Streaming and Distributed Algorithms for Robust Column Subset Selection

Shuli Jiang [1] **Dongyu Li** [1] **Irene Mengze Li** [1] **Arvind V. Mahankali** [1] **David P. Woodruff** [1]

Abstract

We give the first single-pass streaming algorithm for Column Subset Selection with respect to the entrywise ℓ_p-norm with $1 \leq p < 2$. We study the ℓ_p norm loss since it is often considered more robust to noise than the standard Frobenius norm. Given an input matrix $A \in \mathbb{R}^{d\times n}$ ($n \gg d$), our algorithm achieves a multiplicative $k^{\frac{1}{p}-\frac{1}{2}}\text{poly}(\log nd)$-approximation to the error with respect to the *best possible column subset* of size k. Furthermore, the space complexity of the streaming algorithm is optimal up to a logarithmic factor. Our streaming algorithm also extends naturally to a 1-round distributed protocol with nearly optimal communication cost. A key ingredient in our algorithms is a reduction to column subset selection in the $\ell_{p,2}$-norm, which corresponds to the p-norm of the vector of Euclidean norms of each of the columns of A. This enables us to leverage strong coreset constructions for the Euclidean norm, which previously had not been applied in this context. We also give the first provable guarantees for greedy column subset selection in the $\ell_{1,2}$ norm, which can be used as an alternative, practical subroutine in our algorithms. Finally, we show that our algorithms give significant practical advantages on real-world data analysis tasks.

1. Introduction

Column Subset Selection (k-CSS) is a widely studied approach for low-rank approximation and feature selection. In k-CSS, on an input data matrix $A \in \mathbb{R}^{d\times n}$, we seek a small subset A_I of k columns from A such that $\min_V \|A_I V - A\|$ is minimized for some norm $\|\cdot\|$. In contrast to general low-rank approximation, where one finds $U \in \mathbb{R}^{d\times k}$ and $V \in \mathbb{R}^{k\times n}$ such that $\|UV - A\|$ is minimized (Clarkson & Woodruff, 2013; Woodruff, 2014c), k-CSS outputs an actual subset of the columns of A as the left factor U. The main advantage of k-CSS over general low-rank approximation is that the resulting factorization is more interpretable. For instance, the subset A_I can represent salient features of A, while in general low-rank approximation, the left factor U may not be easy to relate to the original dataset. In addition, the subset A_I preserves sparsity of the original matrix A.

k-CSS has been extensively studied in the Frobenius norm (Guruswami & Sinop, 2012; Boutsidis et al., 2014; Boutsidis & Woodruff, 2017; Boutsidis et al., 2008b) and also the operator norm (Halko et al., 2011; Woodruff, 2014a). A number of recent works (Song et al., 2017; Chierichetti et al., 2017; Dan et al., 2019; Ban et al., 2019; Mahankali & Woodruff, 2021) studied this problem in the ℓ_p norm for $1 \leq p < 2$, due to its robustness properties. The ℓ_1 norm, especially, is less sensitive to outliers, and better at handling missing data and non-Gaussian noise, than the Frobenius norm (Song et al., 2017). Using the ℓ_1 norm loss has been shown to lead to improved performance in many real-world applications of low-rank approximation, such as structure-from-motion (Ke & Kanade, 2005) and image denoising (Yu et al., 2012).

In this work, we give algorithms for k-CSS in the ℓ_p norm, or k-CSS$_p$, in the streaming and distributed settings for $1 \leq p < 2$. The streaming algorithm can be used on small devices with memory constraints, when the elements of a large dataset are arriving one at a time and storing the entire stream is not possible. The distributed algorithm is useful when a large dataset is partitioned across multiple devices. Each device only sees a subset of the entire dataset, and it is expensive to communicate or transmit data across devices.

1.1. Background: k-CSS$_p$ in the Streaming Model

We study the column-update streaming model (See Definition 2.1) where the dataset A is a $d \times n$ matrix with $n \gg d$, and the columns of A arrive one by one. In this setting, our goal is to maintain a good subset of columns of A, while using space that can be linear in d and k, but sublinear in n. This model is relevant in settings where memory is limited, and the algorithm can only make one pass over the input data, e.g. (Drineas & Kannan, 2003).

Authors are listed in alphabetical order. [1]School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA. Correspondence to: David P. Woodruff <dwoodruf@andrew.cmu.edu>, Shuli Jiang <shulij@andrew.cmu.edu>, Arvind V. Mahankali <amahanka@andrew.cmu.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

k-CSS and low-rank approximation with the Frobenius norm loss have been extensively studied in the column-update streaming model (or equivalently the row-update streaming model), e.g. (Clarkson & Woodruff, 2009; Liberty, 2013; Ghashami & Phillips; Woodruff, 2014b; Altschuler et al., 2016; Boutsidis et al., 2016). However, k-CSS in the streaming model has not been studied in the more robust ℓ_p norm, and it is not clear how to adapt existing offline k-CSS$_p$ or ℓ_p low-rank approximation algorithms into streaming k-CSS$_p$ algorithms. In fact, the only work which studies ℓ_p low-rank approximation in the column-update streaming model is (Song et al., 2017). They obtain a poly$(k, \log d)$-approximate algorithm with $\widetilde{O}(dk)$ space, and using the techniques of (Wang & Woodruff, 2019) this approximation factor can be further improved. However, it is not clear how to turn this streaming algorithm of (Song et al., 2017) into a streaming k-CSS$_p$ algorithm.

In addition, it is not clear how to adapt many known algorithms for ℓ_p low-rank approximation and column subset selection into one-pass streaming algorithms for k-CSS$_p$. For instance, one family of offline algorithms for k-CSS$_p$, studied by (Chierichetti et al., 2017; Dan et al., 2019; Song et al., 2019b; Mahankali & Woodruff, 2021), requires $O(\log n)$ iterations. In each of these iterations, the algorithm selects $\widetilde{O}(k)$ columns of A and uses their span to approximate a constant fraction of the remaining columns, which are then discarded. Since these rounds are adaptive, this algorithm would require $O(\log n)$ passes over the columns of A if implemented as a streaming algorithm, making it unsuitable.

1.2. Streaming k-CSS$_p$: Our Results

In this work, we give the first non-trivial single-pass algorithm for k-CSS$_p$ in the column-update streaming model (Algorithm 1). The space complexity of our algorithm, $\widetilde{O}(kd)$, is nearly optimal up to a logarithmic factor, since for k columns, each having d entries, $\Omega(dk)$ words of space are needed. Our algorithm is bi-criteria, meaning it outputs a column subset of size $\widetilde{O}(k)$ (where $\widetilde{O}$ hides a poly$(\log k)$ factor) for a target rank k. Bi-criteria relaxation is standard in most low-rank approximation algorithms to achieve polynomial running time, since obtaining a solution with rank exactly k or exactly k columns can be NP hard, e.g., in the ℓ_1 norm case (Gillis & Vavasis, 2015). Furthermore, we note that our algorithm achieves an $\widetilde{O}(k^{1/p-1/2})$-approximation to the error from *the optimal column subset* of size k, instead of the optimal rank-k approximation error (i.e., $\min_{\text{rank}-k A_k} \|A_k - A\|_p$). (Song et al., 2017) shows any matrix $A \in \mathbb{R}^{d\times n}$ has a subset of $O(k \log k)$ columns which span an $\widetilde{O}(k^{1/p-1/2})$-approximation to the optimal rank-k approximation error. Thus our algorithm is able to achieve an $\widetilde{O}(k^{2/p-1})$-approximation relative to the optimal rank-k approximation error.

1.3. Streaming k-CSS$_p$: Our Techniques

Our first key insight is that we need to maintain a small subset of columns with size independent of n that globally approximates all columns of A well throughout the stream under the desired norm. A good data summarization technique for this is a *strong coreset*, which can be a subsampled and reweighted subset of columns that preserves the cost of projecting onto all subspaces (i.e., the span of the columns we ultimately choose in our subset). However, *strong coresets* are not known to exist in the ℓ_p norm for $1 \le p < 2$. Thus, we reduce to low rank approximation in the $\ell_{p,2}$ norm, which is the sum of the p-th powers of the Euclidean norms of all columns (see Definition 2.3). Strong coresets in the $\ell_{p,2}$ norm have been extensively studied and developed, see, e.g., (Sohler & Woodruff, 2018). However, to the best of our knowledge, $\ell_{p,2}$ strong coresets have not been used for developing ℓ_p low-rank approximation or k-CSS algorithms prior to our work.

The next question to address is how to construct and maintain *strong coresets* of A during the stream. First we observe that *strong coresets* are mergeable. If two coresets C_1, C_2 provide a $(1 \pm \epsilon)$-approximation to the $\ell_{p,2}$-norm cost of projecting two sets of columns A_M and A_N respectively, to any subspace, then the coreset of $C_1 \cup C_2$ gives a $(1 \pm \epsilon)^2$-approximation to the cost of projecting $A_M \cup A_N$ onto any subspace. Thus, we can construct *strong coresets* for batches of input columns from A and merge these coresets to save space while processing the stream. In order to reduce the number of merges, and hence the approximation error, we apply the Merge-and-Reduce framework (see, e.g., (McGregor, 2014)). Our algorithm greedily merges the *strong coresets* in a binary tree fashion, where the leaves correspond to batches of the input columns and the root is the single *strong coreset* remaining at the end. This further enables us to maintain only $O(\log n)$ coresets of size $\widetilde{O}(k)$ throughout the stream, and hence achieve a space complexity of $\widetilde{O}(kd)$ words.

One problem with reducing to the $\ell_{p,2}$ norm is that this leads to an approximation factor of $d^{1/p-1/2}$. Our second key insight is to apply a dimensionality reduction technique in the ℓ_p norm, which reduces the row dimension from d to $k$poly$(\log nd)$ via data-independent (i.e., oblivious) sketching matrices of i.i.d. p-stable random variables (see Definition 2.4), which only increases the approximation error by a factor of $O(\log nd)$. The overall approximation error is thus reduced to $O(k^{1/p-1/2}\text{poly}(\log nd))$.

As a result, our algorithm constructs coresets for the sketched columns instead of the original columns. However, since we do not know which subset of columns will be selected a priori, we need approximation guarantees of dimensionality reduction via oblivious skecthing matrices for all possible subsets of columns. We combine a net argument

with a union bound over all *possible* subspaces spanned by column subsets of A of size $\widetilde{O}(k)$ (see Lemma 2). Previous arguments involving sketching for low-rank approximation algorithms, such as those by (Song et al., 2017; Ban et al., 2019; Mahankali & Woodruff, 2021), only consider a single subspace at a time.

At the end of the stream we will have a single coreset of size $k \cdot \text{poly}(\log nd)$. To further reduce the size of the set of columns output, we introduce an $O(1)$-approximate bi-criteria column subset selection algorithm in the $\ell_{p,2}$ norm (k-CSS$_{p,2}$; see Section 3.3) that selects $k\text{poly}(\log k)$ columns from the coreset as the final output.

1.4. Distributed k-CSS$_p$: Results and Techniques

Our streaming algorithm and techniques can be extended to an efficient one-round distributed protocol for k-CSS$_p$ ($1 \leq p < 2$). We consider the column partition model in the distributed setting (see Definition 2.2), where s servers communicate to a central coordinator via 2-way channels. This model can simulate arbitrary point-to-point communication by having the coordinator forward a message from one server to another; this increases the total communication by a factor of 2 and an additive $\log s$ bits per message to identify the destination server.

Distributed low-rank approximation arises naturally when a dataset is too large to store on one machine, takes a prohibitively long time for a single machine to compute a rank-k approximation, or is collected simultaneously on multiple machines. The column partition model arises naturally in many real world scenarios such as federated learning (Farahat et al., 2013; Altschuler et al., 2016; Liang et al., 2014). Despite the flurry of recent work on k-CSS$_p$, this problem remains largely unexplored in the distributed setting. This should be contrasted to Frobenius norm column subset selection and low-rank approximation, for which a number of results in the distributed model are known, see, e.g., (Altschuler et al., 2016; Balcan et al., 2015; 2016; Boutsidis et al., 2016).

In this work, we give the first one-round distributed protocol for k-CSS$_p$ (Algorithm 3). Each server sends the coordinator a *strong coreset* of columns. To reduce the number of columns output, our protocol applies the $O(1)$-approximate bi-criteria k-CSS$_{p,2}$ algorithm to give $k\text{poly}(\log k)$ output columns, independent of s, n, and d. The communication cost of our algorithm, $\widetilde{O}(sdk)$ is optimal up to a logarithmic factor. Our distributed protocol is also a bi-criteria algorithm outputting $\widetilde{O}(k)$ columns and achieving an $\widetilde{O}(k^{1/p-1/2})$-approximation relative to the error of the optimal column subset.

1.5. Comparison with Alternative Approaches in the Distributed Setting

If one only wants to obtain a good left factor U, and not necessarily a column subset of A, in the column partition model, one could simply sketch the columns of A_i by applying an oblivious sketching matrix S on each server. Each server sends $A_i \cdot S$ to the coordinator. The coordinator obtains $U = AS$ as a column-wise concatenation of the A_iS. (Song et al., 2017) shows that AS achieves an $\widetilde{O}(\sqrt{k})$ approximation to the optimal rank-k error, and this protocol only requires $\widetilde{O}(sdk)$ communication, $O(1)$ rounds and polynomial running time. However, while AS is a good left factor, it does not correspond to an actual subset of columns of A.

Obtaining a subset of columns that approximates A well with respect to the p-norm in a distributed setting is non-trivial. One approach due to (Song et al., 2017) is to take the matrix AS described above, sample rows according to the Lewis weights (Cohen & Peng, 2015) of AS to get a right factor V, which is in the row span of A, and then use the Lewis weights of V to sample columns of A. Unfortunately, this protocol only achieves a loose $\widetilde{O}(k^{3/2})$ approximation to the optimal rank-k error (Song et al., 2017). Moreover, it is not known how to do Lewis weight sampling in a distributed setting. Alternatively, one could first apply k-CSS$_p$ on A_i to obtain factors U_i and V_i on each server, and then send the coordinator all the U_i and V_i. The coordinator then column-wise stacks the U_iV_i to obtain $U \cdot V$ and selects $\widetilde{O}(k)$ columns from $U \cdot V$. Even though this protocol applies to all $p \geq 1$, it achieves a loose $O(k^2)$ approximation to the optimal rank-k error and requires a prohibitive $O(n + d)$ communication cost[1]. One could instead try to just communicate the matrices U_i to the coordinator, which results in much less communication, but this no longer gives a good approximation. Indeed, while each U_i serves as a good approximation locally, there may be columns that are locally not important, but become globally important when all of the matrices A_i are put together. What is really needed here is a small *strong coreset* C_i for each A_i so that if one concatenates all of the C_i to obtain C, *any* good column subset of the coreset C corresponds to a good column subset for A.

1.6. Greedy k-CSS and Empirical Evaluations

We also propose an offline, greedy algorithm to select columns in the $\ell_{p,2}$ norm, $\forall p \in [1, 2)$ (see Section 6), which can be used as an alternative subroutine in both of our algorithms, and show the provable additive error guarantees for this algorithm. Similar error guarantees were known for the Frobenius norm (Altschuler et al., 2016), though nothing was known for the $\ell_{p,2}$ norm. We implement both of our

[1] We give this protocol and the analysis in the supplementary.

streaming and distributed algorithms in the ℓ_1 norm and experiment with real-world text document and genetic analysis applications. We compare the $O(1)$-approximate bi-criteria k-CSS$_{1,2}$ (denoted regular CSS$_{1,2}$) and the greedy k-CSS$_{1,2}$ as subroutines of our streaming and distributed algorithms, and show that greedy k-CSS$_{1,2}$ yields an improvement in practice. Furthermore, we compare our $O(1)$-approximate k-CSS$_{1,2}$ subroutine against one k-CSS$_2$ algorithm as active learning algorithms on a noisy image classification task to show that the ℓ_1 norm loss is indeed more robust for k-CSS to non-Gaussian noise.

Note that regular CSS$_{p,2}$ gives a stronger multiplicative $O(1)$ approximation to the optimal rank-k error, while greedy CSS$_{p,2}$ gives an additive approximation to the error from the best column subset. However, in practice, we observe that greedy CSS$_{1,2}$ gives lower approximation error than regular CSS$_{1,2}$ in the ℓ_1 norm, though it can require significantly longer running time than regular CSS$_{1,2}$. An additional advantage of greedy CSS$_{p,2}$ is that it is simpler and easier to implement.

1.7. Technical Novelty

We highlight several novel techniques and contributions:

- **Non-standard Net Argument:** To apply dimensionality reduction techniques via p-stable random variables to reduce the final approximation error, on the lower bound side, we need to show that p-stable random variables do not reduce the approximation error (i.e. no contraction), with respect to *all possible* column subsets of A, with high probability, in Lemma 2. While the net arguments are widely used, our proof of Lemma 2 is non-standard: we union bound over all possible subspaces defined on *subsets* of size $\widetilde{O}(k)$. This is more similar to the Restricted Isometry Property (RIP), which is novel in this context, but we only need a one-sided RIP since we only require no contraction; on the upper bound side, we just argue with constant probability the *single* optimal column subset and its corresponding right factor do not dilate much.
- ***Strong Coresets* for CSS:** Strong coresets for the $\ell_{p,2}$ norm have not been used for entrywise ℓ_p norm column subset selection, or *even for ℓ_p low rank approximation*, for $p \neq 2$. This is perhaps because strong coresets for subspace approximation, i.e., coresets that work for all query subspaces simultaneously, are not known to exist for sums of p-th powers of ℓ_p-distances for $p \neq 2$; our work provides a workaround to this. By switching to the Euclidean norm in a low-dimensional space we can use strong coresets for the $\ell_{p,2}$ norm with a small distortion.
- **Greedy $\ell_{p,2}$-norm CSS:** We give the first provable guarantees for greedy $\ell_{p,2}$-norm column subset selection. We show that the techniques used to derive guarantees for greedy CSS in the Frobenius norm from (Altschuler et al., 2016) can be extended to the $\ell_{p,2}$ norms, $\forall p \in [1, 2)$. A priori, it is not clear this should work, since for example, $\ell_{1,2}$ norm low rank approximation is NP-hard (Clarkson & Woodruff, 2015) while Frobenius norm low rank approximation can be solved in polynomial time.

2. Problem Setting

Definition 2.1 (Column-Update Streaming Model (Song et al., 2017))**.** Let $A_{*1}, A_{*2}, \cdots, A_{*n}$ be a set of columns from the input matrix $A \in \mathbb{R}^{d\times n}$. In the column-update model, each column of A will occur in the stream exactly once, but the columns can be in an arbitrary order. An algorithm in this model is only allowed a single pass over the columns. At the end of the stream, the algorithm stores some information about A. The space of the algorithm is the total number of words required to store this information during the stream. Here, each word is $O(\log nd)$ bits.

Definition 2.2 (Column Partition Distributed Model (Song et al., 2017))**.** There are s servers, the i-th of which holds matrix $A_i \in \mathbb{R}^{d\times n_i}$ as the input. Suppose $n = \sum_{i=1}^{s} n_i$, and the global data matrix is denoted by $A = [A_1, A_2, \ldots, A_s]$. A is column-partitioned and distributed across s machines. Furthermore, there is a coordinator. The model only allows communication between the servers and the coordinator. The communication cost in this model is the total number of words transferred between machines and the coordinator. Each word is $O(\log snd)$ bits.

Definition 2.3 ($\ell_{p,2}$ norm)**.** For matrix $A \in \mathbb{R}^{d\times n}$, $\|A\|_{p,2} = (\sum_{j=1}^{n} \|A_{*j}\|_2^p)^{1/p}$, where A_{*j} denotes the j-th column.

Definition 2.4 (p-Stable Distribution and Random Variables)**.** Let $X_1, \ldots, X_d$ be random variables drawn i.i.d. from some distribution $\mathcal{D}$. $\mathcal{D}$ is called p-stable if for an arbitrary vector $v \in \mathbb{R}^d$, $\langle v, X\rangle = \|v\|_p Z$ for some Z drawn from $\mathcal{D}$, where $X = [X_1, \ldots, X_d]^T$. $\mathcal{D}$ is called a p-stable distribution — these exist for $p \in (0, 2]$. Though there is no closed form expression for the p-stable distribution in general except for a few values of p, we can efficiently generate a single p-stable random variable in $O(1)$ time using the following method due to (Chambers et al., 1976): if $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ and $r \in [0, 1]$ are sampled uniformly at random, then, $\frac{\sin(p\theta)}{\cos^{1/p}\theta}(\frac{\cos(\theta(1-p))}{\ln(\frac{1}{r})})^{\frac{1-p}{p}}$ follows a p-stable distribution.

3. Preliminaries

In this section, we introduce the dimensionality reduction techniques we make use of: strong coresets for $\ell_{p,2}$-norm

low-rank approximation and oblivious sketching using p-stable random matrices. We begin with a standard relationship between the ℓ_p norm and the $\ell_{p,2}$ norm:

Lemma 1. *For a matrix $A \in \mathbb{R}^{d\times n}$ and $p \in [1,2)$, $\|A\|_{p,2} \le \|A\|_p \le d^{\frac{1}{p}-\frac{1}{2}}\|A\|_{p,2}$.*

3.1. Dimensionality Reduction in the ℓ_p norm

To reduce the row dimension d, we left-multiply A by an oblivious sketching matrix S with i.i.d. p-stable entries so that our approximation error only increases by an $\widetilde{O}(k^{\frac{1}{p}-\frac{1}{2}})$ factor instead of an $O(d^{\frac{1}{p}-\frac{1}{2}})$ factor when we switch to the $\ell_{p,2}$ norm. The following lemma shows that for all column subsets A_T and right factors V, the approximation error when using these to fit A does not shrink after multiplying by S (i.e., this holds simultaneously for all A_T and V):

Lemma 2 (Sketched Error Lower Bound)**.** *Let $A \in \mathbb{R}^{d\times n}$ and $k \in \mathbb{N}$. Let $t = k \cdot poly(\log(nd))$, and let $S \in \mathbb{R}^{t\times d}$ be a matrix whose entries are i.i.d. standard p-stable random variables, rescaled by $\Theta(1/t^{\frac{1}{p}})$. Then, with probability $1-o(1)$, for **all** $T \subset [n]$ with $|T| = k \cdot poly(\log k)$ and for all $V \in \mathbb{R}^{|T|\times n}$,*

$$\|A_T V - A\|_p \le \|SA_T V - SA\|_p$$

We also recall the following upper bound for oblivious sketching from (Song et al., 2017) for a *fixed* subset of columns A_T and a *fixed* V.

Lemma 3 (Sketched Error Upper Bound (Lemma E.11 of (Song et al., 2017)))**.** *Let $A \in \mathbb{R}^{d\times n}$ and $k \in \mathbb{N}$. Let $t = k \cdot poly(\log(nd))$, and let $S \in \mathbb{R}^{t\times d}$ be a matrix whose entries are i.i.d. standard p-stable random variables, rescaled by $\Theta(1/t^{\frac{1}{p}})$. Then, for a fixed subset $T \subset [n]$ of columns with $|T| = k \cdot poly(\log k)$ and a fixed $V \in \mathbb{R}^{|T|\times n}$, with probability $1-o(1)$, we have*

$$\min_V \|SA_T V - SA\|_p \le \min_V O(\log^{1/p}(nd))\|A_T V - A\|_p$$

3.2. Strong Coresets in the $\ell_{p,2}$ Norm

As mentioned above, strong coresets for $\ell_{p,2}$-norm low-rank approximation are reweighted column subsets which preserve the $\ell_{p,2}$-norm approximation error incurred by any rank-k projection. Our construction of strong coresets follows (Sohler & Woodruff, 2018), which is based on Lewis weights (Cohen & Peng, 2015) sampling. Note that (Sohler & Woodruff, 2018) only states such strong coresets hold with constant probability. But in our applications, we need to union bound over multiple constructions of strong coresets, so need a lower failure probability. The only reason the coresets of (Sohler & Woodruff, 2018) hold only with constant probability is because they rely on the sampling result of (Cohen & Peng, 2015), which is stated for constant probability. However, the results of (Cohen & Peng, 2015) are a somewhat arbitrary instantiation of the failure probabilities of the earlier ℓ_p-Lewis weights sampling results in (Bourgain et al., 1989) in the functional analysis literature. That work performs ℓ_p-Lewis weight sampling and we show how to obtain failure probability δ with a $\log(1/\delta)$ dependence in Section B.1 of the supplementary material.

Lemma 4 (Strong Coresets in $\ell_{p,2}$ norm (Sohler & Woodruff, 2018))**.** *Let $A \in \mathbb{R}^{d\times n}$, $k \in \mathbb{N}$, $p \in [1,2)$, and $\varepsilon, \delta \in (0,1)$. Then, in $\widetilde{O}(nd)$ time, one can find a sampling and reweighting matrix T with $O(\frac{d}{\varepsilon^2} poly(\log(d/\varepsilon), \log(1/\delta)))$ columns, such that, with probability $1-\delta$, for all rank-k matrices U,*

$$\min_{\text{rank-}k\ V} \|UV - AT\|_{p,2} = (1\pm\varepsilon) \min_{\text{rank-}k\ V} \|UV - A\|_{p,2}$$

*where AT is called a **strong coreset** of A.*

3.3. $O(1)$-approximate Bi-criteria k-CSS$_{p,2}$

We introduce an $O(1)$-approximation bi-criteria algorithm, which is a modification of the algorithm from (Clarkson & Woodruff, 2015). The number of output columns is $\widetilde{O}(k)$ instead of $O(k^2)$ since we use ℓ_p Lewis weights instead of ℓ_p leverage scores. Details are in the supplementary material.

Theorem 1 (Bicriteria $O(1)$-Approximation Algorithm for k-CSS$_{p,2}$)**.** *Let $A \in \mathbb{R}^{d\times n}$ and $k \in \mathbb{N}$. There is an algorithm with $(nnz(A)+d^2)\cdot k \cdot poly(\log k)$ runtime that outputs a rescaled subset of columns $U \in \mathbb{R}^{d\times\widetilde{O}(k)}$ of A and a right factor $V \in \mathbb{R}^{\widetilde{O}(k)\times n}$ for which $V = \min_V \|UV - A\|_{p,2}$, such that with probability $1-o(1)$,*

$$\|UV - A\|_{p,2} \le O(1) \cdot \min_{\text{rank-}k\ A_k} \|A_k - A\|_{p,2}$$

4. A Streaming Algorithm for k-CSS$_p$

Our one-pass streaming algorithm (Algorithm 1) is based on the Merge-and-Reduce framework. The n columns of the input matrix A are partitioned into $\lceil n/r\rceil$ batches of length $r = k \cdot \text{poly}(\log nd)$. See the supplementary material for an illustration. These $\lceil n/r\rceil$ batches can be viewed as the leaves of a binary tree and are considered to be at level 0 of the tree. A merge operation (Algorithm 2) is used as a subroutine of Algorithm 1 — it computes a strong coreset of two sets of columns corresponding to the two children nodes. Each node in the binary tree represents a set of columns. Starting from level 0, every pair of neighboring batches of columns will be merged, until there is only one coreset of columns left at the root, i.e. level $\log(n/k)$. During the stream, the nodes are greedily merged. The streaming algorithm constructs strong coresets and merges the sketched columns SA_{*j} (list C), while keeping a list

Algorithm 1 A one-pass streaming algorithm for bi-criteria k-CSS$_p$ in the column-update streaming model.

Input: A matrix $A \in \mathbb{R}^{d\times n}$ whose columns arrive one at a time, $p \in [1,2)$, rank $k \in \mathbb{N}$ and batch size r.
Output: A subset of $\widetilde{O}(k)$ columns A_I.
Generate a dense p-stable sketching matrix $S \in \mathbb{R}^{k\text{poly}(\log(nd))\times d}$.
A list of strong coresets and the level number $C \leftarrow \{\}$.
A list of columns corresponding to the list of strong coresets and the level number $D \leftarrow \{\}$.
A list of sketched columns $M \leftarrow \{\}$.
A list of columns $L \leftarrow \{\}$.
for Each column A_{*j} seen in the data stream **do**
  $M \leftarrow M \cup SA_{*j}$
  $L \leftarrow L \cup A_{*j}$
  if length of M == r **then**
    $C \leftarrow C \cup (M, 0), D \leftarrow D \cup L$
    $C, D \leftarrow$ **Recursive Merge**(C, D) {// Algorithm 2}
    $M \leftarrow \{\}, L \leftarrow \{\}$
  end if
end for
$C \leftarrow C \cup (M, 0), D \leftarrow D \cup L$
$C, D \leftarrow$ **Recursive Merge**(C, D) {// Algorithm 2}
Apply k-CSS$_{p,2}$ on the single strong coreset left in C to obtain the indices I of the subset of selected columns with size $O(k \times \text{poly}(\log k))$. Recover the original columns of A by mapping indices I to columns in D to get the subset of columns A_I.

of corresponding columns A_{*j} (list D) at the same time, in order to recover the original columns of A as the final output.

Theorem 2 (A One-pass Streaming Algorithm for k-CSS$_p$)**.** *In the column-update streaming model, let $A \in \mathbb{R}^{d\times n}$ be the data matrix whose columns arrive one at each time in a data stream. Given $p \in [1,2)$ and a desired rank $k \in \mathbb{N}$, Algorithm 1 outputs a subset of columns $A_I \in \mathbb{R}^{d\times k\text{poly}(\log(k))}$ in $\widetilde{O}(nnz(A)k + nk + k^3)$ time, such that with probability $1 - o(1)$,*

$$\min_V \|A_I V - A\|_p \le \widetilde{O}(k^{1/p-1/2}) \min_{L\subset[n],|L|=k} \|A_L V - A\|_p$$

Moreover, Algorithm 1 only needs to process all columns of A once and uses $\widetilde{O}(dk)$ space throughout the stream.

Proof. We give a brief sketch of the proof (the full proof is in the supplementary). We first need **Lemma** 5 below to show how the approximation error propagates through each level induced by the merge operator. It gives the approximation error of a strong coreset C_0 computed at level l with respect to the union of all sets of columns represented as the leaves of the subtree rooted at C_0.

Algorithm 2 Recursive Merge

Input: A list C of strong coresets and their corresponding level numbers. A list D of (unsketched) columns of A corresponding to the sketched columns in C.
Output: New C, where the list of strong coresets is greedily merged, and the corresponding new D.
if length of C == 1 **then**
  Return C, D.
else
  Let $(C_{-2}, l_{-2}), (C_{-1}, l_{-1})$ be the second to last and last sets of columns C_{-2}, C_{-1} with their corresponding level l_{-2}, l_{-1} from list C.
  if l_{-2} == l_{-1} **then**
    Remove $(C_{-2}, l_{-2}), (C_{-1}, l_{-1})$ from C.
    Remove the corresponding D_{-2}, D_{-1} from D.
    Compute a strong coreset C_0 of (i.e., select columns from) $C_{-2} \cup C_{-1}$. Record the indices I of the columns selected in C_0.
    Map indices I to columns in $D_{-2} \cup D_{-1}$ to form a new subset of columns D_0.
    $C \leftarrow C \cup (C_0, l_{-1} + 1), D \leftarrow D \cup D_0$.
    Recursive Merge(C, D).
  else
    Return C, D.
  end if
end if

Lemma 5 (Approximation Error from Merging)**.** *Let C_0 be the strong coreset of size $\widetilde{O}(k)$ (See Lemma 4) at level l constructed from a union of its two children $C_{-1} \cup C_{-2}$, with $\frac{k}{\gamma^2} \cdot poly(\log(nd/\gamma))$ columns, where $\gamma \in (0,1)$. Then with probability $1 - \frac{1}{n^2}$, for all rank-k matrices U,*

$$\min_V \|UV - C_0\|_{p,2} = (1\pm\gamma) \min_V \|UV - (C_{-1}\cup C_{-2})\|_{p,2}$$

Let M be the union of all sets of sketched columns represented as the leaves of the subtree rooted at C_0 (and assume the subtree has size q). Then with probability $1 - \frac{q}{n^2}$, for all rank-k matrices U,

$$\min_V \|UV - C_0\|_{p,2} = (1 \pm \gamma)^l \min_V \|UV - M\|_{p,2}$$

If at the leaves (level 0), we construct $(1 \pm \frac{\varepsilon}{\log n})$-approximate coresets for the input columns from the stream, the final single coreset left at the root level (level $\log(n/k)$) will be a $(1 \pm \varepsilon)$-approximate coreset for *all* columns of SA. The k-CSS$_{p,2}$ algorithm that selects $O(k\text{poly}(\log k))$ columns from this coreset gives an $O(1)$-approximation by Theorem 1. By Lemmas 1, 2 and 3, the final approximation error of this algorithm is dominated by the one from the relaxation to the $\ell_{p,2}$ norm, which leads to an overall $\widetilde{O}(k^{1/p-1/2})$ approximation factor. Note that the space com-

plexity is $\widetilde{O}(dk)$ since each coreset has $\widetilde{O}(k)$ columns and we only keep coresets for at most $O(\log n)$ of the nodes of the tree, at a single time. The running time is dominated by the $O(n/k)$ merging operators throughout the stream and the k-CSS$_{p,2}$ algorithm. A detailed analysis is in the supplementary. □

5. A Distributed Protocol for k-CSS$_p$

Theorem 3 (A One-round Protocol for Distributed k-CSS$_p$)**.** *In the column partition model, let $A \in \mathbb{R}^{d\times n}$ be the data matrix whose columns are partitioned across s servers and suppose server i holds a subset of columns $A_i \in \mathbb{R}^{d\times n_i}$, where $n = \sum_{i\in[s]} n_i$. Then, given $p \in [1,2)$ and a desired rank $k \in \mathbb{N}$, Algorithm 3 outputs a subset of columns $A_I \in \mathbb{R}^{d\times k\,\mathrm{poly}(\log(k))}$ in $\widetilde{O}(nnz(A)k + kd + k^3)$ time, such that with probability $1 - o(1)$,*

$$\min_V \|A_I V - A\|_p \le \widetilde{O}(k^{1/p-1/2}) \min_{L\subset[n],|L|=k} \|A_L V - A\|_p$$

Moreover, Algorithm 3 uses one round of communication and $\widetilde{O}(sdk)$ words of communication.

The analysis of the protocol is similar to the analysis of our streaming algorithm (Section 4). We give a detailed analysis in the supplementary.

6. Greedy k-CSS$_{p,2}$

We propose a greedy algorithm for k-CSS$_{p,2}$ (Algorithm 4). In each iteration, the algorithm samples a subset A_C of $O(\frac{n}{k}\log(\frac{1}{\delta}))$ columns from the input matrix $A \in \mathbb{R}^{d\times n}$ and picks the column among A_C that reduces the approximation error the most. We give the first provable guarantees for this algorithm below, the proof of which is in the supplementary. [2] We also empirically compare this algorithm to the k-CSS$_{p,2}$ algorithm mentioned above, in Section 7.

Theorem 4 (Greedy k-CSS$_{1,2}$)**.** *Let $p \in [1,2)$. Let $A \in \mathbb{R}^{d\times n}$ be the data matrix and $k \in \mathbb{N}$ be the desired rank. Let A_L be the best possible subset of k columns, i.e., $A_L = \mathrm{argmin}_{A_L} \min_V \|A_L V - A\|_{p,2}$. Let σ be the minimum non-zero singular value of the matrix B of normalized columns of A_L, (i.e., the j-th column of B is $B_{*j} = (A_L)_{*j}/\|(A_L)_{*j}\|_2$). Let $T \subset [n]$ be the subset of output column indices selected by **Algorithm 4**, for $\epsilon, \delta \in (0,1)$, for $|T| = \Omega(\frac{k}{p\sigma^2\epsilon^2})$, with probability $1-\delta$,*

$$\mathbb{E}[\min_V \|A_T V - A\|_{p,2}] \le \min_V \|A_L V - A\|_{p,2} + \epsilon\|A\|_{p,2}$$

The overall running time is $O(\frac{n}{p\sigma^2\epsilon^2}\log(\frac{1}{\delta}) \cdot (\frac{dk^2}{p^2\sigma^4\epsilon^4} + \frac{ndk}{p\sigma^2\epsilon^2}))$.

[2] The analysis is based on the analysis by (Altschuler et al., 2016) of greedy k-CSS$_2$.

Algorithm 3 A one-round protocol for bi-criteria k-CSS$_p$ in the column partition model

Initial State:
Server i holds matrix $A_i \in \mathbb{R}^{d\times n_i}$, $\forall i \in [s]$.
Coordinator:
Generate a dense p-stable sketching matrix $S \in \mathbb{R}^{k\,\mathrm{poly}(\log(nd))\times d}$.
Send S to all servers.
Server i:
Compute SA_i.
Let the number of samples in the coreset be $t = O(k \cdot \mathrm{poly}(\log(nd)))$. Construct a coreset of SA_i under the $\ell_{p,2}$ norm by applying a sampling matrix D_i of size $n_i \times t$ and a diagonal reweighting matrix W_i of size $t \times t$.
Let $T_i = D_i W_i$. Send SA_iT_i along with A_iD_i to the coordinator.
Coordinator:
Column-wise stack SA_iT_i to obtain $SAT = [SA_1T_1, SA_2T_2, \ldots, SA_sT_s]$.
Apply k-CSS$_{p,2}$ on SAT to obtain the indices I of the subset of selected columns with size $O(k \cdot \mathrm{poly}(\log k))$.
Since D_i's are sampling matrices, the coordinator can recover the original columns of A by mapping indices I to A_iD_i's.
Denote the final selected subset of columns by A_I. Send A_I to all servers.
Server i:
Solve $\min_{V_i} \|A_I V_i - A_i\|_p$ to obtain the right factor V_i.
A_I and V will be factors of a rank-$k \cdot \mathrm{poly}(\log k)$ factorization of A, where V is the (implicit) column-wise concatenation of the V_i.

7. Experiments[3]

7.1. Streaming and Distributed k-CSS$_1$

7.1 Streaming and Distributed k-CSS$_1$

We implement both of our streaming and distributed k-CSS$_1$ algorithms, with subroutines **regular** k-CSS$_{1,2}$ (Section 3.3) and **greedy** k-CSS$_{1,2}$ (Section 6). Given a target rank k, we set the number of output columns to be k. We compare against a commonly used baseline for low-rank approximation (Song et al., 2019a; Chierichetti et al., 2017), **SVD** (rank-k singular value decomposition), and a **uniform** random baseline. In the streaming setting, the **uniform** baseline first collects k columns from the data stream and on each of the following input columns, it decides whether to keep or discard the new column with equal probability. If it keeps the new columns, it will pick one existing column to replace uniformly at random. In the distributed setting, the **uniform** baseline simply uniformly at random selects k columns.

[3] The source code is available at: https://github.com/11hifish/robust_css.

Algorithm 4 Greedy k-CSS$_{p,2}$.

Input: The data matrix $A \in \mathbb{R}^{d\times n}$. A desired rank $k \in \mathbb{N}$ and $p \in [1, 2)$. The number of columns to be selected $r \leq n$. Failure probability $\delta \in (0, 1)$.
Output: A subset of r columns A_T.
Indices of selected columns $T \leftarrow \{\}$.
for $i = 1$ to r **do**
  $C \leftarrow$ Sample $\frac{n}{k}\log(\frac{1}{\delta})$ indices from $\{1, 2, \ldots, n\} \setminus T$ uniformly at random.
  Column index $j^* \leftarrow \text{argmin}_{j\in C}(\min_V \|A_{T\cup j}V - A\|_{p,2})$
  $T \leftarrow T \cup j^*$.
end for
Map indices T to get the selected columns A_T.

We apply the proposed algorithms on one synthetic data to show when SVD (and hence all the existing k-CSS$_2$ algorithms) fails to find a good subset of columns in the ℓ_1 norm. We also apply the proposed algorithms to two real-world applications, where k-CSS$_2$ was previously used to analyze the most representative set of words among a text corpus or the most informative genes from genetic sequences, e.g. (Mahoney & Drineas, 2009; Boutsidis et al., 2008a).

Datasets. 1) `Synthetic` has a matrix A of size $(k+n)\times(k+n)$ with rank k and a fixed number n, where the top left $k \times k$ submatrix is the identity matrix multiplied by $n^{\frac{3}{2}}$, and the bottom right $n \times n$ submatrix has all 1's. The optimal k columns consist of one of the last n columns along with $k-1$ of the first k columns, incurring an error of $n^{\frac{3}{2}}$ in the ℓ_1 norm. SVD, however, will not cover any of the last n columns, and thus will get an ℓ_1 error of n^2. We set $n = 1000$ in the experiments. 2) `TechTC`[4] contains 139 documents processed in a bag-of-words representation with a dictionary of 18446 words, which naturally results in a sparse matrix. 3) `Gene`[5] contains 5000 different RNA-Seq gene expressions from 400 cancer patients, which gives a dense data matrix with $\geq 85\%$ non-zero entries.

Setup. For an input data matrix $A \in \mathbb{R}^{d\times n}$, we set the number of rows of our 1-stable (Cauchy) sketching matrix to be $0.5d$ in both settings. In the streaming setting, we set the batch size to be $5k$ and maintain a list of coresets of size $2k$. In the distributed setting, we set the number of servers to be 5 and each server sends the coordinator a coreset of size $2k$. For each of our experimetnts, we conduct 10 random runs and report the mean ℓ_1 error ratio $\frac{\min_V \|A_IV-A\|_1}{\|A\|_1}$ and the mean time (in seconds) to obtain A_I along with one standard deviation, where A_I is the output set of columns. Note that the input columns of the data matrix are randomly permuted in the streaming setting for each run.

Implementation Details. Our algorithms are implemented with Python Ray[6], a high-level framework for parallel and distributed computing, and are thus highly scalable. All the experiments are conducted on AWS EC2 c5a.8xlarge machines with 32 vCPUs and 64GB EBS memory.

Results. The results for the streaming setting and the distributed setting are presented in Figure 1 and Figure 2 respectively. We note that **SVD** works in neither streaming nor distributed settings and thus the running time of **SVD** is not directly comparable to the other algorithms. The performance of **SVD** and **uniform** is highly dependent on the actual data and does not have worst case guarantees, while the performance of our algorithm is stable and gives relatively low ℓ_1 error ratio across different datasets. **greedy** k-CSS$_{1,2}$ gives lower error ratio compared to **regular** k-CSS$_{1,2}$ as one would expect, but the time it takes significantly increases as the number of output columns increases, especially in the distributed setting, while **regular** k-CSS$_{1,2}$ takes comparable time to the **uniform** random baseline in most settings and is thus more scalable.

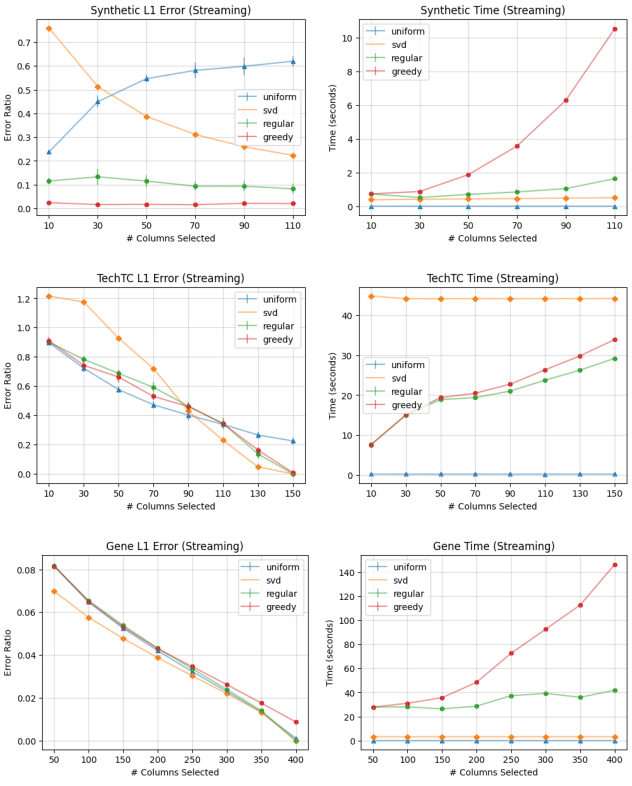

Figure 1. Streaming results.

[4] http://gabrilovich.com/resources/data/techtc/techtc300/techtc300.html

[5] https://archive.ics.uci.edu/ml/datasets/gene+expression+cancer+RNA-Seq

[6] https://docs.ray.io/en/master/

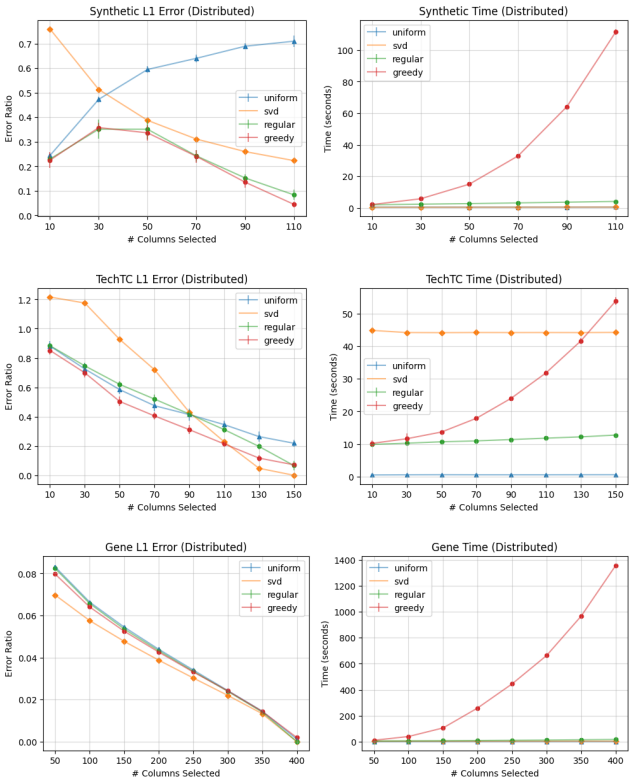

Figure 2. Distributed results.

7.2. Robustness of the ℓ_1 norm

7.2 Robustness of the ℓ_1 norm

We further highlight the robustness of the ℓ_1 norm loss to non-Gaussian noise compared to the Frobenius norm loss for k-CSS on an image classification task. k-CSS was previously used as an active learning algorithm to select the most representative samples to acquire training labels in supervised learning, when acquiring such labels is expensive (Shen et al., 2011; Kaushal et al., 2018).

We apply one Frobenius norm k-CSS$_2$ algorithm (Boutsidis et al., 2008a), our regular k-CSS$_{1,2}$ algorithm (Section 3.3) and a random baseline to select a subset of k training samples to train a linear regression model to classify images from the `COIL20`[7] dataset. `COIL20` contains a total of 1440 images. Each image has 32×32 pixels with 256 gray levels per pixel. We randomly split the dataset into 80% training set (1152 samples) and 20% testing set (288 samples). We then mask 40% of the pixels of each training sample with a uniformly random noise value in $[0, 256)$. We conduct 20 random runs and report the average Mean Squared Error (MSE) and one standard deviation for each algorithm on the testing set.

[7] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php

Algorithm / k samples	Random	k-CSS$_2$	k-CSS$_{1,2}$
200	16.29 ± 2.94	17.64 ± 2.97	$\mathbf{15.99 \pm 2.55}$
300	15.94 ± 2.17	17.92 ± 3.59	$\mathbf{15.41 \pm 2.44}$
400	14.27 ± 1.68	16.49 ± 4.04	$\mathbf{13.84 \pm 1.55}$
500	14.03 ± 1.33	14.59 ± 1.85	$\mathbf{13.53 \pm 1.16}$
600	14.45 ± 1.91	15.56 ± 3.19	$\mathbf{13.68 \pm 1.40}$

Table 1. Image classification results: average MSE and one std.

The results are summarized in Table 1. k-CSS$_{1,2}$ gives a slightly lower average MSE score and a lower variance, compared to k-CSS$_2$ and the random baseline on noisy training data. This suggests that the ℓ_1 norm loss is more robust compared to the Frobenius norm loss for k-CSS, which agrees with previous observations from other algorithms and applications.

8. Conclusion

In this work, we give the first one-pass streaming algorithm for k-CSS$_p$ ($1 \leq p < 2$) in the column-update model and the first one-round distributed protocol in the column-partition model. Both of our algorithms achieve $\widetilde{O}(k^{1/p-1/2})$-approximation to the optimal column subset. The streaming algorithm uses nearly optimal space complexity of $\widetilde{O}(kd)$, and the distributed protocol uses nearly optimal $\widetilde{O}(sdk)$ communication cost. We introduce novel analysis techniques for k-CSS. To achieve a good approximation factor, we use dense p-stable sketching and work with the $\ell_{p,2}$ norm, which enables us to use an efficient construction of strong coresets and an $O(1)$-approximation bi-criteria k-CSS$_{p,2}$ algorithm as a subroutine of our algorithms. We further propose a greedy alternative for k-CSS$_{p,2}$ and show the first additive error upper bound. Our experimental results confirm that our algorithms give stable low ℓ_1 error in both distributed and streaming settings. We further demonstrate the robustness of the ℓ_1 norm loss for k-CSS.

Acknowledgements

D. Woodruff would like to thank partial support from NSF grant No. CCF-1815840, Office of Naval Research grant N00014-18-1-256, and a Simons Investigator Award. A. Mahankali would like to thank partial support from the SURF award from CMU's Undergraduate Research Office.

References

Altschuler, J., Bhaskara, A., Fu, G., Mirrokni, V., Rostamizadeh, A., and Zadimoghaddam, M. Greedy column subset selection: New bounds and distributed algorithms. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 2539–2548. JMLR.org, 2016.

Balcan, M., Liang, Y., Song, L., Woodruff, D. P., and Xie, B.

Distributed kernel principal component analysis. *CoRR*, abs/1503.06858, 2015.

Balcan, M., Liang, Y., Song, L., Woodruff, D. P., and Xie, B. Communication efficient distributed kernel principal component analysis. In Krishnapuram, B., Shah, M., Smola, A. J., Aggarwal, C. C., Shen, D., and Rastogi, R. (eds.), *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 725–734. ACM, 2016.

Ban, F., Bhattiprolu, V., Bringmann, K., Kolev, P., Lee, E., and Woodruff, D. P. A PTAS for ℓ_p-low rank approximation. In Chan, T. M. (ed.), *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pp. 747–766. SIAM, 2019.

Bourgain, J., Lindenstrauss, J., and Milman, V. Approximation of zonoids by zonotopes. *Acta Mathematica*, 162 (none):73 – 141, 1989.

Boutsidis, C. and Woodruff, D. P. Optimal cur matrix decompositions. *SIAM Journal on Computing*, 46(2):543–589, 2017.

Boutsidis, C., Mahoney, M. W., and Drineas, P. Unsupervised feature selection for principal components analysis. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pp. 61–69, New York, NY, USA, 2008a. Association for Computing Machinery.

Boutsidis, C., Mahoney, M. W., and Drineas, P. An improved approximation algorithm for the column subset selection problem. *CoRR*, abs/0812.4293, 2008b. URL http://arxiv.org/abs/0812.4293.

Boutsidis, C., Drineas, P., and Magdon-Ismail, M. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

Boutsidis, C., Woodruff, D. P., and Zhong, P. Optimal principal component analysis in distributed and streaming models. In Wichs, D. and Mansour, Y. (eds.), *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pp. 236–249. ACM, 2016.

Chambers, J. M., Mallows, C. L., and Stuck, B. W. A method for simulating stable random variables. *Journal of the American Statistical Association*, 71(354):340–344, 1976.

Chierichetti, F., Gollapudi, S., Kumar, R., Lattanzi, S., Panigrahy, R., and Woodruff, D. P. Algorithms for ℓ_p Low-Rank Approximation. 2017.

Clarkson, K. L. and Woodruff, D. P. Numerical linear algebra in the streaming model. In Mitzenmacher, M. (ed.), *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pp. 205–214. ACM, 2009.

Clarkson, K. L. and Woodruff, D. P. Low rank approximation and regression in input sparsity time. In Boneh, D., Roughgarden, T., and Feigenbaum, J. (eds.), *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pp. 81–90. ACM, 2013.

Clarkson, K. L. and Woodruff, D. P. Input sparsity and hardness for robust subspace approximation. In *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, FOCS '15, pp. 310–329, USA, 2015. IEEE Computer Society. ISBN 9781467381918. doi: 10.1109/FOCS.2015.27.

Cohen, M. B. and Peng, R. Lp row sampling by lewis weights. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pp. 183–192, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450335362. doi: 10.1145/2746539.2746567.

Dan, C., Wang, H., Zhang, H., Zhou, Y., and Ravikumar, P. K. Optimal analysis of subset-selection based l_p low-rank approximation. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 2541–2552. Curran Associates, Inc., 2019.

Drineas, P. and Kannan, R. Pass efficient algorithms for approximating large matrices. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA*, pp. 223–232. ACM/SIAM, 2003.

Farahat, A. K., Elgohary, A., Ghodsi, A., and Kamel, M. S. Distributed column subset selection on mapreduce. In Xiong, H., Karypis, G., Thuraisingham, B. M., Cook, D. J., and Wu, X. (eds.), *2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013*, pp. 171–180. IEEE Computer Society, 2013.

Ghashami, M. and Phillips, J. M. Relative errors for deterministic low-rank matrix approximations. In Chekuri, C. (ed.), *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pp. 707–717. SIAM.

Gillis, N. and Vavasis, S. A. On the complexity of robust PCA and ℓ_1-norm low-rank matrix approximation. *CoRR*, abs/1509.09236, 2015.

Guruswami, V. and Sinop, A. K. Optimal column-based

low-rank matrix reconstruction. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pp. 1207–1214. SIAM, 2012.

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

Kaushal, V., Sahoo, A., Doctor, K., Raju, N., Shetty, S., Singh, P., Iyer, R., and Ramakrishnan, G. Learning from less data: Diversified subset selection and active learning in image classification tasks, 2018.

Ke, Q. and Kanade, T. Robust l_1 norm factorization in the presence of outliers and missing data by alternative convex programming. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pp. 739–746. IEEE Computer Society, 2005.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Liang, Y., Balcan, M., Kanchanapally, V., and Woodruff, D. P. Improved distributed principal component analysis. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 3113–3121, 2014.

Liberty, E. Simple and deterministic matrix sketching. In Dhillon, I. S., Koren, Y., Ghani, R., Senator, T. E., Bradley, P., Parekh, R., He, J., Grossman, R. L., and Uthurusamy, R. (eds.), *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, pp. 581–588. ACM, 2013.

Mahankali, A. V. and Woodruff, D. P. Optimal ℓ_1 column subset selection and a fast ptas for low rank approximation. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 560–578. SIAM, 2021.

Mahoney, M. and Drineas, P. Cur matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences of the United States of America*, 106: 697–702, 02 2009.

McGregor, A. Graph stream algorithms: A survey. *SIGMOD Rec.*, 43(1):9–20, May 2014. ISSN 0163-5808.

Shen, J., Ju, B., Jiang, T., Ren, J., Zheng, M., Yao, C., and Li, L. Column subset selection for active learning in image classification. *Neurocomputing*, 74:3785–3792, 11 2011.

Sohler, C. and Woodruff, D. P. Strong coresets for k-median and subspace approximation: Goodbye dimension. In Thorup, M. (ed.), *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pp. 802–813. IEEE Computer Society, 2018.

Song, Z., Woodruff, D. P., and Zhong, P. Low rank approximation with entrywise l1-norm error. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pp. 688–701, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450345286. doi: 10.1145/3055399.3055431.

Song, Z., Woodruff, D. P., and Zhong, P. Average case column subset selection for entrywise ℓ_1-norm loss. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 10111–10121, 2019a.

Song, Z., Woodruff, D. P., and Zhong, P. Towards a zero-one law for column subset selection. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 6120–6131, 2019b.

Wang, R. and Woodruff, D. P. Tight bounds for ℓ_p oblivious subspace embeddings. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pp. 1825–1843, 2019.

Woodruff, D. P. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014a.

Woodruff, D. P. Low rank approximation lower bounds in row-update streams. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 1781–1789, 2014b.

Woodruff, D. P. Sketching as a tool for numerical linear algebra. *Found. Trends Theor. Comput. Sci.*, 10(1-2): 1–157, 2014c.

Yu, L., Zhang, M., and Ding, C. H. Q. An efficient algorithm for l1-norm principal component analysis. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2012, Kyoto, Japan, March 25-30, 2012*, pp. 1377–1380. IEEE, 2012.