Empirical Performance of Approximate Algorithms for Low Rank Approximation

Dimitris Konomis (dkonomis@cs.cmu.edu)
Machine Learning Department (MLD)
School of Computer Science (SCS)
Carnegie Mellon University (CMU)

Data Analysis Project
DAP Committee: Manuela Veloso, David P. Woodruff

August 31 2018

Abstract

Low rank approximation is arguably one of the most well-studied problems in numerical linear algebra, with diverse applications to clustering, data mining, distance matrix completion, information retrieval, learning mixtures of distributions, recommendation systems and web search. Formally, given a rank-r matrix $A \in \mathbb{R}^{n \times m}$, a parameter k, and a norm p, one seeks a rank-$\leq k$ matrix $A' \in \mathbb{R}^{n \times m}$ that "well-approximates" A with respect to the norm p: i.e. minimizes $\|A - A'\|_p$. When $p = F$ (Frobenius norm) the problem can be solved using the truncated SVD algorithm, whereas when $p = 1$ (entrywise ℓ_1 norm) the problem is NP-hard and there exists no closed-form solution. In both cases, today's massive datasets prohibit the computation of A' in a reasonable amount of time. Recent advances in numerical linear algebra (sketching) have allowed the development of approximation algorithms that allow for a fast but approximate solution. This data analysis project studies the empirical performance of state-of-the-art in theory approximation algorithms for the ℓ_F and ℓ_1 variants of the low-rank approximation problem on a variety of synthetic and real datasets.

1 Introduction

Low rank approximation is arguably one of the most well-studied problems in numerical linear algebra, with diverse applications to clustering, data mining, distance matrix completion information retrieval, learning mixtures of distributions recommendation systems and web search. In practice one often has a low rank matrix which has been corrupted with noise of bounded norm, and low rank approximation allows one to approximately recover the original matrix. Low rank approximation may also help explain a dataset, revealing low dimensional structure in high dimensional data. Formally, given a rank-r matrix $A \in \mathbb{R}^{n \times m}$, the ℓ_p low-rank approximation is formulated as the following optimization problem:

$$\begin{aligned} \min_{A'} \quad & \|A - A'\|_p \\ \text{s.t.} \quad & rank(A') \leq k \end{aligned} \tag{1}$$

When $p = 2$ (spectral norm) or $p = F$ (Frobenius norm), the problem has a closed form solution, which can be obtained through the singular value decomposition of A. The runtime complexity of the SVD algorithm is $O(min(n^2m, m^2n))$, which is theoretically polynomial (fast) but slow for todays massive datasets. Clarkson and Woodruff [4] developed a randomized approximation algorithm for ℓ_F low-rank approximation that runs in $O(\text{nnz}(A) + (n + d)\text{poly}(k/\epsilon))$ running time, and achieves a $(1 + \epsilon)$-approximation factor. That is, given the optimal solution A^*, the algorithm outputs A' such that: $\|A - A'\|_F \leq (1 + \epsilon)\|A - A^*\|_F$.

As for regression, using the entrywise ℓ_1 norm ($p = 1$) is a more robust and less sensitive to outliers loss function. Unfortunately, this problem not only does not have a closed form solution, but is in fact NP-hard [2]. Song, Woodruff, and Zhong [3] recently proposed a randomized algorithm for entrywise ℓ_1 norm low rank approximation, with an approximation factor of $\alpha = \log n \cdot \text{poly}(k)$ and running time $O(\text{nnz}(A) + m \cdot \text{poly}(k))$. This algorithm also heavily relies on sketching, where the underlying sketching matrices are combinations of CountSketch and Cauchy random matrices. Formally, if A^* is a real minimizer, then the algorithm outputs A' such that: $\|A - A'\|_1 \leq \log m \cdot \text{poly}(k)\|A - A^*\|_1$.

We implement the approximate-ℓ_1 and approximate ℓ_F algorithms and evaluate their performance on a wide variety of both synthetic and real datasets. We believe our empirical results provide strong evidence of the practical value of the recent theoretical breakthroughs in sketching-based methods for low rank approximation.

The rest of this paper is organized as follows: section 2 introduces sketching and some important definitions and theorems, section 3 presents the approximate-ℓ_F algorithm and analyzes its running time and approximation guarantees, section 4 presents the approximate-ℓ_1 algorithm and analyzes its running time and approximation guarantees, section 5 presents thorough experimental results and section 6 contains acknowledgements.

2 Preliminaries

Linear matrix sketching is a technique where given a big matrix, one first compresses it to a much smaller matrix by multiplying it by a random matrix with certain properties. Much of the expensive computation can then be performed on the smaller matrix, thereby accelerating the solution for the original problem. Linear sketching has been successfully used in the development of efficient approximate algorithms for regression, low rank approximation graph sparsification problems and many of their variants.

Definition 1 (**ℓ_2-subspace embedding**)**.** A $(1 \pm \epsilon)$ ℓ_2-subspace embedding for the column space of a matrix $A \in \mathbb{R}^{n \times m}$ is a matrix S for which, for all $x \in \mathbb{R}^m$ the following holds:

$$\|SAx\|_2^2 = (1 \pm \epsilon)\|Ax\|_2^2 \tag{2}$$

Definition 2 (**(ϵ, δ) oblivious ℓ_2-subspace embedding**)**.** Suppose Π is a distribution on random matrices $S \in \mathbb{R}^{r \times n}$ where $r = f(n, m, \epsilon, \delta)$. Suppose that for any fixed matrix $A \in \mathbb{R}^{n \times m}$, with probability at least $1 - \delta$, a matrix S drawn from distribution Π is a $(1 \pm \epsilon)$ ℓ_2-subspace embedding for A. Then Π is called an (ϵ, δ) oblivious ℓ_2-subspace embedding.

Definition 3 (JLT)**.** A random matrix $S \in \mathbb{R}^{k \times n}$ forms a Johnson-Lindenstrauss transform with parameters ϵ, δ, f or $JLT(\epsilon, \delta, f)$ for short, if with probability at least $1 - \delta$, for any f-element subset $V \subseteq \mathbb{R}^n$:

$$\forall v, v' \in V : \|\langle Sv, Sv' \rangle\|_1 \leq \epsilon \|v\|_2 \|v'\|_2 \tag{3}$$

Theorem 1. *Let $0 < \epsilon, \delta < 1$ and $S = \frac{1}{\sqrt{k}} R \in \mathbb{R}^{k \times n}$ where the entries of $R \in \mathbb{R}^{k \times n}$ are independent standard normal variables. Setting $k = \Theta(\frac{1}{\epsilon^2}(m + \log(\frac{1}{\delta})))$, for any fixed matrix $A \in \mathbb{R}^{n \times m}$, with probability at least $1 - \delta$, S is a $(1 \pm \epsilon)$ ℓ_2-subspace embedding for A:*

$$\forall x \in \mathbb{R}^m : \|SAx\|_2^2 = (1 \pm \epsilon)\|Ax\|_2^2 \tag{4}$$

Theorem 2 (**Subsampled Randomized Hadamard Transform, (PHD)**)**.** *Let $S = \frac{1}{\sqrt{kn}} P \cdot H_n \cdot D$, where $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix with i.i.d. diagonal entries $D_{i,i} = 1$ with probability 1/2 and $D_{i,i} = -1$ with probability 1/2. $H_n \in \mathbb{R}^{n \times n}$ is the Hadamard matrix of size $n = 2^t$, for some $t > 0$, with $H_{i,j} = \frac{1}{\sqrt{n}}(-1)^{\sum_{z=1}^{\log n} i_z \cdot j_z}$ where $(i_{\log n}, ..., i_1)$ and $(j_{\log n}, ..., j_1)$ are the binary representations of i and j. The matrix $P \in \mathbb{R}^{r \times n}$ samples r coordinates of an n-dimensional vector uniformly at random. Setting $r = \Omega(\frac{\log m}{\epsilon^2}(\sqrt{m} + \sqrt{\log n})^2)$, with probability at least 0.99, for any fixed matrix $U \in \mathbb{R}^{n \times m}$ with orthonormal columns:*

$$\|I_m - U^T S^T S U\|_2 \leq \epsilon \tag{5}$$

Finally for any vector $x \in \mathbb{R}^n$, $S \cdot x$ can be computed in $O(n \log r)$ time.

Definition 4 (**Sparse Embedding Matrix (CountSketch)**)**.** The CountSketch matrix is a sparse matrix $S \in \mathbb{R}^{r \times n}$, constructed in the following way. Firstly, for each of the n columns $S_{*,i}$, we (i) independently choose a uniformly random row $h(i) \in \{1, 2, ..., \ldots, r\}$ and (ii) choose uniformly a random element $\sigma(i) \in \{-1, 1\}$. Secondly, we set $S_{h(i),i} = \sigma(i)$ and $S_{i,j} = 0$ for all $j \neq i$. Therefore S has a single non-zero entry per column.

Definition 5 (**Well-conditioned basis for the 1-norm**). Consider a matrix $A \in \mathbb{R}^{n\times m}$. A matrix $U \in \mathbb{R}^{n\times m}$ is an $(\alpha, \beta, 1)$-well-conditioned basis for the column space of A if:

$$\|U\|_1 \leq \alpha \tag{6a}$$

$$\forall x \in \mathbb{R}^m : \|x\|_\infty \leq \beta \|Ux\|_1 \tag{6b}$$

Definition 6. A matrix $S \in \mathbb{R}^{s\times n}$ is an ℓ_1-subspace embedding for a matrix $A \in \mathbb{R}^{n\times m}$ if there exist constants $c_1, c_2 > 0$ such that:

$$\forall x \in \mathbb{R}^m : \|Ax\|_1 \leq \|SAx\|_1 \leq d^{c_1}\|Ax\|_1 \tag{7}$$

and S has at most $s = d^{c_2}$ rows.

Theorem 3. *A matrix $S \in \mathbb{R}^{r\times n}$ of i.i.d. Cauchy random variables with $r = O(d \log d)$ rows is an ℓ_1-subspace embedding with constant probability: with probability at least $\frac{9}{10}$, $\forall x \in \mathbb{R}^m$:*

$$\|Ax\|_1 \leq \frac{4\|SAx\|_1}{r} = O(d \log d)\|Ax\|_1 \tag{8}$$

Definition 7 (**(ϵ, δ, ℓ)-JL Property**). A distribution $\mathcal{D}$ on matrices $S \in \mathbb{R}^{k\times m}$ has the (ϵ, δ, ℓ)-JL moment property if $\forall x \in \mathbb{R}^m$, with $\|x\|_2 = 1$:

$$\mathbb{E}_{S\sim\mathcal{D}}[|\|Sx\|_2^2 - 1|^\ell] \leq \epsilon^\ell \cdot \delta \tag{9}$$

Definition 8 (**Approximate Matrix Product Property**). For $\epsilon, \delta \in (0, 1/2)$, let $\mathcal{D}$ be a distribution over matrices with m columns that satisfies the (ϵ, δ, ℓ)-JL moment property for some $\ell \geq 2$. Then, for any matrices $A \in \mathbb{R}^{m\times a}, B \in \mathbb{R}^{m\times b}$:

$$Pr_{S\sim\mathcal{D}}[\|A^T S^T SB - A^T B\|_F > 3\epsilon\|A\|_F\|B\|_F] \leq \delta \tag{10}$$

Theorem 4. *Let $S \in \mathbb{R}^{r\times n}$ be a sparse embedding matrix (CountSketch), with $r = \Omega(\frac{2}{\epsilon^2\delta})$ rows. Then S satisfies the $(\epsilon, \delta, 2)$-JL moment property. Further, this holds if the hash function $h : [n] \to [r]$ is only 2-wise independent and the sign function $\sigma : [n] \to \{-1, 1\}$ is 4-wise independent.*

Definition 9 (**Affine Embeddings**). Consider matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{n\times m'}$. The generalized regression problem seeks to find the matrix $X \in \mathbb{R}^{m\times m'}$ that minimizes $\|A \cdot X - B\|_F$. A random matrix S is an affine embedding if, with high probability:

$$\|SAX - SB\|_F \leq (1 \pm \epsilon)\|AX - B\|_F \tag{11}$$

It turns out that S is an affine embedding if it satisfies the properties:

1. S is an ℓ_2-subspace embedding for A
2. S satisfies approximate matrix product property

3. S preserves the Frobenius norm of a matrix, i.e. with high probability $\|SA\|_F^2 \leq (1 \pm \epsilon)\|A\|_F^2$

Theorem 5. *Let $S \in \mathbb{R}^{r \times n}$ be a sparse embedding matrix (CountSketch). Setting $r = O(\frac{m^2}{\epsilon^2} poly(\log(\frac{m}{\epsilon})))$ rows, we have that for any fixed matrix $A \in \mathbb{R}^{n \times m}$, with probability at least 0.99, S is a $(1 \pm \epsilon)$ ℓ_2-subspace embedding for A. Furthermore, $S \cdot A$ can be computed in $O(nnz(A))$ time.*

3 Approximate ℓ_F Low-Rank Approximation

For the ℓ_F low-rank approximation problem, an approximate solution is simply a matrix A' such that:

$$\|A - A'\|_F^2 \leq (1 + \epsilon)\|A - A_k\|_F^2 \tag{12}$$

Woodruff and Clarkson [4] developed a randomized approximate algorithm for ℓ_F low-rank approximation, that achieves a $O(nnz(A) + (n + d)poly(k/\epsilon))$ runtime complexity (where $nnz(A)$ is the number of non-zero entries of A).

3.1 Existence of good solutions in the row-span of SA

Let us consider the hypothetical regression problem $min_X \|A_k X - A\|_F^2$. Because $A_k X$ has rank at most k and A_k is the best rank-k approximation to A, the optimal solution to the above problem is to set X to be the identity matrix: $X = I$. Hence the minimum cost of problem (5) is $\|A_k - A\|_2^2$.

We choose S to be an affine-embedding. We sketch problem (5) on the left using S:

$$\|SA_k X - SA\|_F^2 = (1 \pm \epsilon)\|A_k X - A\|_F^2 \tag{13}$$

The minimizer of the problem $min_X \|SA_k X - SA\|_F^2$ is clearly $X_S^* = (SA_k)^+(SA)$ and lives in the row span of the matrix SA. We also observe that:

$$\begin{aligned}(1 \pm \epsilon)\|A_k(SA_k)^+(SA) - A\|_F^2 &= \|SA_k(SA_k)^+SA - SA\|_F^2 \\ &\leq \|SA_k I - SA\|_F^2 \\ &= (1 \pm \epsilon)\|A_k I - A\|_F^2 \\ &= (1 \pm \epsilon)\|A_k - A\|_F^2, \end{aligned} \tag{14}$$

where the first and second equality follow from the fact that S is an affine embedding and the inequality from the fact that X_S^* is the minimizer of problem (6). Re-arranging terms, we get:

$$\begin{aligned}\|A_k(SA_k)^+SA - A\|_F^2 &\leq \frac{1 + \epsilon}{1 - \epsilon}\|A_k - A\|_F^2 \\ &\approx (1 + 2\epsilon)\|A_k - A\|_F^2, \end{aligned} \tag{15}$$

from which we conclude that good rank-k approximations of A_k exist in the row-span of SA.

3.2 Computing a good rank-k approximation of A in the row-span of SA

We have only shown that $A_k(SA_k)^+SA$ is an ϵ approximation of the optimal A_k. Rank-k matrices in the row-span of SA are of the form XSA for some rank-k matrix X. We observe that:

$$\begin{aligned} min_{rank-kX}\|XSA - A\|_F^2 &\leq \|A_k(SA_k)^+SA - A\|_F^2 \\ &\leq (1 \pm \epsilon)\|A_k - A\|_F^2, \end{aligned} \tag{16}$$

where the first inequality follows because we are minimizing $\|XSA - A\|_F^2$ over rank-k matrices X and the second one from proper-scaling of the number of rows of S ($poly(k/\epsilon)$).

Observe however, by the Pythagorean Theorem, that:

$$\|XSA - A\|_F^2 = \|A(SA)^+SA - A\|_F^2 + \|XSA - A(SA)^+SA\|_F^2, \tag{17}$$

where the rows of $A(SA)^+SA$ are the projections of the rows of A onto the row-span of SA. The first term of (11) does not depend on X. Hence we only need to solve the problem $min_{rank-kX}\|XSA - A(SA)^+SA\|_F^2$.

We next consider the singular value decomposition of matrix SA, let it be $SA = U\Sigma V^T$. We can compute this fast in $m \cdot poly(k/\epsilon)$ time, since SA is a small $poly(k/\epsilon) \times m$ matrix. We then get, successively:

$$\begin{aligned} min_{rank-kX}\|XSA - A(SA)^+SA\|_F^2 &= min_{rank-kX}\|XU\Sigma V^T - A(SA)^+U\Sigma V^T\|_F^2 \\ &= min_{rank-kX}\|XU\Sigma - A(SA)^+U\Sigma\|_F^2 \\ &= min_{rank-kY}\|Y - A(SA)^+U\Sigma\|_F^2, \end{aligned} \tag{18}$$

where the second equality follows from the fact that V^T has orthonormal rows and the third equality from a simple change of variables ($Y = XU\Sigma$). (Technically, since A is a high-rank matrix, otherwise we would ignore S and work with the row-span of A, $U\Sigma$ is with constant high probability invertible and thus minimizing over $XU\Sigma$ with rank-k X is equivalent to minimizing over $Y = XU\Sigma$ with rank-k Y). Problem (8) can be solved if perform truncated SVD to the product $A(SA)^+U\Sigma$. Even though we can compute fast $U\Sigma$ and $(SA)^+ = V\Sigma^+U^T$, multiplying on the left by A is too costly to afford.

3.3 Sketching the Sketch

From the above discussion, it is clear that the bottleneck lies in the projection of the rows of A onto the row-span of SA. The rows of $A(SA)^TSA$ are exactly the projections of the rows of A onto the row-span of SA. The **novelty** of the algorithm lies in **approximately** projecting the rows of A onto the row-span of SA.

We consider another affine embedding, R, with d rows and $poly(k/\epsilon)$ columns. We know that:

$$\|XSAR - AR\|_F^2 = (1 \pm \epsilon)\|XSA - A\|_F^2 \tag{19}$$

By minimizing $\|XSAR - AR\|_F^2$ we still obtain a $(1 \pm \epsilon)$-approximation algorithm, as:

$$\begin{aligned} min_{rank-kX}\|XSAR - AR\|_F^2 &= (1 \pm \epsilon) min_{rank-kX}\|XSA - A\|_F^2 \\ &\leq (1 \pm \epsilon)^2\|A_k - A\|_F^2 \\ &= (1 \pm O(\epsilon)) min_{A'}\|A - A'\|_F^2 \end{aligned} \tag{20}$$

The minimizer of $\|XSAR - AR\|_F^2$ is this time given by $X_{SR}^* = (SAR)^+AR$ and by using the Pythagorean theorem one more time we get:

$$\|XSAR - AR\|_F^2 = \|A(SAR)^+SAR - AR\|_F^2 + \|XSAR - A(SAR)^+SAR\|_F^2, \tag{21}$$

Again, the first term does not depend on X and thus we concern ourselves with solving $min_{rank-kY}\|Y - AR(SAR)^+SAR\|_F^2$ (where $Y = XSAR$,

Finally, how are we guaranteed that Y takes the form $XSAR$? Suppose for the sake of contradiction, that this does not hold true. Then there exists a row of Y that does not belong to the row-span of SAR. Define $Y' = Y(SAR)^+SAR$. Then:

$$\begin{aligned} \|Y' - AR(SAR)^+SAR\|_F^2 &= \|Y(SAR)^+SAR - AR(SAR)^+SAR\| \\ &= \|Y(SAR)^+SAR - AR(SAR)^+(SAR(SAR)^+SAR)\| \\ &= \|(Y - AR(SAR)^+SAR)(SAR)^+SAR\|_F^2 \\ &\leq \|Y - AR(SAR)^+SAR\|_F^2, \end{aligned} \tag{22}$$

where the second equality follows from pseudo-inverse property $AA^+A = A$ and the inequality from applying the inequality $\|BP\|_F^2 \leq \|BP\|_F^2 + \|B(I - P)\|_F^2 = \|B\|_F^2$ for projection matrix $P = (SAR)^+(SAR)$ and $B = Y - AR(SAR)^+SAR$.

). Now computing the truncated SVD of $AR(SAR)^+SAR$ can be done fast given that all the factors are small matrices (in at least one dimension). The overall algorithm can be described as follows:

Algorithm 1 Approximate ℓ_F Low-Rank Approximation Algorithm.

```
1: function APPROX-RANK-K($A$)
2:     Generate affine embedding $S$.
3:     Generate affine embedding $R$.
4:     Compute $SA$.
5:     Compute $AR$ and $SAR$.
6:     Solve $min_{rank-kY} \|Y - AR(SAR)^{+}SAR\|_F^2$ exactly, using SVD.
7:     Output $Y(SAR)^{+}SA$.
8: end function
```

4 Approximate ℓ_1 low-rank approximation

Woodruff, Song & Zhong [3] recently proposed an approximate randomized algorithm for entrywise ℓ_1 norm low rank approximation, with an approximation factor of $\alpha = \log m \cdot poly(k)$ and runtime $O(nnz(A) + m \cdot poly(k))$. Formally, if A^* is the real minimizer, the algorithm outputs A' such that:

$$\|A - A'\|_F \leq \log m \cdot poly(k) \|A - A^*\|_1 \tag{23}$$

Woodruff, Song & Zhong [3] recently proposed an approximate randomized algorithm for entrywise ℓ_1 norm low rank approximation, with an approximation factor of $\alpha = \log n \cdot poly(k)$ and runtime $O(nnz(A) + m \cdot poly(k))$. This algorithm also heavily relies on sketching, but the underlying sketching matrices are combinations of CountSketch and Cauchy random matrices (as opposed to CountSketch and Gaussian random matrices). Formally, if A^* is the real minimizer, the algorithm outputs A' such that:

$$\|A - A'\|_F \leq \log m \cdot poly(k) \|A - A^*\|_1 \tag{24}$$

4.1 Sketching by a Cauchy Random Matrix

The entrywise-ℓ_1 norm of matrix $A \in \mathbb{R}^{n \times m}$ can be defined as:

$$\|A\|_1 = \sum_{j=1}^{m} \|A_{*,j}\| \tag{25}$$

Suppose $S \in \mathbb{R}^{r \times n}$ is a matrix of i.i.d Cauchy random variables, scaled by $1/r$. We now prove that, with constant probability:

$$\|SA\|_1 = O(\log m) \|A\|_1 \tag{26}$$

From the 1-stability of the Cauchy distribution, we get that each element of the j-th column of $SA \in \mathbb{R}^{r \times m}$, $SA_{i,j}$ is:

$$SA_{i,j} = \frac{1}{r} \|A_{*,j}\|_1 Z_i, \tag{27}$$

where Z_i is a Cauchy random variable.

Therefore, we have:

$$\|SA_{*,j}\| = \frac{1}{r}\|A_{*,j}\|\sum_{i=1}^{r}|Z_{i,j}|, \tag{28}$$

and where $|Z_{i,j}|$ is a half-Cauchy random variable.

We define $Z'_{i,j} = 1$ if $|Z_{i,j}| \leq m^3$ and $Z'_{i,j} = 0$ otherwise. Let $E_{i,j}$ be the event that $Z'_{i,j} = 1$, which happens with probability $p_{i,j}$.

We then have:

$$\begin{aligned} E[Z_{i,j}|E_{i,j}] &= E[Z'_{i,j}|E_{i,j}] \\ &= \int_0^{m^3} \frac{2z}{\pi(1+z^2)p_{i,j}}dz \\ &= \dots \\ &= \Theta(\log m) \end{aligned} \tag{29}$$

If we call E the event that for all $i, j \in [r] \times [n]$ the event $E_{i,j}$ occurs, we have:

$$P[\bar{E}] \leq \frac{m^2 \log m}{m^3} = \frac{\log m}{m}, \tag{30}$$

and thus:

$$P[E] \geq 1 - \frac{\log m}{m} \tag{31}$$

Furthermore:

$$\begin{aligned} E[Z_{i,j}|E_{i,j}] &= E[Z'_{i,j}|E_{i,j}, E]P[E|E_{i,j}] + E[Z'_{i,j}|E_{i,j}, \bar{E}][Z'_{i,j}|E_{i,j}, \bar{E}] \\ &\geq P[E|E_{i,j}, E]P[E|E_{i,j}] \\ &= \dots \\ &\geq E[Z'_{i,j}|E](1 - \frac{\log m}{m}) \end{aligned} \tag{32}$$

We get $E[Z'_{i,j}|E] = O(\log m)$ and therefore:

$$\|SA_{*,j}\| = O(\log m)\|A_{*,j}\|_1, \tag{33}$$

with constant probability, by a simple Markov Bound.

By taking a union bound over the m events $\|SA_{*,j}\| = O(\log m)\|A\|_1$, we finally get that, with very high probability (at least 9/10 for appropriate constants):

$$\|SA\|_1 = O(\log m)\|A\|_1 \tag{34}$$

4.2 A $\sqrt{k \log k} \log m$ approximation in the rowspan of SA

It can also be proved for any fixed $n \times k$ matrix U, if S is an $r \times n$ matrix of i.i.d Cauchy random variables, with $r = O(k \log(k))$, then with probability at least $9/10$, simultaneously for all $x \in \mathbb{R}^m$:

$$\|Ux\|_1 \leq \|SUx\|_1 = O(k \log(k))\|Ux\|_1 \tag{35}$$

Let us now consider a fixed $U \in \mathbb{R}^{n \times k}$ and let the matrices $V^*, V' \in \mathbb{R}^{k \times n}$ be defined as $V' = argmin_V \|SUV - SA\|_1$ and $V^* = argmin_V \|UV - A\|_1$ respectively. We observe that:

$$\begin{aligned} \|SUV' - SA\|_1 &\geq \|SUV' - SUV^*\|_1 - \|SUV^* - SA\|_1 \\ &= \sum_{j=1}^{k} \|SU(V' - V^*)_{*,j}\|_1 - \|SUV^* - SA\|_1 \\ &\geq \sum_{j=1}^{k} \|U(V' - V^*)_{*,j}\|_1 - \|SUV^* - SA\|_1 \\ &= \|U(V' - V^*)\|_1 - \|SUV^* - SA\|_1 \\ &\geq \|UV' - A\|_1 - \|UV^* - A\|_1 - \|SUV^* - SA\|_1, \end{aligned} \tag{36}$$

where the first inequality follows directly from the triangle inequality, the second inequality uses the property described by equation (30), and the third inequality follows again from the triangle inequality.

After re-arranging terms in the previous inequality and using the definitions of V^*, V', we can prove, that with high probability:

$$\|UV' - A\|_1 \leq O(\log(m)) \min_V \|UV - A\|_1 \tag{37}$$

Given the problem $\min_V \|SUV - SA\|_1$, consider its relaxation:

$$\min_V \|SUV - SA\|_{1,2} = \min_V \sum_{i=1}^{m} \|(SUV - SA)_{*,i}\|_2 \tag{38}$$

It is straighforward to observe that fixing U, one can find the i-th columns of V by solving the ℓ_2 regression $\min_{V_{*,i}} \|(SUV - SA)_{*,i}\|_2$. The solution to this problem is $V''_{*,i} = (SU)^+(SA)_{*,i}$, where SU^+ is the pseudo-inverse of matrix $SU \in \mathbb{R}^{s \times k}$ and $(SA)_{*,i}$ is the i-th column of matrix $SA \in \mathbb{R}^{r \times m}$. Combining the solutions for each column $V''_{*,i} \in \mathbb{R}^r$, we obtain the following expression for $V'' = (SU)^+ SA$; it lies in the rowspan of $SA \in \mathbb{R}^{r \times m}$.

One can then use the definition of V' and V'', the Cauchy-Shwartz inequality and the fact that for $x \in \mathbb{R}^m$, $\|x\|_2 \leq \|x\|_1$ and prove that:

$$\|SUV'' - SA\|_1 \leq \sqrt{r} \min_V \|SUV - SA\|_1 \tag{39}$$

4.3 Sketch & Sketch & Replace ℓ_1 by ℓ_F

Assume that $A^* = U^*V^*$ is the minimizer of the original problem $\min_{rank-kA'} \|A - A'\|_1$ and that we know U^*. If we then consider the hypothetical regression problem $\min_V \|U^*V - A\|_1$, we know that if S is an $(r = k \log k) \times n$ matrix of i.i.d. Cauchy random variables, there is a matrix $UV'' \in \mathbb{R}^{n \times m}$, with the matrix V'' lying in the row-span of $SA \in \mathbb{R}^{r \times m}$ and for which the following inequality holds:

$$\|U^*V'' - A\|_1 = O(\sqrt{r} \log m) \min_V \|U^*V - A\|_1 \tag{40}$$

Therefore we can write $V'' = XSA$ for some unknown matrix $X \in \mathbb{R}^{k \times r}$, and if X^* is the minimizer of the problem $\min_X \|U^*XSA - A\|_1$ then the matrix U^*X^*SA would be an $O(\sqrt{r}\log m)$-approximate entrywise-ℓ_1 low rank approximation. The ba news is that we do not know or have a way of guessing the matrix $U^* \in \mathbb{R}^{n \times k}$.

The good news however is that we can sketch the problem on the right by another matrix $R \in \mathbb{R}^{n \times r}$, where $r = kpoly(\log k)$ and conclude that there exists a matrix of the form $ARYXSA$, where $Y \in \mathbb{R}^{r \times k}$, which satisfies:

$$\|ARYXSA - A\|_1 \leq poly(k \log m) \min_{rank-kA'} \|A - A'\|_1 \tag{41}$$

We therefore focus on solving the problem $\min_{Y,X} \|ARYXSA - A\|_1$. It turns out, that we can sketch both on the left and right by two Cauchy matrices of i.i.d random variables, $T_L \in \mathbb{R}^{t \times n}$ and $T_R \in \mathbb{R}^{n \times t}$. If $t = kpoly(\log k)$, then for the minimizers $Y' \in \mathbb{R}^{r \times k}$, and $X' \in \mathbb{R}^{k \times s}$ of the problem $\min_{Y,X} \|T_L ARYXSAT_R - T_L AT_R\|_1$, with very high probability, it is true that:

$$\|T_L ARY'X'SAT_R - T_L AT_R\|_1 \leq poly(k \log n) \min_{rank-kA'} \|A - A'\|_1 \tag{42}$$

We also note that the problem $\min_{Y,X} \|T_L ARYXSAT_R - T_L AT_R\|_1$ is very small and does not depend on n. Also, the dimensions of matrices $T_L AR$, SAT_R and $T_L AT_R$ and the unknown matrices Y, X are all $kpoly(\log k)$.

Let instead $X'' \in \mathbb{R}^{k \times s}$ and $Y'' \in \mathbb{R}^{r \times k}$ be the minimizers of the problem $\min_{Y,X} \|T_L ARYXSAT_R - T_L AT_R\|_F$, for which there exists a closed-form solution. Moreover:

$$\begin{aligned}
\|T_L ARY''X''SAT_R - T_L AT_R\|_F &\leq \|T_L ARY'X'SAT_R - T_L AT_R\|_F \\
&\leq \sqrt{kpoly(\log k) \cdot kpoly(\log k)} \|T_L ARY'X'SAT_R - T_L AT_R\|_1 \\
&= kpoly(\log k) \|T_L ARY'X'SAT_R - T_L AT_R\|_1 \\
&\leq kpoly(\log k) poly(k \log m) \min_{rank-kA'} \|A - A'\|_1 \\
&= \beta \cdot poly(k \log m) \min_{rank-kA'} \|A - A'\|_1, \quad (43)
\end{aligned}$$

where the first inequality follows from the definition of Y'', X'', the second inequality follows from the inequality:

$$\|M\|_F \leq \sqrt{s \cdot t}\|M\|_1, \tag{44}$$

for any matrix $M \in \mathbb{R}^{s \times t}$, the third inequality is just equation (30) and finally the last equality follows from setting $\beta = kpoly(\log k)$.

The overall algorithm then is:

Algorithm 2 Approximate ℓ_1 Low-Rank Approximation Algorithm.

```
1:  function APPROX-RANK-K($A$)
2:      Generate Cauchy matrix $S$.
3:      Compute $SA$
4:      Compute $C_{i,*} = argmin_x \|x^T SA - A_{i,*}\|_1$
5:      Compute $B = C \cdot SA$
6:      Generate Cauchy sketches $T_L, R, D, T_R$.
7:      Compute $SA$.
8:      Compute $T_L BR$, $DBT_R$ and $T_L BT_R$.
9:      Solve $min_{X,Y} \|T_L BRXYDBT_R - T_L BT_R\|_F$ exactly.
10:     Output $BRX, YDB$.
11: end function
```

5 Experiments

We implemented the approximate-ℓ_F and approximate-ℓ_1 algorithms and evaluated them on both synthetic and real-world datasets. We performed 2 different experiments on 3 synthetic datasets and 2 real-world datasets. In the first experiment, we vary the rank k of the desired approximation, whereas in the second experiment we fix a rank $k = 50$ and vary the number of rows/columns of the sketching matrices. For evaluating the approximate-ℓ_F algorithm we used $s = c \cdot \frac{k}{\epsilon}$ rows/columns of sketching matrices S and R with $\epsilon = 0.1$, whereas for evaluating the ℓ_1-approximate algorithm we used $s = c \cdot k \log k$ as the small dimension for the sketching matrices S, R, T_L and T_R.

5.1 Datasets

5.1.1 Synthetic Datasets

We generated 3 sparse synthetic datasets: a matrix with $n = 10^5$ and $m = 10^3$ random i.i.d. normal variables, with $\|M\|_0 = \text{nnz}(M) = 0.50nm$ (sparse), $\|M\|_0 = \text{nnz}(M) = 0.50nm$ (semi-dense) and $|M\|_0 = \text{nnz}(M) = 0.75nm$ (dense) nonzero elements, respectively.

5.1.2 Real Datasets

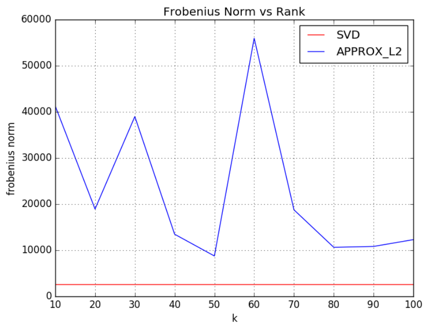
As our first real-world dataset, we use the MovieLens 1M dataset [9], which contains 1 million ratings (0-5) from 6000 users on 4000 movies. As our second real-world dataset, we use the NIPS dataset [8], which contains the distribution of words in the full text of the NIPS conference papers published from 1987 to 2015. The dataset is in the form of a 11463 x 5812 matrix of word counts, containing 11463 words and 5811 NIPS conference papers. Each column contains the number of times each word appears in the corresponding document.

5.2 Programming Environment

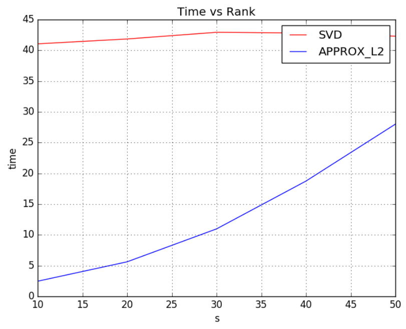
The experiments were executed on a MacBook Pro with a 2.5GHz Intel Core i7 processor and 16GB 1600 MHz DDR3 memory. The implementation was done in Python using the numpy library.

5.3 Results

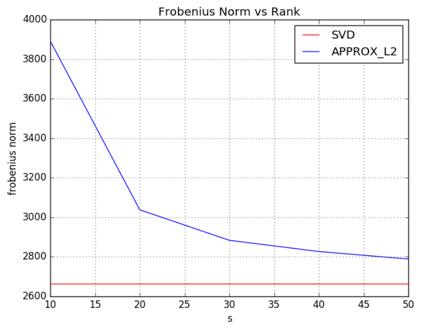
5.3.1 Sparse Synthetic Dataset



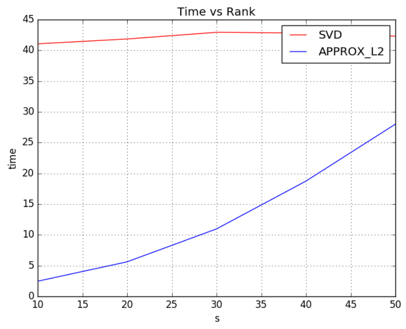
(a) $\|\|_F$ vs k



(b) Time vs k

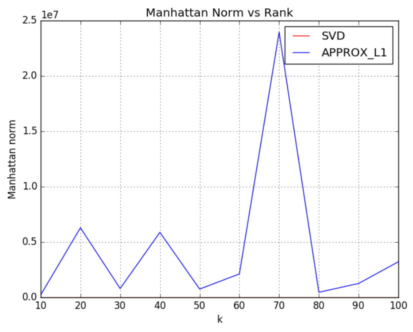


(c) $\|\|_F$ vs c

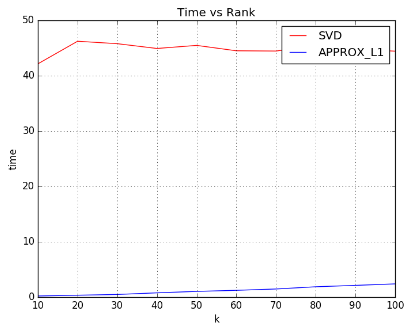


(d) Time vs c

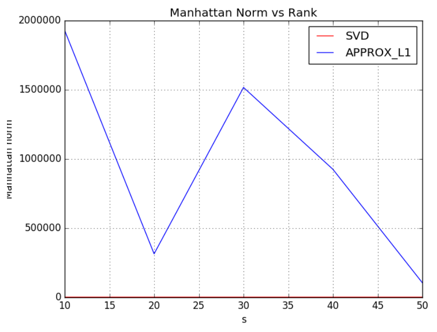
Figure 1: ℓ_F-approximate algorithm on MovieLens. Plots illustrate ℓ_F Norm & Time vs (i) rank k and (ii) number of rows/columns $s = c \cdot \frac{k}{\epsilon^2}$ of sketching matrices S and R. First experiment ((a), (b)): $k \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.



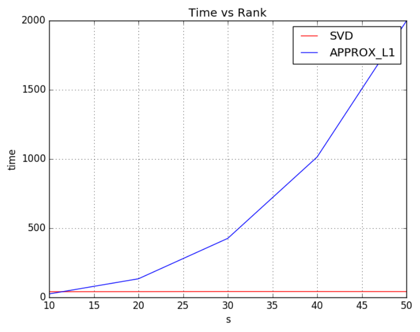
(a) $\|\|_F$ vs k



(b) Time vs k



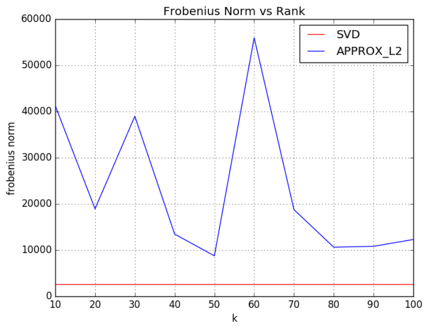
(c) $\|\|_F$ vs c



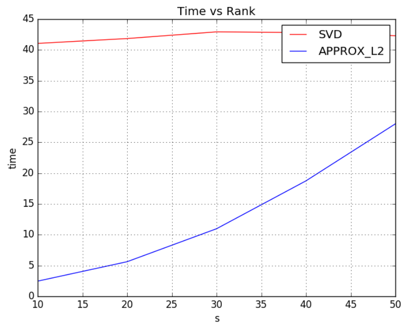
(d) Time vs c

Figure 2: ℓ_1 algorithm on MovieLens. Plots illustrate ℓ_1 Norm & Time vs (i) rank k and (ii) small dimension $s = c \cdot k \log k$ of Sketching matrices S, R, T_L and T_R. First experiment ((a),(b)): $k \in \{10, 20, 30, 40, 50\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.

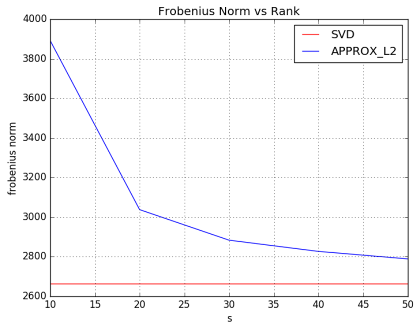
5.3.2 Semi-Sparse Synthetic Dataset



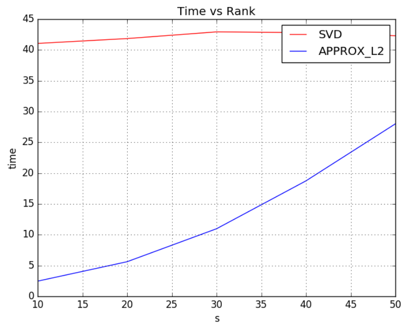
(a) $\|\|_F$ vs k



(b) Time vs k

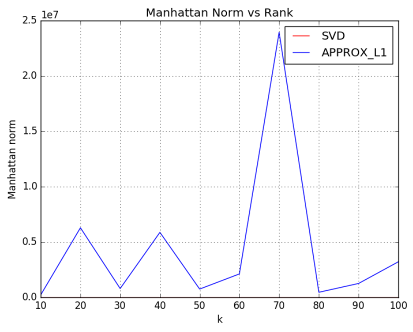


(c) $\|\|_F$ vs c

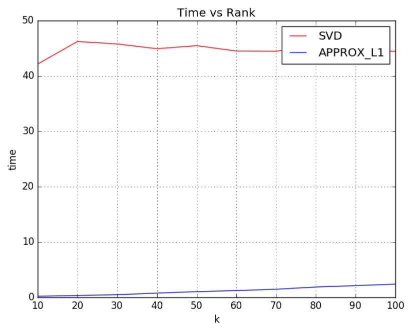


(d) Time vs c

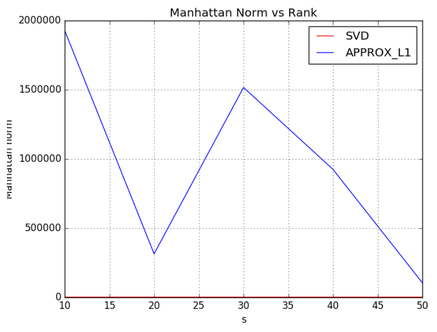
Figure 3: ℓ_F-approximate algorithm on MovieLens. Plots illustrate ℓ_F Norm & Time vs (i) rank k and (ii) number of rows/columns $s = c \cdot \frac{k}{\epsilon^2}$ of sketching matrices S and R. First experiment ((a), (b)): $k \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.



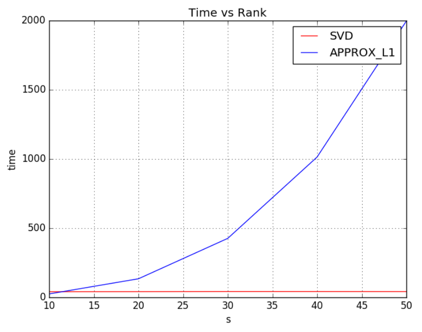
(a) $\|\|_F$ vs k



(b) Time vs k



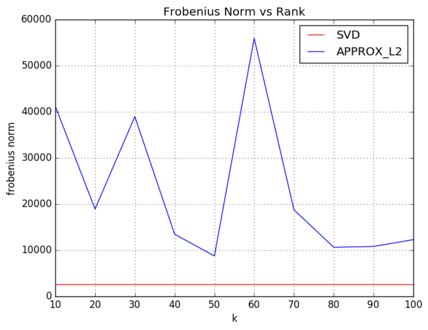
(c) $\|\|_F$ vs c



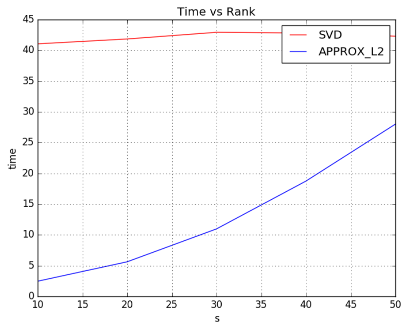
(d) Time vs c

Figure 4: ℓ_1 algorithm on MovieLens. Plots illustrate ℓ_1 Norm & Time vs (i) rank k and (ii) small dimension $s = c \cdot k \log k$ of Sketching matrices S, R, T_L and T_R. First experiment ((a),(b)): $k \in \{10, 20, 30, 40, 50\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.

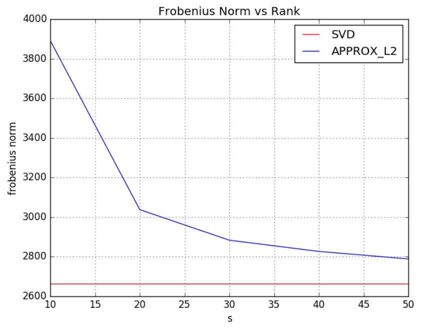
5.3.3 Dense Synthetic Dataset



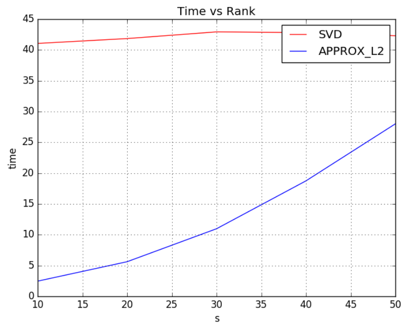
(a) $\|\|_F$ vs k



(b) Time vs k

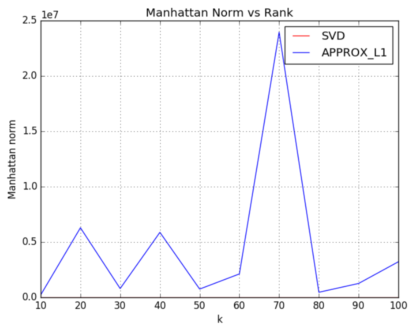


(c) $\|\|_F$ vs c

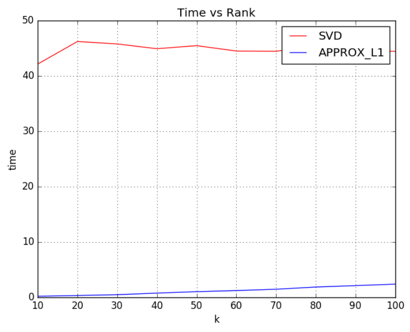


(d) Time vs c

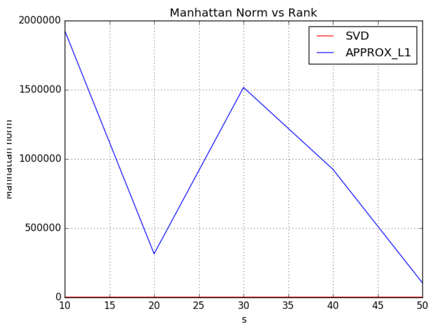
Figure 5: ℓ_F-approximate algorithm on MovieLens. Plots illustrate ℓ_F Norm & Time vs (i) rank k and (ii) number of rows/columns $s = c \cdot \frac{k}{\epsilon^2}$ of sketching matrices S and R. First experiment ((a), (b)): $k \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.



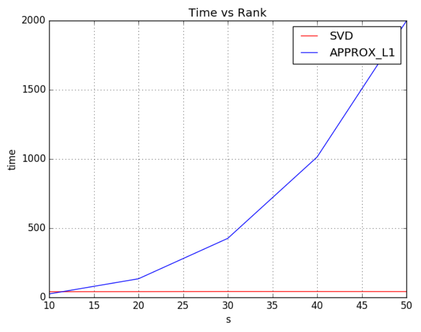
(a) $\|\|_F$ vs k



(b) Time vs k



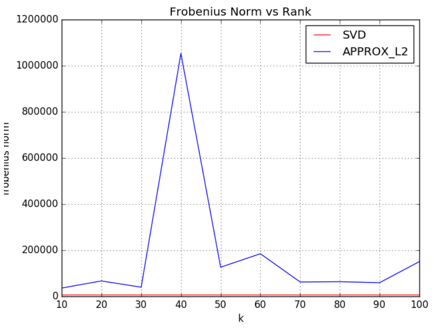
(c) $\|\|_F$ vs c



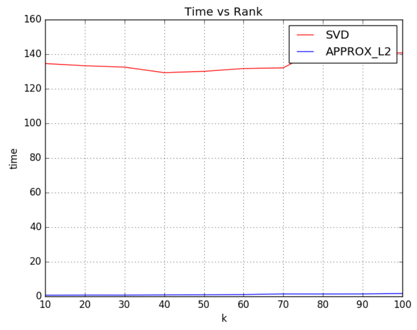
(d) Time vs c

Figure 6: ℓ_1 algorithm on MovieLens. Plots illustrate ℓ_1 Norm & Time vs (i) rank k and (ii) small dimension $s = c \cdot k \log k$ of Sketching matrices S, R, T_L and T_R. First experiment ((a),(b)): $k \in \{10, 20, 30, 40, 50\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.

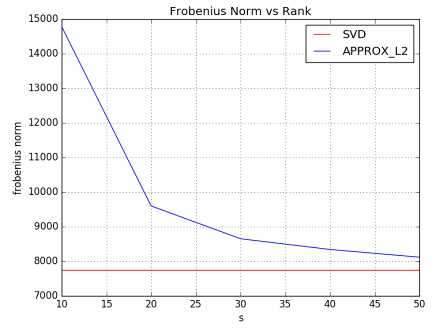
5.3.4 NIPS Dataset



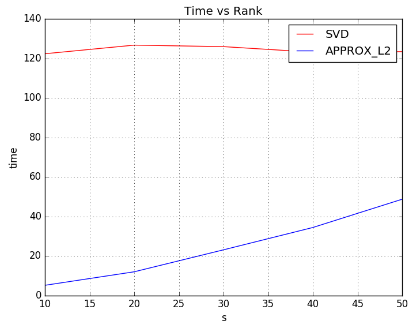
(a) $\|\|_F$ vs k



(b) Time vs k

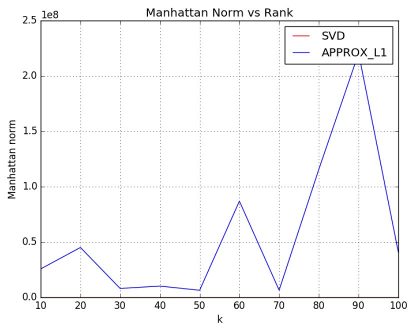


(c) $\|\|_F$ vs c

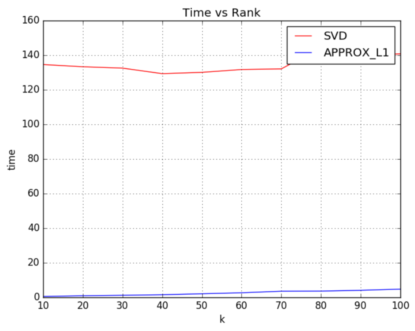


(d) Time vs c

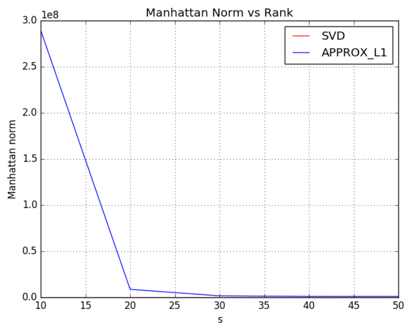
Figure 7: ℓ_F-approximate algorithm on NIPS. Plots illustrate ℓ_F Norm & Time vs (i) rank k and (ii) number of rows/columns $s = c \cdot \frac{k}{\epsilon^2}$ of sketching matrices S and R. First experiment ((a), (b)): $k \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.



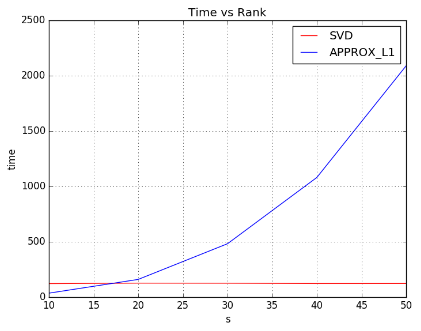
(a) $\|\|_F$ vs k



(b) Time vs k



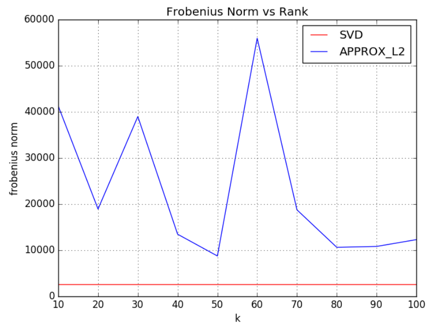
(c) $\|\|_F$ vs c



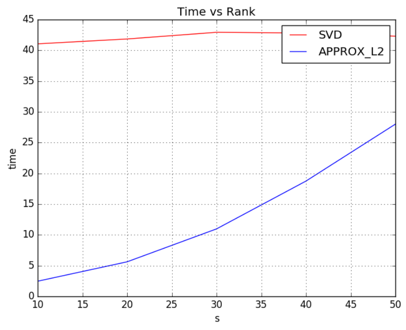
(d) Time vs c

Figure 8: ℓ_1-approximate algorithm on NIPS. Plots illustrate ℓ_1 Norm & Time vs (i) rank k and (ii) number of rows/columns $s = c \cdot k \log k$ of sketching matrices S, R, T_L and T_R. First experiment ((a),(b)): $k \in \{10, 20, 30, 40, 50\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.

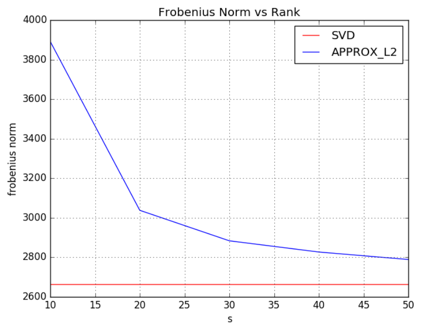
5.3.5 MovieLens Dataset



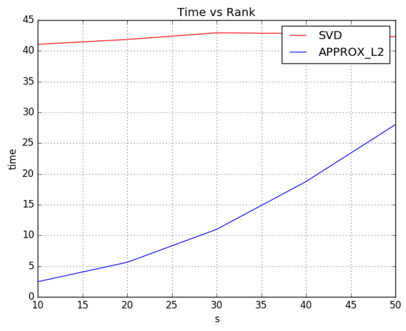
(a) $\|\|_F$ vs k



(b) Time vs k

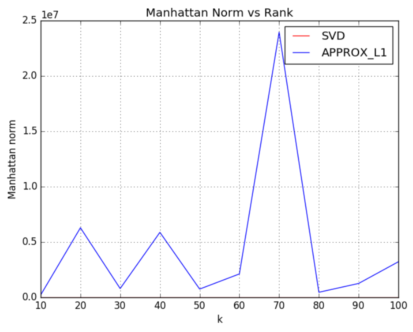


(c) $\|\|_F$ vs c

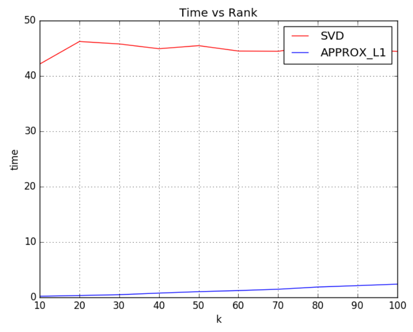


(d) Time vs c

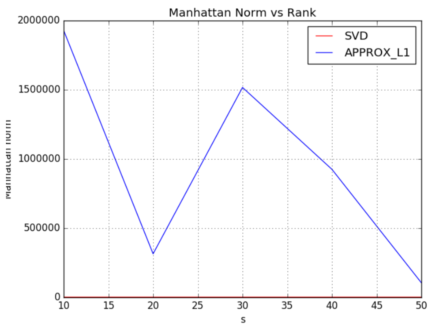
Figure 9: ℓ_F-approximate algorithm on MovieLens. Plots illustrate ℓ_F Norm & Time vs (i) rank k and (ii) number of rows/columns $s = c \cdot \frac{k}{\epsilon^2}$ of sketching matrices S and R. First experiment ((a), (b)): $k \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.



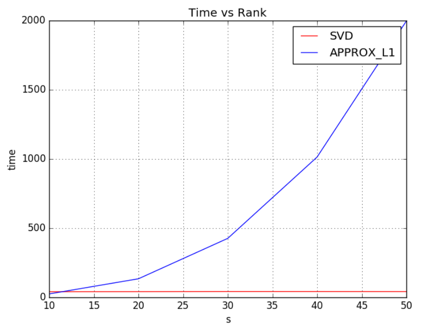
(a) $\|\|_F$ vs k



(b) Time vs k



(c) $\|\|_F$ vs c



(d) Time vs c

Figure 10: ℓ_1 algorithm on MovieLens. Plots illustrate ℓ_1 Norm & Time vs (i) rank k and (ii) small dimension $s = c \cdot k \log k$ of Sketching matrices S, R, T_L and T_R. First experiment ((a),(b)): $k \in \{10, 20, 30, 40, 50\}$ and $c = 1$. Second experiment ((c),(d)): rank $k = 50$ and $c \in \{10, 20, 30, 40, 50\}$. Both experiments use $\epsilon = 0.1$.

5.4 Discussion

The approximate-ℓ_F algorithm succeeds as we vary k. The norm $\|A - A_F\|_F = c\|A - A_k\|_F$, for c a constant in the range [10,20]. The quality of the ℓ_F-algorithm approximation factor improves and its running time increases as we increase the number s of rows of the sketching matrices.

The approximate-ℓ_1 algorithm succeeds as we vary k. The norm $\|A - A_1\|_1 = c\|A - A_k\|_F$, for c a constant in the range $[10^5, 10^8]$. Observe that since computing the optimum $\|A - U^*V^*\|_1$ is NP-hard and requires the use of a polynomial system solver, we compare against the Frobenius norm $\|A - A_k\|_F$. We know that $\|A - U^*V^*\|_1 \leq \sqrt{n \cdot m}\|A - U^*V^*\|_F$. Also the ℓ_1-approximate algorithm does not have a constant factor approximation but rather a $\log m \cdot k \cdot \text{poly}(\log k)$ factor approximation. As expected, the running time of the approximate-ℓ_1

algorithm increases as we increase the number s of rows of the sketching matrices.

6 Acknowledgements

First, I would like to thank professor Phil B. Gibbons for helping me explore different research directions and figure out an area I am strong in and passionate about. Second, I would like to thank professors Gary Miller and David Woodruff for introducing me to Randomized Numerical Linear Algebra. Thirdly, I would like to thank my advisor Manuela Veloso for her endless suppport and mentorship. Finally, I would like to thank the “Alexander S. Onassis" Foundation for a graduate fellowship that partially supports my graduate studies in the U.S.

References

[1] David P. Woodruff. Sketching as a Tool for Numerical Linear Algebra. CoRR, 1411.4357, 2014. http://arxiv.org/abs/1411.4357.

[2] Nicolas Gillis and Stephen A Vavasis. On the complexity of robust pca and ℓ_1-norm low-rank matrix approximation. arXiv preprint arXiv:1509.09236, 2015.

[3] Zhao Song, David P. Woodruff, Peilin Zhong. Low Rank Approximation with Entrywise ℓ_1-Norm Error. arXiv preprint arXiv:1611.00898

[4] Kenneth L. Clarkson and David P. Woodruff. Low Rank Approximation and Regression in input Sparsity Time. arXiv preprint arXiv:1207.6365v4, 2013.

[5] Flavio Chierichetti, Sreenivas Gollapudi, Ravi Kumar, Silvio Lattanzi, RIna Panigrahy, David P. Woodruff. Algorithms for ℓ_p Low Rank Approximation. arXiv preprint arXiv:1705:06730v1, 2017.

[6] Karl Bringmann, Pavel Kolev, David P. Woodruff. Approximation Algorithms for ℓ_0-Low Rank Approximation. arXiv preprint arXiv:1710.11253

[7] Noga Alon, Rina Panigrahy, and Sergey Yekhanin. Deterministic approximation algorithms for the nearest codeword problem. In Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings, pages 339–351, 2009.

[8] Perrone V., Jenkins P. A., Spano D., Teh Y. W. Poisson Random Fields for Dynamic Feature Models. arXiv preprint arXiv:1611.07460, 2013.

[9] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872

[10] Daniel M. Kane and Jelani Nelson. Sparser johnson-lindenstrauss transforms. J. ACM, 61(1):4, 2014.