A Scalable Approach to Column-Based Low-Rank Matrix Approximation[*]

Yifan Pi[1], **Haoruo Peng**[2], **Shuchang Zhou**[3,4], **Zhihua Zhang**[5]

[1]Institute for Theoretical Computer Science, IIIS, Tsinghua University, Beijing, China, 100084

[2]Department of Computer Science & Technology, Tsinghua University, Beijing, China, 100084

[3]State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, 100190

[4]Google Inc., Beijing, China, 100084

[5] College of Computer Science & Technology, Zhejiang University, Hangzhou, China, 310027

piyifan@gmail.com penghaoruo@hotmail.com shuchang.zhou@gmail.com zhzhang@cs.zju.edu.cn

Abstract

In this paper, we address the column-based low-rank matrix approximation problem using a novel parallel approach. Our approach is based on the divide-and-combine idea. We first perform column selection on submatrices of an original data matrix in parallel, and then combine the selected columns into the final output. Our approach enjoys a theoretical relative-error upper bound. In addition, our column-based low-rank approximation partitions data in a deterministic way and makes no assumptions about matrix coherence. Compared with other traditional methods, our approach is scalable on large-scale matrices. Finally, experiments on both simulated and real world data show that our approach is both efficient and effective.

1 Introduction

In multivariate data analysis, the data in question is usually defined as an $m \times n$ feature-instance matrix; that is, each column represents an instance of m features. Examples of feature-instance datasets include environmental measurements and genomes, words and documents, face images and people, etc. Typically, finding a parsimonious approximation of a data matrix assists in the efficient recognition of valuable information behind the instances. A classical approach is to use the singular value decomposition (SVD) to obtain a low rank approximation of the matrix, such as in the case of eigenface and eigengene.

Although the SVD method is widely used for low-rank approximation of a data matrix, it is hard to assign the resulting singular vectors meaning in terms of the field from which the data is generated. This makes the task of understanding and interpreting the data in question very difficult. Compared to SVD, column-based approximation methods, representing a data matrix in terms of a small number of actual columns and/or actual rows of the matrix, have been shown to have advantage of analyzing data [Goreinov *et al.*, 1997a; 1997b; Tyrtyshnikov, 2000; Drineas, 2003; Drineas and Mahoney, 2005; Drineas *et al.*, 2006].

Many algorithms, both randomized and deterministic, have been developed to find the proper columns of the column-based approximation [Boutsidis *et al.*, 2009; Kannan and Vempala, 2009; Boutsidis *et al.*, 2011; Mahoney, 2011; Çivril and Magdon-Ismail, 2012; Guruswami and Sinop, 2012]. Drineas *et al.* [2008] devised randomized algorithms with relative error by sampling sufficiently many columns. In particular, Drineas *et al.*'s algorithm has $(1+\epsilon)$ relative-error ratio with high probability (w.h.p.). Unfortunately, relative-error bound algorithms for a general matrix often require the use of SVD as a subprocess when performing computation on the data matrix. This limits scalability of such algorithms.

To solve the scalability problem resulted from SVD, numerous algorithms have been proposed. While some approaches focus on speeding up underlying numerical iterations [Golub and Loan, 1996], most are numerically unstable and hard to implement. Randomized algorithms with fast running times and robust performance have enjoyed much interest in the research community, e.g., see Halko *et al.* [2011] for a review. A recent unpublished work claims to approximate low-rank SVD randomly in input sparsity time with relative error bound [Clarkson and Woodruff, 2012].

We propose a novel method to perform column-based approximation on large scale data matrices. Our work is motivated by the divide-and-combine methodology, which was also studied by Mackey *et al.* [2011] when improving scalability of both matrix completion and robust PCA. Informally,

[*]Z.Z. is the corresponding author, and he acknowledges support from the Natural Science Foundations of China (No. 61070239) and the Google visiting faculty program. This work was done when Y.P. and H.P. were research interns in Google Beijing. Y.P. was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61061130540.

given some conditions, we divide the original matrix into smaller submatrices, factorize these submatrices in parallel, then combine the results from the factorization to obtain a final matrix consistent with the conditions (in this work, our conditions correspond to some specific number of columns).

Our framework is fully deterministic as long as the base algorithms used are deterministic. Our approach does not require the data matrix to satisfy any coherence conditions, so it applies to general matrices. We note that the size of matrices becomes $m \times \lceil\sqrt{nk}\rceil$ rather than $m \times n$, where k is a low rank parameter which is often regarded as a small constant in practice [Mahoney and Drineas, 2009]. Thus, our framework is scalable. In addition, any sparse input matrix can be handled by our algorithm flexibly. During the execution of the algorithm, only small-size dense matrices will be produced at any given step.

The rest of this paper is organized as follows. In Section 2 we present some preliminary backgrounds. In Section 3 we describe our approach. Section 4 establishes a relative error bound on general matrices. Section 5 gives an empirical analysis of our approach. Finally, we conclude our work in Section 6.

2 Preliminaries

This section presents some notational conventions that we will use in this paper. The norm function $\|\cdot\|_\xi$ can be either the Frobenius norm (denoted $\|\cdot\|_{\mathrm{F}}$) or the spectral norm (denoted $\|\cdot\|_2$). Let $\mathbf{I}_n$ be the $n \times n$ identity matrix. For a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$, we denote its rank by $\mathrm{rank}(\mathbf{M})$ and its Moore-Penrose pseudoinverse by $\mathbf{M}^+$. Let $\sigma_1(\mathbf{M}) \geq \sigma_2(\mathbf{M}) \geq ... \geq \sigma_r(\mathbf{M}) > 0$ be r positive singular values of $\mathbf{M}$. Let $\mathbf{u}_i(\mathbf{M}) \in \mathbb{R}^m$ and $\mathbf{v}_i(\mathbf{M}) \in \mathbb{R}^n$ be respectively the left singular vector and right singular vector of $\mathbf{M}$ associated with σ_i for all $1 \leq i \leq r$. Given a matrix $\mathbf{C} \in \mathbb{R}^{m\times r}$ with $r > k$, $\Pi^\xi_{\mathbf{C},k}(\mathbf{M})$ minimizes the residual $\|\mathbf{M} - \widehat{\mathbf{M}}\|_\xi$ over all $\widehat{\mathbf{M}}$ in the column space of $\mathbf{C}$ that have rank at most k. For column submatrices, we follow the Matlab notation: $\mathbf{M}_{(:,J)}$ represents the submatrix formed by columns in index set J of $\mathbf{M}$. Note that we also use $\mathbf{C}$ and $\mathbf{B}$ to represent column submatrices if the base matrix is implicit according to the context. We often use k as the rank parameter and r as the number of columns to be chosen.

We now introduce three matrix factorization algorithms that we will refer to as *base algorithms* for short.

2.1 Truncated Singular Value Decomposition

Singular Value Decomposition (SVD) is often used when we want to have a concise approximation of a data matrix. It is a well-known result that for $\mathbf{M} \in \mathbb{R}^{m\times n}$ and $k \leq \min\{m, n\}$,

$$\mathbf{M}_k \triangleq \sum_{i=1}^{k} \sigma_i(\mathbf{M})\mathbf{u}_i(\mathbf{M})\mathbf{v}_i(\mathbf{M})^\top = \operatorname*{argmin}_{\mathrm{rank}(\mathbf{X})\leq k} \|\mathbf{M} - \mathbf{X}\|_\xi.$$

The classical approach to such a best rank-k approximation of $\mathbf{M}$ is to truncate the result of the full SVD to obtain the first top k singular values and singular vectors. The running time for this approach is $O(mn\min\{m, n\})$. There are also other numerical methods such as the Krylov subspace method [Golub and Loan, 1996] to calculate the best rank-k approximation in $O(mnk)$ time.

2.2 Column-based Matrix Factorization (CX)

In contrast to the k-rank approximation of the matrix in the SVD approach, the CX approach focuses on choosing a subset of r columns for some $k \leq r \leq n$, whose span should approximate $\mathbf{M}$ as close as possible to $\mathbf{M}_k$. Suppose those columns we choose form a matrix $\mathbf{C} \in \mathbb{R}^{m\times r}$. We aim to reconstruct $\mathbf{M}$ via the column space of $\mathbf{C}$. We note that a common way of reconstructing the matrix $\mathbf{M}$ is to compute $\mathbf{CC}^+\mathbf{M}$, but the rank of this matrix may be larger than k. In order to ensure the reconstructed matrix also has rank k, Boutsidis *et al.* [2011] devised methods to find a matrix $\Pi^\xi_{\mathbf{C},k}(\mathbf{M})$. Unless mentioned explicitly, we assume the reconstructed matrix to be $\mathbf{CC}^+\mathbf{M}$ below. However, in our theoretical analysis on the error upper bound, we consider the case when the reconstructed matrix is $\Pi^\xi_{\mathbf{C},k}(\mathbf{M})$ because $\|\mathbf{M} - \mathbf{CC}^+\mathbf{M}\|_\xi \leq \|\mathbf{M} - \Pi^\xi_{\mathbf{C},k}(\mathbf{M})\|_\xi$ as long as $k \leq r$.

Letting $\mathbf{C}$ be the matrix consisting of r columns of $\mathbf{M}$, we want to relate either $\|\mathbf{M} - \mathbf{CC}^+\mathbf{M}\|_\xi$ or $\|\mathbf{M} - \Pi^\xi_{\mathbf{C},k}(\mathbf{M})\|_\xi$ to $\|\mathbf{M} - \mathbf{M}_k\|_\xi$. Based on the relative-error approximation, we have $\|\mathbf{M} - \mathbf{CC}^+\mathbf{M}\|_\xi \leq \|\mathbf{M} - \Pi^\xi_{\mathbf{C},k}(\mathbf{M})\|_\xi \leq \Delta\|\mathbf{M} - \mathbf{M}_k\|_\xi$ for some Δ which is a function of r, k and the size of the matrix. If an algorithm is randomized, the above bound is obtained in the expectation sense. Note that as r becomes larger, Δ becomes smaller. This gives rise to a more accurate approximation. Please see Figure 1 of Guruswami and Sinop [2012] for comparison on the error bounds and the running times of various CX algorithms.

There are also algorithms that yield an additive error bound [Kannan and Vempala, 2009], i.e., $\|\mathbf{M} - \mathbf{M}_k\|_\xi + \varepsilon\|\mathbf{M}\|_\xi$. These algorithms run in linear time by using some trivial sampling methods, but they result in a huge error when $\|\mathbf{M}\|_\xi$ is much larger than $\|\mathbf{M} - \mathbf{M}_k\|_\xi$. In this paper we focus on algorithms that produce a relative-error bound.

2.3 Interpolative Decomposition (ID)

Interpolative Decomposition (ID) selects a collection of k columns from a rank-k matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ that span the column space of $\mathbf{M}$. Formally, ID finds a column index set $J = \{j_1, .., j_k\}$ such that $\mathbf{M} = \mathbf{M}_{(:,J)}\mathbf{X}$, where $\mathbf{X} \in \mathbb{R}^{k\times n}$ satisfies $\mathbf{X}_{(:,J)} = \mathbf{I}_k$ and has no entry with magnitude larger than two. When the Gu-Eisenstat deterministic algorithm is used to produce ID [Gu and Eisenstat, 1996; Cheng *et al.*, 2005], the running time is close to $O(mnk)$ for an $m \times n$ rank-k matrix, though it can be higher (i.e., $O(mn\min\{m, n\})$) in rare cases [Halko *et al.*, 2011].

3 Methodology

In this section, we first give the framework of our divide-and-combine column-based matrix approximation method, and then present its running time analysis and discuss the related work.

3.1 Framework

Given a matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$, our purpose is to choose r columns to approximate $\mathbf{M}_k$ ($k \leq \mathrm{rank}(\mathbf{M})$, $k \leq r \leq n$). Our method consists of four stages which are described as follows.

Divide Given an integer $t > 0$ such that $\lfloor n/t \rfloor \geq k$, we split $\mathbf{M}$ into t l-column submatrices in any order [1], i.e., $\mathbf{M} = [\mathbf{M}^{(1)}, ..., \mathbf{M}^{(t)}]$. Unlike the method of Mackey *et al.* [2011], our approach does not need to randomly permute the order of columns. Thus, determinism of our approach only depends on the base algorithms.

Approximation We perform t rank-k approximation algorithms (e.g., the best rank-k approximation through SVD) on $\{\mathbf{M}^{(1)}, ..., \mathbf{M}^{(t)}\}$ in parallel. It yields t rank-k matrices $\{\widehat{\mathbf{M}}^{(1)}, ..., \widehat{\mathbf{M}}^{(t)}\}$.

Selection In parallel, we run the ID algorithm on each $\widehat{\mathbf{M}}^{(i)}$ (a total of t parallel instances). This results in, each i, $\widehat{\mathbf{M}}^{(i)} = \widehat{\mathbf{C}}^{(i)}\widehat{\mathbf{X}}^{(i)}$ where $\widehat{\mathbf{X}}^{(i)}$ is a $k \times l$ matrix such that no entry of $\widehat{\mathbf{X}}^{(i)}$ has magnitude larger than two. We let J_i denote the set of indices of the columns in $\widehat{\mathbf{C}}^{(i)}$ that corresponds to columns in $\widehat{\mathbf{M}}^{(i)}$.

Ensemble We first form an ensemble matrix $\mathbf{B} = \mathbf{M}_{(:, J_1 \cup ... \cup J_t)} \in \mathbb{R}^{m \times s}$ where $s = kt$.[2] Next, we run the column based base approximation algorithm on $\mathbf{B}$. [3] We accordingly have $\mathbf{B} \approx \mathbf{C}\mathbf{C}^{+}\mathbf{B}$ where $\mathbf{C}$ contains r columns of $\mathbf{M}$. $\mathbf{C}$ is the final output, i.e., $\mathbf{M} \approx \mathbf{C}\mathbf{C}^{+}\mathbf{M}$.

Adaptive Sampling (Optional) The purpose of this step is to sample more columns to $\mathbf{C}$ to obtain a tighter theoretical error bound. We can omit this step in practice. This step was introduced in [Deshpande *et al.*, 2006]. At first, we compute $\mathbf{M}'$ as $\mathbf{M}' = \mathbf{M} - \mathbf{C}\mathbf{C}^{+}\mathbf{M}$. Since $\mathbf{C}$ is a small matrix, the pseudoinverse of $\mathbf{C}$ and the matrix multiplications can be computed very fast. Let $p_i = \|\mathbf{m}'_\mathbf{i}\|_2^2 / \|\mathbf{M}'\|_\mathrm{F}^2$ where $\mathbf{m}'_\mathbf{i}$ is the i-th column of $\mathbf{M}'$. Then, sample further s' columns from $\mathbf{M}$ in s' i.i.d. trials, where in each trial the i-th column is chosen with probability p_i. Finally, add these s' columns to $\mathbf{C}$ to obtain the new output. The choice of the parameter s' is discussed in Section 4.

Note that the rank-k approximation of $\mathbf{M}^{(i)}$ is generally returned in factorized form, i.e., $\widehat{\mathbf{M}}^{(i)} = \widehat{\mathbf{R}}^{(i)}\widehat{\mathbf{T}}^{(i)}$ where $\widehat{\mathbf{R}}^{(i)} \in \mathbb{R}^{m \times k}$ and $\widehat{\mathbf{T}}^{(i)} \in \mathbb{R}^{k \times l}$. Thus, when the ID algorithm is performed on $\widehat{\mathbf{T}}^{(i)}$, we find an index set J_i such that $\widehat{\mathbf{T}}^{(i)} = \widehat{\mathbf{T}}^{(i)}_{(:,J_i)}\widehat{\mathbf{X}}^{(i)}$. From this, we know $\widehat{\mathbf{M}}^{(i)} = \widehat{\mathbf{R}}^{(i)}\widehat{\mathbf{T}}^{(i)}_{(:,J_i)}\widehat{\mathbf{X}}^{(i)} = \widehat{\mathbf{M}}^{(i)}_{(:,J_i)}\widehat{\mathbf{X}}^{(i)} = \widehat{\mathbf{C}}^{(i)}\widehat{\mathbf{X}}^{(i)}$. The trick was devised by Halko *et al.* [2011] to perform the ID on the smaller matrix. We also employ this trick in the selection stage.

Additionally, note that while $\mathbf{B}$ in the ensemble stage can be used as prediction of the column space of $\mathbf{M}$, it may contain too many columns and become large when $\mathbf{M}$ is split into more matrices. Thus, in the ensemble stage $\mathbf{B}$ should be resampled to obtain r columns in order to maintain consistency with the base CX algorithm.

It is worth pointing out that our framework can also handle the case that input matrix $\mathbf{M}$ is sparse. Particularly, in the approximation stage, we can employ a suitable algorithm for the sparse matrix. As for the selection stage, there is no problem because we only need to do ID on a small dense matrix $\widehat{\mathbf{T}}^{(i)}$ as mentioned earlier. Finally, in the ensemble stage, the matrix $\mathbf{B}$ is sparse because it is a submatrix of $\mathbf{M}$.

3.2 Running Time Analysis

Let the running time be $T_C(m,n)$ for the base CX algorithm, $T_I(m,n)$ for the ID, and $T_L(m,n)$ for a base rank-k matrix approximation method. Then the total running time for choosing r columns to approximate $\mathbf{M} \in \mathbb{R}^{m \times n}$ of rank k is

$$T_{\text{choose}} \sim t[T_L(m,l) + T_I(k,l)] + T_C(m,kt).$$

Note that by introducing parallelism the factor t in the term $t[T_L(m,l)+T_I(k,l)]$ becomes negligible. In practice, our approach extends the CX algorithms to also work on large matrices that do not fit into the main memory. Our algorithm performs extremely fast with a relative-error bound when $k \ll \min\{m,n\}$ (which is common in practice). For example, when $t = \lceil \sqrt{n/k} \rceil$, we see that all matrices we need to handle are of size at most $m \times \lceil \sqrt{kn} \rceil$, much smaller than the original $m \times n$ matrix. In addition, when kt is still large we can further perform the divide-and-combine strategy on $\mathbf{B}$ to obtain an approximation which is also relative-error bounded in terms of Theorems 1 and 2 given in Section 4.

3.3 Related Work

It is worth noting that Halko *et al.* [2011] proposed a compressed factorization method of large-scale matrices. The method uses a randomized technique to get the sketch of the column space of the matrix. Then it projects the matrix on the sketched space and performs factorization. Our approach keeps the main advantages of this method, while introducing several differences. First, we can use any underlying rank-k approximation (including the method in [Halko *et al.*, 2011] and [Clarkson and Woodruff, 2012]), ID and CX algorithms. Second, our approach does not depend on randomness. Finally, our approach is easily parallelized and resolves the memory problem of a large-scale matrix.

There is also an approximation method so-called CUR similar to CX [Drineas *et al.*, 2008]. Our approach adapts to the calculation of CUR decomposition by doing CX on $\mathbf{M}$ and $\mathbf{M}^\top$, respectively, to obtain a column submatrix $\mathbf{C}$ and a row submatrix $\mathbf{R}$. We approximate $\mathbf{M}$ with $\mathbf{C}\mathbf{C}^{+}\mathbf{M}\mathbf{R}^{+}\mathbf{R} = \mathbf{C}\mathbf{U}\mathbf{R}$. A similar error bound to the CUR decomposition under our approach is easily deduced by the equation (6) in [Mahoney and Drineas, 2009] when a bound of the CX algorithm

[1]For simplicity, assume that $n \bmod t = 0$, and hence, $l = n/t$. Note that for an arbitrary n and t, $\mathbf{M}$ can always be partitioned into t submatrices, each with either $\lfloor n/t \rfloor$ or $\lceil n/t \rceil$ columns.

[2]s may be less than r if t is small. If so, we just choose whole $\mathbf{B}$, yielding the same error bound in the theoretical analysis.

[3]Ideally, we should use $\widehat{\mathbf{B}} = [\widehat{\mathbf{C}}_1, ..., \widehat{\mathbf{C}}_t]$. However, the $\widehat{\mathbf{C}}_i$'s are not real columns of $\mathbf{M}$. In the theoretical analysis, we will show that even our approach works on the $\mathbf{B}$, we obtain a good approximation.

is given. We note that the bound on CUR may be a factor of 2 larger than that of the CX algorithm.

4 Theoretical Analysis

In this section, we present theoretical results that establish a relative error bound of our approach. We first give a brief discussion on the type of the bound. Next, we introduce a main theorem about the error bound. Finally, we analyze our framework. The analysis shows that our approach enjoys a relative-error bound as long as base algorithms used also give relative-error bounds.

4.1 Main Results

In this subsection, we present two theorems, one based on the Frobenius norm and the other based on the spectral norm. The theorems show that the ensemble stage can obtain an accurate result on a block low rank matrix when the base CX algorithm yields a bounded approximation.

Theorem 1. *Let $t = n/l$ and $l \geq k$. Suppose $\mathbf{L} = [\mathbf{L}^{(1)}, ..., \mathbf{L}^{(t)}] \in \mathbb{R}^{m\times n}$ where $\mathbf{L}^{(i)} \in \mathbb{R}^{m\times l}$ and $\text{rank}(\mathbf{L}^{(i)}) \leq k$. We apply an ID algorithm on $\mathbf{L}^{(i)}$ to obtain $\mathbf{L}^{(i)} = \widehat{\mathbf{C}}^{(i)}\widehat{\mathbf{X}}^{(i)}$ for $i = 1, \ldots, t$. Denote $\widehat{\mathbf{B}} = [\widehat{\mathbf{C}}^{(1)}, ..., \widehat{\mathbf{C}}^{(t)}] \in \mathbb{R}^{m\times s}$. For a fixed matrix $\mathbf{B}$ and a random matrix $\mathbf{C} \in \mathbb{R}^{m\times r}$, $k \leq r \leq s$, and $\mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{B}} - \Pi^F_{\mathbf{C},k}(\mathbf{B})\|_F] \leq \Delta_F$, then $\mathbb{E}_{\mathbf{C}}[\|\mathbf{L} - \Pi^F_{\mathbf{C},k}(\mathbf{L})\|_F] \leq 2\sqrt{kl}\Delta_F$.*

Proof. Let $\widehat{\mathbf{X}} = diag(\widehat{\mathbf{X}}^{(1)}, ..., \widehat{\mathbf{X}}^{(t)}) \in \mathbb{R}^{s\times n}$. We know that $\mathbf{L} = [\mathbf{L}^{(1)}, ..., \mathbf{L}^{(t)}] = [\widehat{\mathbf{C}}^{(1)}\widehat{\mathbf{X}}^{(1)}, ..., \widehat{\mathbf{C}}^{(t)}\widehat{\mathbf{X}}^{(t)}] = \widehat{\mathbf{B}}\widehat{\mathbf{X}}$. Thus, $\mathbb{E}_{\mathbf{C}}[\|\mathbf{L} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{L})\|_{\mathrm{F}}] \overset{(a)}{\leq} \mathbb{E}_{\mathbf{C}}[\|\mathbf{L} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\widehat{\mathbf{X}}\|_{\mathrm{F}}]$ $= \mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{B}}\widehat{\mathbf{X}} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\widehat{\mathbf{X}}\|_{\mathrm{F}}] \overset{(b)}{\leq} \|\widehat{\mathbf{X}}\|_2\mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{B}} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\|_{\mathrm{F}}]$. (a) follows from $\text{rank}[\Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\widehat{\mathbf{X}}] \leq \text{rank}[\Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})] \leq k$. (b) follows from the strong submultiplicativity of the Frobenius norm. Furthermore, we have $\|\widehat{\mathbf{X}}\|_2 \overset{(a)}{=} \max_i \|\widehat{\mathbf{X}}^{(i)}\|_2 \leq \max_i \|\widehat{\mathbf{X}}^{(i)}\|_{\mathrm{F}} \overset{(b)}{\leq} 2\sqrt{kl}$. (a) follows from the simple property of the spectral norm. (b) follows from the fact that the magnitude of entries in $\mathbf{X}^{(i)}$ is bounded by 2. □

Theorem 2. *We use the same notation as in Theorem 1. If $\mathbf{C}$ holds the property that $\mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{C}} - \Pi^2_{\mathbf{C},k}(\widehat{\mathbf{C}})\|_2] \leq \Delta_S$, then $\mathbb{E}_{\mathbf{C}}[\|\mathbf{L} - \Pi^2_{\mathbf{C},k}(\mathbf{L})\|_2] \leq 2\sqrt{kl}\Delta_S$.*

Once we replace the Frobenius norm with the spectral norm, the proof is immediately obtained from that of Theorem 1.

4.2 Theoretical Analysis for CX

We also give two theorems respectively based on the Frobenius norm and the spectral norm, which ensure our approach yields a relative-error bound. Note that Δ_* stated in the following theorems is the error parameter given by the base algorithm which depends on r, k and the size of the data matrix. Please refer to Figure 1 in [Guruswami and Sinop, 2012] and [Halko *et al.*, 2011] for examples.

Theorem 3. *Let $t = n/l$ and $l \geq k$. Based on the same notation as in Section 3.1, the base rank-k approximation algorithm returns a matrix $\widehat{\mathbf{M}}^{(i)}$ of rank at most k such that $\mathbb{E}_{\widehat{\mathbf{M}}^{(i)}}[\|\mathbf{M}^{(i)} - \widehat{\mathbf{M}}^{(i)}\|_F^2] \leq \Delta^2_{LF}\|\mathbf{M}^{(i)} - \mathbf{M}^{(i)}_k\|_F^2$ for some $\Delta_{LF} \geq 1$. The base CX algorithm yields $\mathbb{E}_{\mathbf{C}}[\|\mathbf{B} - \Pi^F_{\mathbf{C},k}(\mathbf{B})\|_F|\mathbf{B}] \leq \Delta_{CF}\|\mathbf{B} - \mathbf{B}_k\|_F$ for $\Delta_{CF} \geq 1$ by choosing r columns $\mathbf{C}$ from $\mathbf{B}$. Then $\mathbb{E}_{\mathbf{C},\widehat{\mathbf{M}}}[\|\mathbf{M} - \Pi^F_{\mathbf{C},k}(\mathbf{M})\|_F] \leq [2\sqrt{kl}(\Delta_{LF} + \Delta_{CF}) + \Delta_{LF}]\|\mathbf{M} - \mathbf{M}_k\|_F$.*

Proof. Split $\mathbf{M}_k = [\mathbf{N}^{(1)}, ..., \mathbf{N}^{(t)}]$ as the same way as $\mathbf{M} = [\mathbf{M}^{(1)}, ..., \mathbf{M}^{(t)}]$. and note that $\text{rank}(\mathbf{N}^{(i)}) \leq k$ for each i. Then, $\mathbb{E}_{\widehat{\mathbf{M}}}[\|\mathbf{M} - \widehat{\mathbf{M}}\|^2_{\mathrm{F}}] = \mathbb{E}_{\widehat{\mathbf{M}}}\left[\sum_{i=1}^t \|\mathbf{M}^{(i)} - \widehat{\mathbf{M}}^{(i)}\|^2_{\mathrm{F}}\right]$ $\leq \sum_{i=1}^t \Delta^2_{LF}\|\mathbf{M}^{(i)} - \mathbf{M}^{(i)}_k\|^2_{\mathrm{F}} \overset{(a)}{\leq} \Delta^2_{LF}\sum_{i=1}^t \|\mathbf{M}^{(i)} - \mathbf{N}^{(i)}\|^2_{\mathrm{F}} = \Delta^2_{LF}\|\mathbf{M} - \mathbf{M}_k\|^2_{\mathrm{F}}$. (a) follows because $\mathbf{M}^{(i)}_k$ is the best rank-k approximation of $\mathbf{M}^{(i)}$, and $\text{rank}(\mathbf{N}^{(i)}) \leq k$. It follows that $\mathbb{E}_{\widehat{\mathbf{M}}}[\|\mathbf{M} - \widehat{\mathbf{M}}\|_{\mathrm{F}}] \leq \sqrt{\mathbb{E}_{\widehat{\mathbf{M}}}[\|\mathbf{M} - \widehat{\mathbf{M}}\|^2_{\mathrm{F}}]} \leq \Delta_{LF}\|\mathbf{M} - \mathbf{M}_k\|_{\mathrm{F}}$.

When $\widehat{\mathbf{M}}$ is fixed, we know $\widehat{\mathbf{B}}$ and $\mathbf{B}$ are determined because the ID algorithm is deterministic (see Section 2.3). Consider the error introduced by replacing $\widehat{\mathbf{B}}$ with $\mathbf{B}$, and perform the CX approximation on $\mathbf{B}$. That is,

$$\begin{aligned}
&\mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{B}} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\|_{\mathrm{F}}|\widehat{\mathbf{M}}] \\
\leq\ &\|\mathbf{B} - \widehat{\mathbf{B}}\|_{\mathrm{F}} + \mathbb{E}_{\mathbf{C}}[\|\mathbf{B} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\|_{\mathrm{F}}|\widehat{\mathbf{M}}] \\
\overset{(a)}{\leq}\ &\|\mathbf{M} - \widehat{\mathbf{M}}\|_{\mathrm{F}} + \mathbb{E}_{\mathbf{C}}[\|\mathbf{B} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\|_{\mathrm{F}}|\widehat{\mathbf{M}}] \\
\leq\ &\|\mathbf{M} - \widehat{\mathbf{M}}\|_{\mathrm{F}} + \Delta_{CF}\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}} \\
\overset{(b)}{\leq}\ &\|\mathbf{M} - \widehat{\mathbf{M}}\|_{\mathrm{F}} + \Delta_{CF}\|\mathbf{M} - \mathbf{M}_k\|_{\mathrm{F}}.
\end{aligned}$$

(a) follows because $\mathbf{B}$ and $\widehat{\mathbf{B}}$ are the submatrix of $\mathbf{M}$ and $\widehat{\mathbf{M}}$ respectively. (b) follows because $\mathbf{B}$ is always the submatrix of $\mathbf{M}$. If we apply Theorem 1 on block low-rank matrix $\widehat{\mathbf{M}} = [\widehat{\mathbf{M}}^{(1)}, ..., \widehat{\mathbf{M}}^{(t)}]$ and use the bound on $\mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{B}} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{B})\|_{\mathrm{F}}|\widehat{\mathbf{M}}]$, we have $\mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{M}} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{M})|\widehat{\mathbf{M}}\|_{\mathrm{F}}] \leq 2\sqrt{kl}(\|\mathbf{M} - \widehat{\mathbf{M}}\|_{\mathrm{F}} + \Delta_{CF}\|\mathbf{M} - \mathbf{M}_k\|_{\mathrm{F}})$. Finally,

$$\begin{aligned}
&\mathbb{E}_{\mathbf{C},\widehat{\mathbf{M}}}[\|\mathbf{M} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\mathbf{M})\|_{\mathrm{F}}] \overset{(a)}{\leq} \mathbb{E}_{\mathbf{C},\widehat{\mathbf{M}}}[\|\mathbf{M} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\widehat{\mathbf{M}})\|_{\mathrm{F}}] \\
&= \mathbb{E}_{\widehat{\mathbf{M}}}[\|\mathbf{M} - \widehat{\mathbf{M}}\|_{\mathrm{F}}] + \mathbb{E}_{\widehat{\mathbf{M}}}[\mathbb{E}_{\mathbf{C}}[\|\widehat{\mathbf{M}} - \Pi^{\mathrm{F}}_{\mathbf{C},k}(\widehat{\mathbf{M}})\|_{\mathrm{F}}|\widehat{\mathbf{M}}]] \\
&\leq \Delta_{LF}\|\mathbf{M} - \mathbf{M}_k\|_{\mathrm{F}} + 2\sqrt{kl}(\Delta_{LF} + \Delta_{CF})\|\mathbf{M} - \mathbf{M}_k\|_{\mathrm{F}}.
\end{aligned}$$

(a) follows from $\text{rank}(\Pi^{\mathrm{F}}_{\mathbf{C},k}(\widehat{\mathbf{M}})) \leq k$. □

Theorem 4. *Use the same settings as in Theorem 3, the base rank-k approximation algorithm returns a matrix $\widehat{\mathbf{M}}^{(i)}$ of rank at most k such that $\mathbb{E}_{\widehat{\mathbf{M}}^{(i)}}[\|\mathbf{M}^{(i)} - \widehat{\mathbf{M}}^{(i)}\|_2^2] \leq \Delta^2_{LS}\|\mathbf{M}^{(i)} - \mathbf{M}^{(i)}_{(k)}\|_2^2$ for some $\Delta_{LS} \geq 1$. If the base CX algorithm yields $\mathbb{E}_{\mathbf{C}}[\|\mathbf{B} - \Pi^2_{\mathbf{C},k}(\mathbf{B})\|_2|\mathbf{B}] \leq \Delta_{CS}\|\mathbf{B} - \mathbf{B}_k\|_2$ for $\Delta_{CS} \geq 1$ by choosing r columns $\mathbf{C}$ from $\mathbf{B}$, then $\mathbb{E}_{\widehat{\mathbf{M}},\mathbf{C}}[\|\mathbf{M} - \mathbf{C}\mathbf{C}^+\mathbf{M}\|_2] \leq [2\sqrt{kl}(\sqrt{t}\Delta_{LS} + \Delta_{CS}) + \sqrt{t}\Delta_{LS}]\|\mathbf{M} - \mathbf{M}_k\|_2$.*

The proof is almost identical to that of Theorem 3. We omit the details for the sake of brevity.

4.3 Discussion

We highlight four points on the theoretical results. **(1)** We give an example to interpret the above theorems intuitively. If we use the best rank-k approximation on submatrices, Theorem 3 shows that we obtain a $[2\sqrt{kl}(2+\varepsilon)+1]\|\mathbf{M}-\mathbf{M}_k\|_{\mathrm{F}}$ upper error bound on the Frobenius norm, where ε is the error parameter determined by the base CX algorithm, and l is the number of columns in each submatrix after the divide step. We actually get a relative-error type bound the same as the base CX algorithm. It means if $\|\mathbf{M}-\mathbf{M}_k\|_{\mathrm{F}}$ is small (i.e., the numerical rank of $\mathbf{M}$ is k), then our approach approximate the matrix almost exactly. As noted before, the adaptive sampling can be used to obtain a better theoretical bound. By Lemma 16 of Boutsidis *et al.* [2011], if the parameter s' is at least $k[2\sqrt{kl}(2+\varepsilon)+1]/\varepsilon$, we can make sure that our framework has the same theoretical error bound as the base CX algorithm. For example, let us consider the case that base CX algorithm is Theorem 5 of Boutsidis *et al.* [2011]. In order to get a $(1+\varepsilon)\|\mathbf{M}-\mathbf{M}_k\|_{\mathrm{F}}$ error bound on the Frobenius norm, we need to choose $\Theta(k\sqrt{kl}/\varepsilon)$ columns. **(2)** In general, in order to obtain a $(1+\varepsilon)\|\mathbf{M}-\mathbf{M}_k\|_{\mathrm{F}}$ error bound on the Frobenius norm, we should let the parameter s' in the adaptive sampling be at least $k[2\sqrt{kl}(\Delta_{LF}+\Delta_{CF})+\Delta_{LF}]/\varepsilon$. **(3)** There is a trade-off between the approximation error bound and the speed-up. If l is small (i.e., the number of submatrices is large), the error bound is small. But the running time of the Ensemble stage is longer as the matrix $\mathbf{B}$ becomes larger. **(4)** For flexibility, we analyze the approach with randomized algorithms. If all the base algorithms are deterministic, our consequence automatically becomes deterministic (just throw the expectation away). Note that our analysis only gives the error bound in expectation. It can be adapted to the high-probability low-error form by the tail inequality such as the Markov inequality.

5 Empirical Evaluation

The main purpose of this section is to evaluate the accuracy and speed of our approach on four different types of data. In particular, we want to evaluate the speed of our approach and compare the error to the conventional approach that runs the CX algorithm directly on the original data matrices.

5.1 Set-up

For the rank-k approximation algorithm, we use the truncated SVD (SVD command in Matlab with truncation). For the ID algorithm, we choose Algorithm 5 in [Gu and Eisenstat, 1996]. For the CX algorithm, we choose two: a deterministic one (Theorem 2 of Boutsidis *et al.* [2011], denoted *Det*), and a randomized one (Algorithm 1 of Drineas *et al.* [2008], denoted *Rand*), using the EXACTLY(C) sampling scheme [Drineas *et al.*, 2008] and boosting 20 times. We split the $m \times n$ input matrix into $t = \lceil\sqrt{n/k}\rceil$ pieces for parallelization. We write the code in Matlab with the PARFOR feature to implement parallelization. We measure the error via the ratio of the Frobenius-norm error over the best rank-k approximation, i.e., reconstruction error ratio $\Theta \triangleq \|\mathbf{M}-\mathbf{C}\mathbf{C}^{+}\mathbf{M}\|_{\mathrm{F}}/\|\mathbf{M}-\mathbf{M}_k\|_{\mathrm{F}}$. The time measured is wall-clock time. We do not perform the adaptive sampling in all experiments. We evaluate the first three types of data by using Matlab 7.10.

5.2 Simulated Data

We evaluate our framework on two types of random matrices mentioned in [Çivril and Magdon-Ismail, 2012]: *Log* and *Scaled Random*. We set $m = n = 5000$ and the rank parameter $k = 15$ for the reconstruction error comparison. Note that all reported results are averages over five random examples of the same size. The reconstruction error and the running time are presented in Figure 1 and Table 1 respectively. These figures show that the error is not amplified too much by our framework, and the speed-up is super-linear. Surprisingly, the error of our framework is smaller than the base CX algorithm on *Log* Data. We think it is due to the property of this type of data.

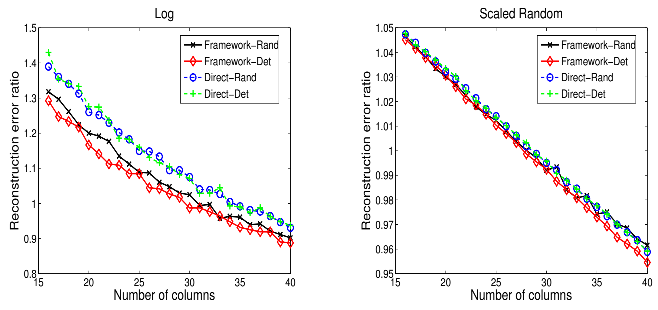

Figure 1: Empirical reconstruction error results for *Log* and *Scaled Random* data. The reconstruction error ratio as a function of number of columns chosen for rank parameter $k = 15$ and $m = n = 5000$ is shown. *Framework-Rand* and *Framework-Det* respectively denote *Rand* and *Det* algorithms under our proposed framework. *Direct-Rand* and *Direct-Det* respectively denote *Rand* and *Det* algorithms running directly on the data matrices.

5.3 Collaborative Filtering Data

A collaborative filtering dataset naturally forms a rating matrix. We use the CX method to approximate the matrix. We evaluate our approach on two public datasets: the Jester dataset [Goldberg *et al.*, 2001] and the MovieLens 10M dataset[4]. To generate suitable matrices, we perform some preprocessing for the two datasets. For the Jester dataset, we only choose the people who rate all 100 jokes to form a 100×14116 matrix. For the MovieLens 10M dataset, we set the unrated entry as a random number uniformly distributed in $[0, 5]$ to form a 10K $\times$ 10K matrix. As shown in Figure 2, the error resulted from the framework is almost the same as the direct performance. In addition, the running time is reduced from 130s to 5s for the two base CX algorithms.

[4]`http://www.grouplens.org`

Table 1: Running times for *Log* and *Scaled Random* data. Here the rank parameter $k = 15$ and the number of chosen columns $r = 30$. The number of available cores is equal to $t = \lceil \sqrt{n/k} \rceil$.

Size of matrix / Algorithm & Data	1000×1000	3000×3000	5000×5000	7000×7000
Direct-Det on *Log*	3.4s	359.7s	1460.6s	4490.2s
Framework-Det on *Log*	1.1s	12.1s	32.7s	63.0s
Direct-Rand on *Log*	3.6s	366.9s	1423.5s	4486.0s
Framework-Rand on *Log*	1.0s	12.1s	31.7s	62.6s
Direct-Det on *Scaled Random*	1.8s	174.7s	473.4s	1250.6s
Framework-Det on *Scaled Random*	1.0s	12.0s	31.6s	73.0s
Direct-Rand on *Scaled Random*	2.0s	177.2s	478.1 s	1260.1s
Framework-Rand on *Scaled Random*	1.3s	11.8s	32.4s	75.2s

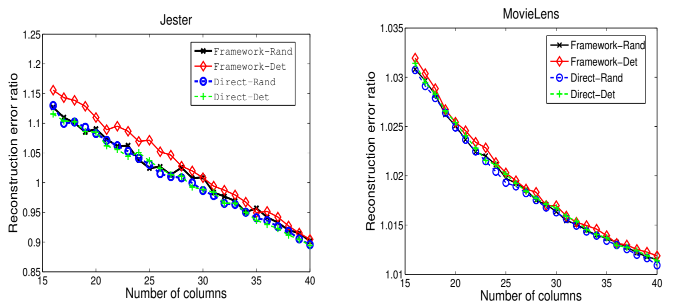

Figure 2: Empirical reconstruction error results for the collaborative filtering data. The results are given as the reconstruction error ratio as a function of number of columns chosen for rank parameter $k = 15$.

5.4 Human Face Data

In image processing, a matrix can be constructed from a set of images with each image reshaped as a column vector [Peng *et al.*, 2011]. Moreover, such a matrix is usually found to be of low rank. We now use the CX algorithm to approximate the matrix. In particular, we study our approach on the cropped images of the Extended Yale Face Database B [Georghiades *et al.*, 2001; Lee *et al.*, 2005] which forms a 32256×2424 matrix. The performance result is shown in Figure 3. The running time is reduced to 6s from 146s after employing our framework.

5.5 PicasaWeb Dataset

We use images from the PicasaWeb Dataset [Wang *et al.*, 2012]. After having extracted the HMAX features (1000 dimension) [Serre *et al.*, 2007] and deleted some images, we form a 1.44 million by 1000 dense data-feature matrix. We use the truncated SVD as the rank-k approximation algorithm and *Rand* as the base CX algorithm. In the other word, our algorithm is the same as *Framework-Rand*. Instead of using Matlab, we implement our framework with the distributed computing framework FlumeJava [Chambers *et al.*, 2010] and the matrix algorithms library Eigen [Guennebaud *et al.*, 2010]. Our method takes less than 10 minutes by 200 12-cores machines to choose 100 columns with $k = 15$.

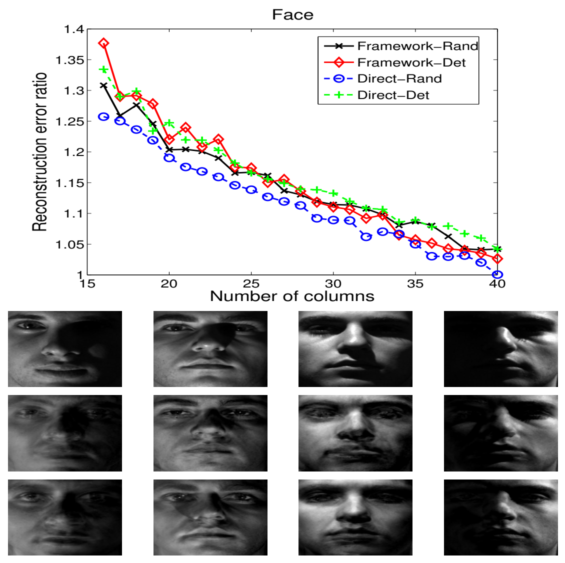

Figure 3: Empirical reconstruction error results for the human face data. The graph corresponds to the reconstruction error ratio as a function of number of columns chosen for rank parameter $k = 15$. The facial images is used for a comparison between original images and reconstructed images when the number of chosen columns is $r = 40$. The faces on top are the originals. Faces on the second and third rows are reconstructed by *Direct-Rand* and *Framework-Rand*, respectively.

6 Conclusion

We have proposed a divide-and-combine approach for handling the scalability problem of column-based approximation methods on large-scale matrices. Our intrinsically parallel approach leads to a significant speed-up on large-scale matrices, and holds a nice error bound theoretically and empirically. Our analysis applies to a general matrix; that is, no additional conditions such as matrix coherence are required. Theoretical analysis and empirical results demonstrate the effectiveness, efficiency and practicality of our approach.

References

[Boutsidis *et al.*, 2009] Christos Boutsidis, Michael W. Mahoney, and Petros Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'09)*, pages 968–977, 2009.

[Boutsidis *et al.*, 2011] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near optimal column-based matrix reconstruction. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS'11)*, pages 305–314, 2011.

[Chambers *et al.*, 2010] Craig Chambers, Ashish Raniwala, Frances Perry, Stephen Adams, Robert R. Henry, Robert Bradshaw, and Nathan Weizenbaum. Flumejava: easy, efficient data-parallel pipelines. In *Proceedings of the 2010 ACM SIGPLAN conference on Programming language design and implementation*, PLDI '10, pages 363–375, New York, NY, USA, 2010. ACM.

[Cheng *et al.*, 2005] H. Cheng, Z. Gimbutas, P. G. Martinsson, and V. Rokhlin. On the compression of low rank matrices. *SIAM J. Sci. Comput.*, 26(4):1389–1404, 2005.

[Çivril and Magdon-Ismail, 2012] A. Çivril and M. Magdon-Ismail. Column subset selection via sparse approximation of SVD. *Theoretical Computer Science*, 421(0):1–14, 2012.

[Clarkson and Woodruff, 2012] Kenneth L. Clarkson and David P. Woodruff. Low rank approximation and regression in input sparsity time. *CoRR*, abs/1207.6365, 2012.

[Deshpande *et al.*, 2006] Amit Deshpande, Luis Rademacher, Santosh Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(1):225–247, 2006.

[Drineas and Mahoney, 2005] Petros Drineas and Michael W. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

[Drineas *et al.*, 2006] Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast monte carlo algorithms for matrices iii: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36(1):184–206, 2006.

[Drineas *et al.*, 2008] Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM J. Matrix Anal. Appl.*, 30(2):844–881, 2008.

[Drineas, 2003] Petros Drineas. Pass-efficient algorithms for approximating large matrices. In *In Proceeding of the 14th Annual ACM-SIAM Symposium on Dicrete Algorithms*, pages 223–232, 2003.

[Georghiades *et al.*, 2001] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

[Goldberg *et al.*, 2001] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[Golub and Loan, 1996] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, third edition, 1996.

[Goreinov *et al.*, 1997a] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra and Its Applications*, 261:1–21, 1997.

[Goreinov *et al.*, 1997b] S. A. Goreinov, N. L. Zamarashkin, and E. E. Tyrtyshnikov. Pseudo-skeleton approximations by matrices of maximal volume. *Mathematical Notes*, 62(4):619–623, 1997.

[Gu and Eisenstat, 1996] Ming Gu and Stanley C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM J. Sci. Comput.*, 17(4):848–869, 1996.

[Guennebaud *et al.*, 2010] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[Guruswami and Sinop, 2012] Venkatesan Guruswami and Ali Kemal Sinop. Optimal column-based low-rank matrix reconstruction. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*, pages 1207–1214, 2012.

[Halko *et al.*, 2011] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, 2011.

[Kannan and Vempala, 2009] Ravindran Kannan and Santosh Vempala. Spectral algorithms. *Foundations and Trends in Theoretical Computer Science*, 4(3-4):157–288, 2009.

[Lee *et al.*, 2005] K.C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 27(5):684–698, 2005.

[Mackey *et al.*, 2011] Lester W. Mackey, Ameet S. Talwalkar, and Michael I. Jordan. Divide-and-conquer matrix factorization. In *Advances in Neural Information Processing Systems 24*, pages 1134–1142, 2011.

[Mahoney and Drineas, 2009] Michael W. Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. *PNAS*, 106:697–702, 2009.

[Mahoney, 2011] Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, Vol. 3: No 2:123–224, 2011.

[Peng *et al.*, 2011] Yigang Peng, Arvind Ganesh, John Wright, Wenli Xu, and Yi Ma. Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(PrePrints), 2011.

[Serre *et al.*, 2007] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Robust object recognition with cortex-like mechanisms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(3):411 –426, march 2007.

[Tyrtyshnikov, 2000] Eugene E. Tyrtyshnikov. Incomplete cross approximation in the mosaic-skeleton method. *Computing*, 64:367–380, 2000.

[Wang *et al.*, 2012] Zhiyu Wang, Fangtao Li, Edward Y. Chang, and Shiqiang Yang. A data-driven study on image feature extraction and fusion. Technical report, Google Inc., 2012.