Orthogonal Rank-One Matrix Pursuit for Matrix Completion

Zheng Wang[1,2], Ming-Jun Lai[3], Zhaosong Lu[4], Wei Fan[5], Jieping Ye[1,2]

[1]Computer Science and Engineering, Arizona State University, AZ 85287, USA
[2]The Biodesign Institute, Arizona State University, AZ 85287, USA
[3]Department of Mathematics, The University of Georgia, Athens, GA 30602, USA
[4]Department of Mathematics, Simon Frasor University, Burnaby, BC, V5A 156, Canada
[5]Huawei Noah's Ark Lab Hong Kong

ABSTRACT

Low rank modeling has found applications in a wide range of machine learning and data mining tasks, such as matrix completion, dimensionality reduction, compressed sensing, multi-class and multi-task learning. Recently, significant efforts have been devoted to the low rank matrix completion problem, as it has important applications in many domains including collaborative filtering, Microarray data imputation, and image inpainting. Many algorithms have been proposed for matrix completion in the past. However, most of these algorithms involve computing singular value decomposition, which is not scalable to large-scale problems. In this paper, we propose 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 fully update the weights for all rank-one matrices obtained up to the current iteration. The computation of the top singular vector pair and the updating of the weights can be implemented efficiently, making the proposed algorithm scalable to large matrices. We further establish the linear convergence of the proposed iterative algorithm. This is quite different from the existing theory for convergence rate of orthogonal greedy algorithms. A linear convergence rate is achieved due to our construction of matrix bases. We empirically evaluate the proposed algorithm on many real-world datasets, including the largest publicly available benchmark dataset Netflix as well as the MovieLens 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.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications Data Mining

KDD '13 Chicago, Illinois USA

General Terms

Algorithm

Keywords

Low rank, singular value decomposition, rank minimization, matrix completion, matching pursuit

1. INTRODUCTION

Low rank matrix learning has attracted significant attentions in machine learning and data mining due to its wide range of applications, such as collaborative filtering, dimensionality reduction, compress sensing, multi-class learning and multi-task learning [1, 2, 6, 8, 19, 31, 35]. 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 $\mathbf{X} \in \mathcal{R}^{n\times m}$ with minimum rank that best approximates the matrix $\mathbf{Y}$ on the observed elements. The mathematical formulation is given by

$$\begin{array}{ll} \min\limits_{\mathbf{X}\in\mathcal{R}^{n\times m}} & rank(\mathbf{X}) \\ s.t. & P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{Y}), \end{array} \tag{1}$$

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, many approximate solutions have been proposed for problem (1) [6, 20, 24]. A widely used convex relaxation of matrix rank is the trace norm or nuclear norm [6]. The matrix trace norm is defined by the Schatten p-norm as $p = 1$. 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 $||\mathbf{X}||_* = \sum_{i=1}^{r} |\sigma_i|$. Then the convex relaxation for problem (1) is given by

$$\begin{array}{ll} \min\limits_{\mathbf{X}\in\mathcal{R}^{n\times m}} & ||\mathbf{X}||_* \\ s.t. & P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{Y}). \end{array} \tag{2}$$

1.1 Related Works

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 for large-scale problems attracts much attention in recent years. Several techniques have been adopted to accelerate the training

speed [5, 15, 17, 27]. Cai et al. [5] propose an algorithm by using soft singular value thresholding. Keshavan et al. [17] and Meka et al. [27] use top k singular pairs and attain more efficient methods.

Another set of approaches solve the trace norm penalized problem:

$$\min_{\mathbf{X}\in\mathcal{R}^{n\times m}} ||P_\Omega(\mathbf{X}) - P_\Omega(\mathbf{Y})||_F^2 + \lambda||\mathbf{X}||_*. \quad (3)$$

Ji et al. [16], Liu et al. [23] and Toh et al.[39] independently propose proximal gradient algorithms to improve the algorithm of [5] by significantly reducing the iteration steps. They obtain an ϵ-accurate solution in $O(1/\sqrt{\epsilon})$ steps. More efficient soft singular vector thresholding algorithms are proposed in [25, 26] by investigating the factorization property of the estimation matrix. Each step of the algorithms requires the computation of a partial SVD for the estimated matrix. In addition, several methods approximate the trace norm using its variational characterizations [29, 35, 43], and proceed by alternating optimization. However these methods lack global convergence guarantees.

All the methods above involve the computation of SVD iteratively, which is not scalable to large-scale problems. Recently, the coordinate gradient descent method has been demonstrated to be efficient in solving sparse learning problems in the vector case [10, 34, 41, 42]. 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. Dudík et al. [8] propose a coordinate gradient descent solution for the trace norm penalized problem. They recast the non-smooth objective in problem (3) as a smooth one in an infinite dimensional rank-one matrix space, then apply the coordinate gradient algorithm on the collection of rank-one matrices. Zhang et al. [44] further improve the efficiency using the boosting method, and the improved algorithm guarantees an ϵ-accuracy within $O(1/\epsilon)$ iterations. Although these algorithms need slightly more iterations than the proximal methods, they are more scalable as they only need to compute the top singular vector pair in each iteration. Note that the top singular vector pair can be computed efficiently by the power method or Lanczos iterations [12]. Jaggi et al. [15] use the same order of iterations as [44] by directly applying the Hazan's algorithm [13]. Tewari et al. [37] solve a more general problem based on a greedy algorithm. Shalev-Shwartz et al. [33] further reduce the number of iterations based on a heuristic without theoretical guarantees.

Most methods based on the top singular vector pair include 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 for constructing the target matrix. The main differences among these methods lie in how they refine the weights. The Jaggi's algorithm (JS) [15] directly applies the Hazan's algorithm, which relies on the Frank-Wolfe algorithm [9]. 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. The greedy efficient component optimization (GECO) [33] optimizes the weights by solving another time consuming optimization problem. It involves a smaller number of iterations than the JS algorithm. However, the sophisticated weight refinement makes it slower in total computational time. Similar to JS, Tewari et al. [37] use a small update step size for a general structure constrained problem. The lifted coordinate gradient descent algorithm (Lifted) [8] updates the rank-one matrix basis with a constant weight in each iteration, and conducts a lasso type algorithm [38] 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 [44]. The matrix norm boosting approach (Boost) [44] 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.

We summarize their common drawbacks as follows:

- The weight refinement steps are inefficient, resulting in a slow convergence rate. The current best convergence rate is $O(1/\epsilon)$. Some refinement steps themselves contain computationally expensive iterations [8, 44], which cannot run on large-scale data.

- They have heuristic-based tunable parameters which are not easy to use. However, these parameters severely affect their convergence speed and the approximation result. In some algorithms, an improper parameter even makes the algorithm diverge [5, 8].

In this paper, we propose a simple and efficient algorithm to solve the low rank matrix completion problem. The key idea is to extend the orthogonal matching pursuit (OMP) procedure [32] from the vector case to matrix case. In our algorithm, we use rank-one matrix as the update basis, which is generated by the left and right top singular vectors of the current approximation residual. And we fully update the weights for all rank-one matrices in the current basis set at the end of each iteration by orthogonally projecting the observation matrix onto their spanning subspace. The main computational cost of the proposed algorithm is to calculate the top singular vector pair of a sparse matrix, which costs $O((n+m)|\Omega|)$ operations in each iteration. Another important 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. Their convergence rate for sparse vector recovery is sub-linear as shown in [22]. See also [7], [36], [40] 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. To the best of our knowledge, this is the fastest algorithm among all related methods. We verify the efficiency of our algorithm empirically in large-scale matrix completion problems, such as MovieLens [28] and Netflix [3, 4].

The main contributions of our paper include:

- 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 get an ϵ-accuracy solution, and in each step we only need to compute the top singular vector pair, which can be computed efficiently.

- Our algorithm has only one free parameter, i.e., the rank of the approximation matrix. The proposed algorithm is guaranteed to converge, i.e., no risk of divergence.

Notions: Let $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_m) \in \Re^{n\times m}$ be an $n \times m$ real matrix, $\Omega \subset \{1, \cdots, n\} \times \{1, \cdots, m\}$ 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 Frobeius norm of $\mathbf{Y}$ is defined as $||\mathbf{Y}||_F = \sqrt{\sum_{i,j} \mathbf{X}_{i,j}^2}$. 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 $\mathbf{X}$ is defined as $\langle \mathbf{X}, \mathbf{Y}\rangle = \langle vec(\mathbf{X}), vec(\mathbf{Y})\rangle$.

2. ORTHOGONAL 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,

$$\mathbf{X} = \mathbf{M}(\boldsymbol{\theta}) = \sum_{i\in\mathcal{I}} \theta_i \mathbf{M}_i, \tag{4}$$

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

The original low rank matrix approximation problem can be reformulated as

$$\begin{array}{ll} \min\limits_{\boldsymbol{\theta}} & ||\boldsymbol{\theta}||_0 \\ s.t. & P_\Omega(\mathbf{M}(\boldsymbol{\theta})) = P_\Omega(\mathbf{Y}), \end{array} \tag{5}$$

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

If we reformulate the problem as

$$\begin{array}{ll} \min\limits_{\boldsymbol{\theta}} & ||P_\Omega(\mathbf{M}(\boldsymbol{\theta})) - P_\Omega(\mathbf{Y})||_F^2 \\ s.t. & ||\boldsymbol{\theta}||_0 \le r, \end{array} \tag{6}$$

we could solve it by an orthogonal matching pursuit type greedy algorithm using rank-one matrices as the basis. 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 pursue the basis; and the other one is to learn the weights of the basis. Before presenting this approach, we introduce some notations that will be used subsequently. 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 $\|\mathbf{A}\| = \sqrt{\langle \mathbf{A}, \mathbf{A}\rangle}$.

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 kth 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

$$\mathbf{X}_{k-1} \;=\; (\mathbf{M}(\boldsymbol{\theta}^{k-1}))_\Omega \;=\; \sum_{i=1}^{k-1} \boldsymbol{\theta}_i^{k-1} (\mathbf{M}_i)_\Omega.$$

Therefore, $\mathbf{M}_k$ can be chosen to be an optimal solution of the following problem:

$$\max_{\mathbf{M}} \{\langle \mathbf{M}, \mathbf{R}_k\rangle : \ \mathrm{rank}(\mathbf{M}) = 1, \ \|\mathbf{M}\|_F = 1\}. \tag{7}$$

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 (7) can be equivalently solved as

$$\max_{\mathbf{u},\mathbf{v}} \{\mathbf{u}^T \mathbf{R}_k \mathbf{v} : \ \|\mathbf{u}\| = \|\mathbf{v}\| = \mathbf{1}\}. \tag{8}$$

Clearly, the optimal solution $(\mathbf{u}_*, \mathbf{v}_*)$ of problem (8) is a pair of top left and right singular vectors of $\mathbf{R}_k$. It can be efficiently computed by the power method; a simple and efficient variant is given in the Appendix. 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 least squares regression problem

$$\min_{\boldsymbol{\theta}\in\Re^k} ||\sum_{i=1}^{k} \theta_i \mathbf{M}_i - \mathbf{Y}||_\Omega^2. \tag{9}$$

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 (9) is given by

$$\boldsymbol{\theta}^k = (\bar{\mathbf{M}}_\mathbf{k}^T \bar{\mathbf{M}}_\mathbf{k})^{-1} \bar{\mathbf{M}}_\mathbf{k}^T \dot{\mathbf{y}}, \tag{10}$$

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 $\bar{M}_k$ is the number of total observed entries. It is computationally expensive to directly calculate the matrix multiplication. We simplify this step by an incremental process, and give the implementation details in the Appendix.

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 approximation matrix or the approximation residual. In particular, one can choose a preferred rank of the approximate solution matrix and run the method for that number of iterations. Alternatively, one can stop the method once the residual $\|\mathbf{R}_k\|$ is less than a tolerance parameter ε. The main steps of Orthogonal Rank-One Matrix Pursuit (OR1MP) are given in algorithm 1.

REMARK 2.1. *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. [21] recently propose another similar algorithm called ADMiRA by extending the compressive sampling matching pursuit (CoSaMP) algorithm [30]. It requires iteratively computing hard singular value thresholding, which is computationally expensive especially for large-scale problems.*

Algorithm 1 Orthogonal Rank-One Matrix Pursuit

Input: $\mathbf{Y}$ and a tolerance parameter ε.
Initialize: Set $\mathbf{X}_0 = 0$, $\boldsymbol{\theta}^0 = 0$ 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 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 $\mathbf{X}_k = \sum_{i=1}^k \boldsymbol{\theta}_i^k(\mathbf{M}_i)_\Omega$ and $k \leftarrow k+1$.
until observed residual $\|R_k\|$ is smaller than ε
Output: Learned matrix $\hat{\mathbf{Y}} = \sum_{i=1}^k \boldsymbol{\theta}_i^k\mathbf{M}_i$.

3. CONVERGENCE ANALYSIS

In this section, we will show that our proposed orthogonal rank-one matrix pursuit algorithm converges in linear time. This main result is given in the following theorem.

THEOREM 3.1. *The orthogonal rank-one matrix pursuit algorithm satisfies*

$$||\mathbf{R}_k|| \;\leq\; \left(\sqrt{1-\frac{1}{\min(m,n)}}\right)^{k-1} \|Y\|_\Omega, \quad \forall k \geq 1.$$

Before proving Theorem 3.1, we need to establish some useful and preparatory properties of our orthogonal rank-one matrix pursuit algorithm.

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 (9). By the first-order optimality condition, one has

$$\langle \mathbf{Y} - \sum_{i=1}^t \boldsymbol{\theta}_i^k\mathbf{M}_i, \mathbf{M}_i\rangle_\Omega = 0 \text{ for } i = 1, \cdots, k,$$

which together with $\mathbf{R}_k = \mathbf{Y}_\Omega - \mathbf{X}_{k-1}$ and $\mathbf{X}_k = \sum_{i=1}^k \boldsymbol{\theta}_i^k(\mathbf{M}_i)_\Omega$ 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 $k \geq 1$,

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

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 (10) 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

$$(\mathbf{M}_i)_\Omega = \sum_{j=1}^{i-1} \boldsymbol{\alpha}_j(\mathbf{M}_j)_\Omega,$$

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 $\boldsymbol{\theta}_k^{k-1} = 0$ and let

$$\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1} + \boldsymbol{\eta}^k,$$

In view of (9), one can observe that

$$\boldsymbol{\eta}^k = \arg\min_{\boldsymbol{\eta}} ||\sum_{i=1}^k \boldsymbol{\eta}_i\mathbf{M}_i - \mathbf{R}_k||_\Omega^2. \tag{11}$$

Let

$$\mathbf{L}_k = \sum_{i=1}^k \boldsymbol{\eta}_i^k(\mathbf{M}_i)_\Omega. \tag{12}$$

By the definition of $\mathbf{X}_k$, one can also observe that

$$\mathbf{X}_k = \mathbf{X}_{k-1} + \mathbf{L}_k,$$
$$\mathbf{R}_{k+1} = \mathbf{R}_k - \mathbf{L}_k.$$

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* (12).

PROOF. Since $\mathbf{L}_k = \sum_{i\leq k} \boldsymbol{\eta}_i^k(\mathbf{M}_i)_\Omega$, it follows from Property 3.2 that $\langle \mathbf{R}_{k+1}, \mathbf{L}_k\rangle = 0$. We then have

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

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 (11), 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 $\mathbf{L}_k = \sum_{i \le k} \eta_i^k (\mathbf{M}_i)_\Omega$, we then obtain

$$||\mathbf{L}_k||^2 = \dot{\mathbf{r}}_k^T \bar{\mathbf{M}}_k (\bar{\mathbf{M}}_k^T \bar{\mathbf{M}}_k)^{-1} \bar{\mathbf{M}}_k^T \dot{\mathbf{r}}_k. \quad (13)$$

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 kth column of the matrix $\bar{M}_k$ and $\mathbf{m}_k$ is the reshaped vector of $(\mathbf{M}_k)_\Omega$. Recall that $\|\mathbf{M}_k\| = \|\mathbf{u}_k \mathbf{v}_k^T\| = 1$. Hence, $\|(\bar{\mathbf{M}}_k)_k\| \le 1$. Due to $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$, $\bar{\mathbf{M}}_k = \mathbf{QU}$ and the definition of $\mathbf{U}$, we have

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

In addition, by Property 3.2, we have

$$\bar{\mathbf{M}}_k^T \dot{\mathbf{r}}_k = [0, \cdots, 0, \langle \mathbf{M}_k, \mathbf{R}_k \rangle]^T. \quad (14)$$

Substituting $\bar{\mathbf{M}}_k = \mathbf{QU}$ into (13), and using $\mathbf{Q^TQ} = \mathbf{I}$ and (14), we obtain that

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

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

We are now ready to prove Theorem 3.1.

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

$$\begin{aligned} \langle \mathbf{M}_k, \mathbf{R}_k \rangle \ &= \ \langle \mathbf{u}^k (\mathbf{v}^k)^T, \mathbf{R}_k \rangle \ = \ \sigma_{\max}(\mathbf{R}_k) \\ &\ge \ \sqrt{\frac{\sum_i \sigma_i^2(\mathbf{R}_k)}{\mathrm{rank}(\mathbf{R}_k)}} \ = \ \sqrt{\frac{\|\mathbf{R}_k\|^2}{\mathrm{rank}(\mathbf{R}_k)}} \ \ge \ \sqrt{\frac{\|\mathbf{R}_k\|^2}{\min(m,n)}}. \end{aligned}$$

Using this inequality and Property 3.5, we obtain that

$$\begin{aligned} ||\mathbf{R}_{k+1}||^2 \ &= \ ||\mathbf{R}_k||^2 - ||\mathbf{L}_k||^2 \ \le \ ||\mathbf{R}_k||^2 - \langle \mathbf{M}_k, \mathbf{R}_k \rangle^2 \\ &\le \ (1 - \tfrac{1}{\min(m,n)}) ||\mathbf{R}_k||^2. \end{aligned}$$

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

$$||\mathbf{R}_k|| \ \le \ \left(\sqrt{1 - \frac{1}{\min(m,n)}} \right)^{k-1} \|\mathbf{Y}\|_{\mathbf{\Omega}}.$$

This completes the proof. □

REMARK 3.6. *In a standard study of the convergence rate of the Orthogonal Match Pursuit(OMP) or Orthogonal Greedy Algorithm(OGA), the term* $|\langle \mathbf{M}_k, \mathbf{R}_k \rangle| \ge \|\mathbf{R}_k\|^2$ *which leads a sublinear convergence. Our* $\mathbf{M}_k$ *is a data dependent construction which is based on the top left and right singular vectors of the residual matrix* $\mathbf{R}_k$*. It has a better estimate which gives us the linear convergence.*

4. EXPERIMENTS

In this section, we compare our orthogonal rank-one matrix pursuit (OR1MP) algorithm with state-of-the-art matrix completion methods. The competing algorithms include: singular value projection (SVP) [27], singular value thresholding (SVT) [6], Jaggi's fast algorithm for trace norm constraint (JS) [15], spectral regularization algorithm (SoftImpute) [26], low rank matrix fitting (LMaFit) [43], boosting type accelerated matrix-norm penalized solver (Boost) [44] and greedy efficient component optimization (GECO) [33]. The first three solve trace norm constrained problems; the next three solve trace norm penalized problems; the last one directly solves the low rank constrained problem. The general greedy method [37] is not included in our comparison, as it includes JS and GECO as special cases for matrix completion. The lifted coordinate descent method (Lifted) [8] is not included in our comparison as it is very sensitive to the parameters and is less efficient than Boost proposed in [44].

The code for most of these methods are available online:

- singular value projection (SVP): http://www.cs.utexas.edu/∼pjain/svp/
- singular value thresholding (SVT): http://svt.stanford.edu/
- spectral regularization algorithm (SoftImpute): http://www-stat.stanford.edu/∼rahulm/software.html
- low rank matrix fitting (LMaFit): http://lmafit.blogs.rice.edu/
- boosting type solver (Boost): http://webdocs.cs.ualberta.ca/∼xinhua2/boosting.zip
- greedy efficient component optimization (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. All the competing methods are implemented in MATLAB[1] and call some external packages for fast 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.

To set the parameters in the following experiments, we follow the recommended settings for competing algorithms. If no recommended parameter value is available, we choose the best one from a candidate set using cross validation. For our OR1MP algorithm, we only need a stop criterion. For simplicity, we stop our algorithm after r iterations. In this way, we approximate the ground truth using a rank r matrix. We present the experimental results using three metrics, *peak signal-to-noise ratio* (PSNR)[14], *normalized mean absolute error* (NMAE) [11] and *root-mean-square error* (RMSE) [18]. PSNR is a test metric specific for images. A higher value in PSNR generally indicates a better quality [14]. NMAE is a metric for recommendation systems. RMSE is a general metric for prediction. NMAE and RMSE measure the approximation error of the corresponding result.

4.1 Efficiency and Convergence

We firstly evaluate the efficiency of our algorithm. The results are reported for image recovery as well as the largest publicly available recommendation dataset, Netflix [3, 4,

[1]We warp GECO into MATLAB.

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

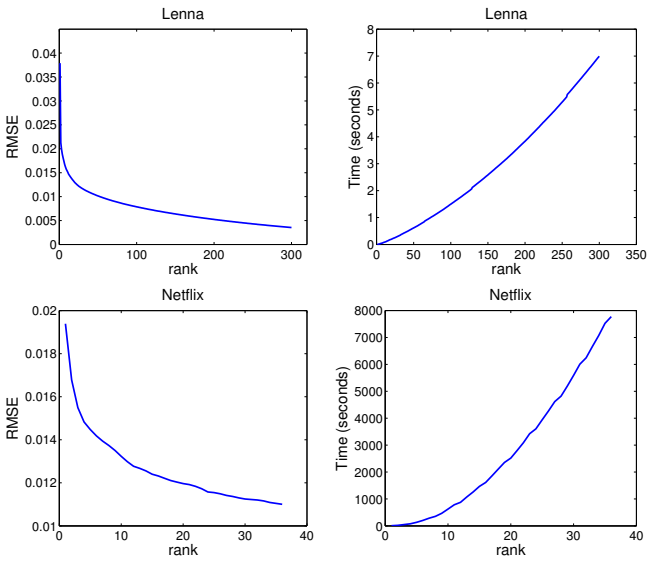

Figure 1: Illustration of convergence of the proposed algorithm on Lenna image and Netflix dataset: the x-axis is the rank, the y-axis is the RMSE (left column), and the running time measured in second (right column).

18]. The Netflix dataset is available at the Netflix website[3]. It has 10^8 ratings of 17,770 movies by 480,189 Netflix customers. This is a large-scale dataset, and most of the competing methods are not applicable for this dataset. The results in Figure 1 show that our method rapidly reduces the approximation error, which is consistent with our theoretical analysis.

4.2 Image Recovery

In the image recovery experiments, we use the following benchmark test images: Lenna, Barbara, Cameraman, Clown, Couple, Crowd, Girl, Man, Peppers[4]. The size of each image is 512×512. We randomly exclude 50% of the pixels in the image, and the remaining ones are used as the observations. The numerical results in terms of the PSNR and the RMSE are separately listed in Table 1 and Table 2. The results show SVT and our OR1MP achieve the best numerical performance. We also present the recovered images for Lenna in Figure 2. Image recovery needs a relatively higher approximation rank; both GECO and Boost fail to find a good recovery in most cases, so we do not include them in the result tables.

4.3 Recommendation

Table 3: Recommendation Data Sets.

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

[3]http://www.netflixprize.com/

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

In the following experiments, we compare the different matrix completion algorithms using large recommendation datasets, Jester [11] and MovieLens [28]. In these experiments, we use six datasets including, Jester1, Jester2, Jester3, MovieLens100K, MovieLens1M, and MovieLens10M. The statistics of these datasets are given in Table 3. 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[5]. They contains 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. The results in RMSE and NMAE are given in Table 4 and Table 5. We also show the running time of different methods in Table 6. We can observe from the above experiments that our algorithm is the fastest among all competing methods to obtain satisfactory results. We can also observe that our method obtains comparable results with a much lower rank.

5. 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. However, our extension is nontrivial as we build up a set of rank one matrices dependent on the given incomplete entries of the unknown matrix. Our algorithm is computationally inexpensive for each matrix pursuit iteration, and finds a satisfactory result in a few iterations. Another advantage of our method is it has 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 our method achieves a linear convergence rate, which is significantly better than the previous known result (a sub-linear convergence rate). We also empirically compare it with state-of-the-art low rank matrix completion algorithms, and our results show that the proposed algorithm is more efficient than competing algorithms. Our method can be easily adapted to optimize other forms of convex functions, by substituting the residual with the gradient. We plan to generalize our theoretical and empirical analysis to other loss functions in the future.

6. REFERENCES

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

[2] F. Bach. Consistency of trace norm minimization. *Journal of Machine Learning Research*, 9:1019–1048, 2008.

[3] R. Bell and Y. Koren. Lessons from the netflix prize challenge. *SIGKDD Explorations*, 9(2), 2007.

[4] J. Bennett and S. Lanning. The netflix prize. In *In Proceedings of KDD Cup and Workshop*, 2007.

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

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

Table 1: Image recovery results measured in terms of the peak signal-to-noise ratio (PSNR).

Data Set	SVT	SVP	SoftImpute	LMaFit	JS	OR1MP
Lena	28.1832	25.4586	26.7022	23.2003	24.5056	28.0115
Barbara	26.9635	25.2598	25.6073	25.9589	23.5322	26.5314
Cameraman	25.6273	25.9444	26.7183	24.8956	24.6238	27.8565
Clown	28.5644	19.0919	26.9788	27.2748	25.2690	28.1963
Couple	23.1765	23.7974	26.1033	25.8252	24.4100	27.0707
Crowd	26.9644	22.2959	25.4135	26.0662	18.6562	26.0535
Girl	29.4688	27.5461	27.7180	27.4164	26.1557	30.0878
Goldhill	28.3097	16.1256	27.1516	22.4485	25.9706	28.5646
Man	27.0223	25.3246	25.7912	25.7417	23.3060	26.5829
Peppers	25.7202	26.0223	26.8475	27.3663	24.0979	28.0781

Table 2: Image recovery results measured in terms of the RMSE.

Data Set	SVT	SVP	SoftImpute	LMaFit	JS	OR1MP
Lena	0.0390	0.0533	0.0462	0.0692	0.0595	0.0398
Barbara	0.0449	0.0546	0.0524	0.0504	0.0666	0.0471
Cameraman	0.0547	0.0504	0.0460	0.0569	0.0587	0.0404
Clown	0.0373	0.1110	0.0448	0.0433	0.0545	0.0389
Couple	0.0677	0.0646	0.0493	0.0511	0.0602	0.0443
Crowd	0.0449	0.0768	0.0536	0.0497	0.1167	0.0498
Girl	0.0336	0.0419	0.0411	0.0426	0.0492	0.0313
Goldhill	0.0384	0.1562	0.0439	0.0754	0.0503	0.0373
Man	0.0446	0.0542	0.0513	0.0516	0.0683	0.0469
Peppers	0.0586	0.0500	0.0455	0.0428	0.0624	0.0395

Table 4: Recommendation results measured in terms of RMSE. Boost fails on MovieLens10M.

Data Set	SVP	SoftImpute	LMaFit	Boost	JS	GECO	OR1MP
Jester1	4.7311	5.1113	4.7623	5.1746	4.4713	4.3680	4.3418
Jester2	4.7608	5.1646	4.7500	5.2319	4.5102	4.3967	4.3649
Jester3	8.6958	5.4348	9.4275	5.3982	4.6866	5.1790	4.9783
MovieLens100K	0.9683	1.0354	1.2308	1.1244	1.0146	1.0243	1.0168
MovieLens1M	0.9085	0.8989	0.9232	1.0850	0.9290	1.1439	0.9595
MovieLens10M	0.8611	0.8534	0.8825	–	0.8928	0.8668	0.8621

Table 5: Recommendation results measured in terms of normalized mean absolute error (NMAE).

Data Set	SVP	SoftImpute	LMaFit	Boost	JS	GECO	OR1MP
Jester1	0.1759	0.2141	0.1772	0.2259	0.1758	0.1683	0.1682
Jester2	0.1761	0.2168	0.1761	0.2196	0.1765	0.1681	0.1681
Jester3	0.3067	0.2314	0.3289	0.2298	0.1933	0.2020	0.1994
MovieLens100K	0.1886	0.2083	0.2309	0.2363	0.2026	0.1992	0.2011
MovieLens1M	0.1765	0.1754	0.1786	0.2259	0.2246	0.1797	0.1901
MovieLens10M	0.1454	0.1447	0.1481	–	0.1775	0.1757	0.1563

Table 6: The running time (measured in seconds) for all methods on all recommendation datasets.

Data Set	SVP	SoftImpute	LMaFit	Boost	JS	GECO	OR1MP
Jester1	18.3495	161.4941	3.6756	93.9142	29.6751	$> 10^4$	1.8317
Jester2	16.8519	152.9600	2.4237	261.7005	28.5228	$> 10^4$	1.6769
Jester3	16.5801	1.5450	8.4513	245.7895	12.9441	$> 10^3$	0.9264
MovieLens100K	1.3237	128.0658	2.7613	2.8669	2.8583	10.8300	0.0418
MovieLens1M	18.9020	59.5600	30.5475	93.9142	13.0972	$> 10^4$	0.8714
MovieLens10M	$> 10^3$	$> 10^3$	$> 10^5$	–	130.1343	$> 10^5$	23.0513

Figure 2: The original, observed images and images recovered by different methods on the Lenna image.

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

[7] R. A. DeVore and V. N. Temlyakov. Some remarks on greedy algorithms. *Advances in computational Mathematics*, 5:173–187, 1996.

[8] M. Dudík, Z. Harchaoui, and J. Malick. Lifted coordinate descent for learning with trace-norm regularization. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

[9] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.

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

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

[12] G. H. Golub and C. F. V. Loan. *Matrix computations (3rd ed.)*. The Johns Hopkins University Press, 1996.

[13] E. Hazan. Sparse approximate solutions to semidefinite programs. In *Proceedings of the 8th Latin American conference on Theoretical informatics*, 2008.

[14] Q. Huynh-Thu and M. Ghanbari. Scope of validity of psnr in image/video quality assessment. *Electronics Letters*, 44(13):800–801, 2008.

[15] M. Jaggi and M. Sulovský. A simple algorithm for nuclear norm regularized problems. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 471–478, 2010.

[16] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, pages 457–464, 2009.

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

[18] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2008.

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

[20] M.-J. Lai, Y. Xu, and W. Yin. Improved iteratively reweighted least squares for unconstrained smoothed ℓ_q minimization,. *SIAM Journal on Numerical Analysis*, 2012.

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

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

[23] Y.-J. Liu, D. Sun, and K.-C. Toh. An implementable proximal point algorithmic framework for nuclear norm minimization. *Mathematical Programming*, 133(1-2):399–436, 2012.

[24] Z. Lu and Y. Zhang. Penalty decomposition methods for rank minimization. *http://arxiv.org/abs/0910.5260*, 2010.

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

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

[27] R. Meka, P. Jain, and I. S. Dhillon. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems (NIPS) 22*, pages 937–945, 2010.

[28] B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan, and J. Riedl. Movielens unplugged: experiences with an occasionally connected recommender system. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 263–266, 2003.

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

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

[31] S. Negahban and M. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010.

[32] Y. C. Pati, R. Rezaiifar, Y. C. P. R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of the 27th Annual Asilomar Conference on Signals, Systems, and Computers*, pages 40–44, 1993.

[33] S. Shalev-Shwartz, A. Gonen, and O. Shamir. Large-scale convex minimization with a low-rank constraint. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 329–336, 2011.

[34] S. Shalev-Shwartz and A. Tewari. Stochastic methods for l1 regularized loss minimization. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, pages 929–936, 2009.

[35] N. Srebro, J. Rennie, and T. Jaakkola. Maximum margin matrix factorizations. In *Advances in Neural Information Processing Systems (NIPS) 17*, 2005.

[36] V. N. Temlyakov. Greedy approximation. *Acta Numerica*, 17:235–409, 2008.

[37] A. Tewari, P. Ravikumar, and I. S. Dhillon. Greedy algorithms for structurally constrained high dimensional problems. In *Advances in Neural Information Processing Systems (NIPS) 23*, 2011.

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

[39] K.-C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *Optimization Online*, 2009.

[40] J. A. Tropp. Greed is good: algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50:2231–2242, 2004.

[41] T. T. Wu and K. Lange. Coordinate descent algorithms for lasso penalized regression. *Annals of Applied Statistics*, 2(1):224–244, 2008.

[42] S. Yun and K.-C. Toh. A coordinate gradient descent method for l1-regularized convex minimization. *Computational Optimization and Applications*, 2011.

[43] W. Y. Zaiwen Wen and Y. Zhang. 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.

[44] X. Zhang, Y. Yu, and D. Schuurmans. Accelerated training for matrix-norm regularization: A boosting approach. In *Advances in Neural Information Processing Systems (NIPS) 24*, 2012.

APPENDIX

A. TOP SINGULAR VECTOR PAIR

In our algorithm we calculate the top singular vector pair by the following problem:

$$\min_{||\mathbf{u}||^2=1,||\mathbf{v}||^2=1} ||\mathbf{R}_k - \sigma \mathbf{u}\mathbf{v}^T||^2,$$

where $\mathbf{R}_k$ is the residual and σ is the dominant singular value. We use the first order condition to solve the problem. With a initial vector $\mathbf{u}$, we iteratively update $\mathbf{v}$ and $\mathbf{u}$ using following steps,

$$\mathbf{v} = \mathbf{u}^T\mathbf{R}_k/||\mathbf{u}||^2 \quad \text{and} \quad \mathbf{u} = \mathbf{R}_k\mathbf{v}^T/||\mathbf{v}||^2.$$

These updates quickly converge. In large-scale problems, we can calculate the approximated singular vector pair with fixed number of iterations.

B. INVERSE MATRIX UPDATE

In our algorithm, we use least square solution to update the weights for the rank-one matrix bases. In this step, we need to calculate a inverse matrix $(\bar{\mathbf{M}}_\mathbf{k}\bar{\mathbf{M}}_\mathbf{k})^{-1}$. To directly compute this inverse is computationally expensive, as the matrix $\bar{\mathbf{M}}_\mathbf{k}$ has large row size. We implement this efficiently with an incremental method. As

$$\bar{\mathbf{M}}_\mathbf{k}^T\bar{\mathbf{M}}_\mathbf{k} = [\bar{\mathbf{M}}_{\mathbf{k-1}}, \dot{\mathbf{m}}_k]^T[\bar{\mathbf{M}}_{\mathbf{k-1}}, \dot{\mathbf{m}}_k]$$

The inverse can be written in block matrix form

$$(\bar{\mathbf{M}}_\mathbf{k}^T\bar{\mathbf{M}}_\mathbf{k})^{-1} = \begin{bmatrix} \bar{\mathbf{M}}_{\mathbf{k-1}}^T\bar{\mathbf{M}}_{\mathbf{k-1}} & \bar{\mathbf{M}}_{\mathbf{k-1}}^T\dot{\mathbf{m}}_k \\ \dot{\mathbf{m}}_k^T\bar{\mathbf{M}}_{\mathbf{k-1}}^T & \dot{\mathbf{m}}_k^T\dot{\mathbf{m}}_k \end{bmatrix}^{-1}$$

Then it is calculated as

$$\begin{bmatrix} \mathbf{A} + d\mathbf{A}\mathbf{b}\mathbf{b}^T\mathbf{A} & -d\mathbf{A}\mathbf{b} \\ -d\mathbf{b}^T\mathbf{A} & d \end{bmatrix}$$

where $\mathbf{A} = \bar{\mathbf{M}}_{\mathbf{k-1}}^T\bar{\mathbf{M}}_{\mathbf{k-1}}$ is the corresponding inverse matrix in the last step, $\mathbf{b} = \bar{\mathbf{M}}_{\mathbf{k-1}}^T\dot{\mathbf{m}}_k$ is a vector with $|\Omega|$ elements, and $d = (\mathbf{b}^T\mathbf{b} - \mathbf{b}^T\mathbf{A}\mathbf{b})^{-1} = 1/(\mathbf{b}^T\mathbf{b} - \mathbf{b}^T\mathbf{A}\mathbf{b})$ is a scalar.

$\bar{\mathbf{M}}_\mathbf{k}^T\dot{\mathbf{y}}$ is also calculated incrementally by $[\bar{\mathbf{M}}_{\mathbf{k-1}}^T\dot{\mathbf{y}}, \dot{\mathbf{m}}_k^T\dot{\mathbf{y}}]$, as $\dot{\mathbf{y}}$ is fixed.