*International Journal of Latest Research in Engineering and Management" (IJLREM)*
*ISSN: 2456-0766*
*www.ijlrem.org ‖ Volume 05 Issue 01 ‖ January 2021 ‖ PP 01-07*

# method for robust principal component analysis based on $l_{1/2}$norm

Baicai Xue[1], Tao Yan[2]
[1](college of science/ Nanjing University of Science and Technology, BAICAI XUE)
[2]((college of science/ Nanjing University of Science and Technology, TAO YAN)

**Abstract:** Robust Principal Component analysis (RPCA) can separate a low-rank matrix and a sparse matrix from a given data matrix. It can be applied in many engineering area, such as Video surveillance,Face Recognition and so on. In this paper, we use $l_{1/2}$ norm to get an improved non-convex model without a regularization term. Based on the new model, we apply the ADMM algorithm to solve the original RPCA problem. The convergence analysis of the method is also presented. Numerical experiments show the efficiency of the new model and method

**Keywords:**RPCA, $l_{1/2}$ norm,non-convex

## 1. Introduction

With the proliferation of high-dimensional data in many fields of scientific engineering, such as image video and biometric, we need to degrade the relevant data, and PCA is one of the most widely used statistical tools for data analysis and dimensional reduction. On the other hand, the existence of errors can cause the matrix low rank estimate to be far from its corresponding true value. In turn, the robustness of PCA has become one of the research directions, that is, the research data can be modeled as low rank part plus sparse part. In general, the robust PCA (RPCA) model is defined as follows:

$$\min_{L,S\in R^{m\times n}} rank(L)+\rho\|S\|_0 \quad s.t. L+S=D. \text{（1）}$$

where$D\in R^{m\times n}$is a given data matrix, $L\in R^{m\times n}$is a low rank matrix, and $S\in R^{m\times n}$ is a sparse matrix.Model (1) is NP-hard and is numerically difficult to handle. However, under certain conditions, (1) can be equivalent to the following convex model：

$$\min_{L,S\in R^{m\times n}} \|L\|_*+\rho\|S\|_1 \quad s.t. L+S=D. \text{（2）}$$

Besides model (1) and model (2), there are other non-convex models. In [1], The authors gives a simple RPCA non-convex model:

$$\min_{U,V}\|UV^T-D\|_1, \text{（3）}$$

and the model does not regularization U and V. Model (3) can be rewritten as the following form:

$$\min_{S}\|S\|_1 \quad s.t. \ S+UV^T=D. \text{（4）}$$

$U\in R^{m\times k}$ ， $V\in R^{n\times k}$.Based on model (4)， the authors proposes an ADMM algorithm to solve the original problem.

On the other hand, the non-convex approximations of $l_0 norm$ can yield better results. Thus, There are many non-convex function approximations of $l_0$ $norm$ are proposed, such as$l_p$ $norm$ （$p\in(0,1)$）[2], SCAD function [3], Logarithm function [4], MCP function [5], Capped $l_1$ function [6], ETP function [7], Geman function [8]， Laplace function[9] and so on. In [10], the authors presented an iterative semi-threshold algorithm to solve the $l_{1/2}$ regularization ， numerical experiments also show the efficiency of the semi-algorithm.

Inspired by the research results above, we replace the$l_1$norm with the $l_{1/2}$ norm based on model (4) in order to get more accuracy solution. We make an assumption that the rank of $L=UV$ does not exceed the specified estimate of $k<max(m,n)$ and introduce the variable $Z$ which satisfies $Z=L$, then we get a new nonconvex model as follows:

$$min\|Z-D\|_{\frac{1}{2}}^{\frac{1}{2}} s.t. \ UV-Z=0. \text{(5)}$$

where $U\in R^{m\times k} V\in R^{k\times n}$.

The paper is organized as follows. In Section Ⅱ, we propose an algorithm based on problem (5) and present the convergence result for the algorithm and the model. The numerical results and the conclusion of this paper are illustrated in Sections Ⅲand Ⅳ, respectively.

## 2. Algorithms and Convergence

We apply the augmented Lagrangian alternating direction method for solving problem (5). First, we define the augmented Lagrangian function of (5) as follows：

$$\mathcal{L}_{\beta}(\mathrm{U},\mathrm{V},\mathrm{Z},\Lambda) = ||(\mathrm{Z}-\mathrm{D})||_{\frac{1}{2}}^{\frac{1}{2}} + < \Lambda, UV-\mathrm{Z} > + \frac{\beta}{2}\left||UV-Z|\right|_{\mathrm{F}}^{2}, \quad (6)$$

Where $\beta > 0$is a penalty parameter and $\Lambda \in R^{m\times n}$ is the Lagrange multiplier corresponding to the constraint $UV - Z = 0$， $< X, Y >= \sum_{i,j} X_{i,j}\, Y_{i,j}$ denotes the inner product between matrices X and Y of equal sizes.Then we can solve problem (5) by solving the following optimization problems:

$$\min_{U,V,Z} \mathcal{L}_{\beta}(U,V,Z,\Lambda^{l}). \quad (7)$$

By the ADMM method, we can minimize the Lagrange function for each variable block of U, V, and Z in turn while fixing the other two blocks with their latest values, and then update the Lagrange multiplier .
With the rule above, we get the following iterative form:

$$U^{l+1} = \mathrm{argmin}_{U\in R^{m\times k}}\, \mathcal{L}_{\beta}\,(\mathrm{U}, V^{l}, Z^{l}, \Lambda^{k}) \quad (8a)$$

$$V^{l+1} = \mathrm{argmin}_{V\in R^{k\times n}}\, \mathcal{L}_{\beta}\,(U^{l+1}, \mathrm{V}, Z^{l}, \Lambda^{l}) \quad (8b)$$

$$Z^{l+1} = \mathrm{argmin}_{Z\in R^{m\times n}}\, \mathcal{L}_{\beta}\,(U^{l+1}, V^{l+1}, Z, \Lambda^{l}) \quad (8c)$$

$$\Lambda^{l+1} = \Lambda^{k} + \gamma\beta(U^{l+1}V^{l+1} - Z^{l+1}). \quad (8d)$$

Where $\gamma > 0$ is a step length. The subproblems (8a) and (8b) are simple least squares problems, whose solutions are：

$$B = Z - \frac{\Lambda}{\beta}, U^{l+1} = BV^{\dagger}, V^{l+1} = (U^{l+1})^{\dagger}B. \quad (9)$$

$X^{\dagger}$ denotes the pseudo-inverse of the matrix X. In addition, (9) can be rewritten as following form[1]:

$$B = Z - \Lambda/\beta, U^{l+1} = \mathrm{orth}(BV^{T}), V^{l+1} = {U^{l+1}}^{T}B.$$

Differ from [1], we use $l_{\frac{1}{2}}$ norm to construct the mode in this paper. Therefore we need to rewritten the solution $Z^{l+1}$ to the subproblem (8c). According to [10], the $l_{\frac{1}{2}}$ norm can be calculated by the iterative half-threshold algorithm,

$$R_{\lambda,\frac{1}{2}}(x) = ((f_{\lambda,\frac{1}{2}}(x_1)，\ f_{\lambda,\frac{1}{2}}(x_2), f_{\lambda,\frac{1}{2}}(x_N))^{T},$$

$$f_{\lambda,\frac{1}{2}}(x) = \frac{2}{3}x_i\left(1+\cos\left(\frac{2}{3}\pi - \frac{2}{3}\varphi\lambda(x_i)\right)\right),$$

$$\varphi\lambda(x_i) = \arccos\left(\frac{\lambda}{8}\left(\frac{|x_i|}{3}\right)^{-\frac{3}{2}}\right),$$

$$h_{\lambda,\frac{1}{2}}(x) = \begin{cases} f_{\lambda,\frac{1}{2}}(x) & |x| > \frac{\sqrt[3]{54}}{4}\ (\lambda)^{\frac{2}{3}} \\ 0 & otherwise \end{cases}.$$

It implies that $Z^{l+1}$ can be expressed as following form:

$$Z^{l+1} = \begin{cases} f_{\frac{1}{\beta},\frac{1}{2}}\left(U^{l+1}V^{l+1} - D + \frac{\Lambda}{\beta}\right) + D & |x| > \frac{\sqrt[3]{54}}{4}\ \left(\frac{1}{\beta}\right)^{\frac{2}{3}} \\ D & otherwise \end{cases}. \quad (10)$$

Summarizing the result (8d), (9) and (10), we give the following algorithm framework.

**Algorithm 1: $l_{1/2}$ method**

1. Input D, initial rank estimate k， parameter $\beta$， $\gamma$
2. set $l$=0.initialize$V^{0} \in R^{k\times n}, Z^{0}, \Lambda^{0} \in R^{m\times n}$
3. while not converge do
4. Compute $U^{l+1}, V^{l+1}$， by(9)， $Z^{l+1}\ by\ (10)$, and $\Lambda^{l+1}\ by\ (8d)$
5. Increment $l$， and possibly re-estimate $k$ and adust sizes of iterates.

Next, we start to give the convergence analysis. It is straightforward to derive the KKT conditions for (5):

$$\Lambda \in \partial_Z(\|Z-D\|_{\frac{1}{2}}^{\frac{1}{2}}),$$
$$\Lambda V^T = 0, U^T\Lambda = 0,$$
$$UV - Z = 0,$$

where, for any $\beta > 0$， $\Lambda \in \partial_Z(\|Z-D\|_{\frac{1}{2}}^{\frac{1}{2}})$is equivalent to

$$Z - D + \frac{\Lambda}{\beta} \in \frac{1}{\beta}\partial_Z\left(\|Z-D\|_{\frac{1}{2}}^{\frac{1}{2}}\right) + Z - D\,. \qquad (11)$$

On the other hand, since $x \in R_0^N$is the solution to the problem

$$\min_{x\epsilon R^N}\{\cos\| y - Ax \| + \lambda \| x \|_{\frac{1}{2}}^{\frac{1}{2}}\},$$

then the first-order optimal condition with respect to variable x can be obtained, i.e.,

$$0 = \mathrm{A}^{\mathrm{T}}(\mathrm{Ax} - \mathrm{y}) + \frac{\lambda}{2}\nabla(\| x \|_{\frac{1}{2}}^{\frac{1}{2}}),$$

where $\nabla\left(\| x \|_{\frac{1}{2}}^{\frac{1}{2}}\right)$is the gradient of $\| x \|_{\frac{1}{2}}^{\frac{1}{2}}$. Multiplying by positive parameter μ and adding $x$ on both sides of formula above, then we have

$$\mathrm{x} + \mu\mathrm{A}^{\mathrm{T}}(\mathrm{y} - \mathrm{Ax}) = \mathrm{x} + \frac{\lambda\mu}{2}\nabla(\| x \|_{\frac{1}{2}}^{\frac{1}{2}}).$$

Because the operator

$$R_{\lambda,\,\frac{1}{2}}(.) = (I + \frac{\lambda}{2}\nabla(\|.\|_{\frac{1}{2}}^{\frac{1}{2}})^{-1}$$

is well defined for any positive real $\lambda$ by the fact that $\nabla(\| x \|_{\frac{1}{2}}^{\frac{1}{2}})$ exists, then from [10], the equation $x = y + \frac{\lambda}{2}\nabla\left(\| y \|_{\frac{1}{2}}^{\frac{1}{2}}\right)$has a unique solution$y^* = R_{\lambda,\,\frac{1}{2}}(x)$for any $x\epsilon D_{\frac{1}{2}}^N$.

Let $x = Z - D + \frac{\Lambda}{\beta}$ and $y = Z - D$**,** from the analysis above, we can rewrite (11)as

$$x = y + \frac{\lambda}{2}\nabla(\| y \|_{\frac{1}{2}}^{\frac{1}{2}}),$$

and the equation has a unique solution $y^* = R_{\lambda,\,\frac{1}{2}}(x)$. Then we have

$$Z - D = (I + \frac{\lambda}{2}\nabla(\| x \|_{\frac{1}{2}}^{\frac{1}{2}})^{-1} = (I + \frac{\lambda}{2}\nabla(\| Z - D + \frac{\Lambda}{\beta} \|_{\frac{1}{2}}^{\frac{1}{2}})^{-1}.$$

Therefore the KKT conditions of (5) can be written as, for any $\beta > 0$,

$$\Lambda V^T = 0, U^T\Lambda = 0,\ UV - Z = 0, \qquad (12a)$$

$$Z - D = (I + \frac{\lambda}{2}\nabla(\| Z - D + \frac{\Lambda}{\beta} \|_{\frac{1}{2}}^{\frac{1}{2}})^{-1}. \qquad (12b)$$

We have the following convergence theorem.

**Theorem 1**:Let $X \triangleq (U, V, Z, \Lambda)$ and $\{X^l\}_{j=1}^{\infty}$ be generated by Algorithm 1， Assume that $\{X^l\}_{j=1}^{\infty}$ is bounded and $\lim_{j\to\infty}(X^{l+1} - X^l) = 0$. Then any accumulation point of $\{X^l\}_{j=1}^{\infty}$ satisfies the KKT conditions (12). In particular, whenever $\{X^l\}_{j=1}^{\infty}$converges， it converges to a *KKT* point of (5).

**Proof:**It follows from (9), (12) and the identities$V^{\dagger}VV^T = V^T$ and $(U^{l+1})^{\intercal}U^{l+1}(U^{l+1})^{\intercal} = (U^{l+1})^{\intercal}$, that

$$(U^{l+1} - U^l)V^l(V^l)^{\intercal} = (Z^l - \frac{\Lambda^l}{\beta} - U^lV^l)(V^l)^{\intercal},$$
$$(U^{l+1})^{\intercal}U^{l+1}(V^{l+1} - V^l) = (U^{l+1})^{\intercal}(Z^l - \frac{\Lambda^j}{\beta} - U^{l+1}V^l),$$

$$Z^{l+1} - Z^l = I + \frac{\beta}{2}\nabla(\| Z - D + \frac{\Lambda}{\beta} \|_{\frac{1}{2}}^{\frac{1}{2}})^{-1} + D - Z^l,$$

$$\Lambda^{l+1} - \Lambda^l = \beta(U^{l+1}V^{l+1} - Z^{l+1}) . \quad (13)$$

Since $\{X^l\}_{l=1}^{\infty}$ is bounded by our assumption, the sequences $\{V^l(V^l)^\top\}_{l=1}^{\infty}$ and $\{(U^{l+1})^\top U^{l+1}\}_{l=1}^{\infty}$ are bounded. Hence $\lim_{l\to\infty}(X^{l+1} - X^l) = 0$ implies that both sides of (12) all tend to 0 as $l$ goes to infinity. Consequently,

$$U^lV^l - Z^l \to 0,\ \Lambda^l(V^l)^\top \to 0,\ (U^l)^\top\Lambda^l \to 0, \quad (14a)$$

$$I + \frac{\beta}{2}\nabla(\| Z - D + \frac{\Lambda}{\beta} \|_{\frac{1}{2}}^{\frac{1}{2}})^{-1} + D - Z^l \to 0, \quad (14b)$$

where the first limit in (14a) is used to derive other limits. That is, the sequence $\{X^l\}_{l=1}^{\infty}$ satisfies KKT conditions (12),from which the conclusions of the proposition follow readily. This completes the proof.

## 3. Numerical Experiments

In this section, we compare the algorithm $l_{1/2}$ method with the LMaFit and IALM methods to verify the validity of the algorithm. First of all, the corresponding marks and experimental instructions are given. $k^*$represents the rank of the low rank matrix $L$, $\|S^*\|_0$is the number of non-zero elements for sparse matrix S， and the density of $S^*$ by $\|S^*\|_0/mn$. We compared the performance of the algorithm involved by the number of cycles(iter), CPU time (seconds), the relative error between the recovered low-rank matrix $L$ and the exact one $L^*$,

$$relerr := \frac{\|L-L^*\|_F^2}{\|L^*\|_F^2} .$$

For random questions, the*iter*, *CPU*, and *relerr*in the numeric results are the averages of the corresponding results. We selectively used the following three stopping rules for all three codes compared：

$$(1)\ relerr < tol1, (2) relchg < tol2 , (3) iter < maxit ,$$

where $relchg := \frac{\|L^j - L^{j-1}\|_F^2}{\|L^{j-1}\|_F^2}$.tol1 and tol2 are prescribed tolerance values, and maxit is a prescribed maximum iteration number with the default value set to 100. We used rules (2) and (3) for comparing recoverability, and we used rules (1) and (3) to compare running speed. In some difficult cases, we do make an exception for IALM when it had trouble to reduce the relative error to a prescribed tolerance as other two codes could. All numerical experiments were performed on a notebook computer with an Inter®Core™i7-8565U1.80GHz processor, 8G memory.

## 4. Numerical Results

Test 1: Random matrices with rank $k^* = 10$ were constructed by the method in the section 3.3in[1]. The sparse matrices were from five images in the Matlab Image Processing toolbox: “blobs”, “circles”, “phantom”, “text” and “rice”. We run our algorithms, LMaFit and IALM, without and with noise added respectively. $l_{1/2}$ method and LMaFit use the stop criterion $tol2 = 10^{-4}$, IALM uses the stop criterion $tol3 = 10^{-4}$. The other operating parameter settings are same as the ones in [1]. The relative errors generated by the algorithms are shown in Tables 1 and 2.The recovered images are shown in Figures 1 and 2, where one is a sparse image of recovery without adding pulse noise, and the other is a sparse image with a pulse noise value of 0.15 that is randomly added with a pulse noise value of 10% pixels.

Table 1：Relative error without noise

| Solver/Pattern | blob | circle | phantom | text | rice |
|---|---|---|---|---|---|
| $l_{1/2}$ method | 5.12e-4 | 2.73e-4 | 1.14e-4 | 2.76e-4 | 2.43e-4 |
| Lmafit | 6.42e-4 | 3.56e-4 | 1.20e-4 | 4.18e-4 | 2.51e-4 |
| IALM | 8.22e-1 | 9.17e-1 | 6.46e-1 | 1.73e-1 | 2.99e-1 |

Table 2：Relative error with a pulse noise value of 0.15

| Solver/Pattern | blob | circle | phantom | text | rice |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| $l_{1/2}$ method | 4.60e-4 | 1.92e-4 | 1.17e-4 | 3.38e-4 | 2.19e-4 |
| Lmafit | 6.21e-4 | 9.09e-4 | 1.93e-4 | 3.65e-4 | 5.80e-4 |
| IALM | 7.18e-1 | 8.54e-1 | 5.29e-1 | 1.37e-1 | 2.80e-1 |

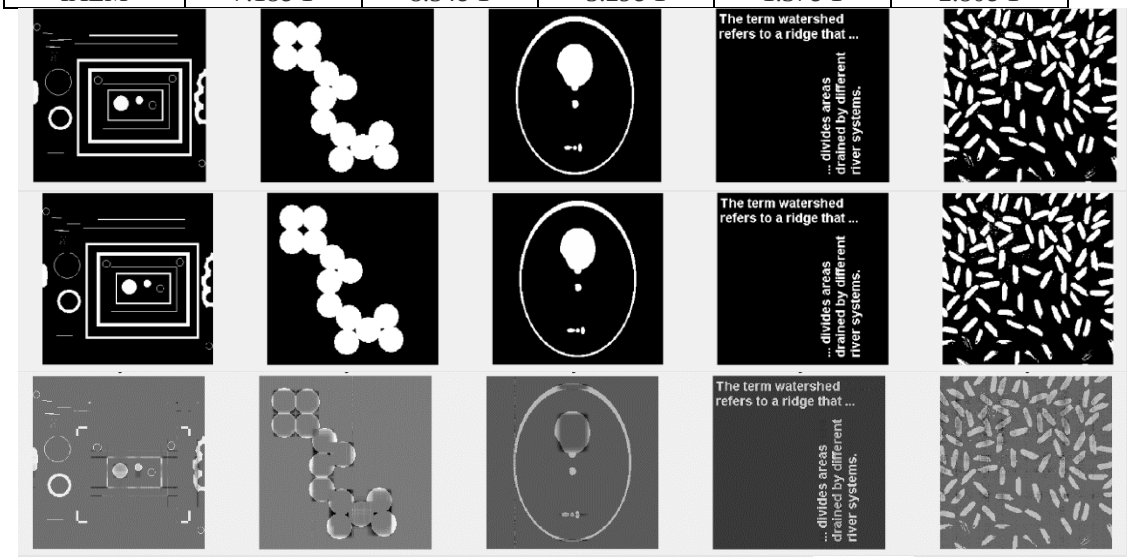


Fig 1：Recovery images that does not have noise. From top to bottom are $l_{1/2}$ method, Lmafit, IALM recovery images

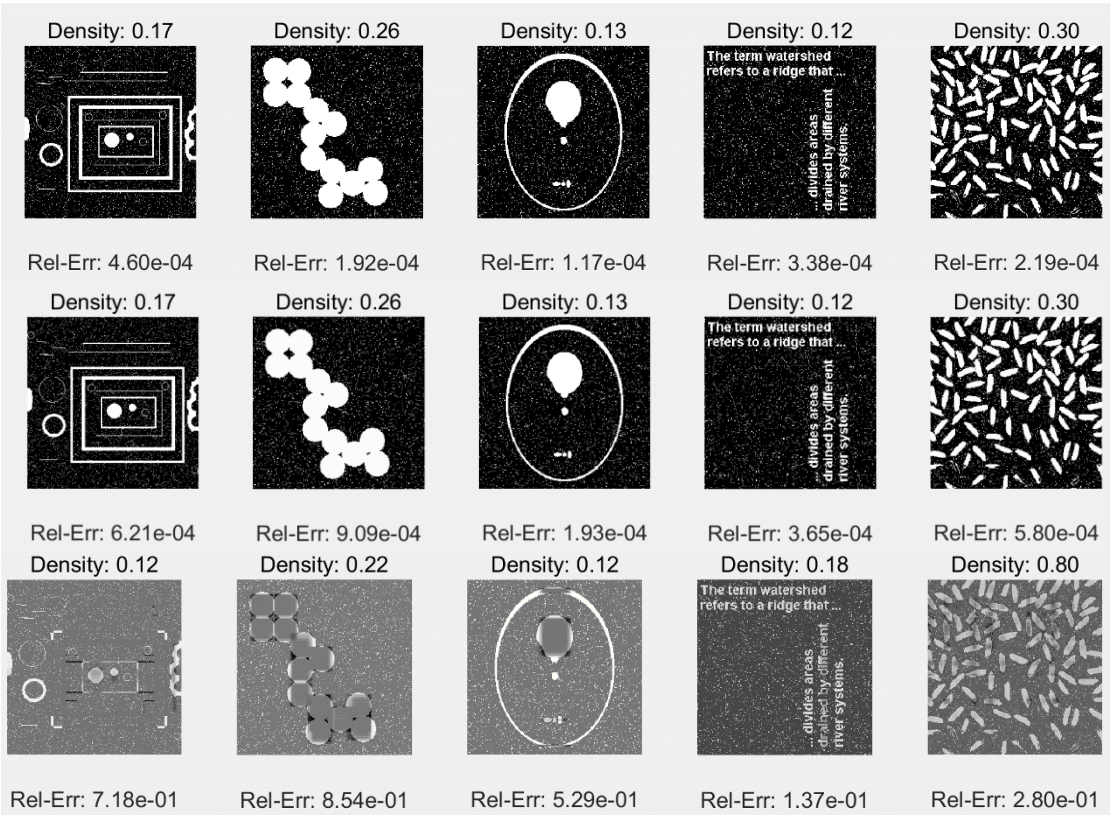


Fig2：Recovery images with 0.15 noise value images. From top to bottom are $l_{1/2}$ method, Lmafit, IALM recovery images

As can be seen from Tables 1 and 2, $l_{1/2}$ method produce less relative errors than the other two methods.Figure 1 and 2 show that image recovery by the $l_{1/2}$ method is better than the results from Lmafit and the IALM algorithm.

Test 2：Considering that low-rank matrices are determinic, sparse matrices are random. Our test is on a "board"image with 256× 256. We reconstruct the image by adding a sparse matrix whose non-zeros are uniformly distributed in [0,1], and the locations of the nonzeros are uniformly randomly sampled. The sparsity level $\|S^*\|_0/m^2$ range from 0.05 to 0.8,with increment 0.05. The results of the comparison are shown in Figure 3. The relative error result is shown in Table 3.

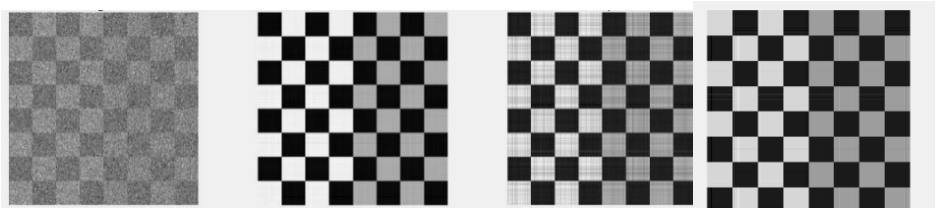

**Fig 3:** From left to right (1) is the reconstructed image with 70% noise, (2) recovered by the $l_{1/2}$method (3) recovered by the IALM algorithm, (4) recovered by the Lmafit.

**Table 3:** results of relative error

| Algorithm | XY RelErr | Z RelErr |
|---|---|---|
| $l_{1/2}$method | 1.807e-2 | 1.247e-2 |
| IALM | 1.597e-1 | 1.091e-1 |
| Lmafit | 2.336e-2 | 1.559e-2 |

From the numerical results, the $l_{1/2}$ method is significantly better than the IALM and slightly better than the Lmafit method in terms of the relative error of the recovery image to determine the noise increase of the low rank matrix.

**Test 3：** We consider the issue of video background extraction in [3]. We compared four video background extraction issues with LMaFit , IALM and $l_{1/2}$ method.The CPU time results are reported in the last three columns of Table 4. *Resolution* denotes the size of each frame in a video clip, and No. frames denotes the numberof frames tested.

**Table 4：** Video separation issue statistics and CPU usage time

| Video | Resolution | No.frames | IALM | LMaFit | $l_{1/2}$method |
|---|---|---|---|---|---|
| RPCA video | 48*48 | 50 | 5.865 | 0.365 | 2.394 |
| demo | 64*36 | 180 | 5.437 | 1.696 | 3.959 |
| Shop | 192*144 | 100 | 480.407 | 13.4204 | 180.970 |
| escalotor | 160*130 | 180 | 421．807 | 23.210 | 168.327 |

From Table 4, $l_{1/2}$ algorithm requires more CPU time than the Lmafit algorithm, less than the IALM algorithm.In our algorithms $l_{1/2}$ of explicit expressions takes longer to evaluate than $L_1$ needs, so the experiment takes more time. But the time required is only about half that of the IALM algorithm.

## 5. Conclusion

In this paper, we constructed a nonconvex model based on $L_{1/2}$ norm for RPCA problem. We also prove the simple convergence results of the algorithm corresponding to the new model. Numerical experiments verify that our model is more recoverable than the LMafit algorithm and less CPU-cost than the IALM algorithm, but our method takes far more time than the Lmafit, which is also a flaw that cannot be ignored. We should to find ways to accelerate and reduce the time it takes for our approach to be found in future studies.

## 6. Acknowledgements

The work was supported by the National Science Foundation of China (Grant no. 11671205)

## References

[1] Y. Shen, Z.Wen, and Y. Zhang. Augmented lagrangian alternating direction method for matrix separation based on low-rank factorization Optimization Methods and Software ,29(2), 2012,1-25

[2] L. Frank and J. Friedman. A statistical view of some chemometrics regression tools. Technometrics 35(2),1993, 109-135.

[3] J. Fan and R. Li. Variable. selection via nonconcave penalized likelihood and its oracle properties. Journal of the American Statistical Association,94(465),2001, 1348-1360.

[4] J. Friedman. Fast sparse regression and classification. International Journal of Forecasting, 28(3) 2012 ，722-738.

[5] C. Zhang. Nearly unbiased variable selection under minimax concave penalty The Annals of Statistics, 38（2）2010.894-942.

[6] T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. Journal of Machine Learning Research, 11(35), 2010, 1081−1107.

[7] C.Gao, N. Wang, Q. Yu, and Z. Zhang，A feasible nonconvex relaxation approach to feature selection. Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011.

[8] J. Trzasko and A.Manduca. Highly Undersampled magnetic resonance image reconstruction via homotopic minimization.TMI, 28（1）,2009，106-121.

[9] D. Geman and C .YangNonlinear image recovery with half-quadratic regularization. IEEE Trans on Image Processing 4(7), 2002, 932-946.

[10] Zongben Xu, Xiangyu Chang,Fengmin Xu,Hai Zhang$L_{1/2}$ regularization: a thresholding representation theory and a fast solver . IEEE Transactions on Neural Networks and Learning Systems 23(7), 2012，1013-1027.