Rank-One Matrix Pursuit for Matrix Completion

Zheng Wang[*] ZHENGWANG@ASU.EDU
Ming-Jun Lai[†] MJLAI@MATH.UGA.EDU
Zhaosong Lu[‡] ZHAOSONG@SFU.CA
Wei Fan[§] DAVID.FANWEI@HUAWEI.COM
Hasan Davulcu[¶] HASANDAVULCU@ASU.EDU
Jieping Ye[*¶] JIEPING.YE@ASU.EDU

[*]The Biodesign Institue, Arizona State University, Tempe, AZ 85287, USA
[†]Department of Mathematics, University of Georgia, Athens, GA 30602, USA
[‡]Department of Mathematics, Simon Fraser University, Burnaby, BC, V5A 156, Canada
[§]Huawei Noah's Ark Lab, Hong Kong Science Park, Shatin, Hong Kong
[¶]School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85281, USA

Abstract

Low rank matrix completion has been applied successfully in a wide range of machine learning applications, such as collaborative filtering, image inpainting and Microarray data imputation. However, many existing algorithms are not scalable to large-scale problems, as they involve computing singular value decomposition. In this paper, we present an efficient and scalable algorithm for matrix completion. The key idea is to extend the well-known orthogonal matching pursuit from the vector case to the matrix case. In each iteration, we pursue a rank-one matrix basis generated by the top singular vector pair of the current approximation residual and update the weights for all rank-one matrices obtained up to the current iteration. We further propose a novel weight updating rule to reduce the time and storage complexity, making the proposed algorithm scalable to large matrices. We establish the linear convergence of the proposed algorithm. The fast convergence is achieved due to the proposed construction of matrix bases and the estimation of the weights. We empirically evaluate the proposed algorithm on many real-world large-scale datasets. Results show that our algorithm is much more efficient than state-of-the-art matrix completion algorithms while achieving similar or better prediction performance.

Proceedings of the 31^{st} International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

1. Introduction

Low rank matrix learning has attracted significant attention in the machine learning community due to its wide range of applications, such as collaborative filtering (Koren et al., 2009; Srebro et al., 2005), compressed sensing (Candès & Recht, 2009), multi-class learning and multi-task learning (Argyriou et al., 2008; Negahban & Wainwright, 2010; Dudík et al., 2012). In this paper, we consider the general form of low rank matrix completion: given a partially observed real-valued matrix $\mathbf{Y} \in \Re^{n\times m}$, the low rank matrix completion problem is to find a matrix $\mathbf{X} \in \Re^{n\times m}$ with minimum rank such that $P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{Y})$, where Ω includes the index pairs (i, j) of all observed entries, and P_Ω is the orthogonal projector onto the span of matrices vanishing outside of Ω. As it is intractable to minimize the matrix rank exactly in the general case, the trace norm or nuclear norm is widely used as a convex relaxation of the matrix rank (Candès & Recht, 2009). It is defined by the Schatten p-norm with $p = 1$. For matrix $\mathbf{X}$ with rank r, its Schatten p-norm is defined by $(\sum_{i=1}^{r} \sigma_i^p)^{1/p}$, where $\{\sigma_i\}$ are the singular values of $\mathbf{X}$. Thus, the trace norm of $\mathbf{X}$ is the ℓ_1 norm of the matrix spectrum as $||\mathbf{X}||_* = \sum_{i=1}^{r} |\sigma_i|$.

Solving the standard low rank or trace norm problem is computationally expensive for large matrices, as it involves computing singular value decomposition (SVD). How to solve these problems efficiently and accurately has attracted much attention in recent years (Avron et al., 2012; Srebro et al., 2005; Cai et al., 2010; Balzano et al., 2010; Keshavan & Oh, 2009; Toh & Yun, 2010; Ji & Ye, 2009; Ma et al., 2011; Mazumder et al., 2010; Mishra et al., 2011; Wen et al., 2010; Lee & Bresler, 2010; Recht & Ré, 2013). Most of these methods still involve the computation of SVD or truncated SVD iteratively, which is

not scalable to large-scale problems (Cai et al., 2010; Keshavan & Oh, 2009; Toh & Yun, 2010; Ma et al., 2011; Mazumder et al., 2010; Lee & Bresler, 2010). Several methods approximate the trace norm using its variational characterizations (Mishra et al., 2011; Srebro et al., 2005; Wen et al., 2010; Recht & Ré, 2013), and proceed by alternating optimization. The linear convergence rate is established theoretically for properly designed alternating optimization algorithm under appropriate initialization (Jain et al., 2013). However, its computational complexity depends on the square of the rank of the estimated matrix. Thus in practical problems, especially for large matrices, it requires the rank of the estimated matrix to be very small, which sacrifices the estimation accuracy.

Recently, the coordinate gradient descent method has been demonstrated to be efficient in solving sparse learning problems in the vector case (Friedman et al., 2010; Shalev-Shwartz & Tewari, 2009). The key idea is to solve a very simple one-dimensional problem (for one coordinate) in each iteration. One natural question is whether and how such method can be applied to solve the matrix completion problem. Some progress has been made recently along this direction (Jaggi & Sulovský, 2010; Tewari et al., 2011; Shalev-Shwartz et al., 2011; Dudík et al., 2012; Zhang et al., 2012). These algorithms proceed in two main steps in each iteration. The first step involves computing the top singular vector pair, and the second step refines the weights of the rank-one matrices formed by all top singular vector pairs obtained up to the current iteration. The main differences among these algorithms lie in how they refine the weights. The Jaggi's algorithm (JS) (Jaggi & Sulovský, 2010) directly applies the Hazan's algorithm, which adapts the Frank-Wolfe algorithm to the matrix case (Hazan, 2008). It updates the weights with a small step size and does not consider further refinement. It does not use all information in each step, which leads to a slow convergence rate. Similar to JS, Tewari et al. (Tewari et al., 2011) use a small update step size for a general structure constrained problem. A more efficient Frank-Wolfe type algorithm is to fully refine the weights, which is claimed to be equivalent to orthogonal matching pursuit (OMP) in a wide range of l_1 ball constrained convex optimization problems (Jaggi, 2013). The greedy efficient component optimization (GECO) (Shalev-Shwartz et al., 2011) applies a similar approach, which optimizes the weights by solving another time consuming optimization problem. It empirically reduces the number of iterations without theoretical guarantees. However, the sophisticated weight refinement leads to a higher total computational cost. The lifted coordinate gradient descent algorithm (Lifted) (Dudík et al., 2012) updates the rank-one matrix basis with a constant weight in each iteration, and conducts a lasso type algorithm (Tibshirani, 1994) to fully correct the weights. The weights for the basis update are difficult to tune: a large value leads to divergence; a small value makes the algorithm slow (Zhang et al., 2012). The matrix norm boosting approach (Boost) (Zhang et al., 2012) learns the update weights and designs a local refinement step by a non-convex optimization problem which is solved by alternating optimization. It has a sub-linear convergence rate.

In this paper, we present a simple and efficient algorithm to solve the low rank matrix completion problem. The key idea is to extend the orthogonal matching pursuit procedure (Pati et al., 1993) from the vector case to the matrix case. In each iteration, a rank-one basis matrix is generated by the left and right top singular vectors of the current approximation residual. In the standard algorithm, we fully update the weights for all rank-one matrices in the current basis set at the end of each iteration by performing an orthogonal projection of the observation matrix onto their spanning subspace. The most time-consuming step of the proposed algorithm is to calculate the top singular vector pair of a sparse matrix, which costs $O(|\Omega|)$ operations in each iteration. An appealing feature of the proposed algorithm is that it has a linear convergence rate. This is quite different from traditional orthogonal matching pursuit or weak orthogonal greedy algorithms, whose convergence rate for sparse vector recovery is sub-linear as shown in (Liu & Temlyakov, 2012). See also (Tropp, 2004) for an extensive study on various greedy algorithms. With this rate of convergence, we only need $O(\log(1/\epsilon))$ iterations for achieving an ϵ-accuracy solution. One drawback of the standard algorithm is that it needs to store all rank-one matrices in the current basis set for full weight updating, which contains $r|\Omega|$ elements in the r-th iteration. This makes the storage complexity of the algorithm dependent on the number of iterations, which restricts the approximation rank especially for large matrices. To tackle this problem, we propose an economic weight updating rule for this algorithm. In this economic algorithm, we only track two matrices in each iteration. One is the current estimated matrix and the other one is the pursued rank-one matrix. When restricted to the observations in Ω, each has $|\Omega|$ nonzero elements. Thus the storage requirement, i.e., $2|\Omega|$, keeps the same in different iterations, which is the same as the greedy algorithms (Jaggi & Sulovský, 2010; Tewari et al., 2011). Interestingly, we show that using this economic updating rule we still retain the linear convergence rate. To the best of our knowledge, our proposed algorithms are the fastest among all related methods. We verify the efficiency of our algorithms empirically on large-scale matrix completion problems.

The main contributions of our paper are:

- We propose a computationally efficient and scalable algorithm for matrix completion, which extends the

orthogonal matching pursuit from the vector case to the matrix case.

- We theoretically prove the linear convergence rate of our algorithm. As a result, we only need $O(\log(1/\epsilon))$ steps to obtain an ϵ-accuracy solution, and in each step we only need to compute the top singular vector pair, which can be computed efficiently.
- We further reduce the storage complexity of our algorithm based on an economic weight updating rule while retaining the linear convergence rate. This algorithm has constant storage complexity which is independent of the approximation rank and is more practical for large-scale problems.
- Our proposed algorithm is free of tuning parameter, except for the accuracy of the solution. And it is guaranteed to converge, i.e., no risk of divergence.

Notations: Let $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_m) \in \Re^{n\times m}$ be an $n \times m$ real matrix, and $\Omega \subset \{1, \cdots, n\} \times \{1, \cdots, m\}$ denote the indices of the observed entries of $\mathbf{Y}$. P_Ω is the projection operator onto the space spanned by the matrices vanishing outside of Ω so that the (i, j)-th component of $P_\Omega(\mathbf{Y})$ equals to $\mathbf{Y}_{i,j}$ for $(i, j) \in \Omega$ and zero otherwise. The Frobenius norm of $\mathbf{Y}$ is defined as $||\mathbf{Y}||_F = \sqrt{\sum_{i,j} \mathbf{Y}_{i,j}^2}$. Let $vec(\mathbf{Y}) = (\mathbf{y}_1^T, \cdots, \mathbf{y}_m^T)^T$ denote a vector reshaped from matrix $\mathbf{Y}$ by concatenating all its column vectors. Let $\dot{\mathbf{y}} = vec(P_\Omega(\mathbf{Y}))$ be the vector by concatenating all observed entries in $\mathbf{Y}$. The inner product of two matrices $\mathbf{X}$ and $\mathbf{Y}$ is defined as $\langle \mathbf{X}, \mathbf{Y}\rangle = \langle vec(\mathbf{X}), vec(\mathbf{Y})\rangle$. Given a matrix $\mathbf{A} \in \Re^{n\times m}$, we denote $P_\Omega(\mathbf{A})$ by $\mathbf{A}_\Omega$. For any two matrices $\mathbf{A}, \mathbf{B} \in \Re^{n\times m}$, we define $\langle \mathbf{A}, \mathbf{B}\rangle_\Omega = \langle \mathbf{A}_\Omega, \mathbf{B}_\Omega\rangle$, $\|\mathbf{A}\|_\Omega = \sqrt{\langle \mathbf{A}, \mathbf{A}\rangle_\Omega}$ and $\|\mathbf{A}\| = \sqrt{\langle \mathbf{A}, \mathbf{A}\rangle}$.

2. Rank-One Matrix Pursuit

It is well-known that any matrix $\mathbf{X} \in \Re^{n\times m}$ can be written as a linear combination of rank-one matrices, that is,

$$\mathbf{X} = \mathbf{M}(\boldsymbol{\theta}) = \sum_{i\in\mathcal{I}} \theta_i \mathbf{M}_i, \tag{1}$$

where $\{\mathbf{M}_i : i \in \mathcal{I}\}$ is the set of all $n \times m$ rank-one matrices with unit Frobenius norm. Clearly, $\boldsymbol{\theta}$ is an infinite dimensional vector. Such a representation can be obtained from the standard SVD of $\mathbf{X}$.

The original low rank matrix approximation problem aims to minimize the zero-norm of θ subject to the constraint:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & ||\boldsymbol{\theta}||_0 \\ s.t. \quad & P_\Omega(\mathbf{M}(\boldsymbol{\theta})) = P_\Omega(\mathbf{Y}), \end{aligned} \tag{2}$$

where $||\boldsymbol{\theta}||_0$ denotes the cardinality of the number of nonzero elements of $\boldsymbol{\theta}$.

If we reformulate the problem as

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & ||P_\Omega(\mathbf{M}(\boldsymbol{\theta})) - P_\Omega(\mathbf{Y})||_F^2 \\ s.t. \quad & ||\boldsymbol{\theta}||_0 \le r, \end{aligned} \tag{3}$$

we could solve it by an orthogonal matching pursuit type greedy algorithm using rank-one matrices as the basis. If the dictionary $\{\mathbf{M}_i : i \in \mathcal{I}\}$ is known and finite, this is equivalent to the compressed sensing problem. However, in our formulation, the size of the dictionary is infinite and the bases are to be constructed during the basis pursuit process. In particular, we are to find a suitable subset with over-complete rank-one matrix coordinates, and learn the weight for each coordinate. This is achieved by executing two steps alternatively: one is to construct the basis, and the other one is to learn the weights of the basis.

Suppose that after the $(k$-1)-th iteration, the rank-one basis matrices $\mathbf{M}_1, \ldots, \mathbf{M}_{k-1}$ and their current weight $\boldsymbol{\theta}^{k-1}$ are already computed. In the k-th iteration, we are to pursue a new rank-one basis matrix $\mathbf{M}_k$ with unit Frobenius norm, which is mostly correlated with the current observed regression residual $\mathbf{R_k} = P_\Omega(\mathbf{Y}) - \mathbf{X}_{k-1}$, where $\mathbf{X}_{k-1} = (\mathbf{M}(\boldsymbol{\theta}^{k-1}))_\Omega = \sum_{i=1}^{k-1} \theta_i^{k-1} (\mathbf{M}_i)_\Omega$. Therefore, $\mathbf{M}_k$ can be chosen to be an optimal solution of the following problem:

$$\max_{\mathbf{M}} \{\langle \mathbf{M}, \mathbf{R}_k\rangle : \ \mathrm{rank}(\mathbf{M}) = 1, \ \|\mathbf{M}\|_F = 1\}. \tag{4}$$

Notice that each rank-one matrix $\mathbf{M}$ with unit Frobenius norm can be written as the product of two unit vectors, namely, $\mathbf{M} = \mathbf{u}\mathbf{v}^T$ for some $\mathbf{u} \in \Re^n$ and $\mathbf{v} \in \Re^m$ with $\|\mathbf{u}\| = \|\mathbf{v}\| = \mathbf{1}$. We then see that problem (4) can be equivalently reformulated as

$$\max_{\mathbf{u},\mathbf{v}} \{\mathbf{u}^T \mathbf{R}_k \mathbf{v} : \ \|\mathbf{u}\| = \|\mathbf{v}\| = \mathbf{1}\}. \tag{5}$$

Clearly, the optimal solution $(\mathbf{u}_*, \mathbf{v}_*)$ of problem (5) is a pair of top left and right singular vectors of $\mathbf{R}_k$. It can be efficiently computed by the power method (Jaggi & Sulovský, 2010; Dudík et al., 2012). The new rank-one basis matrix $\mathbf{M}_k$ is then readily available by setting $\mathbf{M}_k = \mathbf{u}_* \mathbf{v}_*^T$.

After finding the new rank-one basis matrix $\mathbf{M}_k$, we update the weights $\boldsymbol{\theta}^k$ for all currently available basis matrices $\{\mathbf{M}_1, \cdots, \mathbf{M}_k\}$ by solving the following least squares regression problem:

$$\min_{\boldsymbol{\theta}\in\Re^k} ||\sum_{i=1}^{k} \theta_i \mathbf{M}_i - \mathbf{Y}||_\Omega^2. \tag{6}$$

By reshaping the matrices $(\mathbf{Y})_\Omega$ and $(\mathbf{M}_i)_\Omega$ into vectors $\dot{\mathbf{y}}$ and $\dot{\mathbf{m}}_i$, we can easily see that the optimal solution $\boldsymbol{\theta}^k$ of (6) is given by

$$\boldsymbol{\theta}^k = (\bar{\mathbf{M}}_\mathbf{k}^T \bar{\mathbf{M}}_\mathbf{k})^{-1} \bar{\mathbf{M}}_\mathbf{k}^T \dot{\mathbf{y}}, \tag{7}$$

where $\bar{\mathbf{M}}_\mathbf{k} = [\dot{\mathbf{m}}_1, \cdots, \dot{\mathbf{m}}_k]$ is the matrix formed by all reshaped basis vectors. The row size of matrix $\bar{\mathbf{M}}_k$ is the total number of observed entries. It is computationally expensive to directly calculate the matrix multiplication. An incremental update rule can be applied to solve this step efficiently (Wang et al., 2014).

We run the above two steps iteratively until some desired stopping condition is satisfied. We can terminate the method based on the rank of the estimated matrix or the approximation residual. In particular, one can choose a preferred rank of the approximate solution matrix. Alternatively, one can stop the method once the residual $\|\mathbf{R}_k\|$ is less than a tolerance parameter ε. The main steps of Rank-One Matrix Pursuit (R1MP) are given in Algorithm 1.

Remark In our algorithm, we adapt orthogonal matching pursuit on the observed part of the matrix. This is similar to the GECO algorithm. However, GECO constructs the estimated matrix by projecting the observation matrix onto a much larger subspace, which is a product of two subspaces spanned by all left singular vectors and all right singular vectors obtained up to the current iteration. So it has much higher computational complexity. Lee et al. (Lee & Bresler, 2010) recently propose the ADMiRA algorithm, which is also a greedy approach. In each step it first chooses $2r$ components by top-$2r$ truncated SVD and then uses another top-r truncated SVD to obtain a rank-r matrix. Thus, the ADMiRA algorithm is computationally more expensive than the proposed algorithm. The main difference between the proposed algorithm and ADMiRA is somewhat similar to the difference between the OMP (Pati et al., 1993) for learning sparse vectors and CoSaMP (Needell & Tropp, 2010). In addition, the performance guarantees (including recovery guarantee and convergence property) of ADMiRA rely on strong assumptions, i.e., the matrix involved in the loss function satisfies a rank-restricted isometry property, which is not satisfied in matrix completion (Lee & Bresler, 2010). Lee et al. sketch a similar idea as the standard verion of our algorithm in Remark 2.3 without any further analysis, and their theoretical results cannot be easily extended to our algorithm. Another contribution of our work is that we further propose an economic version of the algorithm and analyze its convergence property.

3. Convergence Analysis

In this section, we will show that our proposed rank-one matrix pursuit algorithm achieves a linear convergence rate. This main result is given in the following theorem.

Theorem 3.1. *The rank-one matrix pursuit algorithm satisfies*

$$||\mathbf{R}_k|| \leq \gamma^{k-1}\|\mathbf{Y}\|_\Omega, \quad \forall k \geq 1.$$

γ is a constant in $[0, 1)$.

Algorithm 1 Rank-One Matrix Pursuit (R1MP)

Input: $\mathbf{Y}_\Omega$ and stopping criterion.
Initialize: Set $\mathbf{X}_0 = 0$ and $k = 1$.
repeat
  Step 1: Find a pair of top left and right singular vectors $(\mathbf{u}_k, \mathbf{v}_k)$ of the observed residual matrix $\mathbf{R}_k = \mathbf{Y}_\Omega - \mathbf{X}_{k-1}$ and set $\mathbf{M}_k = \mathbf{u}_k(\mathbf{v}_k)^T$.
  Step 2: Compute the weight $\boldsymbol{\theta}^k$ using the closed form least squares solution $\boldsymbol{\theta}^k = (\bar{\mathbf{M}}_\mathbf{k}^T\bar{\mathbf{M}}_\mathbf{k})^{-1}\bar{\mathbf{M}}_\mathbf{k}^T\dot{\mathbf{y}}$.
  Step 3: Set $\mathbf{X}_k = \sum_{i=1}^k \theta_i^k(\mathbf{M}_i)_\Omega$ and $k \leftarrow k + 1$.
until stopping criterion is satisfied
Output: Constructed matrix $\hat{\mathbf{Y}} = \sum_{i=1}^k \theta_i^k\mathbf{M}_i$.

Before proving Theorem 3.1, we need to establish some useful and preparatory properties of Algorithm 1. The first property says that $\mathbf{R}_{k+1}$ is perpendicular to all previously generated $\mathbf{M}_i$ for $i = 1, \cdots, k$.

Property 3.2. $\langle \mathbf{R}_{k+1}, \mathbf{M}_i\rangle = 0$ *for* $i = 1, \cdots, k$.

Proof. Recall that $\boldsymbol{\theta}^k$ is the optimal solution of problem (6). By the first-order optimality condition, one has $\langle \mathbf{Y} - \sum_{j=1}^k \theta_j^k\mathbf{M}_j, \mathbf{M}_i\rangle_\Omega = 0$ for $i = 1, \cdots, k$, which together with $\mathbf{R}_k = \mathbf{Y}_\Omega - \mathbf{X}_{k-1}$ and $\mathbf{X}_k = \sum_{j=1}^k \theta_j^k(\mathbf{M}_j)_\Omega$ implies that $\langle \mathbf{R}_{k+1}, \mathbf{M}_i\rangle = 0$ for $i = 1, \cdots, k$. □

The following property shows that as the number of rank-one basis matrices $\mathbf{M}_i$ increases during our learning process, the residual $\|\mathbf{R}_k\|$ does not increase.

Property 3.3. $\|\mathbf{R}_{k+1}\| \leq \|\mathbf{R}_k\|$ *for all* $k \geq 1$.

Proof. We observe that for all $k \geq 1$,

$$\begin{aligned}\|\mathbf{R}_{k+1}\|^2 &= \min_{\boldsymbol{\theta}\in\Re^k}\{\|\mathbf{Y} - \textstyle\sum_{i=1}^k \theta_i\mathbf{M}_i\|_\Omega^2\} \\ &\leq \min_{\boldsymbol{\theta}\in\Re^{k-1}}\{\|\mathbf{Y} - \textstyle\sum_{i=1}^{k-1} \theta_i\mathbf{M}_i\|_\Omega^2\} = \|\mathbf{R}_k\|^2,\end{aligned}$$

and hence the conclusion holds. □

We next establish that $\{(\mathbf{M}_i)_\Omega\}_{i=1}^k$ is linearly independent unless $\|\mathbf{R}_k\| = 0$. It follows that formula (7) is well-defined and hence $\boldsymbol{\theta}^k$ is uniquely defined before the algorithm stops.

Property 3.4. *Suppose that* $\mathbf{R}_k \neq 0$ *for some* $k \geq 1$. *Then,* $\bar{\mathbf{M}}_i$ *has a full column rank for all* $i \leq k$.

Proof. Using Property 3.3 and the assumption $\mathbf{R}_k \neq 0$ for some $k \geq 1$, we see that $\mathbf{R}_i \neq 0$ for all $i \leq k$. We now prove this statement by induction on i. Indeed, since $\mathbf{R}_1 \neq 0$, we clearly have $\bar{\mathbf{M}}_1 \neq 0$. Hence the conclusion holds for $i = 1$. We now assume that it holds for $i - 1 < k$ and need to show that it also holds for $i \leq k$. By the induction hypothesis, $\bar{\mathbf{M}}_{i-1}$ has a full column rank. Suppose for contradiction that $\bar{\mathbf{M}}_i$ does not have a full column rank. Then, there exists $\boldsymbol{\alpha} \in \Re^{i-1}$ such that

$(\mathbf{M}_i)_\Omega = \sum_{j=1}^{i-1} \alpha_j (\mathbf{M}_j)_\Omega$, which together with Property 3.2 implies that $\langle \mathbf{R}_i, \mathbf{M}_i \rangle = 0$. It follows that $\sigma_{\max}(\mathbf{R}_i) = u_i^T \mathbf{R}_i v_i = \langle \mathbf{R}_i, \mathbf{M}_i \rangle = 0$, and hence $\mathbf{R}_i = 0$, which contradicts the fact that $\mathbf{R}_j \neq 0$ for all $j \leq k$. Therefore, $\bar{\mathbf{M}}_i$ has a full column rank and the conclusion holds. □

We next build a relationship between two consecutive residuals $\|\mathbf{R}_{k+1}\|$ and $\|\mathbf{R}_k\|$.

For convenience, define $\theta_k^{k-1} = 0$ and let $\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1} + \boldsymbol{\eta}^k$. In view of (6), one can observe that

$$\boldsymbol{\eta}^k = \arg\min_{\boldsymbol{\eta}} || \sum_{i=1}^{k} \eta_i \mathbf{M}_i - \mathbf{R}_k ||_\Omega^2. \tag{8}$$

Let

$$\mathbf{L}_k = \sum_{i=1}^{k} \eta_i^k (\mathbf{M}_i)_\Omega. \tag{9}$$

By the definition of $\mathbf{X}_k$, one can also observe that $\mathbf{X}_k = \mathbf{X}_{k-1} + \mathbf{L}_k$ and $\mathbf{R}_{k+1} = \mathbf{R}_k - \mathbf{L}_k$.

Property 3.5. $||\mathbf{R}_{k+1}||^2 = ||\mathbf{R}_k||^2 - ||\mathbf{L}_k||^2$ *and* $||\mathbf{L}_k||^2 \geq \langle \mathbf{M}_k, \mathbf{R}_k \rangle^2$*, where* $\mathbf{L}_k$ *is defined in* (9)*.*

Proof. Since $\mathbf{L}_k = \sum_{i \leq k} \eta_i^k (\mathbf{M}_i)_\Omega$, it follows from Property 3.2 that $\langle \mathbf{R}_{k+1}, \mathbf{L}_k \rangle = 0$. Thus,

$$\begin{aligned}
&||\mathbf{R}_{k+1}||^2 \\
&= ||\mathbf{R}_k - \mathbf{L}_k||^2 = ||\mathbf{R}_k||^2 - 2\langle \mathbf{R}_k, \mathbf{L}_k \rangle + ||\mathbf{L}_k||^2 \\
&= ||\mathbf{R}_k||^2 - 2\langle \mathbf{R}_{k+1} + \mathbf{L}_k, \mathbf{L}_k \rangle + ||\mathbf{L}_k||^2 \\
&= ||\mathbf{R}_k||^2 - 2\langle \mathbf{L}_k, \mathbf{L}_k \rangle + ||\mathbf{L}_k||^2 \\
&= ||\mathbf{R}_k||^2 - ||\mathbf{L}_k||^2
\end{aligned}$$

We next bound $\|\mathbf{L}_k\|^2$ from below. If $\mathbf{R}_k = 0$, $||\mathbf{L}_k||^2 \geq \langle \mathbf{M}_k, \mathbf{R}_k \rangle^2$ clearly holds. We now suppose throughout the remaining proof that $\mathbf{R}_k \neq 0$. It then follows from Property 3.4 that $\bar{\mathbf{M}}_k$ has a full column rank. Using this fact and (8), we have $\boldsymbol{\eta}^k = \left(\bar{\mathbf{M}}_k^T \bar{\mathbf{M}}_k\right)^{-1} \bar{\mathbf{M}}_k^T \dot{\mathbf{r}}_k$, where $\dot{\mathbf{r}}_k$ is the reshaped residual vector of $\mathbf{R}_k$. Invoking that $\mathbf{L}_k = \sum_{i \leq k} \eta_i^k (\mathbf{M}_i)_\Omega$, we then obtain

$$||\mathbf{L}_k||^2 = \dot{\mathbf{r}}_k^T \bar{\mathbf{M}}_k (\bar{\mathbf{M}}_k^T \bar{\mathbf{M}}_k)^{-1} \bar{\mathbf{M}}_k^T \dot{\mathbf{r}}_k. \tag{10}$$

Let $\bar{\mathbf{M}}_k = \mathbf{QU}$ be the QR factorization of $\bar{\mathbf{M}}_k$, where $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$ and $\mathbf{U}$ is a $k \times k$ nonsingular upper triangular matrix. One can observe that $(\bar{\mathbf{M}}_k)_k = \dot{\mathbf{m}}_k$, where $(\bar{\mathbf{M}}_k)_k$ denotes the k-th column of the matrix $\bar{\mathbf{M}}_k$ and $\dot{\mathbf{m}}_k$ is the reshaped vector of $(\mathbf{M}_k)_\Omega$. Recall that $\|\mathbf{M}_k\| = \|\mathbf{u}_k \mathbf{v}_k^T\| = 1$. Hence, $\|(\bar{\mathbf{M}}_k)_k\| \leq 1$. Due to $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$, $\bar{\mathbf{M}}_k = \mathbf{QU}$ and the definition of $\mathbf{U}$, we have

$$0 < |\mathbf{U}_{kk}| \leq \|\mathbf{U}_k\| = \|(\bar{\mathbf{M}}_k)_k\| \leq 1.$$

In addition, by Property 3.2, we have

$$\bar{\mathbf{M}}_k^T \dot{\mathbf{r}}_k = [0, \cdots, 0, \langle \mathbf{M}_k, \mathbf{R}_k \rangle]^T. \tag{11}$$

Substituting $\bar{\mathbf{M}}_k = \mathbf{QU}$ into (10), and using $\mathbf{Q}^\mathbf{T}\mathbf{Q} = \mathbf{I}$ and (11), we obtain that

$$\begin{aligned}
&\|\mathbf{L}_k\|^2 \\
&= \dot{\mathbf{r}}_k^T \bar{\mathbf{M}}_k (\mathbf{U}^T\mathbf{U})^{-1} \bar{\mathbf{M}}_k^T \dot{\mathbf{r}}_k \\
&= [0, \cdots, 0, \langle \mathbf{M}_k, \mathbf{R}_k \rangle] \, \mathbf{U}^{-1}\mathbf{U}^{-T} \, [0, \cdots, 0, \langle \mathbf{M}_k, \mathbf{R}_k \rangle]^T \\
&= \langle \mathbf{M}_k, \mathbf{R}_k \rangle^2 / (\mathbf{U}_{kk})^2 \;\geq\; \langle \mathbf{M}_k, \mathbf{R}_k \rangle^2.
\end{aligned}$$

The last equality follows since $\mathbf{U}$ is upper triangular and the last inequality is due to $|\mathbf{U}_{kk}| \leq 1$. □

We are now ready to prove Theorem 3.1.

Proof. Using the definition of $\mathbf{M}_k$, we have

$$\langle \mathbf{M}_k, \mathbf{R}_k \rangle \;=\; \langle \mathbf{u}^k (\mathbf{v}^k)^T, \mathbf{R}_k \rangle \;=\; \sigma_*(\mathbf{R}_k),$$

where $\sigma_*(\mathbf{R}_k)$ is the maximum singular value of the residual matrix $\mathbf{R}_k$. Using this inequality and Property 3.5, we obtain that

$$\begin{aligned}
||\mathbf{R}_{k+1}||^2 &= ||\mathbf{R}_k||^2 - ||\mathbf{L}_k||^2 \\
&\leq ||\mathbf{R}_k||^2 - \langle \mathbf{M}_k, \mathbf{R}_k \rangle^2 \\
&= \left(1 - \tfrac{\sigma_*^2(\mathbf{R}_k)}{||\mathbf{R}_k||^2}\right) ||\mathbf{R}_k||^2.
\end{aligned}$$

In view of this relation and the fact that $\|\mathbf{R}_1\| = \|\mathbf{Y}\|_\Omega^\mathbf{2}$, we easily conclude that

$$||\mathbf{R}_k|| \;\leq\; \|\mathbf{Y}\|_\Omega \prod_{i=1}^{k-1} \sqrt{1 - \frac{\sigma_*^2(\mathbf{R}_i)}{\|\mathbf{R}_i\|^2}}.$$

As for each step we have $0 < \frac{1}{rank(\mathbf{R}_i)} \leq \frac{\sigma_*(\mathbf{R}_i)}{\|\mathbf{R}_i\|} \leq 1$, there must exist $0 \leq \gamma < 1$ that satisfies $||\mathbf{R}_k|| \leq \gamma^{k-1} \|\mathbf{Y}\|_\Omega$. This completes the proof. □

Remark In practice, the value of $\frac{\|\mathbf{R}_i\|^2}{\sigma_*^2(\mathbf{R}_i)}$ that controls the convergence speed is much less than $\min(m, n)$. We will emprically verify this in the experiments.

Remark If Ω is the entire set of all indices of $\{(i, j), i = 1, \cdots, m, j = 1, \cdots, n\}$, our rank-one matrix pursuit algorithm equals to standard SVD using the power method.

Remark This convergence is obtained for the optimization residual in the low rank matrix completion problem. We further extend our algorithm to solve the more general matrix sensing problem and analyze the corresponding statistical convergence behavior under mild conditions, such as the rank-restricted isometry property (Lee & Bresler, 2010; Jain et al., 2013). Details are provided in the longer version of this paper (Wang et al., 2014).

4. Economic Rank-One Matrix Pursuit

The proposed R1MP algorithm has to track all pursued bases and save them in the memory. It demands $O(r|\Omega|)$ storage complexity to obtain a rank-r estimated matrix. For large-scale problems, such storage requirement is not negligible and restricts the rank of the matrix to be estimated. To adapt our algorithm to large-scale problems with a large approximation rank, we simplify the orthogonal projection step by only tracking the estimated matrix $\mathbf{X}_{k-1}$ and the rank-one update matrix $\mathbf{M}_k$. In this case, we only need to estimate the weights for these two matrices in Step 2 of our algorithm by solving the following least squares problem:

$$\boldsymbol{\alpha}^k = \arg\min_{\boldsymbol{\alpha}=\{\alpha_1,\alpha_2\}} ||\alpha_1\mathbf{X}_{k-1} + \alpha_2\mathbf{M}_k - \mathbf{Y}||_\Omega^2. \quad (12)$$

This still corrects all weights of the existed bases, though the correction is sub-optimal. If we write the estimated matrix as a linear combination of the bases, we have $\mathbf{X}_k = \sum_{i=1}^k \theta_i^k(\mathbf{M}_i)_\Omega$ with $\theta_k^k = \alpha_2^k$ and $\theta_i^k = \theta_i^{k-1}\alpha_1^k$, for $i < k$. The detailed procedure of this simplified method is given in Algorithm 2.

Algorithm 2 Economic Rank-One Matrix Pursuit (ER1MP)

Input: $\mathbf{Y}_\Omega$ and stopping criterion.
Initialize: Set $\mathbf{X}_0 = 0$ and $k = 1$.
repeat
Step 1: Find a pair of top left and right singular vectors $(\mathbf{u}_k, \mathbf{v}_k)$ of the observed residual matrix $\mathbf{R}_k = \mathbf{Y}_\Omega - \mathbf{X}_{k-1}$ and set $\mathbf{M}_k = \mathbf{u}_k(\mathbf{v}_k)^T$.
Step 2: Compute the optimal weights $\boldsymbol{\alpha}^k$ for $\mathbf{X}_{k-1}$ and $\mathbf{M}_k$ by solving:

$$\arg\min_{\boldsymbol{\alpha}} ||\alpha_1\mathbf{X}_{k-1} + \alpha_2(\mathbf{M}_k)_\Omega - \mathbf{Y}_\Omega||^2.$$

Step 3: Set $\mathbf{X}_k = \alpha_1^k\mathbf{X}_{k-1} + \alpha_2^k(\mathbf{M}_k)_\Omega$; $\theta_k^k = \alpha_2^k$ and $\theta_i^k = \theta_i^{k-1}\alpha_1^k$ for $i < k$; $k \leftarrow k + 1$.
until stopping criterion is satisfied
Output: Constructed matrix $\hat{\mathbf{Y}} = \sum_{i=1}^k \theta_i^k\mathbf{M}_i$.

The proposed economic rank-one matrix pursuit algorithm (ER1MP) uses the same amount of storage as the greedy algorithms (Jaggi & Sulovský, 2010; Tewari et al., 2011), which is significantly smaller than that required by R1MP algorithm. Interestingly, we can show that the ER1MP algorithm still retains the linear convergence rate. The main result is given in the following theorem, and the proof is provided in the long version of this paper (Wang et al., 2014).

Theorem 4.1. *The economic rank-one matrix pursuit algorithm satisfies*

$$||\mathbf{R}_k|| \leq \tilde{\gamma}^{k-1}\|\mathbf{Y}\|_\Omega, \quad \forall k \geq 1.$$

$\tilde{\gamma}$ is a constant in $[0, 1)$.

5. Experiments

In this section, we compare our rank-one matrix pursuit algorithms R1MP and ER1MP with state-of-the-art matrix completion algorithms. The competing algorithms include: singular value projection (SVP) (Jain et al., 2010), singular value thresholding (SVT) (Candès & Recht, 2009), Jaggi's fast algorithm for trace norm constraint (JS) (Jaggi & Sulovský, 2010), spectral regularization algorithm (SoftImpute) (Mazumder et al., 2010), low rank matrix fitting (LMaFit) (Wen et al., 2010), alternating minimization (AltMin) (Jain et al., 2013), boosting type accelerated matrix-norm penalized solver (Boost) (Zhang et al., 2012) and greedy efficient component optimization (GECO) (Shalev-Shwartz et al., 2011). The general greedy method (Tewari et al., 2011) is not included in our comparison, as it includes JS and GECO (included in our comparison) as special cases for matrix completion. The lifted coordinate descent method (Lifted) (Dudík et al., 2012) is not included in our comparison, as it is similar to Boost proposed in (Zhang et al., 2012), but more sensitive to the parameters.

The code for most of these algorithms is available online:

- SVP:
  http://www.cs.utexas.edu/∼pjain/svp/
- SVT:
  http://svt.stanford.edu/
- SoftImpute:
  http://www-stat.stanford.edu/∼rahulm/software.html
- LMaFit:
  http://lmafit.blogs.rice.edu/
- Boost:
  http://webdocs.cs.ualberta.ca/∼xinhua2/boosting.zip
- GECO:
  http://www.cs.huji.ac.il/∼shais/code/geco.zip

We compare these algorithms in two problems, including image recovery and collaborative filtering. The data size for image recovery is relatively small, and the recommendation problem is in large-scale. In the experiments, we follow the recommended settings of the parameters for competing algorithms. If no recommended parameter value is available, we choose the best one from a candidate set using cross validation. For our R1MP and ER1MP algorithms, we only need a stopping criterion. For simplicity, we stop our algorithms after r iterations. In this way, we approximate the ground truth using a rank-r matrix. We present the experimental results using *root-mean-square error* (RMSE) (Jaggi & Sulovský, 2010; Shalev-Shwartz et al., 2011). The experiments are implemented in MATLAB[1]. They call some external packages for fast

[1]GECO is written in C++ and we call its executable file in MATLAB.

Table 1. Image recovery results measured in terms of the RMSE: the value below is the actual value times 100 ($mean \pm std$).

Image	SVT	SVP	SoftImpute	LMaFit	AltMin	JS	R1MP	ER1MP
Lenna	3.86 ± 0.02	5.31 ± 0.14	4.60 ± 0.02	7.45 ± 0.63	4.47 ± 0.10	5.48 ± 0.72	3.90 ± 0.02	3.97 ± 0.02
Barbara	4.48 ± 0.02	5.60 ± 0.08	5.22 ± 0.01	5.16 ± 0.28	5.05 ± 0.06	6.52 ± 0.88	4.63 ± 0.01	4.73 ± 0.02
Clown	3.72 ± 0.03	10.97 ± 0.17	4.48 ± 0.03	4.65 ± 0.67	5.49 ± 0.46	7.30 ± 2.32	3.85 ± 0.03	3.91 ± 0.03
Crowd	4.48 ± 0.02	7.62 ± 0.13	5.35 ± 0.02	4.91 ± 0.05	4.87 ± 0.02	7.38 ± 1.41	4.89 ± 0.03	4.96 ± 0.03
Girl	3.36 ± 0.02	4.45 ± 0.16	4.10 ± 0.01	4.12 ± 0.48	5.07 ± 0.50	4.42 ± 0.46	3.09 ± 0.02	3.12 ± 0.02
Man	4.49 ± 0.03	5.52 ± 0.10	5.16 ± 0.03	5.31 ± 0.13	5.19 ± 0.11	6.25 ± 0.54	4.66 ± 0.03	4.76 ± 0.03

computation of SVD[2] and sparse matrix computations. The experiments are run in a PC with WIN7 system, Intel 4 core 3.4 GHz CPU and 8G RAM.

5.1. Image Recovery

In the image recovery experiments, we use the following benchmark test images: Lenna, Barbara, Clown, Crowd, Girl, Man[3]. The size of each image is 512×512. For each experiment, we present the average RMSE and the corresponding standard derivation of 10 different runs for each competing algorithm. In each run, we randomly exclude 50% of the pixels in the image, and the remaining ones are used as the observations. As the image matrix is not guaranteed to be low rank, we use the rank 200 for the estimation matrix for each experiment. The JS algorithm does not explicitly control the rank, thus we fix its number of iterations to 2000. The numerical results are listed in Table 1. The results show that SVT, our R1MP and ER1MP achieve the best numerical performance. However, our algorithm is much faster and more stable than SVT. For each image, ER1MP uses around 3.5 seconds, but SVT consumes around 400 seconds. Image recovery needs a relatively higher approximation rank; GECO and Boost fail to find a good recovery in some cases, so we do not include them in the table.

5.2. Recommendation

In the following experiments, we compare the different matrix completion algorithms using large recommendation datasets: Jester (Goldberg et al., 2001) and MovieLens (Miller et al., 2003). We use six datasets including: Jester1, Jester2, Jester3, MovieLens100K, MovieLens1M, and MovieLens10M. The statistics of these datasets are given in Table 2. The Jester datasets were collected from a joke recommendation system. They contain anonymous ratings of 100 jokes from the users. The ratings are real values ranging from -10.00 to $+10.00$. The MovieLens datasets were collected from the MovieLens website[4]. They contain anonymous ratings of the movies on this web made by its users. For MovieLens100K and MovieLens1M, there are 5 rating scores (1–5), and for MovieLens10M there are 10 levels of scores with a step size 0.5 in the range of 0.5 to 5. In the following experiments, we randomly split the ratings into training and test sets. Each set contains 50% of the ratings. We compare the prediction results from different methods. In the experiments, we use 100 iterations for the JS algorithm, and for other algorithms we use the same rank for the estimated matrices; the values of the rank are $\{10, 10, 5, 10, 10, 20\}$ for the six corresponding datasets. The results in terms of the RMSE is given in Table 3. We also show the running time of different methods in Table 4. We can observe from the above experiments that our ER1MP algorithm is the fastest among all competing methods to obtain satisfactory results.

Table 2. Characteristics of the recommendation datasets.

Dataset	# row	# column	# rating
Jester1	24983	100	10^6
Jester2	23500	100	10^6
Jester3	24938	100	6×10^5
MovieLens100k	943	1682	10^5
MovieLens1M	6040	3706	10^6
MovieLens10M	69878	10677	10^7

5.3. Convergence and Efficiency

We present the residual curves on the Lenna image in logarithmic scale for our R1MP and ER1MP algorithms in Figure 1. The results show that our algorithms reduce the approximation error in a linear rate. This is consistent with our theoretical analysis. The empirical results verify the linear convergence property of our proposed algorithms.

6. Conclusion

In this paper, we propose an efficient and scalable low rank matrix completion algorithm. The key idea is to extend orthogonal matching pursuit method from the vector case to the matrix case. We also propose a novel weight updating

[2]PROPACK is used in SVP, SVT, SoftImpute and Boost. It is an efficient SVD package, which can be downloaded from http://soi.stanford.edu/~rmunk/PROPACK/

[3]Images are downloaded from http://www.utdallas.edu/~cxc123730/mh_bcs_spl.html

[4]http://movielens.umn.edu

Table 3. Recommendation results measured in terms of the RMSE. Boost fails on the MovieLens10M.

Dataset	SVP	SoftImpute	LMaFit	AltMin	Boost	JS	GECO	R1MP	ER1MP
Jester1	4.7311	5.1113	4.7623	4.8572	5.1746	4.4713	4.3680	4.3418	4.3384
Jester2	4.7608	5.1646	4.7500	4.8616	5.2319	4.5102	4.3967	4.3649	4.3546
Jester3	8.6958	5.4348	9.4275	9.7482	5.3982	4.6866	5.1790	4.9783	5.0145
MovieLens100K	0.9683	1.0354	1.2308	1.0042	1.1244	1.0146	1.0243	1.0168	1.0261
MovieLens1M	0.9085	0.8989	0.9232	0.9382	1.0850	1.0439	0.9290	0.9595	0.9462
MovieLens10M	0.8611	0.8534	0.8625	0.9007	–	0.8728	0.8668	0.8621	0.8692

Table 4. The running time (measured in seconds) of all methods on all recommendation datasets.

Dataset	SVP	SoftImpute	LMaFit	AltMin	Boost	JS	GECO	R1MP	ER1MP
Jester1	18.35	161.49	3.68	11.14	93.91	29.68	$> 10^4$	1.83	0.99
Jester2	16.85	152.96	2.42	10.47	261.70	28.52	$> 10^4$	1.68	0.91
Jester3	16.58	10.55	8.45	12.23	245.79	12.94	$> 10^3$	0.93	0.34
MovieLens100K	1.32	128.07	2.76	3.23	2.87	2.86	10.83	0.04	0.04
MovieLens1M	18.90	59.56	30.55	68.77	93.91	13.10	$> 10^4$	0.87	0.54
MovieLens10M	$> 10^3$	$> 10^3$	154.38	310.82	–	130.13	$> 10^5$	23.05	13.79

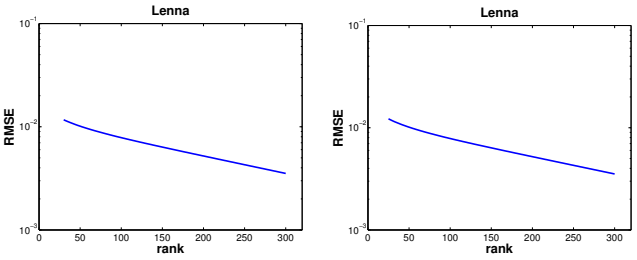

Figure 1. Illustration of the linear convergence of the proposed rank-one matrix pursuit algorithms on the Lenna image: the x-axis is the iteration, and the y-axis is the RMSE in logarithmic scale. The curves are the results for R1MP and ER1MP respectively.

rule under this framework to reduce the storage complexity and make it independent of the approximation rank. Our algorithms are computationally inexpensive for each matrix pursuit iteration, and find satisfactory results in a few iterations. Another advantage of our proposed algorithms is they have only one tunable parameter, which is the rank. It is easy to understand and to use by the user. This becomes especially important in large-scale learning problems. In addition, we rigorously show that both algorithms achieve a linear convergence rate, which is significantly better than the previous known results (a sub-linear convergence rate). We also empirically compare the proposed algorithms with state-of-the-art matrix completion algorithms, and our results show that the proposed algorithms are more efficient than competing algorithms while achieving similar or better prediction performance. We plan to generalize our theoretical and empirical analysis to other loss functions in the future.

7. Acknowledgments

This work was supported in part by China 973 Fundamental R&D Program (No.2014CB340304), NIH (LM010730), and NSF (IIS-0953662, CCF-1025177).

References

Argyriou, A., Evgeniou, T., and Pontil, M. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Avron, H., Kale, S., Kasiviswanathan, S., and Sindhwani, V. Efficient and practical stochastic subgradient descent for nuclear norm regularization. In *ICML*, 2012.

Balzano, L., Nowak, R., and Recht, B. Online identification and tracking of subspaces from highly incomplete information. In *Allerton*, 2010.

Cai, J., Candès, E. J., and Shen, Z. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

Candès, E. J. and Recht, B. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

Dudík, M., Harchaoui, Z., and Malick, J. Lifted coordinate descent for learning with trace-norm regularization. In *AISTATS*, 2012.

Friedman, J. H., Hastie, T., and Tibshirani, R. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

Goldberg, K., Roeder, T., Gupta, D., and Perkins, C. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

Hazan, E. Sparse approximate solutions to semidefinite programs. In *LATIN*, 2008.

Jaggi, M. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *ICML*, 2013.

Jaggi, M. and Sulovský, M. A simple algorithm for nuclear norm regularized problems. In *ICML*, 2010.

Jain, P., Meka, R., and Dhillon, I. S. Guaranteed rank minimization via singular value projection. In *NIPS*, 2010.

Jain, P., Netrapalli, P., and Sanghavi, S. Low-rank matrix completion using alternating minimization. In *STOC*, 2013.

Ji, S. and Ye, J. An accelerated gradient method for trace norm minimization. In *ICML*, 2009.

Keshavan, R. and Oh, S. Optspace: A gradient descent algorithm on grassmann manifold for matrix completion. *http://arxiv.org/abs/0910.5260*, 2009.

Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, 2009.

Lee, K. and Bresler, Y. Admira: atomic decomposition for minimum rank approximation. *IEEE Transactions on Information Theory*, 56(9):4402–4416, 2010.

Liu, E. and Temlyakov, T. N. The orthogonal super greedy algorithm and applications in compressed sensing. *IEEE Transactions on Information Theory*, 58: 2040–2047, 2012.

Ma, S., Goldfarb, D., and Chen, L. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128(1-2):321–353, 2011.

Mazumder, R., Hastie, T., and Tibshirani, R. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 99:2287–2322, August 2010.

Miller, B. N., Albert, I., Lam, S. K., Konstan, J. A., and Riedl, J. Movielens unplugged: experiences with an occasionally connected recommender system. In *IUI*, 2003.

Mishra, B., Meyer, G., Bach, F., and Sepulchre, R. Low-rank optimization with trace norm penalty. *http://arxiv.org/abs/1112.2318*, 2011.

Needell, D. and Tropp, J. A. Cosamp: iterative signal recovery from incomplete and inaccurate samples. *Communications of the ACM*, 53(12):93–100, 2010.

Negahban, S. and Wainwright, M.J. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. In *ICML*, 2010.

Pati, Y. C., Rezaiifar, R., Rezaiifar, Y. C. Pati R., and Krishnaprasad, P. S. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Asilomar SSC*, 1993.

Recht, B. and Ré, C. Parallel stochastic gradient algorithms for large-scale matrix completion. *Mathematical Programming Computation*, 5(2):201–226, 2013.

Shalev-Shwartz, S. and Tewari, A. Stochastic methods for ℓ_1 regularized loss minimization. In *ICML*, 2009.

Shalev-Shwartz, S., Gonen, A., and Shamir, O. Large-scale convex minimization with a low-rank constraint. In *ICML*, 2011.

Srebro, N., Rennie, J., and Jaakkola, T. Maximum margin matrix factorizations. In *NIPS*, 2005.

Tewari, A., Ravikumar, P., and Dhillon, I. S. Greedy algorithms for structurally constrained high dimensional problems. In *NIPS*, 2011.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Toh, K. and Yun, S. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *Pacific Journal of Optimization*, 6:615 – 640, 2010.

Tropp, J. A. Greed is good: algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50:2231–2242, 2004.

Wang, Z., Lai, M., Lu, Z., Fan, W., Davulcu, H., and Ye, J. Orthogonal rank-one matrix pursuit for low rank matrix completion. *http://arxiv.org/abs/1404.1377*, 2014.

Wen, Z., Yin, W., and Zhang, Y. Low-rank factorization model for matrix completion by a non-linear successive over-relaxation algorithm. Rice CAAM Tech Report 10-07, University of Rice, 2010.

Zhang, X., Yu, Y., and Schuurmans, D. Accelerated training for matrix-norm regularization: A boosting approach. In *NIPS*, 2012.