International Journal of Machine Learning and Cybernetics (2020) 11:2327–2339
https://doi.org/10.1007/s13042-020-01121-7

ORIGINAL ARTICLE

Accelerated inexact matrix completion algorithm via closed-form q-thresholding $(q = 1/2, 2/3)$ operator

Zhi Wang[1] · Chao Gao[1] · Xiaohu Luo[1] · Ming Tang[1] · Jianjun Wang[2] · Wu Chen[1]

Received: 12 April 2019 / Accepted: 26 March 2020 / Published online: 17 April 2020

Abstract

l_q $(0 < q < 1)$ regularization is a dominating strategy for matrix completion problems. The main goal of nonconvex l_q regularization based algorithm is to find a so-called low-rank solution.Unfortunately, most existing algorithms suffer from full singular value decomposition (SVD), and thus become inefficient for large-scale matrix completion problems. To alleviate this limitation, in this paper we propose an accelerated inexact algorithm to handle such problem. The key idea is to employ the closed-form q-thresholding $(q = 1/2, 2/3)$ operator to approximate the rank of a matrix. The power method and the special "sparse plus low-rank" structure of the matrix iterates are adopted to allow efficient SVD. Besides, we employ Nesterov's accelerated gradient method and continuation technique to further accelerate the convergence speed of our proposed algorithm. A convergence analysis shows that the sequence $\{X_t\}$ generated by our proposed algorithm is bounded and has at least one accumulation point. Extensive experiments have been conducted to study its recovery performance on synthetic data, image recovery and recommendation problems. All results demonstrate that our proposed algorithm is able to achieve comparable recovery performance, while being faster and more efficient than state-of-the-art methods.

Keywords Matrix completion · l_q regularization · q-thresholding operator · Nesterov's rule · Power method

1 Introduction

In various machine learning and data analysis areas, such as collaborative filtering [1, 2], dimensionality reduction [3], subspace clustering [4], multiple labels learning [5, 6], and image processing [7, 8], one needs to consider the following matrix completion problem:

$$\min_{X\in\mathbb{R}^{m\times n}} rank(X) \quad \text{s.t. } \mathscr{P}_\Omega(X) = \mathscr{P}_\Omega(M), \tag{1}$$

where $M \in \mathbb{R}^{m\times n}$ is the incomplete low-rank matrix to be reconstructed, X is the considered low-rank matrix in $\mathbb{R}^{m\times n}$, $rank(X)$ is the rank of X, Ω is the location of the observed entries, and $\mathscr{P}_\Omega$ is the orthogonal projection onto the span of matrices vanishing outside of Ω. The goal of problem (1) is to find the lowest-rank solution X^* of $\mathscr{P}_\Omega(X) = \mathscr{P}_\Omega(M)$, which is minimal to $rank(X)$. More often, we consider the following regularization problem, which can be cast as:

$$\min_{X\in\mathbb{R}^{m\times n}} \frac{1}{2}\|\mathscr{P}_\Omega(X-M)\|_F^2 + \lambda rank(X), \tag{2}$$

where $\|\cdot\|_F$ is the Frobenius norm, λ is a positive regularization parameter.

However, since the nonconvexity and discontinuous nature of $rank(X)$, problems (1) and (2) are NP-hard [9] and cannot be solved in polynomial time. To alleviate this difficulty, a widely used strategy is to adopt the nuclear norm as a convex surrogate of $rank(\cdot)$. Theoretical studies show that the nuclear norm is the tightest convex lower bound of the rank [9]. Candès and Recht [10] have been proven that the missing values can be perfectly recovered by solving nuclear norm regularization minimization problem if incomplete matrix M satisfies certain assumptions, e.g., $|\Omega| \geq \mathscr{O}(N^{1.2} r\log(N))(N = max(m,n), r = rank(M))$. Generally speaking, the nuclear norm regularization minimization

✉ Zhi Wang
chiw@swu.edu.cn

Jianjun Wang
wjj@swu.edu.cn

Wu Chen
chenwu@swu.edu.cn

[1] College of Computer and Information Science, Southwest University, Chongqing 400715, People's Republic of China

[2] College of Artificial Intelligence, Southwest University, Chongqing 400715, People's Republic of China

problem can be treated as a Semidefinite Program (SDP) problem [11]. However, SDP solvers are only suitable for $m \times n$ matrices with $m, n \leq 100$. In order to overcome this drawback, various methods have been proposed to tackle nuclear norm regularization minimization problem. Examples of first-order methods include singular value thresholding (SVT) algorithm [12] and accelerated proximal gradient with linesearch (APGL) algorithm [13]. Although these two sophisticated algorithms can attain promising results with a strong theoretical guarantee, they all involve expensive SVD operations in each iteration and cannot suitable for large-scale matrices. With the aim of alleviating this shortcoming, fixed point continuation with approximate (FPCA) [14] SVD addresses the same problem as APGL while utilizing a fast Monte Carlo algorithm for SVD calculations. Another state-of-the-art algorithm is Soft-impute algorithm [15] which utilizes a special "sparse plus low-rank" structure associated with the SVT to allow efficient SVD in each iteration. Very recently, Soft-impute algorithm has been accelerated by using Nesterov's rule [16].

Although the nuclear norm regularization minimization problem can be efficiently solved and has been accepted as a powerful tool for the matrix completion problems. Fan [17] pointed out that the nuclear norm penalty shrinks all singular values equally, which leads to over-penalize large singular values. In other words, the nuclear norm may make the solution deviate from the original solution. With the aim of making the larger singular values get less penalized, there has been a significant interest in the use of the nonconvex surrogates of $rank(\cdot)$, such as capped-l_1 penalty, log-sum penalty (LSP), truncated nuclear norm (TNN), smoothly clipped absolute deviation (SCAD), minimax concave penalty (MCP). In [18], the learning formulations with capped-l_1 objective functions are solved by means of multi-stage convex relaxation scheme. In [19], the sparse signal recovery problems are solved by a sequence of weighted l_1-minimization problems. In [20], the matrix completion algorithm based on the TNN is employed in achieving a better approximation to the rank of matrix. In [17], the penalized likelihood methods are proposed to deal with nonparametric regression by select variables and estimate coeffcients simultaneously. In [21], a fast, continuous, nearly unbiased and accurate method is proposed for solving high-dimensional linear regression. Empirically, these nonconvex regularization methods achieve better recovery performance than the convex nuclear norm regularization methods.

Another line of nonconvex surrogates of $rank(\cdot)$ is l_q $(0 < q < 1)$ or Schatten-q quasi-norm, which has received significant interest in the area of matrix completion. The key idea is that it allows a less biased and/or lower rank solution to be found than using the nuclear norm. However, the resultant nonconvex l_q regularization problems are much more difficult to solve. A popular used method for optimization the nonconvex l_q regularization problems is l_q-Proximal-Gradient (l_qPG) algorithm [22]. Since the noncontinuous of objective function ($q = 0$), most existing convergence theory that work with convex cases cannot be applied. By employing the Zangwill's global convergence theory, a non-trivial convergence result of l_qPG is obtained. However, each l_qPG iteration involves computing the approximate solutions [23–26] and SVD steps. Thus, this method is not very accurate and may becomes slow when addressing large-scale matrices. To improve the accuracies and speed of l_qPG algorithm, some improved methods based on Schatten 1/2 quasi-norm [27] and Schatten 2/3 quasi-norm [28] have recently been proposed. Peng et al. [29] proposed a fixed point iterative scheme with the singular value half thresholding operator. The convergence analysis of this method reveals that it is faster and more efficient than l_qPG algorithm. Most recently, by observing that the Schatten 2/3 quasi-norm regularization model has shown better performance than Schatten 1/2 quasi-norm minimization as it requires fewer measurements, Wang et al. [30] build a $L_{2/3}$-PA algorithm for matrix completion. Although, the closed-form thresholding formulas of Schatten 1/2 quasi-norm and Schatten 2/3 quasi-norm regularization problems are obtained, a computationally expensive SVD still be required in each iteration.

With the aim of improving the drawbacks of above, this paper proposes a novel faster and more accurate algorithm for matrix completion. The proposed method extends HFPA and $L_{2/3}$-PA algorithms by using inexact proximal operator and Nesterov's accelerated gradient method. More precisely, our algorithm is based on HFPA and $L_{2/3}$-PA algorithms, but achieves faster convergence rate and better recovery performance than HFPA and $L_{2/3}$-PA algorithms do. Besides, our proposed algorithm is simple and easy to use for large-scale matrix completion problems. The main contributions of our paper include the following:

1. We first propose an efficient and fast algorithm to solve matrix completion problem, which is based on an inexact proximal operator and the Nesterov's accelerated gradient method;
2. In addition, we study the convergence of our proposed method, and show that our proposed algorithm is guaranteed to converge to a critical point of the nonconvex objective. Furthermore, our proposed algorithm is simple and easy to use;
3. Finally, we apply the proposed algorithm to synthetic data, image recovery and large-scale recommendation problems, and achieve faster convergence rate and better recovery performance than most of state-of-the-art algorithms, which demonstrate that our proposed algorithm has the great potentials in matrix completion applications.

The remainder of this paper can be organized as follows. Section 2 describes the related works; Sect. 3 introduces the proposed algorithm; Sect. 4 reports and analyses the experimental results in both speed and quality; Finally, Sect. 5 gives the conclusion of this paper.

Notation We summarize the notations that will be used in this paper. For $X \in \mathbb{R}^{m\times n}$, $\sigma(X) = (\sigma_1(X), \sigma_2(X), \dots, \sigma_r(X))^T$ denotes the vector of singular value of X arranged in nonincreasing order; $Diag(\sigma(X))$ denotes a diagonal matrix whose diagonal vector is $\sigma(X)$. The Frobenius norm and Schatten q quasi-norm of X are defined as $\|X\|_F = (\sum_{i,j} X_{ij}^2)^{1/2} = (\sum_{i=1}^r \sigma_i(X)^2)^{1/2}$ and $\|X\|_q = (\sum_{i=1}^r \sigma_i(X)^q)^{1/q}$, respectively. For $X, Y \in \mathbb{R}^{m\times n}$, $< X, Y >= tr(Y^T X)$ denotes their inner product.

2 Related work

2.1 Proximal algorithms

The APGL algorithm pioneered uses proximal operator [31] to solve matrix completion problems. It considers the following low-rank matrix completion problem:

$$\min_{X\in\mathbb{R}^{m\times n}} F(X) \equiv f(X) + g(X), \tag{3}$$

where f, g are convex, and f is smooth but g is possibly nonsmooth. Besides, f is differentiable with Lipschitz continuous gradient L, i.e., $\| \nabla f(X_1) - \nabla f(X_2) \le L\|X_1 - X_2\|$. At the tth iteration, the APGL algorithm generates X_{t+1} as

$$X_{t+1} = \text{prox}_{\mu g}(Y) = \arg\min_{X\in\mathbb{R}^{m\times n}} \frac{1}{2}\|X - Y\|_F^2 + \mu g(X), \tag{4}$$

where $\mu < 1/L$, and $\text{prox}_{\mu g}(\cdot)$ denotes the proximal operator. By using the Nesterov's accelerated gradient method, Y can be obtained as follows:

$$G_t = (1 + \theta_t)X_t - \theta_t X_{t-1}, \tag{5}$$

$$Y = G_t - \mu\nabla f(G_t), \tag{6}$$

$$\theta_{t+1} = \frac{1 + \sqrt{1 + 4(\theta_t)^2}}{2}. \tag{7}$$

The convergence analysis of APGL algorithm reveals that this algorithm converges at a rate of $\mathcal{O}(1/T^2)$, where T is the iteration number. This is also known to be the best possible rate for the problem (3) [32].

Since the considerable attention of nonconvex and nonsmooth problems in machine learning, it is natural to ask that if the accelerated proximal algorithm can be extended to problems where f and/or g may be nonconvex. The answer is positive. In [33], Li and Lin have extended accelerated proximal algorithm to solve nonconvex and nonsmooth problems. The key idea is adopting a monitor that satisfies the sufficient descent property. Subsequently, they proposed a nonmonotone accelerated proximal gradient algorithm (nmAPG). Although the nmAPG algorithm is much faster than most of state-of-the-art, its convergence rate is still unknown.

2.2 Existing completion methods with l_q regularization

Using the Schatten-q quasi-norm minimization instead of the $rank(\cdot)$ minimization is one of the most successful approaches in the area of matrix completion. The state-of-the-art matrix completion algorithms based on l_q regularization are l_qPG [22], HFPA [29], and $L_{2/3}$-PA [30].

(1) l_qPG: Considering the l_q penalized model:

$$\min_{X\in\mathbb{R}^{m\times n}} F(X) = \frac{1}{2}\|\mathscr{P}_\Omega(X - M)\|_F^2 + \lambda\|X\|_q^q, \tag{8}$$

where $q \in [0, 1]$. It should be note that $q = 0$ is equivalent to the problem (2) and $q = 1$ is equivalent to the nuclear norm penalized problem. It requires computing the following non-trivial optimization problem:

$$\min_x \phi(x) = \frac{1}{2}(z - x)^2 + \lambda|x|^q, \tag{9}$$

where z is a constant and $q \in (0, 1)$. This has the following non-trivial solutions.

Theorem 1 [22] *Let $q \in (0, 1)$, $b = [2\lambda(1 - q)]^{1/(2-q)}$, and $c = b + \lambda q b^{q-1}$. Then the solutions $x^* = \tau_\lambda(z)$ to the problem (9) are:*

$$\tau_\lambda(z) = \begin{cases} 0, & \text{if } |z| < c; \\ \{0, sgn(z)b\}, & \text{if } |z| = c; \\ sgn(z)\hat{x}, & \text{if } |z| > c. \end{cases} \tag{10}$$

where for $|z| > c$, $\hat{x} \in (b, |z|)$ solves:

$$x + \lambda q x^{q-1} = |z| \quad where \quad x > 0. \tag{11}$$

When $|z| > c$ there are two solution to (11) and $\hat{x}$ is the larger one which can be computed from the iteration:

$$x_{t+1} = \rho(x_t) \quad \text{where} \quad \rho(x) = |z| - \lambda q x^{q-1} \tag{12}$$

with the initial condition $x_{(0)} \in [b, |z|]$.

Using the above theorem, Marjanovic et al. have proposed a MM-based algorithm namely l_qPG for iteratively reducing objective function F. Moreover, experiments are performed on matrix completion problems show that l_qPG

algorithm is superior to the state-of-the-art. Although the $l_q PG$ algorithm further accelerated using warm-starting and Nesterov's method, it is still suffer from heavy SVD which takes $\mathscr{O}(mn^2)$ time.

(2) *HFPA*: Recently, Peng et al. proposed HFPA to improve FPCA and $l_q PG$ algorithms by using the Schatten 1/2 quasi-norm. The key idea is that when $q = 1/2$ the solution of problem (9) has an analytical expression [27], that is:

$$\tau_\lambda(z) = \begin{cases} 0, & \text{if } |z| < \frac{\sqrt[3]{54}}{4}\lambda^{2/3}; \\ \{0, \frac{2}{3}z(\cos(\frac{2\pi}{3} - \frac{2}{3}\phi_\lambda(z)))\}, & \text{if } |z| = \frac{\sqrt[3]{54}}{4}\lambda^{2/3}; \\ \frac{2}{3}z\left(\cos\left(\frac{2\pi}{3} - \frac{2}{3}\phi_\lambda(z)\right)\right), & \text{if } |z| > \frac{\sqrt[3]{54}}{4}\lambda^{2/3}. \end{cases} \tag{13}$$

with $\phi_\lambda(z) = \arccos(\frac{\lambda}{8}(\frac{z}{3})^{-3/2})$. The HFPA algorithm consists of two loops. For inner iterations, a special gradient descent method and the matrix half thersholding operator are employed to obtain the approximation solution. In the outer loops, the continuation technique is used to accelerate the convergence speed of HFPA. Besides, the authors further provide the global necessary optimality condition for the $L_{1/2}$ regularization problem and the exact mathematical convergence analysis of the HFPA. Actually, HFPA is a fixed point method, but used for nonconvex optimization problem. Although, a fast Monte Carlo algorithm is adopted in HFPA to approximate SVD procedure, it is still time-consuming on large matrices.

Algorithm 1 Half Norm Fixed Point Algorithm(HFPA) [29]

```
Input: Given the observed data matrix $Y$; Set the parameters $\mu_0 > 0$, $\bar{\lambda} > 0$ and $\eta \in (0,1)$.
Output: rank-$r$ solution $\widehat{X}$
 1: for $k = 1 : maxiter$ do
 2:     $\lambda = \lambda_k$
 3:     while NOT converged do
 4:         Compute $B = X + \mu_0(\mathscr{P}_\Omega(Y) - \mathscr{P}_\Omega(X))$, and its SVD, say $B = UDiag(\sigma)V^T$
 5:         Compute $X = UDiag(H_{\lambda\mu_0}(\sigma))V^T$
 6:     end while
 7:     Output $\sigma_k, r_k = rank(X_k)$ and Update $X_k = X$
 8:     Update $\lambda_{k+1} = \max\{\bar{\lambda}, \min\{\eta\lambda_k, \frac{\sqrt{96}}{9\mu_0}([\sigma(X_k)]_{r_k})^{3/2}\}\}$
 9:     if $\lambda_{k+1} = \bar{\lambda}$, return;
10: end for
11: return $\widehat{X}$
```

(3) $L_{2/3}$-PA: Another state-of-the-art algorithm, $L_{2/3}$-PA, combines a gradient descent method in inner iterations that approximation the solutions by 2/3-thresholding operator with continuation technique that accelerates the convergence speed. The main idea in $L_{2/3}$-PA is to employ the following analytical expression of solutions of problem (9) when $q = 2/3$:

$$\tau_\lambda(z) = \begin{cases} sgn(z)\left(\frac{(|\varphi_\lambda(z)| + \sqrt{\frac{2|z|}{|\varphi_\lambda(z)|} - |\varphi_\lambda(z)|^2})}{2}\right)^3, & \text{if } |z| > \frac{\sqrt[4]{48}}{3}\lambda^{3/4}; \\ 0, & \text{otherwise}; \end{cases} \tag{14}$$

where $\varphi_\lambda(z) = (2/\sqrt{3})\lambda^{1/4}(\cosh(\phi_\lambda(z)/3))^{1/2}$, with $\phi_\lambda(z) = \operatorname{arccosh}(27z^2\lambda^{-3/2}/16)$. Based on the empirical studies, we found that $L_{2/3}$-PA is not very accurate though it improves the recovery performance of HFPA. Moreover, it is not efficient enough as still involve time-consuming SVD steps and cannot suitable for large-scale matrices completion.

2.3 Existing completion methods with inexact proximal operator

In machine learning research, the proximal gradient methods are popular for solving various optimization problems with non-smooth regularization. However, it requires the full SVD to solve the proximal operator, which may be time-consuming. Thus, inexact proximal gradient methods are extremely important.

Based on this line, Yao et al. [34] use the power method to approximate the SVD scheme, and propose Accelerated and Inexact Soft-Impute (AIS-Impute) algorithm for matrix completion. The convergence analysis of the AIS-Impute algorithm illustrates that it still converges at a rate of $\mathscr{O}(1/T^2)$. For the nonconvex problems, a Fast NonConvex Low-rank (FaNCL) [35] algorithm is proposed for matrix completion and robust principal component analysis (RPCA). To improve the convergence speed, the inexact proximal gradient method is also employed in the procedure of FaNCL. Besides, Gu et al. [36] have proposed nonmontone accelerated inexact proximal gradient method (nmAIPG) which extends the nmAPG from exact case to inexact case. They also point out that the inexact version of nmAPG shares the same convergence rate as the nmAPG. Observing that the nmAIPG may requires two proximal steps in each iteration, and can be inefficient for solving large-scale matrices. With the aim of alleviating this shortcoming, the noconvex inexact APG (niAPG) algorithm has been proposed in [37] which requires only one inexact proximal step in each iteration. Thus, the niAPG algorithm is much faster, while achieves the comparable performance as the state-of-the-art.

3 Proposed method

In this section, we introduce our proposed algorithm and discuss some of its basic properties.

3.1 Motivation

In this section, we illustrate that how the proximal gradient algorithm can be solving the following problems:

$$\min_{X \in \mathbb{R}^{m\times n}} F(X) = f(X) + \lambda\|X\|_q^q, \tag{15}$$

where $f(X) = \frac{1}{2}\|\mathscr{P}_\Omega(X-M)\|_F^2$ and $q \in \{\frac{1}{2}, \frac{2}{3}\}$.

First, we give the following definition.

Definition 1 (*q-thresholding operator*) Suppose $x = (x_1, x_2, \dots, x_n)^T$, for any $\lambda > 0$, the q-thresholding operator $\mathscr{T}_\lambda(\cdot)$ is defined as

$$\mathscr{T}_\lambda(x) = (\tau_\lambda(x_1), \tau_\lambda(x_2), \dots, \tau_\lambda(x_n))^T. \tag{16}$$

Now, we consider the following quadratic approximation model of the objective function (15) at Y:

$$\begin{aligned} Q_{\lambda,\mu}(X,Y) = &\frac{1}{2}\|\mathscr{P}_\Omega(Y-M)\|_F^2 + < X-Y, \mathscr{P}_\Omega(Y-M) > \\ &+ \frac{1}{2\mu}\|X-Y\|_F^2 + \lambda\|X\|_q^q, \end{aligned} \tag{17}$$

where $\mu \in (0, 1/L)$. It should be note that $Q_{\lambda,\mu}(X,X) = F(X)$ and $Q_{\lambda,\mu}(X,Y) \ge F(X)$ for any $X, Y \in \mathbb{R}^{m\times n}$. By using simple algebra, Eq. (17) can be recasted as:

$$\begin{aligned} Q_{\lambda,\mu}(X,Y) = &\frac{1}{2\mu}\|X-(Y-\mu\mathscr{P}_\Omega(Y-M))\|_F^2 + \lambda\|X\|_q^q \\ &+ \frac{1}{2}\|\mathscr{P}_\Omega(Y-M)\|_F^2 - \frac{\mu}{2}\|\mathscr{P}_\Omega(Y-M)\|_F^2. \end{aligned} \tag{18}$$

Ignoring the constant terms in (18) and let $Y = X_{t-1}$, the minimizer X_t of $Q_{\lambda,\mu}(X,Y)$ can be obtained by

$$X_t = arg\min_{X\in\mathbb{R}^{m\times n}} \left\{ \frac{1}{2\mu}\|X-(X_{t-1}-\mu\mathscr{P}_\Omega(X_{t-1}-M))\|_F^2 + \lambda\|X\|_q^q \right\}. \tag{19}$$

Thus, the above l_q regularization problem can be solved by the proximal operator as shown in the following lemma.

Lemma 2 [29] *Let* $G = X_{t-1} - \mu\mathscr{P}_\Omega(X_{t-1}-M)$ *and the SVD of* G *is* $U\Sigma V^T$. *Then*

$$X_t = prox_{\lambda\mu,q}(G) = \operatorname{argmin}_{X\in\mathbb{R}^{m\times n}} \frac{1}{2}\|X-G\|_F^2 + \lambda\|X\|_q^q, \tag{20}$$

where $q \in \{\frac{1}{2}, \frac{2}{3}\}$. *Specifically,* $prox_{\lambda\mu,q}(G) = UDiag(\mathscr{T}_{\lambda\mu}(\sigma(G)))V^T$.

3.2 Method for computing inexact proximal operator

As shown in Lemma 2, solving the proximal operator $prox_{\lambda\mu,q}(\cdot)$ only needs the singular values/vectors which are greater than $\frac{\sqrt[3]{54}}{4}(\lambda\mu)^{2/3}$ or $\frac{\sqrt[4]{48}}{3}(\lambda\mu)^{3/4}$. It means that the full SVD in Lemma 2 is time-consuming and unnecessary. In order to overcome this drawback, it is natural to ask that if there is a faster and more effective way to solve this problem.

Fortunately, there are some researcher focus on this problem [35, 38]. They suggest to apply SVD on small matrix instead of the original large matrix, thus the complexity could be dramatically down. The main idea of their methods is to extract a small core matrix by finding orthonormal based with the unitary invariant property. For problem (20), we can obtain similar result.

Proposition 1 *Suppose* G *has* $\hat{k}$ *singular values larger than* $\frac{\sqrt[3]{54}}{4}(\lambda\mu)^{2/3}$ *or* $\frac{\sqrt[4]{48}}{3}(\lambda\mu)^{3/4}$, *and* $G = QQ^TG$, *where* $Q \in \mathbb{R}^{m\times k}, k > \hat{k}$, *is orthogonal. Then*

(i) *range(G)* $\subseteq$ *range(Q);*
(ii) $prox_{\lambda\mu,q}(G) = Qprox_{\lambda\mu,q}(Q^TG)$.

Proof

(i) Since $G = Q(Q^TG)$, we can obtained the first part of Proposition 1 obviously.
(ii) Consider an arbitrary matrix $X \in \mathbb{R}^{m\times n}$ admits the SVD as $X = U\Sigma V^T$, then $QX = (QU)\Sigma V^T$. Since Q is orthogonal and using the unitary invariant norm property, we can get $\|X\|_q^q = \|QX\|_q^q$. Therefore,

$$\begin{aligned} Qprox_{\lambda\mu,q}(Q^TG) &= Q\operatorname{argmin}_{Z\in\mathbb{R}^{m\times n}} \frac{1}{2}\|Z-Q^TG\|_F^2 + \lambda\|Z\|_q^q \\ &= \operatorname{argmin}_{X\in\mathbb{R}^{m\times n}} \frac{1}{2}\|QZ-QQ^TG\|_F^2 + \lambda\|Q^TX\|_q^q, \end{aligned} \tag{21}$$

the second equality follows from $X = QZ$.

Since $\|Q^TX\|_q^q = \|Z\|_q^q = \|QZ\|_q^q = \|X\|_q^q$ and $QQ^T = I$, we get

$$\begin{aligned} Qprox_{\lambda\mu,q}(Q^TG) &= \operatorname{argmin}_{X\in\mathbb{R}^{m\times n}} \frac{1}{2}\|X-G\|_F^2 + \lambda\|X\|_q^q \\ &= prox_{\lambda\mu,q}(G), \end{aligned} \tag{22}$$

which is the second part of Proposition 1. We complete the proof. □

In the traditional way, we need full SVD on the original large matrix and its complexity is $\mathscr{O}(mn^2)$. From Proposition 1, we can easily find that the complexity can be dramatically down as performing SVD on a smaller matrix. Specifically, we perform SVD on matrix Q^TG and its complexity becomes $\mathscr{O}(mk^2)$. Thus, when $k \ll n$, the computation speed can be significantly improved.

It should be note that the proof process of Proposition 1 partially follows but is different from [38]. Actually, we deal with the l_q regularization problem instead of nuclear norm regularization problem.

Next, we will address that how to determine the orthogonal matrix Q. Indeed, orthogonal matrix Q can be approximated by using power method [39], which is wildly used to approximate the SVD in nuclear norm and nonconvex regularization minimization problem [35, 40]. More precisely, we can use Algorithm 2 to obtain the orthogonal matrix Q.

Algorithm 2 PowerMethod(G, R)

```
Input: $G \in \mathbb{R}^{m\times n}$,$R \in \mathbb{R}^{n\times k}$ and $J = 3$.
Output: $Q_J$
1: $Q_1 = QR(GR)$;
2: for $t = 1$ to $J - 1$ do
3:     $Y_{t+1} = G(G^T Q_t)$;
4:     $Q_{t+1} = QR(Y_{t+1})$
5: end for
6: return $Q_J$
```

Similar to [35, 40], we fix the number of iterations to 3 for PowerMethod. The matrix R is used for warm-start which is particularly useful for proximal algorithm. As pointed out by [35], the PROPACK algorithm [41] can also be used to obtain Q and the complexity is the same as PowerMethod. Empirically, PROPACK is much less efficient than PowerMethod. Besides, a fast Monte Carlo algorithm [14, 29, 30] is used for computing an approximate SVD. Although this method greatly reduces the computational effort, it has some tunable parameters that need to be set, which makes it not easy to use.

3.3 Inexact proximal operator

Based on the PowerMethod algorithm, we can compute $prox_{\lambda\mu,q}(\cdot)$ more efficiently. Specifically, as in Algorithm 3, we first adopt the PowerMethod algorithm to obtain orthogonal matrix Q. Then, we perform SVD on $Q^T G$ which is much smaller than the original matrix G as $k \ll n$, and the complexity is reduced from $\mathscr{O}(mn^2)$ to $\mathscr{O}(mk^2)$. The next steps are obtain the approximate $prox_{\lambda\mu,q}(\cdot)$ by solving problem (9), where q is fixed to 1/2 or 2/3.

According to Algorithm 3, the $prox_{\lambda\mu,q}(\cdot)$ step will be inexact. Thus, we should monitor the progress of F to make sure the proposed algorithm converges to a critical point. Motivated by [42], Yao et al. [35, 37] suggest employing the following condition to control the inexactness, that is,

$$F(X_{t+1}) \leq F(X_t) - \frac{\delta}{2}\|X_{t+1} - X_t\|_F^2, \tag{23}$$

where $\delta > 0$ is a constant. Obviously, this condition makes the objective function F always decreased. However, in real application, this monotonically decreasing may makes the algorithm fall into narrow curved valley. A nonmontone condition is also used. Specifically, we accept X_{t+1}, if X_{t+1} makes the value of F smaller than the maximum over previous m ($m > 1$) iterations, that is,

$$F(X_{t+1}) \leq \max_{t=\max(1,k-m),\dots,k} F(X_t) - \frac{\delta}{2}\|X_{t+1} - X_t\|_F^2, \tag{24}$$

Based on the analysis of above, we propose Algorithm 4 to obtain the approximation solution at iteration t.

Algorithm 3 l_q Approximate singular value thresholding: l_qArroxSVT(G, R, λ, μ)

```
Input: $Z \in \mathbb{R}^{m\times n}$,$R \in \mathbb{R}^{n\times k}$, $\lambda > 0$, and $\mu \in (0, 1/L)$.
Output: $\hat{X}$, $V$
1: $Q$ = PowerMethod(G, R);
2: $[U, \Sigma, V] = \text{SVD}(Q^T G)$;
3: for $t = 1, 2, \dots$ do
4:     if $q = \frac{1}{2}$, obtain $\tau_{\lambda\mu}(\Sigma_{tt})$ by 13, or;
5:     if $q = \frac{2}{3}$, obtain $\tau_{\lambda\mu}(\Sigma_{tt})$ by 14;
6: end for
7: $\hat{X} = QUDiag(\mathscr{T}_{\lambda\mu}(\sigma(G)))V^T$
8: return $\hat{X}$ and $V$.
```

Algorithm 4 l_q Inexact proximal step: l_qInexactPS(X, R)

```
Input: $X \in \mathbb{R}^{m\times n}$, and $R \in \mathbb{R}^{m\times k}$ for warm-start;
Output: $\tilde{X}$, $\tilde{Z}$, $\tilde{V}$
1: $G = X - \mu \bigtriangledown f(X)$;
2: $[\tilde{Z}, \tilde{V}] = l_q$ArropxSVT(G, R, $\lambda$, $\mu$);
3: if 23 is satisfied then
4:     $\tilde{X} = \tilde{Z}$;
5: else
6:     $\tilde{X} = X$;
7: end if
8: return $\tilde{X}$, $\tilde{Z}$, and $\tilde{V}$.
```

3.4 The proposed algorithm

In this section, we will introduce our proposed l_q inexact APG (l_qiAPG) algorithm and discuss three techniques to accelerate the convergence of our proposed algorithm.

First, we use Nesterov's accelerated gradient method. In the area of convex optimization, the Nesterov's accelerated gradient method is a widely used technique for speed up the convergence of most machine learning algorithms, e.g., proximal algorithm. Recently, this method has also been employed to accelerate the convergence of nonconvex optimization. nmAPG [33], FaNCL-acc [35] and niAPG [37] are the state-of-the-art algorithms. Thus, similar to nmAPG, we will use the following steps in l_qiAPG algorithm,

$$\theta_{t+1} = \frac{1 + \sqrt{1 + 4\theta_t^2}}{2}, \tag{25}$$

$$Y_{t+1} = X_t + \left(\frac{\theta_t}{\theta_{t+1}}\right)(Z_t - X_t) + \left(\frac{\theta_t - 1}{\theta_{t+1}}\right)(X_t - X_{t-1}). \tag{26}$$

Due to the (25) and (26) strategies, extensive experiments have shown that the convergence rate of l_qiAPG algorithm is significantly improved.

Since the regularization parameter λ plays an important role in our proposed algorithm, the tuning of λ becomes a subtle issue. Fortunately, this problem has been already considered in [29, 30], and a most reliable choice of the optimal regularization parameters of (15) are

$$\lambda^* = \begin{cases} \frac{\sqrt{96}}{9\mu}(\sigma_{r_t}(X_{t+1}))^{3/2}, & \text{if } q = \frac{1}{2}; \\ \frac{\sqrt[3]{108}}{4\mu}(\sigma_{r_t}(X_{t+1}))^{4/3}, & \text{if } q = \frac{2}{3}. \end{cases} \tag{27}$$

Besides, the continuation technique is used in our proposed algorithm. As shown in [14, 30, 35], continuation technique is a commonly used method to improve the convergence speed of machine learning algorithms. The key idea of continuation technique is to choose a decreasing sequence $\lambda_t : \lambda_1 > \lambda_2 > \ldots > \bar{\lambda} > 0$, then at tth iteration, use $\lambda = \lambda_t$. Therefore, based on continuation technique, we suggest use the following regularization parameter at the $(t+1)$th iteration, that is,

$$\lambda_{t+1} = \begin{cases} \max\left\{\bar{\lambda}, \min\left\{\eta\lambda_t, \frac{\sqrt{96}}{9\mu}(\sigma_{r_{t+1}}(X_{t+1}))^{3/2}\right\}\right\}, & \text{if } q = \frac{1}{2}; \\ \max\{\bar{\lambda}, \min\{\eta\lambda_t, \frac{\sqrt[3]{108}}{4\mu}(\sigma_{r_{t+1}}(X_{t+1}))^{4/3}\}\}, & \text{if } q = \frac{2}{3}. \end{cases} \tag{28}$$

where $\eta \in (0, 1)$ is a constant, r_{t+1} is the rank of X_{t+1}, and $\bar{\lambda}$ is a sufficiently small but positive real number, e.g., 10^{-4}.

Now, we outline our proposed algorithm as follows,

Algorithm 5 l_q Inexact APG(l_qiAPG) algorithm

```
Input: Given the observed data matrix $Y$; choose $\mu \in (0, 1/L)$, $\bar{\lambda} > 0$, $\delta > 0$ and $\eta \in (0,1)$;
Output: rank-$r$ solution $X$
 1: initialize $V_0, V_1 \in \mathbb{R}^n$ as random Gaussian matrices, $X_0 = X_1 = 0$, $Y_1 = X_0$, $\lambda_0 = \eta\|\mathscr{P}_\Omega(Y)\|_2$ and
    $\theta_0 = 1$;
 2: for $t = 1, 2, \ldots, T$ do
 3:     $R_t = QR([V_t, V_{t-1}])$;
 4:     $[X_t, Z_t] = l_q\text{InexactPS}(Y_t, R_t)$;
 5:     $\theta_t = \frac{1+\sqrt{1+4\theta_{t-1}^2}}{2}$;
 6:     $Y_{t+1} = X_t + \left(\frac{\theta_{t-1}}{\theta_t}\right)(Z_t - X_t) + \left(\frac{\theta_{t-1}-1}{\theta_t}\right)(X_t - X_{t-1})$;
 7:     Update $\lambda_t$ by (28);
 8:     if $\lambda_t = \bar{\lambda}$ then
 9:         break;
10:     end if
11: end for
12: return $X$.
```

The third technique to accelerate the convergence rate of our proposed algorithm is to use the "sparse plus low-rank" structure [15]. Step 4 in l_qiAPG algorithm performs inexact proximal step. Obviously, for any $t > 1$, there are two cases will be happen. When X_t is equal to Z_t, $Y_{t+1} = X_t + \left(\frac{\theta_t - 1}{\theta_{t+1}}\right)(X_t - X_{t-1})$. Therefore, step 1 of Algorithm 4 has

$$\begin{aligned} G = Y_{t+1} - \mu\mathscr{P}_\Omega(Y_{t+1} - M) = \left(1 + \frac{\theta_t - 1}{\theta_{t+1}}\right)X_t \\ - \left(\frac{\theta_t - 1}{\theta_{t+1}}\right)X_{t-1} - \mu\mathscr{P}_\Omega(Y_{t+1} - M). \end{aligned} \tag{29}$$

The first two terms are low-rank matrices, while the last term is a sparse matrix. This kind of "sparse plus low-rank" structure can speed up matrix multiplications. More precisely, for any $V \in \mathbb{R}^{m\times k}$, GV can be obtained as

$$\begin{aligned} GV = (1 + \alpha_t)U_t\Sigma_t(V_t^T V) - \alpha_t U_{t-1}\Sigma_{t-1}(V_{t-1}^T V) \\ + \mu\mathscr{P}_\Omega(M - Y_t)V, \end{aligned} \tag{30}$$

where $\alpha_t = \frac{\theta_t - 1}{\theta_{t+1}}$. Similarly, for any $U \in \mathbb{R}^{m\times k}$, $U^T G$ can be obtained as

$$\begin{aligned} U^T G = (1 + \alpha_t)(U^T U_t)\Sigma_t V_t^T - \alpha_t(U^T U_{t-1})\Sigma_{t-1}V_{t-1}^T \\ + \mu U^T\mathscr{P}_\Omega(M - Y_t). \end{aligned} \tag{31}$$

When $X_t = X_{t-1}$, Y_{t+1} becomes $X_t + \left(\frac{\theta_t}{\theta_{t+1}}\right)(Z_t - X_t)$. Thus, step 1 of Algorithm 4 has

$$\begin{aligned} G = Y_{t+1} - \mu\mathscr{P}_\Omega(Y_{t+1} - M) = (1 + \alpha_t)X_t \\ - \alpha_t Z_t - \mu\mathscr{P}_\Omega(Y_{t+1} - M), \end{aligned} \tag{32}$$

where $\alpha_t = \frac{\theta_t}{\theta_{t+1}}$. The same as above, we have

$$\begin{aligned} GV = (1 + \alpha_t)U_t\Sigma_t(V_t^T V) - \alpha_t U_{Z_t}\Sigma_{Z_t}(V_{Z_t}^T V) \\ + \mu\mathscr{P}_\Omega(M - Y_t)V, \end{aligned} \tag{33}$$

and

$$\begin{aligned} U^T G = (1 + \alpha_t)(U^T U_t)\Sigma_t V_t^T - \alpha_t(U^T U_{Z_t})\Sigma_{Z_t}V_{Z_t}^T \\ + \mu U^T\mathscr{P}_\Omega(M - Y_t). \end{aligned} \tag{34}$$

It should be note that the l_qiAPG algorithm is different from l_qPG algorithm. First, the l_qiAPG algorithm employs closed-form thresholding formulas, while l_qPG algorithm uses q-thresholding function. The q-thresholding function is often solved by numerical methods and only obtained its approximate solutions. Besides, the l_qiAPG algorithm allows inexact proximal step (Step 4), which makes the algorithm more efficient and faster. Moreover, the l_qiAPG algorithm uses more robust acceleration scheme, in which involves X_t, X_{t-1} and Z_t. Finally, the l_qiAPG algorithm exploits the "sparse plus low-rank" structure to improve the speed of matrix multiplications.

The convergence of $F(X)$ is shown as follows:

Theorem 3 *Let $\{X_t\}$ be the sequence generated by l_qiAGP algorithm, then*

(i) *$\{X_t\}$ is a minimization sequence and bounded, and has at least one accumulation point;*
(ii) *$F(X_t)$ converges to $F(X_*)$, where X_* is any accumulation point of $\{X_t\}$.*

Proof

(i) According to the control condition (23) in l_qiAGP algorithm, for any $t = 1, 2, \dots$, we have

$$F(X_t) \le F(X_{t-1}) \le \cdots \le F(X_0). \tag{35}$$

Thus, $\{X_t\}$ is a minimization sequence of $F(X)$, and $\{F(X_t)\}$ is bounded. Since $\{X_t\} \subset \{X : F(X) \le F(X_0)\}$, $\{X_t\}$ is bounded and has at least one accumulation point.

(ii) From above description, we also have $F(X_t)$ converges to F_*, where F_* is a constant. Suppose an accumulation point of $\{X_t\}$ is X_*. Using the continuity of $F(X)$, we obtain $F(X_t) \to F_* = F(X_*)$ as $t \to \infty$. We complete the proof. □

4 Numerical experiments

In this section, we validate our proposed l_qiAPG algorithm for matrix completion problems by conducting a series of experiments. We compare our proposed method with the following state-of-the-art matrix completion algorithms.

(1) Three nuclear norm minimization algorithms: APGL [13], AIS-Impute [34], and Active [40];
(2) Two low-rank matrix decomposition-based methods: low-rank matrix fitting (LMaFit) [43], and alternating steepest descent algorithm (ASD) [45];
(3) Two methods for solving models with schatten-q regularizers: HFPA [29], and multi-schatten-q norm surrogate (MSS) [46];
(4) Three methods for solving models with nonconvex low-rank regularizers: iterative reweighted nuclear norm (IRNN) [47] algorithm, FaNCL [35], and niAPG [37].

We also tested singular value projection (SVP) [48], rank-one matrix pursuit method (R1MP) [44], iterative reweighted least square(IRucLp) [49], and Soft-AIS [50]. In the following tests, however, these methods are slow or require large memory, so their results are not reported here.

In the following experiments, we follow the recommended settings of the parameters for these algorithms. For our proposed algorithm, we set $\bar{\lambda} = 10^{-4}$, $\mu = 1.99$, and $\eta = 0.75$. To prove the effectiveness of our proposed algorithm, we consider three cases: synthetic data, image recovery and recommendation problems. Besides, all the algorithms are stopped when the difference in objective values between consecutive iterations becomes smaller than 10^{-5}. All the algorithms are implemented in MATLAB R2014a on a Windows server 2008 system with Intel Xeon E5-2680-v4 CPU(3 cores, 2.4 GHz) and 256 GB memory.

4.1 Synthetic data

The test matrix $M \in \mathbb{R}^{m\times n}$ with rank r is generated as $M = M_L M_R + N$, where the entries of random matrices $M_L \in \mathbb{R}^{m\times r}$ and $M_R \in \mathbb{R}^{r\times n}$ are sampled i.i.d. from the standard normal distribution $\mathcal{N}(0, 1)$, and entries of N sampled from $\mathcal{N}(0, 0.1)$. Without loss of generality, we set $m = n$ and $r = 5$. We then sampled a subset Ω of p entries uniformly at random as the observations, where $p = 2mrlog(m)$.

Similar to [35], we evaluate the recovery performance of the algorithms based on the i) normalized mean squared

Table 1 Comparison of different algorithms on synthetic data, NMSE is scaled by 10^{-2}. CPU time is in seconds, and “sr” denotes the sample ratio

	$m = 1000$ sr = 6.91%			$m = 2000$ sr = 3.80%			$m = 3000$ sr = 2.67%			$m = 5000$ sr = 1.70%		
	NMSE	Rank	Time	NMSE	Rank	Time	NMSE	Rank	Time	NMSE	Rank	Time
APGL	3.82	68	45.1803	3.81	104	447.22	3.66	132	2372.0	3.49	179	21,448.0
AIS-Impute	3.05	5	5.0919	3.61	80	259.82	3.53	80	500.96	3.33	80	1579.3
LMaFit	4.14	5	1.4848	4.15	5	3.4542	4.19	5	5.240	4.19	5	10.40
ASD	4.75	5	1.6022	4.81	5	3.2956	4.73	5	5.0718	4.77	5	9.3707
HFPA	4.89	5	5.7976	4.91	5	24.5722	4.92	5	98.6759	4.92	5	222.15
IRNN	1.83	5	92.8488	1.77	5	1046.5	1.70	5	4761.4	1.67	5	23,906.0
FaNCL	1.86	5	2.1539	1.78	5	8.8105	1.73	5	19.1202	1.61	5	39.9434
niAPG	1.87	5	0.5796	1.74	5	2.2092	1.73	5	4.1760	1.67	5	10.9549
l_qiAPG(1/2)	1.84	5	0.4691	1.77	5	1.6351	1.69	5	3.1847	1.64	5	6.8887
l_qiAPG(2/3)	1.84	5	0.4816	1.74	5	1.6496	1.73	5	2.9768	1.66	5	6.8203

error NMSE $= \|\mathscr{P}_{\Omega^\perp}(X - UV)\|_F / \|\mathscr{P}_{\Omega^\perp}(UV)\|_F$, where X is the recovered matrix and $\Omega^\perp$ stands for the unobserved positions; ii) rank of X; and iii) running time. We vary m in the range $\{1000, 2000, 3000, 5000\}$. For each algorithm, we present its average NMSE, rank and running time with 10 runs.

The average NMSE, rank, and running time are reported in Table 1. The results in Table 1 demonstrate that our proposed l_qiAPG is a competitive algorithm. More precisely, l_qiAPG algorithm runs fastest among these algorithms. In terms of accuracy, l_qiAPG attained satisfying performance. In Table 1, we can find that l_qiAPG algorithm achieves most accurate solutions in nearly all problems. We also observe

Table 2 Comparison of different algorithms on benchmark images. CPU time is in seconds

	Barbara		Bridge		Clown		Couple		Crowd		Fingerprint	
	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
APGL	23.74	9.39	23.16	7.78	26.36	7.60	25.57	7.67	24.88	7.82	23.06	7.89
AIS-Impute	23.68	21.22	16.39	1.33	26.35	17.00	16.73	1.33	24.86	20.74	14.87	1.38
Active	23.77	15.44	23.26	19.78	26.33	6.92	25.56	8.39	24.86	10.85	23.06	18.59
LMaFit	18.61	3.56	19.37	0.74	20.5	3.83	21.04	3.74	18.1	4.11	16.44	3.63
ASD	27.03	12.60	25.97	13.13	23.86	9.87	24.87	9.97	28.31	9.33	25.72	9.92
HFPA	23.37	33.63	22.75	33.12	28.45	32.59	26.42	33.35	26.13	33.87	22.58	33.76
MSS	22.34	22.17	21.97	22.16	24.99	22.96	24.27	22.93	23.70	22.40	21.51	22.45
IRNN	23.26	10.80	22.76	23.58	26.06	5.11	25.10	3.88	24.52	5.11	22.81	15.96
FaNCL	23.01	5.75	22.63	10.42	25.95	2.89	24.85	1.69	24.34	2.92	22.61	26.67
niAPG	23.20	0.75	22.76	0.81	25.97	0.72	25.09	0.65	24.55	0.70	22.83	1.03
l_qiAPG(1/2)	22.28	1.43	22.26	2.10	28.20	1.83	26.13	1.92	25.84	1.53	22.69	2.15
l_qiAPG(2/3)	22.19	1.53	22.17	2.24	28.13	1.45	26.22	1.91	25.83	1.97	22.73	2.53

	Houses		Lighthouse		Truck		Trucks		Zelda	
	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
APGL	22.34	8.07	24.82	7.78	27.21	7.43	25.56	7.44	28.72	7.26
AIS-Impute	13.73	1.40	16.43	1.33	21.12	1.38	18.36	1.27	28.66	9.92
Active	22.33	23.81	24.76	10.31	27.22	3.46	25.53	6.95	28.69	2.75
LMaFit	16.21	3.59	20.03	3.69	24.11	3.88	21.87	3.67	23.15	3.55
ASD	26.09	10.12	25.86	9.52	28.14	9.44	27.28	9.71	31.83	12.73
HFPA	21.71	32.98	25.31	32.73	28.60	33.61	26.41	34.31	31.44	33.75
MSS	21.05	23.09	23.65	22.33	25.53	22.15	24.14	23.10	27.34	23.04
IRNN	22.21	9.99	24.43	5.76	26.41	2.63	24.92	6.53	28.26	6.24
FaNCL	21.90	7.72	24.46	7.16	26.32	1.36	24.83	2.75	28.26	1.05
niAPG	22.14	1.00	24.53	0.69	26.41	0.50	24.99	0.54	28.25	0.45
l_qiAPG(1/2)	21.62	2.08	25.05	1.93	29.52	1.38	26.35	2.18	33.37	1.77
l_qiAPG(2/3)	21.66	2.45	24.99	1.65	29.45	1.18	26.44	2.23	33.43	2.18

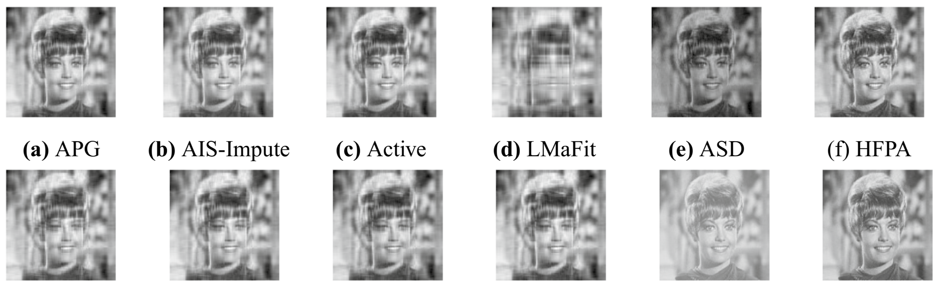


Fig. 1 Results of image recovery by using different algorithms

(a) APG **(b)** AIS-Impute **(c)** Active **(d)** LMaFit **(e)** ASD (f) HFPA

(g) MSS **(h)** IRNN **(i)** FaNCL **(j)** niAPG **(k)** l_qiAPG(1/2) (l) l_qiAPG(2/3) that as the size of the matrix increases, the l_qiAPG algorithm will be more faster than other algorithms. Moreover, l_qiAPG algorithm is able to solve large-scale random matrix completions. Specifically, the running time of l_qiAPG algorithm for solving problem with $m = 10^5$, $sr = 0.12\%$ is within 1159.2 s (NMSE is smaller than 0.0141), while most other algorithms cannot get satisfactory results within this time. Therefore, taking both accuracy and converge speed into consideration, our proposed l_qiAPG algorithm has the best recovery performance among these algorithms.

4.2 Image recovery

In this section, we will apply l_qiAPG algorithm to image inpainting problems. In image inpainting problems, the values of some of the pixels of the image are missing, and our mission is to find out the missing values. If the image is low rank or numerical low rank, we can solve the image inpainting problem as a matrix completion problem. In this tests, we use the following benchmark images: Barbara, Bridge, Clown, Couple, Crowd, Fingerprint, Girlface, Houses, Kiel, Lighthouse, Tank, Truck, Trucks, and Zelda. The size of each image is 512×512. We directly deal with the original images. As the image matrix is not guaranteed to be low rank, we use rank 50 for the estimated matrix for test. We randomly exclude 50% of the pixels in the images, and the remaining ones are used as the observations. We use peak signal-to-noise ratio (PSNR) [51] and running time to evaluate the recovery performance of the algorithms. We represent their average results with 5 runs.

We list the results in terms of the PSNR in Table 2. We also exhibit the results of image recovery by using different algorithms for Zelda in Fig. 1. The results in Fig. 1 demonstrate that our proposed l_qiAPG algorithm performs best among 12 algorithms. We also can easily find from Table 2 that our l_qiAPG, niAPG, HFPA, and ASD algorithms achieve the best results. Although, the results obtained by HFPA and ASD algorithms are slightly better than l_qiAPG. Our algorithm is much faster than HFPA and ASD algorithms. In addition, although niAPG algorithm is slightly faster than our algorithm, the solutions obtained by our algorithm are more accurate. More precisely, l_qiAPG algorithm achieves most accurate solutions 7 times on all images, while niAPG achieves most accurate solutions only 4 times. Again, taking both accuracy and converge speed into consideration, our proposed l_qiAPG algorithm is a competitive algorithm in the field of image recovery.

4.3 Recommendation

To further demonstrate the effectiveness of our proposed method, in this section we apply l_qiAPG algorithm on Jester and MovieLens datasets. We consider six datasets: Jester1, Jester2, Jester3, Jester-all, MovieLens-100K, and MovieLens-1M. The characteristics of these datasets are shown in Table 3. The Jester datasets are collected from a joke recommendation system. The whole data is stored in three excel files with the following characteristics [13].

(1) jester-1: 24,983 users who have rated 36 or more jokes;

Table 3 Characteristics of the recommendation datasets

Dataset	Row	Column	Rating
Jester1	24,983	100	10^6
Jester2	23,500	100	10^6
Jester3	24,983	100	6×10^5
MovieLens100K	943	1682	10^5
MovieLens1M	6040	3706	10^6

Table 4 Comparison of different algorithms on Jester and MovieLens datasets. CPU time is in seconds

	Jester1		Jester2		Jester3		Jester-all		MovieLens100K		MovieLens1M	
	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time
APGL	4.1709	157.53	4.1976	145.35	4.6358	412.55	4.2547	795.66	1.0367	1264.9	–	–
AIS-Impute	4.3158	42.88	4.3471	50.04	5.2217	83.84	4.6815	474.09	1.0390	15.04	0.8880	103.44
LMaFit	5.3163	8.04	5.3831	6.97	5.5149	3.84	5.3901	19.09	–	–	–	–
ASD	4.6943	106.28	4.7283	102.01	5.1348	41.79	4.9944	279.94	1.1552	5.67	0.9992	56.48
HFPA	4.3410	48.59	4.3662	49.14	–	–	4.3449	92.31	1.1992	9.01	1.1117	28.34
MSS	4.3269	110.31	4.3568	104.56	–	–	4.3234	210.43	0.9616	28.79	0.8849	434.75
IRNN	4.3257	412.07	4.3512	370.13	6.8324	274.09	4.6352	1224.6	1.1026	368.84	–	–
FaNCL	4.3210	97.91	4.3589	93.97	4.8349	38.50	4.6251	257.90	1.0871	5.22	0.9190	52.03
niAPG	4.2784	4.18	4.2898	4.45	4.9993	14.39	4.4087	19.96	0.9503	0.62	0.8815	22.98
l_qiAPG(1/2)	4.2867	4.91	4.2970	5.49	4.9315	9.17	4.3992	20.01	0.9682	3.83	0.9079	24.20
l_qiAPG(2/3)	4.3000	7.37	4.3129	8.93	4.9857	12.84	4.4219	19.11	1.0481	5.25	0.9270	38.40

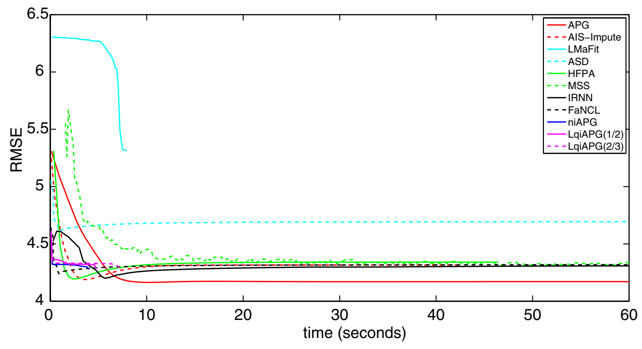
(2) jester-2: 23,500 users who have rated 36 or more jokes;
(3) jester-3: 24,938 users who have rated between 15 and 35 jokes.

The MovieLens datasets are collected from the MovieLens website. The characteristics of these data sets are list as follows [13]:

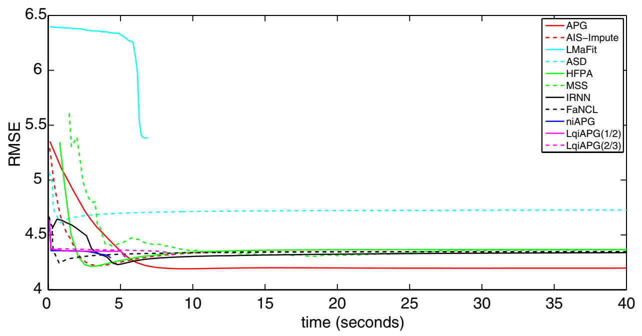
(1) movie-100K: 100,000 ratings for 1682 movies by 943 users;
(2) movie-1M: 1 million ratings for 3900 movies by 6040 users.

The Jester-all is obtained by combining Jester1, Jester2, and Jester3 datasets. In the following test, we follow the setup in [35], and randomly pick up 50% of the observed for training and use the remaining 50% for testing. We use the root mean squared error (RMSE) and running time to evaluate the recovery performance of the algorithms. The RMSE is defined as $\mathrm{RMSE} = \sqrt{\|\mathscr{P}_{\bar{\Omega}}(X - M)\|_F^2 / |\bar{\Omega}|_1}$, where $\bar{\Omega}$ is the test set, X is the recovered matrix. The test of each algorithm is repeated 5 times.

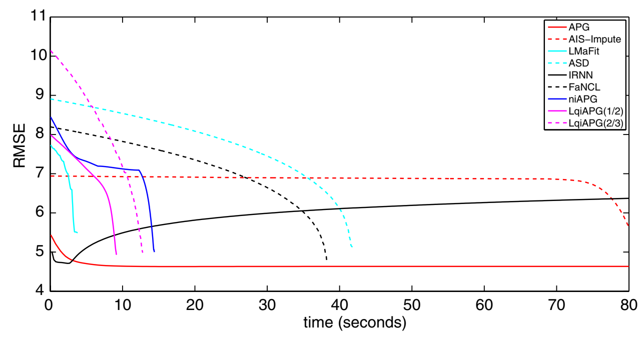
The reconstruction results in terms of RMSE and running time are listed in Table 4. We depict the RMSE along the CPU times and show these results in Fig. 2. As can be seen



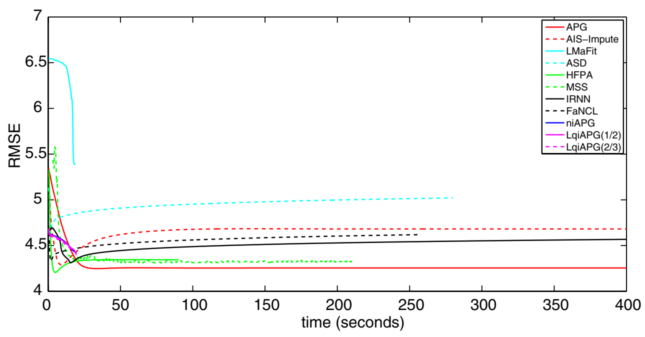
(a) Jester1



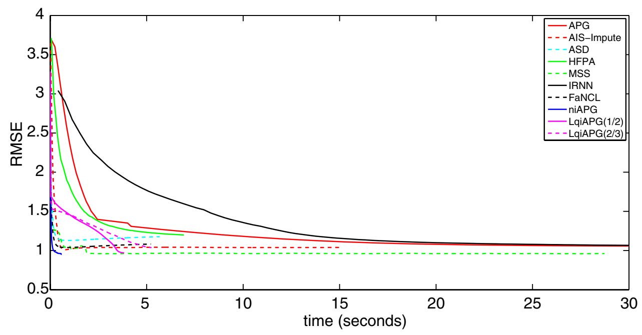
(b) Jester2



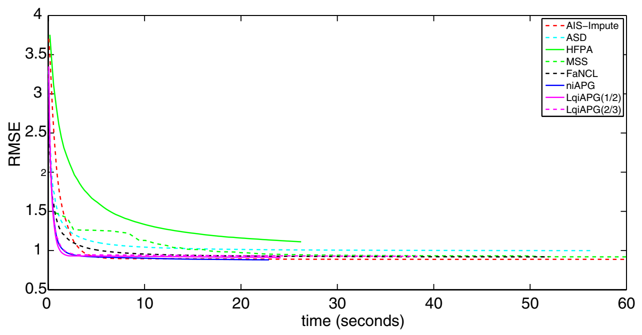
(c) Jester3



(d) Jester-all



(e) MovieLens-100K



(f) MovieLens-1M

Fig. 2 Matrix completion results on Jester and MovieLens datasets by using different algorithms. In the first and second rows, we apply different algorithms on Jester datasets and depict the RMSE along the CPU times. In the last row, we apply different algorithms on MovieLens datasets and depict the RMSE along the CPU times

form Table 4, our proposed algorithm, APGL, and niAPG achieve the lowest RMSE in nearly all problems. We also observe that the results obtained by APGL are slightly better than ours, but our method is much faster. Besides, our proposed algorithm can be runs on all six data sets, while many algorithms only run partial data sets. Furthermore, Fig. 2 demonstrate that our proposed algorithm decreases the RMSE much faster than others. This is the third time to show that our proposed algorithm is a competitive algorithm in the field of matrix completion.

5 Conclusion

In this paper, we focus on the large-scale low-rank matrix completion problems with l_q regularizers and proposed an efficient and fast inexact thresholding algorithm called l_q iAPG algorithm to handle such problems. The key idea is to employ the closed-form q-thresholding operator to approximate the rank of a matrix and power method to approximate the SVD procedure. At the same time, our proposed algorithm inherits the great efficiency advantages of first-order gradient-based methods and is simple and easy to use, which is more suitable for large-scale matrix completion problems. In addition, we adopted three techniques to accelerate the convergence rate of our proposed algorithm, which are Nesterov's accelerated gradient method, continuation technique, and the "sparse plus low-rank" structure. Furthermore, a convergence analysis of the l_qiAPG algorithm has shown that the sequence $\{X_t\}$ generated by l_qiAPG algorithm is bounded and has at least one accumulation point. More important, we also shown that the objective function $F(X)$ converges to $F(X_*)$, where X_* is any accumulation point of $\{X_t\}$. Finally, extensive experiments on matrix completion problems validated that our proposed algorithm is more efficient and faster. Specifically, we compare our proposed algorithm with state-of-the-art algorithms on a series of scenarios, including synthetic data, image recovery and recommendation problems. All results demonstrated that our proposed algorithm is able to achieve comparable recovery performance, while being faster and more efficient than state-of-the-art methods.

Acknowledgements This work is supported in part by the Natural Science Foundation of China under Grant 61273020, and in part by the Fundamental Research Funds for the Central Universities under Grant XDJK2019B063.

References

1. Rennie J, Srebro N (2005) Fast maximum margin matrix factorization for collaborative prediction. In: Proceedings of the 22nd international conference on machine learning (ICML-05), pp 713–719
2. Koren Y (2008) Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 426–434
3. Li X, Wang Z, Gao C, Shi L (2017) Reasoning human emotional responses from large-scale social and public media. Appl Math Comput 310:128–193
4. Peng X, Zhang Y, Tang H (2016) A unified framework for representation-based subspace clustering of out-of-sample and large-scale data. IEEE Trans Neural Netw Learn Syst 27(12):2499–2512
5. Yang L, Nie F, Gao Q (2018) Nuclear-norm based semi-supervised multiple labels learning. Neurocomputing 275:940–947
6. Yang L, Gao Q, Li J, Han J, Shao L (2018) Zero shot learning via low-rank embedded semantic autoencoder. In: Proceedings of the international joint conference on artificial intelligence, pp 2490–2496
7. Cabral R, De la Torre F, Costeira JP, Bernardino A (2015) Matrix completion for weakly-supervised multi-label image classification. IEEE Trans Pattern Anal Mach Intell 37(1):121–135
8. Yang L, Shan C, Gao Q, Gao X, Han J, Cui R (2019) Hyperspectral image denoising via minimizing the partial sum of singular values and superpixel segmentation. Neurocomputing 330:465–482
9. Recht B, Fazel M, Parrilo PA (2010) Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. SIAM Rev 52(3):471–501
10. Candès EJ, Recht B (2009) Exact matrix completion via convex optimization. Found Compt Math 9(6):717–772
11. Fazel M (2002) Matrix rank minimization with applications. Ph.D. thesis, Stanford University
12. Cai J-F, Candès EJ, Shen Z (2010) A singular value thresholding algorithm for matrix completion. SIAM J Optim 20(4):1956–1982
13. Toh K-C, Yun S (2010) An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. Pac J Optim 6(3):615–640
14. Ma S, Goldfarb D, Chen L (2011) Fixed point and Bregman iterative methods for matrix rank minimization. Math Program 128(1–2):321–353
15. Mazumder R, Hastie T, Tibshirani R (2010) Spectral regularization algorithms for learning large incomplete matrices. J Mach Learn Res 11:2287–2322
16. Yao Q, Kwok JT (2015) Accelerated inexact soft-impute for fast largescale matrix completion. In: Proceedings of the international joint conference on artificial intelligence, pp 4002–4008
17. Fan J, Li R (2001) Variable selection via nonconcave penalized likelihood and its Oracle properties. J Am Stat Assoc 96:1348–1361
18. Zhang T (2010) Analysis of multi-stage convex relaxation for sparse regularization. J Mach Learn Res 11:1081–1107
19. Candès EJ, Wakin MB, Boyd SP (2008) Enhancing sparsity by reweighted l_1 minimization. J Fourier Anal Appl 14(5–6):877–905
20. Hu Y, Zhang D, Ye J, Li X, He X (2013) Fast and accurate matrix completion via truncated nuclear norm regularization. IEEE Trans Pattern Anal Mach Intell 35(19):2117–2130
21. Zhang C-H (2010) Nearly unbiased variable selection under minimax concave penalty. Ann Stat 38(2):894–942
22. Marjanovic G, Solo V (2012) On l_q optimization and matrix completion. IEEE Trans Signal Process 60(11):5714–5724
23. Abu Arqub O, Abo-Hammour Z (2014) Numerical solution of systems of second-order boundary value problems using continuous genetic algorithm. Inf Sci 279:396–415
24. Abu Arqub O, AL-Smadi M, Momani S, Hayat M (2016) Numerical solutions of fuzzy differential equtions using reproducing kernel Hilbert space method. Soft Comput 20(8):3283–3302
25. Abu Arqub O, AL-Smadi M, Momani S, Hayat M (2017) Application of reproducing kernel algorithm for solving second-order,

two-point fuzzy boundary value problems. Soft Comput 21(23):7191–7206
26. Abu Arqub O (2017) Adaptation of reproducing kernel algorithm for solving fuzzy Fredholm–Volterra integrodifferential equations. Neural Comput Appl 28(7):1591–1610
27. Xu Z, Chang X, Xu F, Zhang H (2012) $L_{1/2}$ regularization: a thresholding representation theory and a fast solver. IEEE Trans Neural Netw Learn Syst 23(7):1013–1027
28. Cao W, Sun J, Xu Z (2013) Fast image deconvolution using closed-form thresholding formulas of $l_q(q = 1/2, 2/3)$ regularization. J Vis Commun Image R 24(1):31–41
29. Peng D, Xiu N, Yu J (2017) $S_{1/2}$ regularization methods and fixed point algorithms for affine rank minimization problems. Comput Optim Appl 67:543–569
30. Wang Z, Wang W, Wang J, Chen S (2019) Fast and efficient algorithm for matrix completion via closed-form 2/3-thresholding operator. Neurocomputing 330:212–222
31. Qian W, Cao F (2019) Adaptive algorithms for low-rank and sparse matrix recovery with truncated nuclear norm. Int J Mach Learn Cyb 10(6):1341–1355
32. Nesterov Y (2013) Gradient methods for minimizing composite functions. Math Program 140(1):125–161
33. Li H, Lin Z (2015) Accelerated proximal gradient methods for noncovex programming. In: Proceedings of the advances in neural information processing systems, pp 379–387
34. Yao Q, Kwok J (2019) Accelerated and inexact soft-impute for large-scale matrix and tensor completion. IEEE Trans Knowl Data En 31(9):1665–1679
35. Yao Q, Kwok J, Wang T, Liu T (2019) Large-scale low-rank matrix learning with nonconvex regularizers. IEEE Trans Pattern Anal Mach Intell 41(11):2628–2643
36. Gu B, Wang D, Huo Z, Huang H (2018) Inexact proximal gradient methods for non-convex and non-smooth optimization. In: AAAI conference on artificial intelligence
37. Yao Q, Kwok J, Gao F, Chen W, Liu T (2017) Efficient inexact proximal gradient algorithm for nonconvex problems. In: Proceedings of the international joint conference on artificial intelligence, pp 3308–3314
38. Oh T, Matsushita Y, Tai Y, Kweon I (2018) Fast randomized singular value thresholding for low-rank optimization. IEEE Trans Pattern Anal Mach Intell 40(2):376–391
39. Halko N, Martinsson P-G, Tropp J (2011) Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev 53(2):1805–1811
40. Hsieh C.-J, Olsen P (2014) Nuclear norm minimization via active subspace selection. In: Proceedings of the 31st international conference on machine learning (ICML-14), pp 575–583
41. Larsen R (1998) Lanczos bidiagonalization with partial reorthogonalization. Department of Computer Science, Aarhus University, DAIMI PB-357
42. Gong P, Zhang C, Lu Z, Huang J, Ye J (2013) A general iterative shrinkage and tresholding algorithm for non-convex regularized optimization problems. In: Proceedings of the 30th international conference on machine learning (ICML-13), pp 37–45
43. Wen Z, Yin W, Zhang Y (2012) Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. Math Program Comput 44(4):333–361
44. Wang Z, Lai M, Lu Z, Fan W, Davulcu H, Ye J (2015) Orthogonal rank-one matrix pursuit for low rank matrix completion. SIAM J Sci Comput 37(1):A488–A514
45. Tanner J, Wei K (2016) Low rank matrix completion by alternating steepest descent methods. Appl Comput Harmon A 40:417–420
46. Xu C, Lin Z, Zha H (2017) A unified convex surrogate for the schatten-p norm. In: AAAI conference on artificial intelligence
47. Lu C, Tang J, Yan S, Lin Z (2016) Noncovex nonsmooth low rank minimization via iteratively reweighted nuclear norm. IEEE Trans Image Process 25(2):829–839
48. Jain P, Meka R, Dhillon I (2010) Guaranteed rank minimization via singular value projection. In: Proceedings of the advances in neural information processing systems, pp 937–945
49. Lai M, Xu Y, Yin W (2013) Improved iteratively rewighted least squares for unconstrained smoothed l_p minimization. SIAM J Numer Anal 5:927–957
50. Hastie T, Mazumder R, Lee J, Zadeh R (2015) Matrix completion and low-rank SVD via fast alternating least squares. J Mach Learn Res 16:3367–3402
51. Thu Q, Ghanbari M (2008) Scope of validity of PSNR in image/video quality assesment. Electron Lett 44(13):800–801