Matrix completion via alternating direction method

Article *in* IMA Journal of Numerical Analysis · January 2012

DOI: 10.1093/imanum/drq039

3 authors:

Caihua Chen
Nanjing University

Bing-Sheng He
Nanjing University

Xiaoming Yuan
The University of Hong Kong

Matrix Completion via an Alternating Direction Method

CAIHUA CHEN†, BINGSHENG HE‡
Department of Mathematics, Nanjing University, Jiangsu 210093, People's Republic of China

AND

XIAOMING YUAN§
Department of Mathematics, Hong Kong Baptist University, Hong Kong, People's Republic of China

The matrix completion problem is to complete a unknown matrix from a small number of entries, and it captures many applications in diversified areas. Recently, it was shown that completing a low-rank matrix can be well accomplished via solving its convex relaxation problem using the nuclear norm. This paper shows that the alternating direction method (ADM) is applicable for completing a low-rank matrix including the noiseless case, the noisy case and the positive semidefinite case. The ADM approach for the matrix completion problem is easily implementable and very efficient. Numerical comparisons of the ADM approach with some state-of-the-art methods are reported.

Keywords: Matrix completion, convex programming, nuclear norm, low-rank, alternating direction method, noise.

1. Introduction

The matrix completion problem is to complete a unknown matrix from a small number of entries. For many applications, the matrix to be completed is of low rank. Such examples include the well-known Netflix problem (James & Stan (2007); Netflix (2006)), System Identification (Fazel *et al.* (2001); Fazel (2002)), Global Positioning (Candès & Plan (2009)), Computer Vision (Tomasi & Kanade (1992)), Multi-task Learning (Abernethy *et al.* (2006, 2009); Amit *et al.* (2007); Argyriou *et al.* (2008)), Structure-from-motion problem (Chen & Suter (2004); Tomasi & Kanade (1992)), etc. In fact, as pointed out in Candès & Tao (2009), "In general, accurate recovery of a matrix from a small number of entries is impossible, but the knowledge that the unknown matrix has low rank radically changes this premise, making the research for solutions meaningful". Hence, we are particularly interested in the completion of low-rank matrices.

In Candès & Recht (2008)), the authors proved rigorously that most low-rank matrices can be perfectly completed (in the sense of high probability, see Candès & Recht (2008) for the precise meaning) from sets of sampled entries that might have surprisingly small cardinalities, provided that some inco-

†Email: cchenhuayx@gmail.com
‡Email: hebma@nju.edu.cn
§Corresponding author. Email: xmyuan@hkbu.edu.hk

herence assumptions are satisfied. We refer to e.g. Keshavan *et al.* (2009); Recht *et al.* (2007, 2009); Zhu *et al.* (2009) for some recent theoretical results on the matrix completion. Numerically, it was shown in Candès & Recht (2008) that completing a low-rank matrix can be well accomplished by solving the following convex relaxation problem:

$$\begin{array}{ll} \min & \|X\|_* \\ s.t. & X_{ij} = M_{ij}, \quad (i,j) \in \Omega, \end{array} \tag{1.1}$$

where $M \in \mathscr{R}^{m\times n}$ is the unknown matrix to be completed, $\Omega = \{(i,j)|i \in \{1,2,\cdots,m\}, j \in \{1,2,\cdots,n\}\}$ is an index set with the cardinality p; $M_{ij}'s$ $((i,j) \in \Omega)$ represents the sampled (known) entries of M; $\|\cdot\|_*$ is the nuclear norm (also known as trace norm, Ky Fan norm, or Schatten 1-norm) which is defined as the sum of all singular values. We refer to Candès & Tao (2009) for more results about the convex relaxation problem (1.1).

This breakthrough achievement revealing that completing a low-rank matrix can be accomplished by solving (1.1) has immediately inspired many authors to work on efficient numerical algorithms for the convex relaxation model involving nuclear norm (1.1). To mention a few, the singular value thresholding (SVT) algorithm proposed by Cai *et al.* (2008) which is essentially a gradient method for solving the dual of a regularized approximation of (1.1); the fixed point continuation with approximate SVD (FPCA) method in Ma *et al.* (2008) which solves the Lagrangian version (least square regularized with the nuclear norm) of (1.1); the accelerated proximal gradient Lagrangian (APGL) method in Toh & Yun (2009) solving the Lagrangian version of (1.1); the proximal point algorithm (PPA) in Liu *et al.* (2009) solving general nuclear norm minimization problem with linear equality and second order cone constraints; some interior-point method in Candès & Recht (2008); Liu & Vandenberghe (2008); Recht *et al.* (2007) for solving the semidefinite programming reformulation of (1.1); and Recht *et al.* (2007) for a projected subgradient method.

It is obviously interesting to consider the matrix completion problem when the sampled entries are corrupted with a small amount of noise, see e.g. Candès & Plan (2009); Keshavan *et al.* (2009); Schmidt (1986). According to Candès & Plan (2009), the convex relaxation model of the matrix completion with noise is:

$$\begin{array}{ll} \min & \|X\|_* \\ s.t. & \|X_\Omega - M_\Omega\|_F \leqslant \delta, \end{array} \tag{1.2}$$

where $\delta \geqslant 0$ measures the noise level; $\|\cdot\|_F$ denotes the Frobenius norm induced by the standard trace inner product; and $(\cdot)_\Omega : \mathscr{R}^{m\times n} \to \mathscr{R}^{m\times n}$ is the sampling operator defined by $(X_\Omega)_{ij} = X_{ij}$ if $(i,j) \in \Omega$ and 0 otherwise. Note that the model (1.1) is a special case of (1.2) where the noise level $\delta = 0$. Compared to the aplenty literature for solving (1.1), available numerical algorithms for the noisy model (1.2) are evidently less.

An alternative model to study noisy matrix completion problem is the following nuclear norm regularized least squares problem:

$$\min \|X\|_* + \frac{\mu}{2}\|(X-M)_\Omega\|_F^2, \tag{1.3}$$

where $\mu > 0$ is a fixed parameter trading off the low-rank requirement and the violation of constraints. As pointed out in Liu *et al.* (2009), (1.3) is preferable for the particular case where the noisy level δ is ambiguous. Note that the methods in Ma *et al.* (2008); Pong *et al.* (2009); Toh & Yun (2009) also solve (1.3) to generate approximate solutions of the matrix completion problem (1.1).

The objective of this paper is to show that the alternating direction method (ADM) is an easy and efficient approach to solving the noisy model (1.2). More specifically, (1.2) can be easily reformulated into the following linearly constrained convex programming problem:

$$\begin{array}{rl} \min & \|X\|_* \\ s.t. & X - Y = 0 \\ & X \in \mathscr{R}^{m\times n},\ Y \in \mathbb{U} := \{U \in \mathscr{R}^{m\times n} \mid \|U_\Omega - M_\Omega\|_F \leqslant \delta\}, \end{array} \tag{1.4}$$

which falls into the applicable range of ADM perfectly. As we shall analyze, when ADM is applied to solve (1.4), the resulted subproblems all have closed-form solutions. Thus, the ADM approach for (1.4) is easily implementable. The efficiency of ADM will be also clear when we compare it with some state-of-the-art methods in Section 4.

The rest of the paper is organized as follows. In Section 2, we present the ADM approach to solving (1.4) and elaborate on how to derive the closed-form solutions of the subproblems generated at each iteration. In Section 3, we concentrate on an interesting extension of (1.4) where X is additionally required to be positive semidefinite, and develop the corresponding ADM. In Section 4, we implement the ADM approach to solving some test examples, and compare it with some existing influential methods. In Section 5, we present the ADM approach to solving some more general models which are closely related to the model (1.4). Some relevant future works are also specified in this section. Finally, some conclusions are made in Section 6.

2. Alternating direction method

Roughly speaking, ADM is a practical version of the classical augmented Lagrangian method (ALM) for solving linearly constrained convex programming problem with the separate structure whose objective function is in the form of the sum of two individual functions without crossed variables. Ever since its presence in Gabay & Mercier (1976), ADM and its variants have been widely studied in many areas such as convex programming and variational inequalities, see e.g. Bertsekas & Tsitsiklis (1989); Chen & Teboulle (1994); Eckstein (1994); Eckstein & Fukushima (1994); Fukushima (1992); Gabay (1983); Glowinski (1984); Glowinski & Le Tallec (1989); He *et al.* (2002); Kontogiorgis & Meyer (1998); Tseng (1997)). We also refer to e.g. Esser (2009); He *et al.* (2009); Lin *et al.* (2009); Ng *et al.* (2009); Wen *et al.* (2009); Yang & Zhang (2009); Yuan (2009); Yuan & Yang (2009) for some recently-exploited applications of ADM in other areas.

2.1 *Algorithmic framework*

The augmented Lagrangian function of (1.4) is

$$\mathscr{L}_{\mathscr{A}}(X,Y,Z,\beta) := \|X\|_* - \langle Z, X-Y\rangle + \frac{\beta}{2}\|X-Y\|^2,$$

where $Z \in \mathscr{R}^{m\times n}$ is the Lagrange multiplier of the linear constraint; $\beta > 0$ is the penalty parameter for the violation of the linear constraint and $\langle\cdot\rangle$ denotes the standard trace inner product. Applying the classical ALM (see e.g. Bertsekas (1982); Nocedal & Wright (2006)) for (1.4), we obtain the iterative scheme:

$$\left\{ \begin{array}{l} (X^{k+1}, Y^{k+1}) \in \operatorname{argmin}_{X\in\mathscr{R}^{m\times n}, Y\in\mathbb{U}} \{\mathscr{L}_{\mathscr{A}}(X,Y,Z^k,\beta)\}, \\ Z^{k+1} = Z^k - \beta(X^{k+1} - Y^{k+1}), \end{array} \right. \tag{2.1}$$

where (X^k, Y^k, Z^k) is the given triple of iterate. The direct application of ALM, however, treats (1.4) as a generic minimization problem and ignores its favorable separate structure. Hence, the variables X and Y are required to be minimized simultaneously in (2.1).

The idea of ADM is to decompose the minimization task in (2.1) into two easier and smaller subproblems such that the involved variables X and Y can be minimized separately in the alternative order. The decomposed subproblems are usually much easier than the minimization task in (2.1). In fact, for many cases, these decomposed subproblems have closed-form solutions. Thus inner iterative processes approaching solutions of these subproblems can be avoided. This is the main reason why the ADM approach has been shown very efficient for solving some interesting applications in Esser (2009); He *et al.* (2009); Lin *et al.* (2009); Ng *et al.* (2009); Wen *et al.* (2009); Yang & Zhang (2009); Yuan (2009); Yuan & Yang (2009). In particular, applying the original ADM in Gabay & Mercier (1976) for solving (1.4), we have the following iterative scheme:

$$\begin{cases} Y^{k+1} \in \operatorname{argmin}_{Y\in\mathbb{U}}\{\mathscr{L}_{\mathscr{A}}(X^k, Y, Z^k, \beta)\}, \\ X^{k+1} \in \operatorname{argmin}_{X\in\mathscr{R}^{m\times n}}\{\mathscr{L}_{\mathscr{A}}(X, Y^{k+1}, Z^k, \beta)\}, \\ Z^{k+1} = Z^k - \beta(X^{k+1} - Y^{k+1}). \end{cases} \tag{2.2}$$

In Glowinski (1984) and Glowinski & Le Tallec (1989), it was proved that the step for updating the Lagrange multiplier Z^k can be generalized into

$$Z^{k+1} = Z^k - \gamma\beta(X^{k+1} - Y^{k+1}) \quad \text{with} \quad 0 < \gamma < \frac{\sqrt{5}+1}{2}.$$

Hence, by deriving the optimal conditions of the involved subproblems in (2.2), we can easily verify that the iterative scheme of the ADM approach for (1.4) is as follows.

Algorithm: ADM for noisy matrix completion problem (1.2)

Step 1. Solve Y^{k+1} via:

$$\langle Y - Y^{k+1}, Y^{k+1} + \frac{1}{\beta}Z^k - X^k\rangle \geqslant 0, \ \ \forall Y \in \mathbb{U}. \tag{2.3}$$

Step 2. Find X^{k+1} via:

$$0 \in \partial(\|X^{k+1}\|_*) - [Z^k - \beta(X^{k+1} - Y^{k+1})], \tag{2.4}$$

where $\partial(\cdot)$ denotes the subgradient operator of the convex function $\|\cdot\|_*$.

Step 3. Update the multiplier Z^{k+1} via:

$$Z^{k+1} = Z^k - \gamma\beta(X^{k+1} - Y^{k+1}), \quad \gamma \in (0, \frac{\sqrt{5}+1}{2}). \tag{2.5}$$

REMARK 2.1 Some more general ADM-based methods in the literature can be easily extended to solve (1.4). For example, the ADM-based descent method developed in Ye & Yuan (2007). We here omit details of these general ADM type methods for succinctness. In addition, since the convergence of ADM type methods has been well studied in the literatures, e.g. Gabay (1983); Gabay & Mercier (1976); Glowinski (1984); Glowinski & Le Tallec (1989); He & Yang (1998); Ye & Yuan (2007), we here omit the convergence analysis of the proposed ADM (2.3)-(2.5) for (1.4).

2.2 *Subproblems*

According to the analysis above, the computation of each iteration of the ADM approach for (1.4) is dominated by solving the subproblems (2.3) and (2.4). We now elaborate on the strategies of solving these subproblems, and thus derive the computational complexity of the proposed ADM for solving (1.4).

First, the subproblem (2.3) is a linear variational inequality (VI). Based on the classical results on VI, e.g. Eaves (1971), the solution of (2.3) can be given explicitly by:

$$Y^{k+1} = \mathscr{P}_{\mathbb{U}}[X^k - \frac{1}{\beta}Z^k]\,, \tag{2.6}$$

where $\mathscr{P}_{\mathbb{U}}[\cdot]$ denotes the projection operator onto the convex set $\mathbb{U}$. In fact, considering the specific form of $\mathbb{U}$, we can derive that

$$Y^{k+1} = \Big(\min\{\frac{\delta}{\|(B^{k+1}-M)_\Omega\|_F},1\}-1\Big)(B^{k+1}-M)_\Omega + B^{k+1},$$

where $B^{k+1} := X^k - \frac{1}{\beta}Z^k$. For the case of noiseless matrix completion (1.1) where

$$\mathbb{U} = \{U \in \mathscr{R}^{m\times n} | U_{ij} = M_{ij}, \forall (i,j) \in \Omega\},$$

the computation of (2.6) is even easier. In fact, for this case ,we have

$$Y^{k+1} = \begin{cases} M_{ij}, & \text{if } (i,j) \in \Omega, \\ B_{ij}^{k+1}, & \textit{otherwise.} \end{cases}$$

Second, it follows from Higham (1988) (see also Cai *et al.* (2008); Liu *et al.* (2009); Ma *et al.* (2008)) that the solution for (2.4) can be obtained explicitly via the well-known singular value shrinkage computation. More specifically, let

$$A^{k+1} = Y^{k+1} + \frac{1}{\beta}Z^k.$$

Let

$$A^{k+1} = U^{k+1}\Sigma^{k+1}(V^{k+1})^T \quad \text{with} \quad \Sigma^{k+1} = \text{diag}(\sigma_1^{k+1},\ldots,\sigma_{r^{k+1}}^{k+1}) \tag{2.7}$$

be the the singular value decomposition (SVD) of A^{k+1} with the rank r^{k+1}, where $U^{k+1} \in \mathscr{R}^{m\times r^{k+1}}$ and $V^{k+1} \in \mathscr{R}^{n\times r^{k+1}}$ be matrices with orthonormal columns; and the singular values $\sigma_i^{k+1'}s$ be positive. Then, the closed-form solution of (2.4) is given by:

$$X^{k+1} = U^{k+1}\hat{\Sigma}^{k+1}(V^{k+1})^T, \tag{2.8}$$

where $\hat{\Sigma}^{k+1} = \text{diag}(\{(\sigma_i^{k+1} - \frac{1}{\beta})_+\})$ and $(t)_+ := \max\{0,t\}$.

Therefore, when the ADM is applied to solve (1.4), the generated subproblems all have closed-form solutions. This feature makes the implementation of the ADM for (1.4) very easy. Recall that the main computation dominating each iteration of the proposed ADM for (1.4) is to perform the SVD once, which requires $O(n^3)$ flops (see e.g.Golub & Van Loan (1996)).

3. Positive semidefinite matrix completion

In this section, we consider a natural and interesting extension of (1.2) which requires additionally that X should be positive semidefinite. The resulted positive semidefinite matrix completion problem (PSDMCP) captures concrete applications such as completing a covariance matrix. More specifically, the convex relaxation model of PSDMCP is

$$\begin{array}{ll} \min & \langle I_n, X\rangle \\ s.t. & \|X_\Omega - M_\Omega\|_F \leqslant \delta \\ & X \in S_+^n, \end{array} \tag{3.1}$$

where S_+^n denotes the cone of positive semidefinite matrices in the space of symmetric $n\times n$ matrices and I_n is the identity matrix of $\mathscr{R}^{n\times n}$. Note that for the case $X \in S_+^n$, we have $\|X\|_* = \langle I_n, X\rangle$. Hence, the objective function of (3.1) is $\langle I_n, X\rangle$.

By reformulating (3.1) into the following ADM-applicable form:

$$\begin{array}{ll} \min & \langle I_n, X\rangle \\ s.t. & X - Y = 0 \\ & X \in S_+^n, Y \in \mathbb{U} := \{U \in \mathscr{R}^{m\times n} \mid \|U_\Omega - M_\Omega\|_F \leqslant \delta\}, \end{array} \tag{3.2}$$

we can analogously conclude that each iteration of the ADM in Glowinski (1984) for solving (3.2) requires solving the following subproblems:

$$\begin{cases} \langle Y - Y^{k+1}, Z^k - \beta(X^k - Y^{k+1})\rangle \geqslant 0, & \forall Y \in \mathbb{U}; \\ \langle X - X^{k+1}, I_n - Z^k + \beta(X^{k+1} - Y^{k+1})\rangle \geqslant 0, & \forall X \in S_+^n; \\ Z^{k+1} = Z^k - \gamma\beta(X^{k+1} - Y^{k+1}), \gamma \in (0, \frac{\sqrt{5}+1}{2}). \end{cases}$$

Then, based on the theory of VI in Eaves (1971), we can easily derive the following ADM algorithmic framework for solving PSDMCP.

Algorithm: ADM for positive semidefinite matrix completion problem (3.1).

Step 1. Solve Y^{k+1} via:

$$Y^{k+1} = \mathscr{P}_{\mathbb{U}}[X^k - \frac{1}{\beta}Z^k]\,. \tag{3.3}$$

Step 2. Find X^{k+1} via:

$$X^{k+1} = \mathscr{P}_{S_+^n}[Y^{k+1} + \frac{1}{\beta}(Z^k - I_n)]\,. \tag{3.4}$$

Step 3. Update the multiplier Z^{k+1} via:

$$Z^{k+1} = Z^k - \gamma\beta(X^{k+1} - Y^{k+1}), \quad \gamma \in (0, \frac{\sqrt{5}+1}{2}). \tag{3.5}$$

Obviously, the computation of each iteration of the ADM approach for (3.1) is dominated by the subproblem (3.4), which requires $O(n^3)$ flops to perform the eigenvalue decomposition of $Y^{k+1} + \frac{1}{\beta}(Z^k -$

I_n). In fact, for a given symmetric matrix A with rank r,

$$\mathscr{P}_{S^n_+}[A] = V\hat{\Sigma}V^T \text{ with } \quad \Sigma = \mathrm{diag}(\lambda_1, \ldots, \lambda_r) \tag{3.6}$$

where

$$\hat{\Sigma} = \mathrm{diag}(\{(\lambda_i)_+\}) \quad \text{and} \quad A = V\Sigma V^T$$

is the eigenvalue decomposition of A.

4. Numerical results

In this section, we implement the proposed ADM approach to solving the mentioned matrix completion models (1.1), (1.2) and (3.1) to verify the efficiency of ADM empirically. We also compare the ADM approach with some existing influential methods including the SVT in Cai *et al.* (2008), the FPCA in Ma *et al.* (2008), the APGL in Toh & Yun (2009) and the PPA in Liu *et al.* (2009). For the three PPA methods in Liu *et al.* (2009), we focus on the Dual PPA since it performs the best as reported in Liu *et al.* (2009). All the codes of the proposed ADM approach were written by MATLAB R2007b (Version 7.5.0). The MATLAB codes of the SVT, FPCA, APGL and PPA methods were downloaded from the homepages of the respective authors without any change. Thus, without otherwise specification, values of all the involved parameters of these methods are taken as the same values as in the corresponding original codes. All the codes were run on an 3.0GHZ Intel Pentium PC with 1.0GHZ of memory.

When reporting numerical results, we denote by SR the sampling ratio: $SR = p/mn$, where p is the cardinality of Ω. Let X^{opt} be the completed matrix of M obtained by an algorithm. Then the accuracy of the completion is measured by the relative error defined by:

$$\mathrm{REER} = \frac{\|X^{opt} - M\|_F}{\|M\|_F}. \tag{4.1}$$

We use TIME to denote the CPU computing time in seconds, and ITER to represent number of iterations.

4.1 *Some implementation issues*

We here list some details of implementing the proposed ADM approach.

4.1.1 *Stopping Criterion.*

In the implementation of the proposed ADM approach, the stopping criterion is

$$\frac{\|X^{k+1} - X^k\|_F}{\|X^k\|_F} < tol, \tag{4.2}$$

where tol is the given accuracy.

4.1.2 *Scaled models.*

Our numerical experiments indicate that the numerical performance is more stable if the models under consideration (1.1), (1.2) and (3.1) are scaled by a constant. Thus, for our numerical experiments, we attach the objection functions in these models by the scale constant 1.3.

4.1.3 *Parameter selection.* In the implementation of the proposed ADM approach, we set $\gamma = 1.6$ in the step of updating the Lagrange multipliers. For the parameter β, the convergence of ADM is guaranteed for any fixed value $\beta > 0$. Although some strategies allowing β to vary self-adaptively subject to certain criteria have been investigated intensively in the literature (e.g. He *et al.* (2002); Kontogiorgis & Meyer (1998); He & Yang (1998)), in this paper we restrict our discussion for fixed value of β in order to expose the main idea of the ADM approach clearly. We choose $\beta = 2.5/\sqrt{mn}$ for the proposed ADM approach throughout.

4.1.4 *PROPACK for SVD and eigenvalue decomposition.* As shown in Section 2, the SVD (2.7) (eigenvalues decomposition for PSDMCP) dominates the computation of each iteration of the proposed ADM approach. To perform (2.8), we actually only need to compute those singular values larger than $\frac{1}{\beta}$ and their corresponding singular vectors, rather than computing all singular values and singular vectors of $Y^k + \frac{1}{\beta}Z^k$ in (2.7). This enables us to perform the SVD in (2.7) partially, rather than fully.

Our interested methods for comparison, including the SVT, APGL and Dual PPA, all use the well-known PROPACK package (Larsen (2005)) to accomplish the partial SVD. For fair comparison with these methods, we also use PROPACK and exactly the same heuristic of determining the number of singular values and vectors at each iteration as that used in Cai *et al.* (2008); Toh & Yun (2009); Liu *et al.* (2009) to execute the partial SVD for the proposed ADM approach.

4.2 *Numerical results for noiseless matrix completion*

In this subsection, we apply the proposed ADM to solve some examples of the noiseless matrix completion problem (1.1). More specifically, the randomly generated case and the Jester joke problem (Goldberg *et al.* (2001)). Comparisons with some state-of-the-art methods are also reported.

4.2.1 *Randomly generated test problem.* We test the case of (1.1) where $m = n$, and the matrix M to be completed is generated by $M = M_L M_R^T$ where both M_L and M_R are two independently generated $n \times r$ matrices with i.i.d. Gaussian entries. Then, the rank of M is r.

In Table 1, we report the numerical results of ADM, APGL and SVT for different scenarios of (n, r, SR), and tol in (4.2) is set as $2e-4$.

Table 1. ADM, APGL and SVT for noiseless matrix completion (1.1)

M	ADM			APGL			SVT		
(n,r,SR)	TIME	ITER	REER	TIME	ITER	REER	TIME	ITER	REER
(500,10,0.25)	2.73	39	2.46e-4	4.28	34	3.58e-4	12.23	105	1.60e-4
(500,10,0.45)	2.59	24	1.67e-4	4.16	29	1.42e-4	8.94	66	1.25e-4
(500,10,0.65)	2.41	16	7.88e-5	4.36	28	1.31e-4	9.25	51	1.05e-4
(500,10,0.85)	1.56	10	7.94e-5	4.80	28	1.24e-4	7.69	40	1.06e-4
(500,20,0.25)	5.52	33	6.95e-4	7.73	45	9.11e-4	598.12	251	1.41e-4
(500,20,0.45)	3.89	22	1.88e-4	6.19	35	2.15e-4	29.48	90	1.41e-4
(500,20,0.65)	3.33	15	1.81e-4	6.00	31	1.78e-4	13.18	63	1.19e-4
(500,20,0.85)	2.77	11	9.41e-5	5.41	28	1.38e-4	10.75	47	1.12e-4
(1000,10,0.15)	8.31	65	5.58e-4	4.48	35	2.41e-4	34.94	103	1.61e-4
(1000,10,0.25)	7.11	44	3.96e-4	4.80	31	1.81e-4	21.36	68	1.24e-4
(1000,10,0.45)	5.63	24	3.91e-4	6.25	28	1.22e-4	24.07	50	1.35e-4
(1000,10,0.65)	5.50	16	1.27e-4	9.19	28	1.18e-4	26.61	42	9.66e-5
(1000,10,0.85)	3.77	10	2.58e-5	12.22	28	1.28e-4	23.66	35	9.87e-5
(1000,20,0.15)	11.41	53	6.84e-4	11.89	58	4.67e-4	321.24	181	1.59e-4
(1000,20,0.25)	10.02	39	2.83e-4	9.19	38	1.80e-4	57.71	93	1.49e-4
(1000,20,0.45)	8.86	24	1.82e-4	9.86	31	1.34e-4	32.43	60	1.30e-4
(1000,20,0.65)	9.59	16	8.29e-5	12.88	28	1.43e-4	33.21	48	1.07e-4
(1000,20,0.85)	6.91	10	6.10e-5	14.12	28	1.40e-4	33.50	39	1.00e-4
(2000,10,0.15)	20.72	71	9.38e-4	10.08	32	2.11e-4	44.16	66	1.25e-4
(2000,10,0.25)	25.67	48	4.77e-4	13.27	28	1.18e-4	61.62	51	1.06e-4
(2000,10,0.45)	22.19	27	3.41e-5	22.66	28	1.15e-4	67.45	40	1.10e-4
(2000,10,0.65)	22.02	18	6.91e-5	31.77	28	1.10e-4	77.28	35	1.01e-4
(2000,10,0.85)	14.83	11	3.69e-5	48.33	28	1.22e-4	88.50	31	8.76e-5
(2000,20,0.15)	32.75	65	6.45e-4	17.36	38	1.68e-4	104.31	86	1.46e-4
(2000,20,0.25)	33.55	44	4.80e-4	26.42	33	1.48e-4	84.79	63	1.22e-4
(2000,20,0.45)	29.81	24	4.22e-4	31.67	28	1.24e-4	101.20	47	1.13e-4
(2000,20,0.65)	31.03	16	1.20e-4	42.98	28	1.18e-4	116.18	40	1.00e-4
(2000,20,0.85)	25.11	11	2.55e-5	51.31	28	1.29e-4	108.18	34	1.02e-4

Table 1 shows that all the ADM, APGL and SVT methods are able to complete large-scale low-rank matrices with high accuracy from a small fraction of entries. For this test problem, ADM and APGL both perform much faster compared with SVT. For large-scale cases ($n \geqslant 1000$) with small values of SR (e.g. $SR < 0.3$), APGL outperforms ADM. For the cases with medium values of SR (e.g. $0.3 < SR < 0.6$), ADM and APGL perform likely. While for the cases with large values of SR (e.g. $SR > 0.6$), ADM is much faster than APGL.

In the following table, we compare the ADM with the FPCA and Dual PPA for solving the noiseless matrix completion problem (1.1) with higher accuracy, where tol in (4.2) is set as $2e-5$.

Table 2. ADM, FPCA and Dual PPA for noiseless matrix completion (1.1)

M	ADM			FPCA			Dual PPA		
(n, r, SR)	TIME	ITER	REER	TIME	ITER	REER	TIME	ITER	REER
(500,10,0.25)	3.20	50	3.66e-5	8.23	68	8.35e-5	7.02	34	2.21e-5
(500,10,0.45)	2.75	31	1.36e-5	19.00	63	5.39e-6	5.91	27	9.00e-6
(500,10,0.65)	2.25	20	6.88e-6	20.64	67	4.56e-6	5.48	23	3.34e-7
(500,10,0.85)	1.98	13	6.94e-6	36.31	118	4.81e-5	7.05	21	4.03e-7
(500,20,0.25)	6.36	44	1.36e-4	9.42	83	2.11e-4	11.23	44	2.51e-4
(500,20,0.45)	4.19	28	1.71e-5	19.48	67	1.65e-5	8.06	33	3.03e-5
(500,20,0.65)	3.72	19	1.95e-5	20.31	69	1.05e-5	7.83	28	6.13e-6
(500,20,0.85)	3.65	14	1.08e-5	82.66	287	4.88e-5	7.30	24	5.00e-6
(1000,10,0.15)	10.27	84	5.91e-5	16.73	72	2.54e-4	7.97	36	4.35e-5
(1000,10,0.25)	9.42	58	3.27e-5	30.80	63	4.49e-5	10.01	30	2.37e-6
(1000,10,0.45)	7.64	33	1.50e-5	129.83	61	7.29e-7	13.32	25	1.35e-6
(1000,10,0.65)	6.17	20	1.47e-5	135.30	63	7.48e-7	13.62	19	1.74e-5
(1000,10,0.85)	6.09	13	2.36e-6	188.42	81	2.27e-5	21.13	23	6.91e-8
(1000,20,0.15)	15.53	71	9.17e-5	24.61	101	3.95e-4	17.36	45	1.70e-4
(1000,20,0.25)	12.92	50	3.45e-5	32.77	67	8.35e-5	18.36	34	4.36e-5
(1000,20,0.45)	11.11	31	1.41e-5	142.05	67	2.12e-6	23.49	30	4.54e-6
(1000,20,0.65)	9.75	20	6.52e-6	146.20	68	2.30e-6	26.30	25	9.31e-6
(1000,20,0.85)	8.72	12	1.20e-5	235.98	103	4.18e-5	31.09	24	1.36e-6
(2000,10,0.15)	33.09	96	7.90e-5	88.92	66	1.46e-4	22.01	31	3.02e-6
(2000,10,0.25)	31.09	62	4.62e-5	232.92	67	1.67e-5	28.56	25	7.24e-7
(2000,10,0.45)	27.06	34	2.09e-5	1016.08	60	3.08e-7	51.31	25	2.30e-7
(2000,10,0.65)	23.69	22	6.84e-6	1048.30	62	2.73e-7	51.91	19	5.30e-6
(2000,10,0.85)	19.55	13	3.85e-6	1352.23	73	8.74e-6	61.58	18	7.38e-6
(2000,20,0.15)	46.39	89	3.31e-5	100.41	71	2.28e-4	43.44	60	2.14e-5
(2000,20,0.25)	43.27	58	3.88e-5	254.84	64	3.00e-5	44.58	26	8.99e-5
(2000,20,0.45)	38.73	33	1.67e-5	1031.58	61	6.33e-7	71.61	27	4.35e-6
(2000,20,0.65)	31.25	20	1.50e-5	1192.22	64	6.70e-7	97.17	26	3.44e-6
(2000,20,0.85)	30.36	13	2.43e-6	1494.03	80	1.86e-5	113.75	24	1.33e-6

Table 2 shows that the ADM, FPCA and Dual PPA are all capable of recovering the noiseless matrix accurately and efficiently. In terms of the completion speed, both ADM and Dual PPA outperform FPCA substantially. For the cases with small values of SR (e.g. $SR < 0.3$), ADM and Dual PPA perform likely. While for the cases with medium and large values of SR (e.g., $SR > 0.3$), ADM is faster than Dual PPA.

4.2.2 *Jester Joke Problem.* In this subsection, we implement the ADM, APGL and FPCA to solve the noiseless matrix completion problem (1.1) arising in the well-known Jester joke data set, which contains 4.1 millions of ratings for 100 jokes from 73,421 users. The data set is available on

http://www.ieor.berkeley.edu/∼goldberg/jester-data/,

and we refer to, e.g. Goldberg *et al.* (2001); Ma *et al.* (2008); Toh & Yun (2009), for the numerical study on this problem. In particular, the whole data set of the Jester joke problem consists of the following three categories:

(1) jester-1: Data from 24,983 users who have rated 36 or more jokes;
(2) jester-2: Data from 23,500 users who have rated 36 or more jokes;
(3) jester-3: Data from 24,938 users who have rated between 15 and 35 jokes.

For the sake of convenience, we use jester-all to denote the whole date set containing all of the above data sets.

Since the number of users is much larger than that of jokes, as in Ma *et al.* (2008), we select randomly parts of the users' ratings from the entire set. In particular, we denote by n_u the number of select users' ratings, and we test the cases that $n_u = 1000$ and 2000. Moreover, since some entries of M are not available, we are not able to compute the relative error defined by (4.1). To measure the accuracy of the completed matrix, as in Goldberg *et al.* (2001), we define the Mean Absolute Error (MAE) of the approximate matrix X by

$$MAE = \frac{\sum_{(i,j)\in\Omega} |X_{ij} - M_{ij}|}{|\Omega|},$$

where Ω is the sample set and $|\Omega|$ is the cardinality of Ω. Then, the Normalized Mean Absolute Error (abbreviated as NMAE) is used to measure the accuracy of the approximated completion X:

$$NMAE = \frac{MAE}{r_{max} - r_{min}},$$

where r_{min} and r_{max} are lower and upper bounds of the ratings. In the Jester joke problem, the range of rating is $[-10\ 10]$. Hence, $r_{max} - r_{min} = 20$.

To implement the proposed ADM approach (2.3)-(2.5), we take (0,0,0) as the initial iterate and $\beta = 0.25/\sqrt{n_u}$. In the following table, C_s and ε_s are parameters for determining numbers of singular values to be computed in the involved approximate SVD subroutine of FPCA, see Ma *et al.* (2008) for details. Considering that the ranks of matrices arising in the Jester joke problem are usually pretty high, we set the values of C_s and ε_s differently from the "FPCA_MRM_pub" package in Ma *et al.* (2008). Throughout private communication, the first author of Ma *et al.* (2008) suggested us to select $\varepsilon_s = 1e-4$, $C_s = 100$ ($n_u = 1000$) and $C_s = 200(n_u = 2000)$. In the following table, σ_{max} and σ_{min} denote the largest and smallest positive singular values of the recovered matrices, respectively; and Rank denotes the rank of the recovered matrices.

Table 3. ADM, FPCA and APGL for Jester Joke Problem

Examples	n_u	SR	Algorithm	C_s	ε_s	TIME	NMAE	Rank	σ_{min}	σ_{max}
jester-1	1000	0.6927	ADM			6.17	0.163	100	36.74	784.99
jester-1	1000	0.6927	APGL			9.72	0.212	100	21.81	818.98
jester-1	1000	0.6927	FPCA	100	1e-4	477.95	0.199	85	45.22	864.16
jester-1	2000	0.7029	ADM			10.14	0.164	100	60.79	1149.16
jester-1	2000	0.7029	APGL			15.36	0.216	100	43.02	1199.32
jester-1	2000	0.7029	FPCA	200	1e-4	782.36	0.163	100	59.05	1176.44
jester-2	1000	0.6958	ADM			6.03	0.163	100	35.33	804.54
jester-2	1000	0.6958	APGL			9.53	0.213	100	20.62	841.40
jester-2	1000	0.6958	FPCA	100	1e-4	469.42	0.197	83	48.35	891.15
jester-2	2000	0.7100	ADM			10.67	0.161	100	66.00	1180.70
jester-2	2000	0.7100	APGL			15.81	0.218	100	45.04	1229.80
jester-2	2000	0.7100	FPCA	200	1e-4	700.88	0.161	100	63.51	1219.34
jester-3	1000	0.2288	ADM			8.66	0.188	54	14.36	578.50
jester-3	1000	0.2288	APGL			4.72	0.191	88	73.39	589.06
jester-3	1000	0.2288	FPCA	100	1e-4	147.08	0.191	37	56.64	641.28
jester-3	2000	0.2265	ADM			16.78	0.189	62	0.93	836.35
jester-3	2000	0.2265	APGL			7.86	0.190	88	0.33	861.19
jester-3	2000	0.2256	FPCA	200	1e-4	514.23	0.190	42	86.72	889.99
jester-all	1000	0.5186	ADM			6.53	0.175	100	21.79	727.43
jester-all	1000	0.5186	APGL			8.70	0.199	100	11.51	774.61
jester-all	1000	0.5186	FPCA	100	1e-4	430.63	0.226	71	46.42	946.09
jester-all	2000	0.5374	ADM			10.13	0.174	100	52.92	1047.41
jester-all	2000	0.5374	APGL			14.30	0.208	100	29.33	1119.63
jester-all	2000	0.5374	FPCA	200	1e-4	1312.50	0.248	89	46.54	1253.50

Table 3 shows that both ADM and APGL complete the matrix arising from Jester joke data set much faster than FPCA. For all the tested cases except for "jester-3", ADM is faster than APGL. In addition, among these three methods, ADM can obtain solutions with highest accuracy for almost all the tested cases. For the cases "jester-1" and "jester-2", solutions obtained by APGL are of the lowest accuracy. Thus, the promising performance of ADM is well illustrated in Table 3.

4.3 *Numerical results for noisy matrix completion*

In this section, we implement the proposed ADM to solve the noisy matrix completion problem (1.2), and compare it with the APGL, FPCA, SVT and Dual PPA.

The matrix M to be completed is generated by $M = MR + \sigma\Sigma$, where $MR = M_L M_R^T$ with M_L and M_R being two independently generated $n \times r$ matrices with i.i.d. Gaussian entries; the noise matrix Σ is a $n \times n$ matrix with i.i.d. Gaussian entries and $\sigma > 0$ is a constant. For our numerical experiments, we take

$$\sigma = 10^{-2} \text{ and } \delta = \frac{1}{10} \cdot \|(\sigma\Sigma)_\Omega\|_F$$

where the choice of δ comes from experience.

To apply the proposed ADM (2.3)-(2.5), we take (0,0,0) as the initial iterate. In the following table, we report numerical performance of ADM, APGL, PFCA, SVT and Dual PPA for solving different scenarios of (1.2), where tol in (4.2) is set as $2e-4$.

Table 4. ADM, APGL, FPCA, SVT and Dual PPA for (1.2) with $\sigma = 10^{-2}$

M	ADM			APGL			FPCA			SVT			Dual PPA		
(n,r,SR)	TIME	ITER	REER	TIME	ITER	REER	TIME	ITER	REER	TIME	ITER	REER	TIME	ITER	REER
(500,10,0.25)	4.00	45	1.41e-3	5.45	41	1.40e-3	49.08	600	1.69e-3	7.46	63	2.37e-3	6.41	33	1.38e-3
(500,10,0.45)	2.64	27	9.95e-4	4.89	30	9.83e-4	161.92	600	1.35e-3	5.94	43	1.54e-3	6.67	27	9.83e-4
(500,10,0.65)	2.34	17	8.80e-4	4.69	28	8.19e-4	165.47	610	1.44e-3	6.41	35	1.16e-3	6.02	24	7.73e-4
(500,10,0.85)	2.14	11	7.55e-4	5.95	28	6.97e-4	168.72	610	2.05e-3	6.87	29	9.45e-4	8.45	23	6.99e-4
(500,20,0.25)	5.30	41	3.86e-3	7.63	46	1.78e-3	46.69	600	1.85e-3	265.36	155	2.29e-3	12.89	47	1.46e-3
(500,20,0.45)	4.11	24	1.01e-3	6.11	34	1.04e-3	160.72	600	1.31e-3	19.16	59	1.62e-3	8.70	32	9.92e-4
(500,20,0.65)	3.59	16	8.68e-4	6.33	31	8.05e-4	169.52	610	1.45e-3	9.20	43	1.24e-3	8.16	21	6.30e-4
(500,20,0.85)	3.44	11	7.17e-4	6.05	28	6.96e-4	178.70	610	2.42e-3	8.53	34	9.66e-4	9.05	27	6.95e-4
(1000,10,0.15)	10.08	74	1.30e-3	5.36	35	1.27e-3	115.56	600	1.57e-3	20.19	62	2.17e-3	8.17	31	1.22e-3
(1000,10,0.25)	8.86	45	1.04e-3	5.97	31	9.39e-4	282.89	600	1.18e-3	14.55	45	1.48e-3	11.73	29	9.17e-4
(1000,10,0.45)	7.17	27	8.73e-4	7.48	28	6.93e-4	1323.41	610	9.23e-4	17.69	36	9.97e-4	15.45	24	6.76e-4
(1000,10,0.65)	6.63	17	6.04e-4	10.73	28	5.58e-4	1338.23	610	9.70e-4	19.93	31	7.87e-4	19.55	23	5.63e-4
(1000,10,0.85)	5.78	11	5.88e-4	14.86	28	5.07e-4	1426.72	610	1.30e-3	19.90	17	6.51e-4	25.62	24	4.90e-4
(1000,20,0.15)	12.69	64	2.72e-3	11.05	54	1.38e-3	119.91	600	1.64e-3	122.52	110	2.22e-3	19.11	49	1.35e-3
(1000,20,0.25)	12.33	44	9.88e-4	9.13	38	9.83e-4	276.00	600	1.19e-3	36.41	62	1.59e-3	18.05	33	9.63e-4
(1000,20,0.45)	11.05	27	7.52e-4	12.33	31	6.87e-4	1301.33	610	8.97e-4	23.18	43	1.06e-3	27.20	30	6.80e-4
(1000,20,0.65)	9.27	17	5.99e-4	12.48	28	5.72e-4	1315.59	610	9.59e-4	25.15	36	8.05e-4	31.45	28	5.64e-4
(1000,20,0.85)	8.72	11	5.30e-4	16.48	28	5.02e-4	1395.55	610	1.46e-3	27.76	30	6.72e-4	42.58	29	4.96e-4
(2000,10,0.15)	24.66	75	1.15e-3	12.14	30	8.85e-4	672.73	600	1.06e-3	30.28	45	1.31e-3	25.42	28	9.41e-3
(2000,10,0.25)	29.50	50	1.07e-3	16.09	28	6.72e-4	2284.70	610	8.29e-4	42.36	37	3.15e-3	35.84	25	6.46e-4
(2000,10,0.45)	24.13	27	7.93e-4	25.91	28	4.90e-4	10719.31	610	6.38e-4	52.76	31	6.64e-4	55.36	24	4.82e-4
(2000,10,0.65)	22.72	17	4.42e-4	38.30	28	4.06e-4	10718.30	610	6.54e-4	61.22	28	5.34e-4	80.05	24	3.97e-4
(2000,10,0.85)	14.94	9	3.47e-4	46.31	28	3.63e-4	11588.77	620	9.15e-4	78.39	25	4.50e-4	105.88	24	3.52e-4
(2000,20,0.15)	40.23	74	9.27e-4	17.88	38	8.88e-4	687.09	600	1.07e-3	71.16	59	1.42e-3	47.27	40	8.72e-4
(2000,20,0.25)	36.83	45	8.56e-4	25.36	34	6.62e-4	2226.44	610	8.45e-4	61.99	46	9.82e-4	59.27	32	6.58e-4
(2000,20,0.45)	35.88	27	6.65e-4	31.48	28	4.92e-4	10446.23	610	6.40e-4	79.40	37	6.73e-4	95.75	32	4.88e-4
(2000,20,0.65)	30.95	17	4.31e-4	40.73	28	4.15e-4	10653.56	610	6.84e-4	84.74	32	5.53e-4	126.31	30	4.05e-4
(2000,20,0.85)	29.23	11	4.22e-4	55.00	28	3.69e-4	11473.95	620	9.51e-4	97.49	28	4.70e-4	162.44	29	3.55e-4

According to the comparison in Table 4, ADM, APGL and Dual PPA are faster than FPCA and SVT, and the first three methods are capable of obtaining solutions with higher accuracy. Similar as the noiseless matrix completion problem (1.1), APGL performs the best for the case where n is large ($n \geqslant 1000$) and the value of SR is small (e.g., $SR < 0.3$). For these cases, both ADM and Dual PPA are also very competitive. For the cases with medium values of SR (e.g., $0.3 < SR < 0.6$), ADM and APGL perform likely and they outstand the others obviously. For the cases with large values of SR (e.g., $SR > 0.6$), the superiority of ADM is very evident.

4.4 *Numerical results for PSDMCP*

In this subsection, we implement the proposed the ADM approach (3.3)-(3.5) to solve the PSDMCP (3.1) with/without noise.

As in Section 4.3, the matrix M to be completed in (3.1) is generated by $M = MR + \sigma\Sigma$, where $MR = M_L M_R^T$ with M_L and M_R being two independently generated $n \times r$ matrices with i.i.d. Gaussian entries; and the noise matrix Σ is a $n \times n$ matrix with i.i.d. Gaussian entries and $\sigma > 0$ is a constant. In (1.2), we take

$$\delta = \frac{1}{10} \cdot \|(\sigma\Sigma)_\Omega\|_F.$$

When implementing the proposed ADM approach (3.3)-(3.5), we take $(0,0,0)$ as the initial iterate and use the PROPACK to realize partial eigenvalue decomposition. In the following table, we report the numerical performance of the proposed ADM approach for solving the noiseless case $\sigma = 0$ and the noise case $\sigma = 10^{-2}$ of PSDMCP (3.1), where tol in (4.2) is set as $2e-4$.

Table 5. ADM for (3.1) with $\sigma = 0$ and $\sigma = 10^{-2}$

M	$\sigma = 0$			$\sigma = 10^{-2}$		
(n,r,SR)	TIME	ITER	REER	TIME	ITER	REER
(1000,10,0.25)	15.20	44	1.07e-3	16.61	44	2.49e-3
(1000,10,0.45)	11.03	27	3.84e-4	11.34	27	1.96e-3
(1000,10,0.65)	6.72	16	3.12e-4	6.8	16	1.63e-3
(1000,20,0.25)	19.14	40	9.44e-4	20.88	40	3.72e-3
(1000,20,0.45)	12.30	24	6.43e-4	13.44	24	2.35e-3
(1000,20,0.65)	8.52	16	3.45e-4	8.88	16	2.19e-3
(2000,10,0.25)	65.20	48	1.32e-3	65.32	48	2.70e-3
(2000,10,0.45)	41.05	27	6.88e-4	41.66	27	1.89e-3
(2000,10,0.65)	29.42	18	1.95e-4	28.66	18	1.20e-3
(2000,20,0.25)	74.3	44	1.75e-3	80.31	44	2.82e-3
(2000,20,0.45)	48.3	27	5.73e-4	49.41	27	2.15e-3
(2000,20,0.65)	30.06	16	4.66e-4	30.36	16	1.78e-3

The data in Table 5 indicates that the proposed ADM is also very efficient for the PSDMCP. In addition, the iterative time required by the noiseless case and the noise case differs slightly, showing that the proposed ADM is pretty robust for the corruption of noise on sample data.

5. Discussions

In this section, we show that the proposed ADM approach can be extended to some more general problems which are closely related to the matrix completion problem. Thus, the applicable range of the ADM approach is significantly broadened.

5.1 *Nuclear-norm-regularized least squares problem*

By introducing the auxiliary variable Y, we can easily reformulate the nuclear-norm-regularized least squares problem (1.3) into the following ADM-applicable form:

$$\begin{array}{ll} \min & \|X\|_* + \frac{\mu}{2}\|(Y-M)_\Omega\|_F^2 \\ s.t. & X - Y = 0 \\ & X \in \mathscr{R}^{m\times n},\ Y \in \mathscr{R}^{m\times n}, \end{array} \tag{5.1}$$

Based on the analogous analysis for the matrix completion problem, we can derive the following ADM algorithmic framework for solving (5.1).

Algorithm: ADM for the nuclear-norm-regularized least squares problem(1.3).

Step 1. Find X^{k+1} via:

$$0 \in \partial(\|X^{k+1}\|_*) - [Z^k - \beta(X^{k+1} - Y^k)]. \tag{5.2}$$

Step 2 Solve Y^{k+1} via:

$$Y^{k+1} = X^{k+1} - \frac{1}{\beta}Z^k + \frac{\mu}{\mu+\beta}(M - (X^{k+1} - \frac{1}{\beta}Z^k))_\Omega; \tag{5.3}$$

Step 3. Update the multiplier Z^{k+1} via:

$$Z^{k+1} = Z^k - \gamma\beta(X^{k+1} - Y^{k+1}), \quad \gamma \in (0, \frac{\sqrt{5}+1}{2}). \tag{5.4}$$

Now, we report some preliminary numerical results for the implementation of the proposed ADM approach (5.2)-(5.4) for solving (1.3). We generate the matrix M to be completed exactly as the way in Section 4.3, and we take $\mu = 10^4$ in (1.3). When implementing the proposed ADM, we take $(0,0,0)$ as the initial iterate. In the following table, we report some scenarios for $\sigma = 0$ and $\sigma = 10^{-2}$ where *tol* in (4.2) is set as $2e-4$..

Table 6. ADM for (1.3) with $\sigma = 10^{-2}$

M	$\sigma = 0$			$\sigma = 10^{-2}$		
(n,r,SR)	TIME	ITER	REER	TIME	ITER	REER
(1000,10,0.25)	8.00	45	7.35e-4	7.75	45	1.17e-3
(1000,10,0.45)	6.81	27	2.27e-4	6.84	27	7.04e-4
(1000,10,0.65)	5.91	17	7.76e-5	5.92	17	5.68e-4
(1000,20,0.25)	11.77	45	3.09e-4	11.77	45	1.00e-3
(1000,20,0.45)	10.22	27	1.21e-4	10.56	27	6.97e-4
(1000,20,0.65)	8.56	17	4.89e-5	8.77	17	5.66e-4
(2000,10,0.25)	28.20	50	5.29e-4	28.09	50	8.27e-4
(2000,10,0.45)	24.25	27	3.16e-4	24.27	27	5.64e-4
(2000,10,0.65)	20.31	17	1.15e-4	20.33	17	4.11e-4
(2000,20,0.25)	36.72	45	7.98e-4	36.52	45	1.03e-3
(2000,20,0.45)	35.08	27	2.05e-4	34.77	27	4.34e-4
(2000,20,0.65)	29.56	17	8.22e-5	29.83	17	4.00e-4

The data in Table 6 shows the efficiency and effectiveness of the proposed ADM approach for solving the nuclear-norm-regularized least squares problem (1.3). Also, the proposed ADM is pretty robust for the corruption of noise on sample data since the iterative time required by the noiseless case and the noise case differs slightly.

5.2 *Nuclear-norm minimization problem*

It is easy to notice that the models (1.1) and (1.2) are both special cases of the following nuclear norm minimization problem:

$$\begin{array}{ll} \min & \|X\|_* \\ s.t. & \|\mathscr{A}(X)-b\|_F \leqslant \delta, \\ & X \in \mathscr{R}^{m\times n}, \end{array} \tag{5.5}$$

where $\mathscr{A}:\mathscr{R}^{m\times n}\to\mathscr{R}^m$ is a linear operator and $b\in\mathscr{R}^m$. We refer to, e.g. Fazel (2002); Recht *et al.* (2007), for various applications captured by the general model (5.5).

As what we have done for the matrix completion problem, we can readily derive the ADM-oriented reformulation of (5.5) in the following form:

$$\begin{array}{ll} \min & \|X\|_* \\ s.t. & \mathscr{A}(X)-y=b, \\ & X \in \mathscr{R}^{m\times n},\ y\in\mathscr{B}_\delta := \{y\in\mathscr{R}^m \,|\, \|y\|\leqslant\delta\}, \end{array} \tag{5.6}$$

By attaching the Lagrangian multiplier $\lambda^k\in\mathscr{R}^m$ to the linear constraint of (5.6), the ADM algorithmic framework for solving (5.6) is as follows.

Algorithm: ADM for nuclear-norm minimization problem (5.5).

Step 1 Solve Y^{k+1} via:

$$\langle y-y^{k+1},\lambda^k-\beta(\mathscr{A}(X^k)-y^{k+1}-b)\rangle \geqslant 0, \quad \forall y\in\mathscr{B}_\delta; \tag{5.7}$$

Step 2. Find X^{k+1} via:

$$0\in\partial(\|X^{k+1}\|_*)-\mathscr{A}^*(\lambda^k-\beta(\mathscr{A}(X^{k+1})-y^{k+1}-b)), \tag{5.8}$$

where $\mathscr{A}^*$ is the adjoint operator of $\mathscr{A}$.

Step 3. Update the multiplier Z^{k+1} via:

$$\lambda^{k+1}=\lambda^k-\gamma\beta(\mathscr{A}(X^{k+1})-y^{k+1}-b), \quad \gamma\in(0,\frac{\sqrt{5}+1}{2}). \tag{5.9}$$

Compared to the models of the matrix completion problem, the main difficulty of the application of ADM for (5.5) is that the subproblem (5.8) is not eligible for explicit solution any longer. The well-studied Linearized technique (e.g. Osher *et al.* (2008)), inexact ADM technique (e.g. Yang & Zhang (2009) for various interesting l_1 problems in compressive sensing) and the classical proximal point regularization (e.g. Martinet (1970); Rockafellar (1976)) are expected to tackle this difficulty well. Due to this nontrivial treatment and the principle aim of exposing the applicability of ADM to the matrix completion problem in this paper, we would address the application of ADM for (5.5) in another paper.

6. Conclusions

This paper shows that the alternating direction method (ADM) is an easy and efficient approach to accomplishing the task of completing a low-rank matrix. The ADM approach is easily implementable because that all the resulted subproblems have closed-form solutions, and efficient because that even some large-scale matrices can be completed accurately and fast. For many cases, the ADM approach outperforms substantially some state-of-the-art algorithms in the literature.

Funding

The second author was support in part by NSFC grant 10971095 and the NSF of Province Jiangsu grant BK2008255; and the third author was supported in part by RGC grant 203009 and NSFC grant 10701055.

REFERENCES

ABERNETHY, F., EVGENIOU, T. & VERT, J. P. (2006) Low-rank matrix factorization with attributes. *Technical Report N24/06/MM*. Ecole des Mines de Paris: Ecole des Mines de Paris.

ABERNETHY, F., EVGENIOU, T. & VERT, J. P. (2009) A new approach to collaborative filtering:operator estimation with spectral regularization. *Journal of Machine Learning Research*, **10**, 803–826.

AMIT, Y., FINK, M., SREBRO, N. & ULLMAN, S. (2007) Uncovering shared structures in multiclass classification. *Proceeding of the International Conference on Machine Learning*. New York: ACM, pp. 17–24.

ARGYRIOU, A., EVGENIOU, T. & PONTIL, M. (2008) Convex multi-task feature learning. *Machine Learning*, **73(3)**, 243–272.

BERTSEKAS, D. P. (1982) *Constrained Optimization and Lagrange Multiplier methods*. New York: Academic Press.

BERTSEKAS, D. P. & TSITSIKLIS, J. N. (1989) *Parallel and distributed computation: Numerical methods*. Englewood Cliffs, N.J.: Prentice Hall.

CAI, J. F., CANDÈS, E. J. & SHEN, Z. W. (2008) A singular value thresholding algorithm for matrix completion. *manuscript*.

CANDÈS, E. J. & PLAN, Y. (2009) Matrix completion with noise. *manuscript*.

CANDÈS, E. J. & RECHT, B. (2008) Exact matrix completion via convex optimization. *submitted*.

CANDÈS, E. J. & TAO, T. (2009) The power of convex relaxation: Near-optimial matrix completion. *submitted*.

CHEN, G. & TEBOULLE, M. (1994) A proximal-based decomposition method for convex minimization problems. *Math. Programming*, **64**, 81–101.

CHEN, P. & SUTER, D. (2004) Recovering the missing components in a large noisy low-rank matrix: application to sfm source. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26(8)**, 1051–1063.

EAVES, B. C. (1971) On the basic theorem of complementarity. *Math. Programming*, **1**, 68–75.

ECKSTEIN, J. (1994) Some saddle-function splitting methods for convex programming. *Optim. Methods Softw.*, **4**, 75–83.

ECKSTEIN, J. & FUKUSHIMA, M. (1994) Some reformulations and applications of the alternating direction method of multipliers. *Large scale optimization (Gainesville, FL, 1993)*. Dordrecht: Kluwer Acad. Publ., pp. 115–134.

ESSER, E. (2009) Applications of lagrangian-based alternating direction methods and connections to split bregman. *Manuscript*. available on http://www.math.ucla.edu/applied/cam/.

FAZEL, M., HINDI, H. & BOYD, S. (2001) A rank minimization heuristic with application to minimum order system approximation. *Proceedings American Control Conference*. *Proceedings American Control Conference*., pp. 4734–4739.

FAZEL, M. (2002) Matrix rank minimization with applications. *Ph.D. thesis*, Stanford University, USA.

FUKUSHIMA, M. (1992) Application of the alternating direction method of multipliers to separable convex programming problems. *Comput. Optim. Appl.*, **1**, 93–111.

GABAY, D. (1983) Application of the method of multipliers to varuational inequalities. *Augmented Lagrangian methods: Application to the numerical solution of Boundary-Value Problem* (M. Fortin & R. Glowinski eds). Amsterdam: North-holland, pp. 299–331.

GABAY, D. & MERCIER, B. (1976) A dual algorithm for the solution of nonlinear variational problems via finite element approximations. *Computational Mathematics with Applications*, **2**, 17–40.

GLOWINSKI, R. (1984) *Numerical methods for nonlinear variational problems*. New York: Springer-Verlag.

GLOWINSKI, R. & LE TALLEC, P. (1989) *Augmented Lagrangian and Operator Splitting Methods in Nonlinear Mechanics*. SIAM Studies in Applied Mathematics, vol. 9. PA: Philadelphia.

GOLDBERG, K., ROEDER, T., GUPTA, D. & PERKINS, C. (2001) Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, **4**, 133–151.

GOLUB, G. H. & VAN LOAN, C. F. (1996) *Matrix Computation(the third edition)*. Baltimore: The Johns Hopkins University Press.

HE, B., LIAO, L.-Z., HAN, D. & YANG, H. (2002) A new inexact alternating directions method for monotone variational inequalities. *Math. Program.*, **92**, 103–118.

HE, B. S., XU, M. H. & M., Y. X. (2009) Solving large-scale least squares covariance matrix problems by alternating direction methods. *Submitted*.

HE, B. & YANG, H. (1998) Some convergence properties of a method of multipliers for linearly constrained monotone variational inequalities. *Oper. Res. Lett.*, **23**, 151–161.

HIGHAM, N. J. (1988) Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra Appl.*, **103**, 103–118.

JAMES, B. & STAN, L. (2007) The netflix prize. *Proceedings of KDD Cup and Workshop*. available online at http://www.cs.uic.edu/ liub/KDD-cup-2007/proceedings.html:

KESHAVAN, R. H., MONTANARI, A. & OH, S. (2009) Matrix completion from noisy entries. *manuscript*.

KONTOGIORGIS, S. & MEYER, R. R. (1998) A variable-penalty alternating directions method for convex optimization. *Math. Programming*, **83**, 29–53.

LARSEN, R. M. (2005) Propack–software for large and sparse svd calculations. *available on http://sun.stanford.edu/ rmunk/PROPACK/*.

LIN, Z. C., CHEN, M. M., WU, L. Q. & Y., M. (2009) The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *manuscript*.

LIU, Y. J., SUN, D. F. & TOH, K. C. (2009) An implementable proximal point algorithmic framework for nuclear norm minimization. *manuscript*.

LIU, Z. & VANDENBERGHE, L. (2008) Interior-point method for nuclear norm approximation with application to system identification. *manuscript*.

MA, S. Q., GOLDFARB, D. & CHEN, L. F. (2008) Fixed point and bregman iterative mehods for matrix rank minization. *Mathematical Programming*. to appear.

MARTINET, B. (1970) Régularisation d'inéquations variationnelles par approximations successives. *Rev. Française Informat. Recherche Opérationnelle*, **4**, 154–158.

NETFLIX (2006) Netflix prize. *available online at http://www.netflixprize.com/*.

NG, M., WEISS, P. A. & YUAN, X. M. (2009) Solving constrained total-variation problems via alternating direction methods. *Submitted*.

NOCEDAL, J. & WRIGHT, S. J. (2006) *Numerical Optimization*. New York: Spriger Verlag.

OSHER, S., MAO, Y., DONG, B. & YIN, W. (2008) Fast linearized bregman iteration for compressive sensing and sparse denoising. *Communications in Mathematical Sciences*.

PONG, T. K., TSENG, P., JI, S. & YE, J. (2009) Trace norm regularization: Reformulations, algorithms, and multi-task learning. *available on http://www.math.washington.edu/ tseng/papers.html*.

RECHT, B., FAZEL, M. & PARRILO, P. A. (2007) Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*. to appear.

RECHT, B., XU, W. & HASSIBI, B. (2009) Null space conditions and thresholds for rank minimization. *available on http://pages.cs.wisc.edu/ brecht/publications.html*.

ROCKAFELLAR, R. T. (1976) Monotone operators and the proximal point algorithm. *SIAM J. Control Optimization*, **14**, 877–898.

SCHMIDT, R. O. (1986) Multiple emitter location and singal parameter estimation. *IEEE Trans. Ant. and Proop.*, **34(3)**, 276–280.

TOH, K. C. & YUN, S. W. (2009) An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *available on http://www.math.nus.edu.sg/ mattohkc/publist.html*.

TOMASI, C. & KANADE, T. (1992) Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, **9(2)**, 137–154.

TSENG, P. (1997) Alternating projection-proximal methods for convex programming and variational inequalities. *SIAM J. Optim.*, **7**, 951–965.

WEN, Z. W., GOLDFARB, D. & YIN, W. (2009) Alternating direciotn augmented lagrangina methods for semidefinite porgramming. *manuscript*.

YANG, J. & ZHANG, Y. (2009) Alternating direction algorithms for l1-problems in compressive sensing. *TR09-37,CAAM, Rice University*. Houston: Houston.

YE, C. H. & YUAN, X. M. (2007) A descent method for stuctured monotone variational inequalities. *Optimization Methods and Software*, **22**, 329–338.

YUAN, X. M. (2009) Alternating direction direction methods for sparse covariance selection. *submitted*.

YUAN, X. M. & YANG, J. (2009) Sparese and low-rank matrix decomposition via alternating direction methods. *manuscript*.

ZHU, Z. S., SO, A. M. C. & YE, Y. Y. (2009) Fast and near-optimal matrix completion via randomized basis pursuit. *manuscript*.