

# 通过随机排序的交替方向乘子法的矩阵恢复

李吉 赵丽娜\* 侯旭珂

(北京化工大学理学院, 北京 100029)

**摘要:** 为了解决交替方向乘子法(ADMM)在求解广义的鲁棒主成分分析(G-RPCA)模型时结果不收敛的问题, 提出用随机排序的交替方向乘子法(RP-ADMM)来求解这一模型, 并且通过数值模拟和实例验证证明了该算法的有效性。结果表明, 该算法求解 G-RPCA 模型较目前已有的算法速度更快、鲁棒性更高; 在处理同时被稀疏大噪声和稠密小噪声污染的图片时, 能较理想地分离出图像的低秩部分、大噪声部分和小噪声部分。

**关键词:** 广义鲁棒主成分分析; 随机排序的交替方向乘子法(RP-ADMM); 矩阵恢复; 去噪

**中图分类号:** TP391.41 **DOI:** 10.13543/j.bhxbzr.2017.03.021

## 引言

信息爆炸的时代, 每时每刻都会产生海量的信息。为了更好地对海量数据进行挖掘和运用, 需要先对数据进行预处理, 即去除数据中的各种噪声, 保留真实有效的数据。主成分分析(principal component analysis, PCA)<sup>[1]</sup>是一种非常经典的降维方法, 在许多领域中被广泛应用。PCA 常用奇异值分解算法来求解, 能有效去除数据中的稠密小噪声, 实现降维的目的, 但是当需要分离出数据中的稀疏大噪声时, PCA 方法不再适用。鲁棒主成分分析(robust PCA, RPCA)<sup>[1-4]</sup>能有效分离出数据中的稀疏大噪声, 目前一些主流算法, 如迭代阈值算法(iterative thresholding, IT)<sup>[5]</sup>, 加速的近端梯度算法(accelerated proximal gradient, APG)<sup>[6]</sup>和增广的拉格朗日乘子法(augmented Lagrange multiplier, ALM)<sup>[7]</sup>等能对其有效求解, 但是这些算法都只能分离出其中的大噪声, 而无法分离出稠密小噪声。

本文提出用随机排序的交替方向乘子法(randomly permuted ADMM, RP-ADMM)来求解更一般的RPCA模型(G-RPCA), 即在数据中同时存在稀疏大噪声和稠密小噪声的情形, 同时证明了当RP-ADMM被应用到G-RPCA模型中时, 结果是收敛的。

## 1 广义的鲁棒主成分分析及算法

### 1.1 广义的鲁棒主成分分析

广义的鲁棒主成分分析(G-RPCA)是一种矩阵恢复模型, 它能从含有混合噪声的高维数据矩阵中分离出稀疏大噪声和稠密小噪声, 从而得到有价值的低维数据矩阵。G-RPCA 能写成如下形式:

$$\begin{cases} \min_{\mathbf{A}, \mathbf{E}} \|\mathbf{A}\|_* + \lambda \|\mathbf{E}\|_1 + \gamma \|\mathbf{F}\|_F^2 \\ \text{subject to } \mathbf{A} + \mathbf{E} + \mathbf{F} = \mathbf{D} \end{cases} \quad (1)$$

其中,  $\|\cdot\|_F^2$  为  $F$  范数, 表示稠密小噪声;  $\|\cdot\|_*$  为核范数, 表示低秩矩阵;  $\|\cdot\|_1$  为 1 范数, 表示稀疏大噪声。

### 1.2 随机排序的交替方向乘子法

随机排序的交替方向乘子法是一种迭代方法, 由交替方向乘子法(ADMM)<sup>[8]</sup>改进而来。ADMM 适用的凸优化问题如式(2)所示:

$$\begin{cases} \min f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) \\ \text{subject to } \mathbf{A}_1\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2 = \mathbf{b} \end{cases} \quad (2)$$

其中  $f_1(\mathbf{x}_1)$  和  $f_2(\mathbf{x}_2)$  是两个可分离的目标函数,  $\mathbf{A}_1\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2 = \mathbf{b}$  是一个线性约束条件<sup>[9]</sup>。

为了求解式(2)需先写出其增广 Lagrangian 函数形式:

$$L(\mathbf{x}_1, \mathbf{x}_2; \mu) = f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) - \mu(\mathbf{A}_1\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2 - \mathbf{b}) + \frac{\beta}{2} \|\mathbf{A}_1\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2 - \mathbf{b}\|^2 \quad (3)$$

其中  $\mu$  是 Lagrangian 乘子,  $\beta > 0$  为惩罚因子。循环迭代更新原来的变量  $\mathbf{x}_1, \mathbf{x}_2$

收稿日期: 2016-11-21

基金项目: 国家自然科学基金(11301021/11571031)

第一作者: 男, 1991年生, 硕士生

\* 通讯联系人

E-mail: zhaoln@mail.buct.edu.cn

$$\begin{cases} \mathbf{x}_1^{k+1} = \arg \min_{\mathbf{x}_1 \in \chi_1} L(\mathbf{x}_1, \mathbf{x}_2^k; \mu^k) \\ \mathbf{x}_2^{k+1} = \arg \min_{\mathbf{x}_2 \in \chi_2} L(\mathbf{x}_2, \mathbf{x}_1^{k+1}; \mu^k) \\ \mu^{k+1} = \mu^k - \beta(A_1 \mathbf{x}_1^{k+1} + A_2 \mathbf{x}_2^{k+1} - \mathbf{b}) \end{cases}$$

RP-ADMM 算法考虑的是模型(2)中目标函数为 3 个或 3 个以上时的情形,以 3 个为例:

$$\begin{cases} \min f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) + f_3(\mathbf{x}_3) \\ \text{subject to } A_1 \mathbf{x}_1 + A_2 \mathbf{x}_2 + A_3 \mathbf{x}_3 = \mathbf{b} \end{cases} \quad (4)$$

将式(4)写成增广 Lagrangian 函数的形式:

$$L(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3; \mu) = f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) + f_3(\mathbf{x}_3) - \mu(A_1 \mathbf{x}_1 + A_2 \mathbf{x}_2 + A_3 \mathbf{x}_3 - \mathbf{b}) + \frac{\beta}{2} \|A_1 \mathbf{x}_1 + A_2 \mathbf{x}_2 + A_3 \mathbf{x}_3 - \mathbf{b}\|^2 \quad (5)$$

RP-ADMM 和 ADMM 的区别在于,RP-ADMM 算法在每次迭代前都会随机抽取一个排序,例如:  $\sigma = (2, 3, 1)$ ,  $L(\mathbf{x}_{\sigma(1)}, \mathbf{x}_{\sigma(2)}, \mathbf{x}_{\sigma(3)}; \mu) = L(\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_1; \mu)$  表示先固定  $\mathbf{x}_3, \mathbf{x}_1$  求解出  $\mathbf{x}_2$ ,再固定  $\mathbf{x}_2, \mathbf{x}_1$  求解出  $\mathbf{x}_3$ ,最后固定  $\mathbf{x}_2, \mathbf{x}_3$  求解出  $\mathbf{x}_1$ 。

### 1.3 模型推导及算法流程

为了求解 G-RPCA 模型的迭代过程,需要先将式(1)写成增广 Lagrangian 函数的形式:

$$L(\mathbf{A}, \mathbf{E}, \mathbf{F}, \mathbf{Y}, \mu) = \|\mathbf{A}\|_* + \lambda \|\mathbf{E}\|_1 + \frac{\gamma}{2} \|\mathbf{F}\|_F^2 + \langle \mathbf{Y}, \mathbf{D} - \mathbf{A} - \mathbf{E} - \mathbf{F} \rangle + \frac{\mu}{2} \|\mathbf{D} - \mathbf{A} - \mathbf{E} - \mathbf{F}\|_F^2 \quad (6)$$

其中  $\mathbf{F}$  为稠密的小噪声矩阵,  $\|\cdot\|_F$  为  $F$  范数。

根据式(6)得到如下迭代格式的推导过程。

首先固定  $\mathbf{E}_k$  和  $\mathbf{F}_k$ ,更新  $\mathbf{A}_{k+1}$ :

$$\arg \min_{\mathbf{A}_{k+1}} \|\mathbf{A}_{k+1}\|_* + \frac{\mu}{2} \|\mathbf{D} - \mathbf{A}_{k+1} - \mathbf{E}_k - \mathbf{F}_k + \mu_k^{-1} \mathbf{Y}_k\|_F^2 = D_{\mu_k^{-1}}(\mathbf{D} - \mathbf{E}_k - \mathbf{F}_k + \mu_k^{-1} \mathbf{Y}_k) \quad (7)$$

其中,  $D_{\mu_k^{-1}}(\cdot) = US_{\mu_k^{-1}}(\Sigma) V^T$ ; 对  $\mathbf{D} - \mathbf{E}_k - \mathbf{F}_k + \mu_k^{-1} \mathbf{Y}_k$  进行奇异值分解得到矩阵  $\mathbf{U}, \Sigma$  和  $\mathbf{V}, S_{\mu_k^{-1}}(\cdot)$  为收缩算子,

$$S_{\mu_k^{-1}}(x) = \begin{cases} x - \mu_k^{-1}, & x > \mu_k^{-1} \\ x + \mu_k^{-1}, & x < -\mu_k^{-1} \\ 0, & \text{其他} \end{cases}$$

其次固定  $\mathbf{A}_k$  和  $\mathbf{F}_k$ ,更新  $\mathbf{E}_{k+1}$

$$\arg \min_{\mathbf{E}_{k+1}} \lambda \|\mathbf{E}_{k+1}\|_1 + \frac{\mu_k}{2} \|\mathbf{D} - \mathbf{A}_k - \mathbf{E}_{k+1} - \mathbf{F}_k + \mu_k^{-1} \mathbf{Y}_k\|_F^2 = S_{\frac{\lambda}{\mu_k}}(\mathbf{D} - \mathbf{A}_k - \mathbf{F}_k + \mu_k^{-1} \mathbf{Y}_k) \quad (8)$$

最后固定  $\mathbf{A}_k$  和  $\mathbf{E}_k$  更新  $\mathbf{F}_{k+1}$

$$\arg \min_{\mathbf{F}_{k+1}} \frac{\gamma}{2} \|\mathbf{F}_{k+1}\|_F^2 + \frac{\mu}{2} \|\mathbf{D} - \mathbf{A}_k - \mathbf{E}_k - \mathbf{F}_{k+1} + \mu_k^{-1} \mathbf{Y}_k\|_F^2 = \frac{\mu_k}{\mu_k + \gamma} (\mathbf{D} - \mathbf{A}_k - \mathbf{E}_k + \mu_k \mathbf{Y}_k) \quad (9)$$

由式(7) ~ (9) 并结合 RP-ADMM 得到如下 G-RPCA 的算法流程。

1) 输入数据矩阵  $\mathbf{D}$

2) 初始化

令  $\mathbf{Y}_0, \mu_0 > 0, \rho > 0, \gamma = 0.5, k = 0, \mathbf{A}_0 = \mathbf{0}, \mathbf{E}_0 = \mathbf{0}, \mathbf{F}_0 = \mathbf{0}$ ;

3) 产生一个(1,2,3)的随机排序  $\sigma$ ;

4) 遍历  $\sigma$ , 如  $\sigma(i) = 1$ , 则转步骤 5), 如  $\sigma(i) = 2$ , 则转步骤 6), 如果  $\sigma(i) = 3$ , 则转步骤 7);

5) 应用式(7)得到  $\mathbf{A}_{k+1}$ , 令  $\mathbf{A}_k = \mathbf{A}_{k+1}$ ;

6) 应用式(8)得到  $\mathbf{E}_{k+1}$ , 令  $\mathbf{E}_k = \mathbf{E}_{k+1}$ ;

7) 应用式(9)得到  $\mathbf{F}_{k+1}$ , 令  $\mathbf{F}_k = \mathbf{F}_{k+1}$ ;

8)  $\mathbf{Y}_{k+1} = \mathbf{Y}_k + \mu_k(\mathbf{D} - \mathbf{A}_k - \mathbf{E}_k - \mathbf{F}_k)$ ,  $\mu_{k+1} = \rho \mu_k$ ,  $k = k + 1$ ;

9) 检查是否收敛, 如不收敛转步骤 3), 收敛则输出。

10) 输出  $\mathbf{A}_k^*, \mathbf{E}_k^*, \mathbf{F}_k^*$ 。

### 1.4 收敛性证明

证明 RP-ADMM 算法的收敛性。

1) 利用 ADMM 对式(6)进行  $k$  次迭代时有式(10)成立:

$$\begin{cases} -\mathbf{a}_1^T \boldsymbol{\lambda}^k + \mathbf{a}_1^T (\mathbf{a}_1 \mathbf{x}_1^{k+1} + \mathbf{a}_2 \mathbf{x}_2^k + \mathbf{a}_3 \mathbf{x}_3^k - \mathbf{b}) = 0 \\ -\mathbf{a}_2^T \boldsymbol{\lambda}^k + \mathbf{a}_2^T (\mathbf{a}_1 \mathbf{x}_1^{k+1} + \mathbf{a}_2 \mathbf{x}_2^{k+1} + \mathbf{a}_3 \mathbf{x}_3^k - \mathbf{b}) = 0 \\ -\mathbf{a}_3^T \boldsymbol{\lambda}^k + \mathbf{a}_3^T (\mathbf{a}_1 \mathbf{x}_1^{k+1} + \mathbf{a}_2 \mathbf{x}_2^{k+1} + \mathbf{a}_3 \mathbf{x}_3^{k+1} - \mathbf{b}) = 0 \\ (\mathbf{a}_1 \mathbf{x}_1^{k+1} + \mathbf{a}_2 \mathbf{x}_2^{k+1} + \mathbf{a}_3 \mathbf{x}_3^{k+1} - \mathbf{b}) + \boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k = 0 \end{cases} \quad (10)$$

其中  $\boldsymbol{\lambda}$  表示拉格朗日乘子。

设  $\mathbf{y}^k = [\mathbf{x}_1^k; \mathbf{x}_2^k; \mathbf{x}_3^k; (\boldsymbol{\lambda}^k)^T] \in \mathbb{R}^{6 \times 1}$ , 则式(10)改写为

$$\begin{bmatrix} \mathbf{a}_1^T \mathbf{a}_1 & 0 & 0 & 0 \\ \mathbf{a}_2^T \mathbf{a}_1 & \mathbf{a}_2^T \mathbf{a}_2 & 0 & 0 \\ \mathbf{a}_3^T \mathbf{a}_1 & \mathbf{a}_3^T \mathbf{a}_2 & \mathbf{a}_3^T \mathbf{a}_3 & 0 \\ \mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_3 & \mathbf{I}_{3 \times 3} \end{bmatrix} \mathbf{y}^{k+1} = \begin{bmatrix} 0 & -\mathbf{a}_1^T \mathbf{a}_2 & -\mathbf{a}_1^T \mathbf{a}_3 & \mathbf{a}_1^T \\ 0 & 0 & -\mathbf{a}_2^T \mathbf{a}_3 & \mathbf{a}_2^T \\ 0 & 0 & 0 & \mathbf{a}_3^T \\ 0 & 0 & 0 & \mathbf{I}_{3 \times 3} \end{bmatrix} \mathbf{y}^k + \begin{bmatrix} \mathbf{A}^T \mathbf{b} \\ \mathbf{b} \end{bmatrix} \quad (11)$$

定义

$$L = \begin{bmatrix} a_1^T a_1 & 0 & 0 \\ a_2^T a_1 & a_2^T a_2 & 0 \\ a_3^T a_1 & a_3^T a_2 & a_3^T a_3 \end{bmatrix}$$
$$R = \begin{bmatrix} 0 & -a_1^T a_2 & -a_1^T a_3 \\ 0 & 0 & -a_2^T a_3 \\ 0 & 0 & 0 \end{bmatrix}$$

则  $L$  和  $R$  的关系为:  $L - R = A^T A$ , 其中  $A$  为  $Ax = y$  的系数矩阵。

定义

$$\bar{L} = \begin{bmatrix} L & 0 \\ A & I_{3 \times 3} \end{bmatrix}, \bar{R} = \begin{bmatrix} R & A^T \\ 0 & I_{3 \times 3} \end{bmatrix}, \bar{b} = \begin{bmatrix} A^T b \\ b \end{bmatrix},$$

则式 (10) 可写为

$$y^{k+1} = (\bar{L})^{-1} \bar{R} y^k + \bar{L}^{-1} \bar{b}$$

因为  $\rho((\bar{L})^{-1} \bar{R}) > 1^{[10]}$  ( $\rho(\cdot)$  表示谱半径), 即谱半径大于 1, 所以 ADMM 对于式 (6) 并不收敛。

2) 应用 RP-ADMM 求解式 (6), 当迭代到第  $k$  次时其格式变为

$$y^{k+1} = \bar{L}_\sigma^{-1} \bar{R}_\sigma y^k + \bar{L}_\sigma^{-1} \bar{b}$$

其中  $\sigma \in \Gamma = \{\sigma | \text{序列}(1, 2, 3) \text{ 的随机排序}\}$

**引理 1**<sup>[11]</sup> 假设  $A = [A_1, \dots, A_n] \in \mathbb{R}^{N \times N}$  是非奇异的, 对于任意的排序, 定义:

$$M = E_\sigma(\bar{L}_\sigma^{-1} \bar{R}_\sigma) = \frac{1}{n!} \sum_{\sigma \in \Gamma} (\bar{L}_\sigma^{-1} \bar{R}_\sigma)$$

因为  $\Gamma$  中的变量是服从均匀随机变量的, 所以  $E$  是关于  $\Gamma$  的期望。因此  $\rho(M) < 1$ 。

综合 1) 和 2) 的结果, 得出  $M$  的谱半径小于 1, 证明 RP-ADMM 算法是收敛的, 结论得证。

## 2 数值模拟

为测试 G-RPCA 模型分离稀疏大噪声和稠密小噪声的能力, 采用  $n$  阶方阵作测试。低秩矩阵  $A \in \mathbb{R}^{n \times n}$  可由元素服从 i. i. d 标准正态分布的两个独立随机矩阵  $M_L^{n \times r}$  和  $M_R^{r \times n}$  的乘积产生; 稀疏大误差矩阵  $E \in \mathbb{R}^{n \times n}$  的支集元素在区间  $[-500, 500]$  上随机均匀取值; 致密的高斯小噪声矩阵  $N \in \mathbb{R}^{n \times n}$  的元素服从 i. i. d 标准正态分布。由此可得到受稀疏大噪声和稠密小噪声污染的矩阵  $D = A + E + \sigma N$  ( $\sigma = 0.001$ ) ( $F = \sigma N$ )。分别改变矩阵  $D$  的维数 ( $m = 100:100:2000$ )、低秩矩阵  $A$  的秩 ( $\text{rank}(A) = 5:5:60$ ) 和高斯噪声的大小 ( $\sigma = 0.005:0.005:0.06$ ), 得到 G-RPCA、ALM 和 APG 这 3 种算法的性能比较。图 1 为

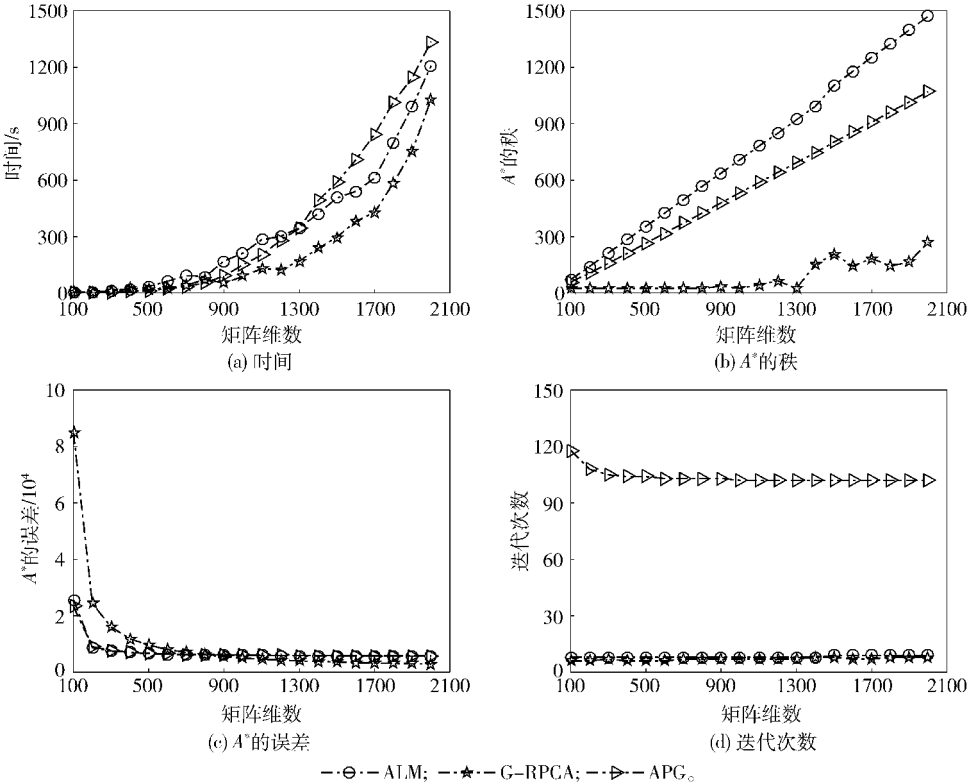


图 1 不同矩阵维数下 3 种算法的性能比较

Fig. 1 Performance comparison of three algorithms with different matrix dimensions

设定高斯噪声 0.05 和低秩矩阵的秩 15,改变矩阵维数的结果;图 2 为设定矩阵维数 1500 阶和高斯噪声 0.05,

改变低秩矩阵的秩的结果;图 3 为设定矩阵维数 1500 阶和低秩矩阵的秩 15,改变高斯噪声的结果。

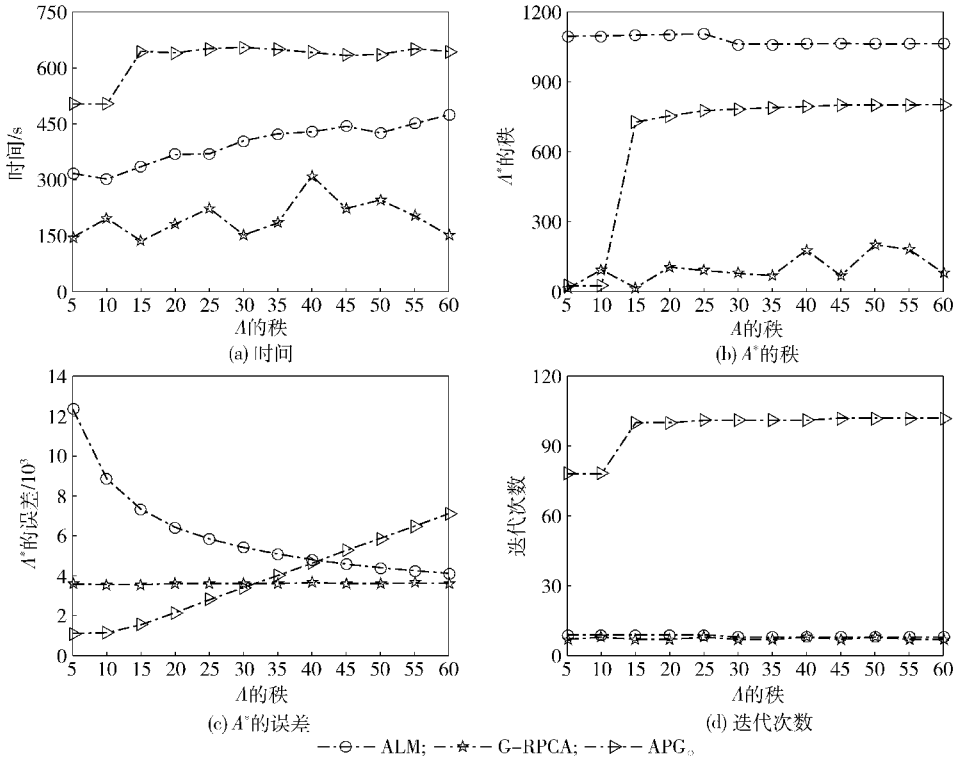


图 2 改变低秩矩阵的秩时 3 种算法的性能比较

Fig. 2 Performance comparison of three algorithms when changing the rank of the lower rank matrix

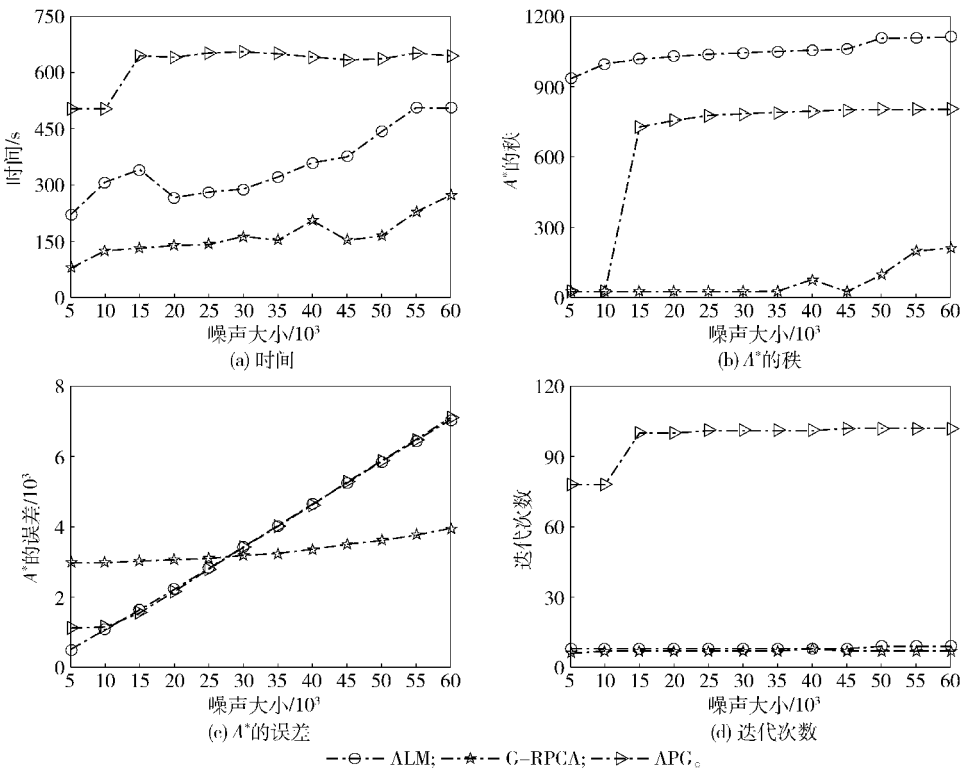


图 3 不同噪声值下 3 种算法的性能比较

Fig. 3 Performance comparison of three algorithms with different noise

从图1中可以看出,随着矩阵维数的增加,低秩部分的秩并没有显著的变化,这说明G-RPCA模型所恢复出的低秩部分鲁棒性很好。在图2和图3中,不管是随着噪声的增大还是矩阵秩的改变,G-RPCA模型所恢复出的矩阵相对于其他算法都更加鲁棒且其迭代次数也更少。

### 3 含有混合噪声的图像分离

任选一幅图像对其添加混合噪声,然后分别应用G-RPCA方法和ALM方法去噪,结果如图4、5所示。

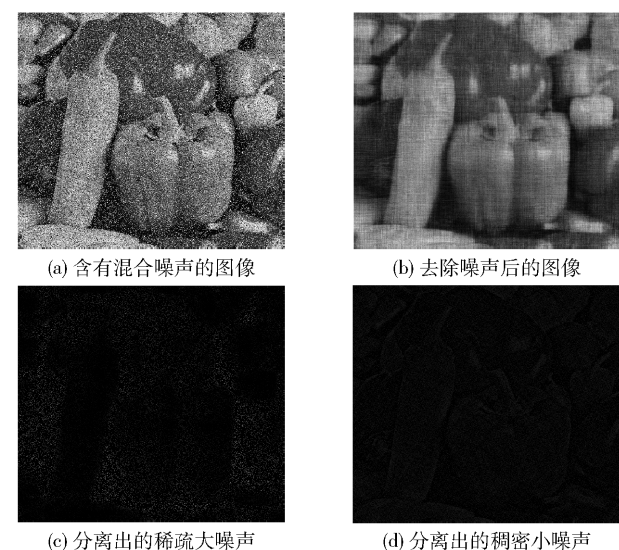


图4 G-RPCA 图像去噪

Fig.4 G-RPCA image denoising

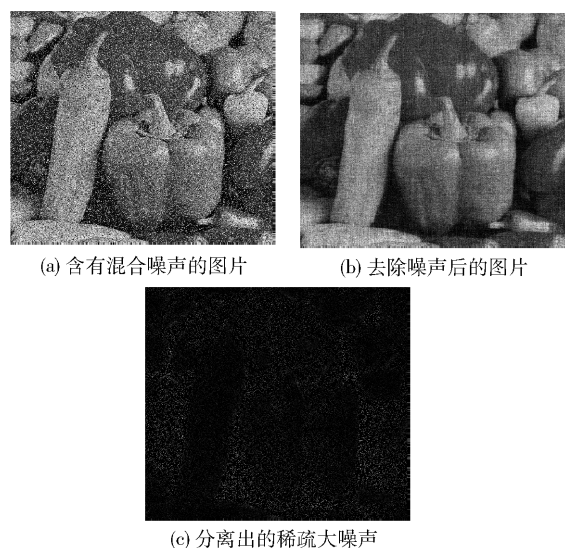


图5 ALM 图像去噪

Fig.5 ALM image denoising

对比图4、5可以明显看出,G-RPCA去除噪声后的图片比ALM更清晰,说明G-RPCA方法能有效分离出稠密的小噪声。

### 4 结束语

本文对于G-RPCA模型基于随机排序的交替方向乘子法提出一种可行的求解方法。通过数值实验可以表明该算法在处理混合噪声问题上,无论从时间成本还是从所恢复出的低秩矩阵秩的鲁棒性上都优于其他两种通用的算法。图像实例也证明了该算法能够分别分离出稀疏大噪声和稠密的小噪声,在去除混合噪声方面具有一定优势。

### 参考文献:

- [1] Wright J, Peng Y G, Ma Y. Robust principal component analysis: exact recovery of corrupted low-rank matrices via convex optimization [C] // Twenty-Third Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2009: 2080-2088.
- [2] Candès E J, Li X D, Ma Y, et al. Robust principal component analysis? [J]. Journal of the ACM, 2011, 58 (3): 11-48.
- [3] Xu H, Caramanis C, Sanghavi S. Robust PCA via outlier pursuit [C] // Twenty-Third Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2010: 2496-2504.
- [4] Candès E J, Tao T. The power of convex relaxation: near-optimal matrix completion [J]. IEEE Transactions on Information Theory, 2010, 56(1): 2053-2080.
- [5] Candès E J, Recht B. Exact matrix completion via convex optimization [J]. Foundations of Computational Mathematics, 2009, 9(2): 717-772.
- [6] Lin Z C, Ganesh A, Wright J, et al. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix [J]. Journal of the Marine Biological Association of the United Kingdom, 2009, 56(3): 707-722.
- [7] Lin Z C, Chen M M, Ma Y. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices [J]. Journal of the Marine Biological Association of the United Kingdom, 2010, 55(3): 607-626.
- [8] Boyd S, Parikh N, Chu E, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers [J]. Foundations and Trends in Machine Learning, 2010, 3(1): 1-122.
- [9] Xie W S, Yang Y F, Zhou B. An ADMM algorithm for second-order TV-based MR image reconstruction [J].



- Numerical Algorithms, 2014, 67(4): 827–843.
- [10] Chen C H, He B S, Ye Y Y, et al. The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent[J]. Mathematical Programming, 2016, 155(1): 57–79.
- [11] Sun D, Toh K C, Yang L. A convergent 3-block semi-proximal alternating direction method of multipliers for conic programming with 4-type of constraints[J]. Society for Industrial and Applied Mathematic, 2015, 25(2): 882–915.

## Matrix recovery by randomly permuted alternating direction method of multipliers (ADMM)

LI Ji   ZHAO LiNa\*   HOU XuKe

(Faculty of Science, Beijing University of Chemical Technology, Beijing 100029, China)

**Abstract:** In order to solve the problem that the alternating direction method of multipliers (ADMM) does not converge when solving the generalized robust principal component analysis (G-RPCA) model, this paper proposes randomly permuted ADMM (RP-ADMM) process to solve this model. The effectiveness of the algorithm was proved by numerical experiments and case verification. Numerical experiments showed that the proposed algorithm is faster and more robust than existing algorithms when solving the G-RPCA model. When dealing with pictures that are simultaneously polluted by sparse large noise and dense small noise, the algorithm can effectively isolate the low rank part of the image, the large noise part and the small noise part.

**Key words:** generalized robust principal component analysis (PCA); randomly permuted alternating direction method of multipliers (RP-ADMM); matrix recovery; denoising

(责任编辑:汪 琴)