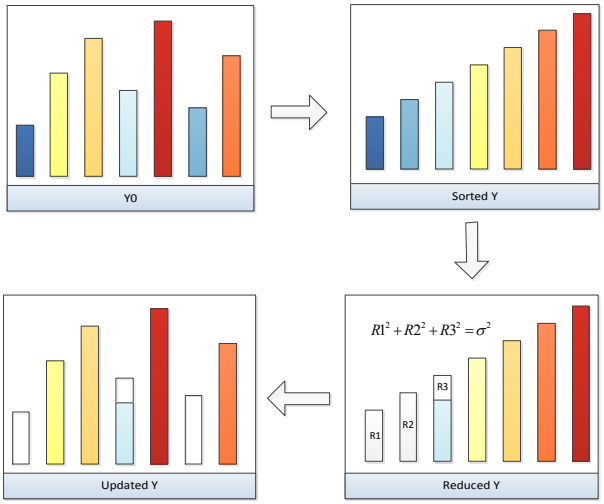

Figure 1: Illustration of Algorithm 2. Y is initialized as Z. We first sort the entries of Y to form a non-decreasing sequence $\{Y_i\}$, then reduce Y as much as possible sequentially within the threshold σ, finally put Y back in the original order.

4.2 Computing the optimal X

When Y is fixed, the optimal X can be obtained via solving the following equivalent problem:

$$
\begin{aligned}
\underset{X}{\text{minimize}} \quad & \frac{1}{\theta_1} \sum_i \min\{\sigma_i(X), \theta_1\}, \\
\text{subject to} \quad & \|X - Z\|_F^2 \leq \sigma^2.
\end{aligned}
\tag{17}
$$

Let $Z = U\Sigma V^T$ be the SVD of Z. For any feasible solution X, let $\tilde{U}\tilde{\Sigma}\tilde{V}^T$ be its SVD. We can observe that:

$$
\begin{aligned}
&\|X - Z\|_F^2 \\
=&\|X\|_F^2 + \|Z\|_F^2 - 2Tr(X^T Z) \\
=&\|U\tilde{\Sigma}V^T\|_F^2 + \|Z\|_F^2 - 2Tr(X^T Z) \\
\geq&\|U\tilde{\Sigma}V^T\|_F^2 + \|Z\|_F^2 - 2\sum_i \sigma_i(X)\sigma_i(Z) \\
=&\|U\tilde{\Sigma}V^T\|_F^2 + \|Z\|_F^2 - 2Tr((U\tilde{\Sigma}V^T)^T U\Sigma V^T) \\
=&\|U\tilde{\Sigma}V^T - Z\|_F^2,
\end{aligned}
$$

where the second equation follows from the fact that the Frobenius norm is unitary-invariant and we use the Von Neumann's trace inequality [15] to obtain the first inequality. The conclusion above essentially shows that the optimal X

shares the same left and right singular vectors with Z. Using the unitary-invariant property of the Frobenius norm we can conclude that (17) is equivalent to:

$$\begin{aligned} \underset{\sigma(X)}{\text{minimize}} \quad & \frac{1}{\theta_1}\sum_i \min\{\sigma_i(X), \theta_1\}, \\ \text{subject to} \quad & \sum_i (\sigma_i(X) - \sigma_i(Z))^2 \leq \sigma^2, \end{aligned} \tag{18}$$

which is exactly in the same form as (16) and therefore can also be computed via Algorithm 2.

5. EXPERIMENTAL RESULTS

In this section, we compare our proposed algorithms with ALM [13] and NSA [1] on synthetic data and real-world data sets. The ALM algorithm formulated RPCA as a convex problem with an equality constraint and applied augmented lagrange multiplier method to solve it. The NSA algorithm formulated the RPCA as a convex problem with an inequality constraint, and investigated a non-smooth augmented lagrange algorithm to solve it.

Our experiments were executed on a PC with Intel Core2 Quad Q8400 2.66G CPU and 8G RAM. The ALM code was downloaded from the Perception and Decision Laboratory, University of Illinois (`http://perception.csl.illinois.edu/`), and the NSA code was kindly provided to us by the author. We implemented both of the DC framework and fast alternating algorithm in MATLAB.

5.1 Synthetic data

In this experiment, we compare different algorithms on synthetic data. We generate the rank-r matrix X^0 as a product of LR^T, where L and R are independent $n \times r$ matrices whose elements are i.i.d. random variables sampled from standard Gaussian distributions. We generate Y^0 as a sparse matrix whose support is chosen uniformly at random, and whose non-zero entries are i.i.d. random variables sampled uniformly in the interval $[-100, 100]$. We set the standard Gaussian noise level at ρ to simulate $\xi \sim \mathcal{N}(0, \rho^2)$. The matrix $A = X^0 + Y^0 + \xi$ is the input to the algorithms.

In our empirical study, we set $\theta_1 = \theta_2 = 0.01$ and $\sigma^2 = \rho\sqrt{n + \sqrt{8n}}$ as in [20] through all our experiments and utilize the NSA results to initialize X and Y in our algorithms. Note that it is very common (in fact it is recommended) to use convex relaxation solutions as initial solutions for non-convex formulations [10, 25].

There are three key parameters in the process of generating A, namely, n, ρ and c_r: n is the dimension of A; c_r represents the ratio between the rank and the dimension of X^0, i.e., $r = n \times c_r$; c_p denotes the density of non-zero entries in Y^0, i.e., $\|Y\|_0 = c_p \times n$. To verify the efficacy and robustness of different approaches, we proceed our experiments in three directions, by varying different parameters:

- We fix ρ and c_r, and vary n in the set S_1, where $S_1 = \{100, 200, 500\}$.
- We fix c_r and n, and vary ρ in the set S_2, where $S_2 = \{0.0001, 0.001, 0.01\}$.
- We fix n and ρ, and vary c_r in the set S_3, where $S_3 = \{0.01, 0.02, 0.05, 0.1\}$.

Following the aforementioned three directions, the experimental results of different approaches are presented in Table 1, Table 2 and Table 3. In each table, comprehensive results of the recovery errors related to X and Y as well as the accuracy of capturing the sparse location are demonstrated. In particular, the computation time with respect to dimensions is summarized in Table 1.

Under all these conditions, our alternating algorithm outperforms ALM and NSA in terms of the rank of X and the sparsity of Y. In particular, the rank of X produced by our alternating algorithm is consistent with the value we generated through c_r in the case of $\rho = 0.001$. Moreover, for capturing the sparse locations in matrix Y, our DC algorithm and alternating algorithm both perform much better than ALM and NSA. Especially, in the case of $\rho = 0.001$, $n = 100$, compared to ALM (20.35%) and NSA (52.57%), we obtain 99.45% (DC) and 98.73% (alternating) accuracy which nearly capture all of the sparse locations in Y. Furthermore, our alternating algorithm achieves 81.42% accuracy in the case of $\rho = 0.01$, showing its robustness to noise.

We can observe that the computation time of our DC framework is longer than ALM and NSA. However, our alternating algorithm has comparable execution time and is much faster than DC framework. Therefore, in the following real-world applications, we only focus on our alternating algorithm.

5.2 Foreground and background separation on surveillance Video

In this experiment, we apply different approaches on the background separation for an airport surveillance video [12]. The dataset contains a sequence of 201 grayscale frames of size 144×176 during a time period. To form the matrix A, we stack the columns of each frame into a vector and concatenate the vectors together. We manually add the Gaussian noise by assuming only impulse noise contained in the video. The Gaussian noise is set at $20dB$ signal-to-noise ratio (SNR) and $\rho = \frac{\|A\|_F}{\sqrt{144\times176\times201\times10^{\mathrm{SNR}/20}}}$ [1]. The input A' is generated through $A' = A + \rho G$, where $G \in \mathbb{R}^{25344\times201}$ and each entry of G is generated i.i.d. from the standard Gaussian distribution.

The results of different algorithms are presented in Table 4. In addition, we show the results of three frames of the video in Figure 2. Each of Figure 2(a), Figure 2(b), Figure 2(c) represents the results of ALM, NSA and our algorithm respectively. In each figure, the first row represents the 15-th, 150-th and 200-th frame after adding noise, the second row represents background and the third row relates to the people in the video. Notice that there is one person who is identified as part of background; this is due to the fact that he/she did not move at all during the period we focus on.

From Figure 2, we conclude that all of the three algorithms can successfully extract background from surveillance video. Even though the visual qualities of background and foreground are similar among different algorithms, the numerical measurements in Table 4 demonstrate that our alternating approach performs better than both ALM and NSA in terms of the low rank and sparsity. In particular, the rank of our X is 67, which is about half of the rank of X computed by ALM or NSA.

Table 1: Synthetic results for varying n with fixed $\rho = 0.001$ and $c_r = 0.05$. Our algorithms perform much better than ALM and NSA in terms of capturing sparse locations in Y.

Dimension	Alg	rank(X)	$\|Y\|_0$	time(sec)	$\frac{\|X-X^0\|_F}{\|X^0\|_F}$	$\frac{\|Y-Y^0\|_F}{\|Y^0\|_F}$	$\frac{\|A-X-Y\|_F}{\|A\|_F}$	$\sum I(Y_{ij} = Y^0_{ij})$
n=100	ALM	60	8535	0.3869	2.72e-4	6.46e-5	1.83e-6	0.2035
	NSA	57	5228	0.6135	2.98e-4	5.63e-5	8.93e-6	0.5257
	OurDC	100	539	93.8845	4.19e-4	6.79e-5	8.41e-5	**0.9945**
	OurAL	5	612	4.3162	3.30e-3	6.11e-4	8.38e-5	**0.9873**
n=200	ALM	119	34224	9.1882	1.87e-4	6.10e-5	1.92e-6	0.1930
	NSA	116	21704	2.1024	2.12e-4	5.34e-5	5.87e-6	0.5060
	OurDC	200	3605	552.8801	2.74e-4	3.78e-5	4.66e-5	**0.9586**
	OurAL	10	7148	5.9051	4.52e-4	1.19e-4	4.71e-5	**0.8699**
n=500	ALM	299	214001	73.7280	1.16e-4	6.18e-5	1.91e-6	0.1927
	NSA	259	150302	19.3789	1.16e-4	6.55e-5	3.51e-6	0.4475
	OurDC	500	58745	5259.9872	1.64e-4	2.71e-5	2.25e-5	**0.8139**
	OurAL	50	73997	94.8372	1.25e-4	7.96e-5	2.25e-5	**0.7527**

Table 2: Synthetic results for varying ρ with fixed $n = 100$ and $c_r = 0.05$. Our algorithms perform much better than ALM and NSA in terms of capturing sparse locations in Y.

Gaussian noise	Alg	rank(X)	$\|Y\|_0$	$\frac{\|X-X^0\|_F}{\|X^0\|_F}$	$\frac{\|Y-Y^0\|_F}{\|Y^0\|_F}$	$\frac{\|A-X-Y\|_F}{\|A\|_F}$	$\sum I(Y_{ij} = Y^0_{ij})$
$\rho = 0.0001$	ALM	5	493	1.57e-5	1.86e-6	7.20e-6	0.9995
	NSA	56	5252	3.13e-5	5.44e-6	9.00e-7	0.5233
	OurDC	100	490	1.32e-4	2.06e-5	2.63e-5	**0.9999**
	OurAL	5	504	1.74e-4	1.24e-5	2.52e-5	**0.9984**
$\rho = 0.001$	ALM	60	8535	2.72e-4	6.46e-5	1.83e-6	0.2035
	NSA	57	5228	2.98e-5	5.63e-5	8.93e-6	0.5257
	OurDC	100	539	4.19e-4	6.79e-5	8.41e-5	**0.9945**
	OurAL	5	612	3.30e-3	6.11e-4	8.38e-5	**0.9873**
$\rho = 0.01$	ALM	68	8107	3.00e-3	5.83e-4	7.85e-6	0.2387
	NSA	57	5290	2.90e-3	5.21e-4	9.01e-5	0.5204
	OurDC	100	4014	2.71e-3	4.82e-4	2.59e-4	**0.6478**
	OurAL	14	2352	3.40e-3	7.09e-4	2.68e-4	**0.8142**

Table 3: Synthetic results for varying c_r with fixed $n = 100$ and $\rho = 0.001$. Our algorithms perform much better than ALM and NSA in terms of capturing sparse locations in Y.

Rank ratio	Alg	rank(X)	$\|Y\|_0$	$\frac{\|X-X^0\|_F}{\|X^0\|_F}$	$\frac{\|Y-Y^0\|_F}{\|Y^0\|_F}$	$\frac{\|A-X-Y\|_F}{\|A\|_F}$	$\sum I(Y_{ij} = Y^0_{ij})$
$c_r = 0.01$	ALM	58	8332	4.38e-4	5.70e-5	6.08e-6	0.2159
	NSA	57	5308	4.90e-4	4.93e-5	9.10e-6	0.5183
	OurDC	100	499	6.37e-4	7.28e-5	8.38e-5	**0.9991**
	OurAL	1	635	2.11e-3	1.75e-4	8.55e-5	**0.9856**
$c_r = 0.02$	ALM	59	8429	3.43e-4	6.10e-5	3.42e-6	0.2060
	NSA	57	5295	3.89e-4	5.21e-5	9.11e-6	0.5194
	OurDC	100	498	5.61e-4	7.12e-5	8.22e-5	**0.9989**
	OurAL	2	630	4.10e-3	4.80e-4	8.56e-5	**0.9859**
$c_r = 0.05$	ALM	60	8535	2.72e-4	6.46e-5	1.83e-6	0.2035
	NSA	57	5228	2.98e-5	5.63e-5	8.93e-6	0.5257
	OurDC	100	539	4.19e-4	6.79e-5	8.41e-5	**0.9945**
	OurAL	5	612	3.30e-3	6.11e-4	8.38e-5	**0.9873**
$c_r = 0.1$	ALM	63	8424	2.28e-4	6.57e-5	1.39e-6	0.2063
	NSA	56	5243	1.30e-3	3.12e-4	8.65e-6	0.5244
	OurDC	100	727	3.88e-4	6.40e-5	8.54e-5	**0.9758**
	OurAL	10	621	3.11e-3	7.68e-4	8.13e-5	**0.9866**

(a) ALM (b) NSA (c) Our

Figure 2: Background extraction results of different algorithms. In each subfigure, the 15-th, 150-th and 200-th flame after adding noise are shown in the first row. The low-rank recoveries and the sparse components extracted by different algorithms are shown in the second and in the last row, respectively.

(a) Subject01 (b) Subject05 (c) Subject10

Figure 3: Shadow removal results of different algorithms (Top: ALM, Middle: NSA, Bottom: our proposed algorithm). In each subfigure, the YaleB images after adding noise are shown in the left column. The low-rank recoveries of different algorithms are shown in the middle column, and the sparse errors corresponding to illumination are shown in the right column.

Table 4: Recovery results of airport surveillance. Our algorithm produces X with a lower rank and Y with a smaller ℓ_0-norm, while keeping the relative error comparable with ALM and NSA.

Alg	rank(X)	$\|Y\|_0$	$\frac{\|X+Y-A\|_F}{\|A\|_F}$
ALM	119	4900870	2.8e-7
NSA	129	4915415	6.8e-4
Our	**67**	**4732014**	4.4e-5

5.3 Shadows removal from face images

Another interesting application of RPCA is to remove shadows and specularities from face images [6]. Often times, portraits are taken under different illuminations which introduce errors to face recognition. If we have enough images under different illuminations of the same face, we can apply RPCA to extract the features of the face and remove the illumination errors.

We compare different algorithms on the YaleB face data [11]. The images of the dataset are of size 192×168, and there are 64 illuminations for each subject. Illuminations vary from both azimuth and elevation, so the shadows for each subject are different in both location and intensity. Considering one subject, the matrix A is constructed by concatenating 64 images under all illuminations together. We add $20dB$ signal-to-noise ratio which is similar to surveillance video. We obtain a noisy A' through $A' = A + \rho G$, where $G \in \mathbb{R}^{32256\times 64}$ and each entry of G is generated i.i.d. from the standard Gaussian distribution. In this application, the low rank part would represent human face, while the sparse component is the shadow induced by different illuminations.

Considering Subject01, Subject05 and Subject10 in the dataset, the results of all three algorithms are shown in Figure 3. For better comparison, each subfigure represents one subject. And the three columns represent observation image, low-rank recovery and sparsity illumination from left to right. Though the visual quality of different algorithms are similar, our algorithm obtains a lower rank of X and a smaller ℓ_0-norm of Y than ALM and NSA, while keeping the relative error comparable, as demonstrated in Table 5.

Table 5: YaleFaceB recovery results. Our algorithm produces X with a lower rank and Y with a smaller ℓ_0-norm, while keeping the relative error comparable with ALM and NSA.

	Alg	rank(X)	$\|Y\|_0$	$\frac{\|X+Y-A\|_F}{\|A\|_F}$
Subject01	ALM	28	1832343	0.0637
	NSA	50	1835919	0.0634
	Our	**27**	**1707272**	0.0637
Subject05	ALM	29	1823015	0.0637
	NSA	49	1828608	0.0634
	Our	**26**	**1698480**	0.0637
Subject10	ALM	28	1848333	0.0637
	NSA	48	1827221	0.0635
	Our	**26**	**1703188**	0.0637

6. CONCLUSION AND FUTURE WORK

This paper investigates a non-convex formulation for Robust Principle Component Analysis (RPCA) by the capped trace norm and the capped ℓ_1-norm. We develop a DC framework as well as a fast alternating algorithm to solve the non-convex formulation. In the DC framework, ADMM is applied to solve the sub-problem, while in our fast alternating algorithm, a greedy algorithm for solving the sub-problem is developed based on the combinatorial optimization. We have performed extensive experiments on both synthetic and real-world datasets. Results show that our proposed approaches perform better in recovering the low-rank part and the sparse component of a given matrix than existing work.

The current work assumes that the complete data is given, i.e., there are no missing entries in the data. However, in many real applications the data may come with missing values. We plan to extend our proposed algorithms to solve the RPCA problem with missing entries in the future. In addition, we plan to study the theoretical properties of the proposed non-convex formulation.

7. ACKNOWLEDGEMENT

The authors are grateful to Professor Necdet Serhat Aybat from Pennsylvania State University for providing the NSA code for comparison. This work was supported in part by NIH (LM010730) and NSF (IIS-0953662).

8. REFERENCES

[1] N. Aybat, D. Goldfarb, and G. Iyengar. Fast first-order methods for stable principal component pursuit. *arXiv preprint arXiv:1105.2126*, 2011.

[2] L. Benoît, J. Mairal, F. Bach, and J. Ponce. Sparse image representation with epitomes. In *CVPR*. IEEE, 2011.

[3] 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*, 2011.

[4] J. Cai, E. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *Arxiv preprint Arxiv:0810.3286*, 2008.

[5] J. Cai, E. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 2010.

[6] E. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of ACM*, 2009.

[7] V. Chandrasekaran, S. Sanghavi, P. Parrilo, and A. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 2011.

[8] J. Chen, J. Liu, and J. Ye. Learning incoherent sparse and low-rank patterns from multiple tasks. *TKDD*, 2012.

[9] D. Donoho. De-noising by soft-thresholding. *IEEE Trans, Information Theory*, 1995.

[10] J. Fan, L. Xue, and H. Zou. Strong oracle optimality of folded concave penalized estimation. *arXiv preprint arXiv:1210.5992*, 2012.

[11] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *PAMI*, 2001.

[12] L. Li, W. Huang, I. Gu, and Q. Tian. Statistical modeling of complex backgrounds for foreground object detection. *IEEE Trans, Image Processing*, 2004.

[13] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *Arxiv preprint arXiv:1009.5055*, 2010.

[14] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. *JMLR*, 2011.

[15] L. Mirsky. A trace inequality of john von neumann. *Monatshefte für Mathematik*, 1975.

[16] Y. Nesterov. Introductory lectures on convex programming volume i: Basic course. 1998.

[17] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. In *CVPR*. IEEE, 2010.

[18] B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 2010.

[19] A. Singer and M. Cucuringu. Uniqueness of low-rank matrix completion by rigidity theory. *SIAM Journal on Matrix Analysis and Applications*, 2010.

[20] M. Tao and X. Yuan. Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM Journal on Optimization*, 2011.

[21] P. Tao and L. An. Convex analysis approach to dc programming: Theory, algorithms and applications. *Acta Mathematica Vietnamica*, 1997.

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

[23] H. Xu, C. Caramanis, and S. Sanghavi. Robust pca via outlier pursuit. *arXiv preprint arXiv:1010.4237*, 2010.

[24] J. Ye and J. Liu. Sparse methods for biomedical data. *ACM SIGKDD Explorations Newsletter*, 2012.

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