Non-Adaptive Adaptive Sampling on Turnstile Streams

Sepideh Mahabadi* Ilya Razenshteyn† David P. Woodruff‡ Samson Zhou§

April 24, 2020

Abstract

Adaptive sampling is a useful algorithmic tool for data summarization problems in the classical centralized setting, where the entire dataset is available to the single processor performing the computation. Adaptive sampling repeatedly selects rows of an underlying matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$, where $n \gg d$, with probabilities proportional to their distances to the subspace of the previously selected rows. Intuitively, adaptive sampling seems to be limited to trivial multi-pass algorithms in the streaming model of computation due to its inherently sequential nature of assigning sampling probabilities to each row only after the previous iteration is completed. Surprisingly, we show this is not the case by giving the first one-pass algorithms for adaptive sampling on turnstile streams and using space $\text{poly}(d, k, \log n)$, where k is the number of adaptive sampling rounds to be performed.

Our adaptive sampling procedure has a number of applications to various data summarization problems that either improve state-of-the-art or have only been previously studied in the more relaxed row-arrival model. We give the first relative-error algorithm for column subset selection on turnstile streams. We show our adaptive sampling algorithm also gives the first relative-error algorithm for subspace approximation on turnstile streams that returns k noisy rows of $\mathbf{A}$. The quality of the output can be improved to a $(1+\epsilon)$-approximation at the tradeoff of a bicriteria algorithm that outputs a larger number of rows. We then give the first algorithm for projective clustering on turnstile streams that uses space sublinear in n. In fact, we use space $\text{poly}\left(d, k, s, \frac{1}{\epsilon}, \log n\right)$ to output a $(1+\epsilon)$-approximation, where s is the number of k-dimensional subspaces. Our adaptive sampling primitive also provides the first algorithm for volume maximization on turnstile streams. We complement our volume maximization algorithmic results with lower bounds that are tight up to lower order terms, even for multi-pass algorithms. By a similar construction, we also obtain lower bounds for volume maximization in the row-arrival model, which we match with competitive upper bounds.

*Toyota Technological Institute at Chicago (TTIC). E-mail: mahabadi@ttic.edu
†Microsoft Research. E-mail: ilyaraz@microsoft.com
‡School of Computer Science, Carnegie Mellon University. E-mail: dwoodruf@cs.cmu.edu
§School of Computer Science, Carnegie Mellon University. E-mail: samsonzhou@gmail.com

1 Introduction

Data summarization is a fundamental task in data mining, machine learning, statistics, and applied mathematics. The goal is to find a set S of k rows of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that optimizes some predetermined function that quantifies how well S represents $\mathbf{A}$. For example, row subset selection seeks S to well-approximate $\mathbf{A}$ with respect to the spectral or Frobenius norm, subspace approximation asks to minimize the sum of the distances of the rows of $\mathbf{A}$ from S, while volume maximization wants to maximize the volume of the parallelepiped spanned by the rows of S. Due to their applications in data science, data summarization problems are particularly attractive to study for big data models.

The streaming model of computation is an increasingly popular model for describing large datasets whose overwhelming size places restrictions on space available to algorithms. For turnstile streams that implicitly define $\mathbf{A}$, the matrix initially starts as the all zeros matrix and receives a large number of updates to its coordinates. Once the updates are processed, they cannot be accessed again and hence any information not stored is lost forever. The goal is then to perform some data summarization task after the stream is completed without storing $\mathbf{A}$ in its entirety.

Adaptive sampling is a useful algorithmic paradigm that yields many data summarization algorithms in the centralized setting [DV06, DV07, DRVW06]. The idea is that S begins as the empty set and some row $\mathbf{A}_i$ of $\mathbf{A}$ is sampled with probability $\frac{\|\mathbf{A}_i\|_2^p}{\|\mathbf{A}\|_{p,2}^p}$, where $p \in \{1,2\}$ and $\|\mathbf{A}\|_{p,q} = \left(\sum_{i=1}^n \left(\sum_{j=1}^d |A_{i,j}|^q\right)^{\frac{p}{q}}\right)^{\frac{1}{p}}$. As S is populated, the algorithm *adaptively samples* rows of $\mathbf{A}$, so that at each iteration, row $\mathbf{A}_i$ is sampled with probability proportional to the p^{th} power of the distance of the row from S. That is, $\mathbf{A}_i$ is sampled with probability $\frac{\left\|\mathbf{A}_i(\mathbb{I}-\mathbf{M}^\dagger\mathbf{M})\right\|_2^p}{\left\|\mathbf{A}(\mathbb{I}-\mathbf{M}^\dagger\mathbf{M})\right\|_{p,2}^p}$, where $\mathbf{M}$ is the matrix formed by the rows in S. The procedure is repeated k times until we obtain k rows of $\mathbf{A}$, which then forms our summary of the matrix $\mathbf{A}$. Unfortunately, adaptive sampling seems like an inherently sequential procedure and thus the extent of its capabilities has not been explored in the streaming model.

1.1 Our Contributions

In this paper, we show that although adaptive sampling seems like an iterative procedure, we do not need multiple passes over the stream to perform adaptive sampling. This is particularly surprising since *any* row $\mathbf{A}_i$ of $\mathbf{A}$ can be made irrelevant, i.e., zero probability of being sampled, in future rounds if some row along the same direction of $\mathbf{A}_i$ is sampled in the present round. Yet somehow we must still output rows of $\mathbf{A}$ while storing a sublinear number of rows more or less non-adaptively. The challenge seems compounded by the turnstile model, since updates can be made to arbitrary elements of the matrix, but somehow we need to recover the rows with the largest norms. For example, if the last update in the stream is substantially larger than the previous updates, an adaptive sampler must return the entire row, even though this update could be an entry in any row of $\mathbf{A}$.

To build our adaptive sampler, we first give an algorithm that performs a single round of sampling. Namely, given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that is defined over a turnstile stream and post-processing query access to a matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$, we first give $L_{p,q}$ samplers for $\mathbf{AP}$.

Theorem 1.1 *Let $\epsilon > 0$, $q = 2$, and $p \in \{1,2\}$. There exists a one-pass streaming algorithm that*

takes rows of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ *as a turnstile stream and post-processing query access to matrix* $\mathbf{P} \in \mathbb{R}^{d\times d}$ *after the stream, and with high probability, samples an index* $i \in [n]$ *with probability* $(1 \pm \mathcal{O}(\epsilon)) \frac{\|\mathbf{A}_i\mathbf{P}\|_q^p}{\|\mathbf{AP}\|_{p,q}^p} + \frac{1}{\text{poly}(n)}$. *The algorithm uses* $\text{poly}\left(d, \frac{1}{\epsilon}, \log n\right)$ *bits of space. (See Theorem 2.7 and Theorem A.4.)*

We remark that our techniques can be extended to $p \in (0, 2]$ but we only require $p \in \{1, 2\}$ for the purposes of our applications. Now, suppose we want to perform adaptive sampling of a row of $\mathbf{A} \in \mathbb{R}^{n\times d}$ with probability proportional to its distance or squared distance from some subspace $\mathbf{H} \in \mathbb{R}^{i\times d}$, where i is any integer. Then by taking $\mathbf{P} = \mathbb{I} - \mathbf{H}^\dagger\mathbf{H}$ and either $L_{1,2}$ or $L_{2,2}$ sampling, we select rows of $\mathbf{A}$ with probability roughly proportional to the distance or squared distance from $\mathbf{H}$. We can thus simulate k rounds of adaptive sampling in a stream, despite its seemingly inherent sequential nature.

Theorem 1.2 *Let* $\mathbf{A} \in \mathbb{R}^{n\times d}$ *be a matrix and* q_S *be the probability of selecting a set* $S \subset [n]$ *of* k *rows of* $\mathbf{A}$ *according to* k *rounds of adaptive sampling with respect to either the distances to the selected subspace in each iteration or the squared distances to the selected subspace in each iteration. There exists an algorithm that takes inputs* $\mathbf{A}$ *through a turnstile stream and* $\epsilon > 0$*, and outputs a set* $S \subset [n]$ *of* k *indices such that if* p_S *is the probability of the algorithm outputting* S*, then* $\sum_S |p_S - q_S| \leq \epsilon$. *The algorithm uses* $\text{poly}\left(d, k, \frac{1}{\epsilon}, \log n\right)$ *bits of space. (See Theorem 3.4 and Theorem A.7.)*

In other words, our output distribution is close in total variation distance to the desired adaptive sampling distribution.

Our algorithm is the first to perform adaptive sampling on a stream; existing implementations require extended access to the matrix, such as in the centralized or distributed models, for subsequent rounds of sampling. Moreover, if the set S of k indices output by our algorithm is $s_1, \ldots, s_k$, then our algorithm also returns a set of rows $\mathbf{r}_1, \ldots, \mathbf{r}_k$ so that if $\mathbf{R}_0 = \emptyset$ and $\mathbf{R}_i = \mathbf{r}_1 \circ \ldots \circ \mathbf{r}_i$ for $i \in [k]$, then $\mathbf{r}_i = \mathbf{u}_{s_i} + \mathbf{v}_i$, where $\mathbf{u}_{s_i} = \mathbf{A}_{s_i}(\mathbb{I} - \mathbf{R}_i^\dagger\mathbf{R}_i)$ is the projection of the sampled row to the space orthogonal to the previously selected rows, and $\mathbf{v}_i$ is some small noisy vector formed by linear combinations of other rows in $\mathbf{A}$ such that $\|\mathbf{v}_i\|_2 \leq \epsilon \|\mathbf{u}_{s_i}\|_2$.

Thus we do not return the true rows of $\mathbf{A}$ corresponding to the indices in S, but we output a small noisy perturbation to each of the rows, which we call noisy rows and suffices for a number of applications previously unexplored in turnstile streams. Crucially, the noisy perturbation in each of our output rows can be bounded in norm not only relative to the norm of the true row, but also relative to the residual. In fact, our previous example of a long stream of small updates followed a single arbitrarily large update shows that it is impossible to return the true rows of $\mathbf{A}$ in sublinear space. Since the arbitrarily large update can apply to *any* entry of the matrix, the only way an algorithm can return the entire row containing the entry is if the entire matrix is stored.

Column subset selection. In the row/column subset selection problem, the inputs are the matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ and an integer $k > 0$, and the goal is to select k rows/columns of $\mathbf{A}$ to form a matrix $\mathbf{M}$ to minimize $\left\|\mathbf{A} - \mathbf{AM}^\dagger\mathbf{M}\right\|_F$ or $\left\|\mathbf{A} - \mathbf{MM}^\dagger\mathbf{A}\right\|_F^2$. For the sake of presentation, we focus on the row subset selection problem for the remainder of this section. Since the matrix $\mathbf{M}$ has rank at most k, then $\left\|\mathbf{A} - \mathbf{AM}^\dagger\mathbf{M}\right\|_F \geq \|\mathbf{A} - \mathbf{A}_k^*\|_F$, where $\mathbf{A}_k^*$ is the best rank k approximation to $\mathbf{A}$. Hence, we would ideally like to obtain some guarantee for $\left\|\mathbf{A} - \mathbf{AM}^\dagger\mathbf{M}\right\|_F$ relative to $\|\mathbf{A} - \mathbf{A}_k^*\|_F$. Such relative error algorithms were given in the centralized setting [DRVW06, BMD09, GS12] and

for row-arrival streams [CMM17, BDM+18], but no such results were previously known for turnstile streams. Our adaptive sampling framework thus provides the first algorithm on turnstile streams with relative error guarantees.

Theorem 1.3 *Given a matrix* $\mathbf{A} \in \mathbb{R}^{n\times d}$ *that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set* $\mathbf{M}$ *of* k *(noisy) rows of* $\mathbf{A}$ *such that*

$$\mathbf{Pr}\left[\left\|\mathbf{A}-\mathbf{A}\mathbf{M}^{\dagger}\mathbf{M}\right\|_F^2 \leq 16(k+1)!\,\|\mathbf{A}-\mathbf{A}_k^*\|_F^2\right] \geq \frac{2}{3}.$$

The algorithm uses $\mathrm{poly}(d,k,\log n)$ *bits of space. (See Theorem 4.3.)*

Subspace approximation. In the subspace approximation problem, the inputs are the matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ and an integer $k > 0$ and the goal is to output a k-dimensional linear subspace $\mathbf{H}$ that minimizes $(\sum_{i=1}^n d(\mathbf{A}_i,\mathbf{H})^p)^{\frac{1}{p}}$, where $p \in \{1,2\}$ and $d(\mathbf{A}_i,\mathbf{H}) = \left\|\mathbf{A}_i(\mathbb{I}-\mathbf{H}^{\dagger}\mathbf{H})\right\|_2$ is the distance from $\mathbf{A}_i$ to the subspace $\mathbf{H}$. A number of algorithms for the subspace approximation were given for the centralized setting [DV07, FMSW10, SV12, CW15] and more recently, [LSW18] gave the first algorithm for subspace approximation on turnstile streams. The algorithm of [LSW18] is based on sketching techniques and although it offers a superior $(1+\epsilon)$-approximation, their subspace has a larger number of rows and the rows may not originate from $\mathbf{A}$, whereas we select k noisy rows of the matrix $\mathbf{A}$ to form the subspace.

Theorem 1.4 *Given* $p \in \{1,2\}$ *and a matrix* $\mathbf{A} \in \mathbb{R}^{n\times d}$ *that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set* $\mathbf{Z}$ *of* k *(noisy) rows of* $\mathbf{A}$ *such that*

$$\mathbf{Pr}\left[\left(\sum_{i=1}^n d(\mathbf{A}_i,\mathbf{Z})^p\right)^{\frac{1}{p}} \leq 4(k+1)!\left(\sum_{i=1}^n d(\mathbf{A}_i,\mathbf{A}_k^*)^p\right)^{\frac{1}{p}}\right] \geq \frac{2}{3},$$

where $\mathbf{A}_k^*$ *is the best rank* k *solution to the subspace approximation problem. The algorithm uses* $\mathrm{poly}\,(d,k,\log n)$ *bits of space. (See Theorem 4.5 and Theorem 4.6.)*

Our adaptive sampling procedure also gives a bicriteria algorithm for a better approximation but allows the dimension of the subspace to be larger.

Theorem 1.5 *Given* $p \in \{1,2\}$, $\epsilon > 0$, *and a matrix* $\mathbf{A} \in \mathbb{R}^{n\times d}$ *that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set* $\mathbf{Z}$ *of* $\mathrm{poly}\left(k,\frac{1}{\epsilon},\log\frac{k}{\epsilon}\right)$ *(noisy) rows of* $\mathbf{A}$ *such that*

$$\mathbf{Pr}\left[\left(\sum_{i=1}^n d(\mathbf{A}_i,\mathbf{Z})^p\right)^{\frac{1}{p}} \leq (1+\epsilon)\left(\sum_{i=1}^n d(\mathbf{A}_i,\mathbf{A}_k^*)^p\right)^{\frac{1}{p}}\right] \geq \frac{2}{3},$$

where $\mathbf{A}_k^*$ *is the best rank* k *solution to the subspace approximation problem. The algorithm uses* $\mathrm{poly}(d,k,\frac{1}{\epsilon},\log n)$ *bits of space. (See Theorem 4.8.)*

Projective clustering. Projective clustering is an important problem for bioinformatics, computer vision, data mining, and unsupervised learning [Pro17]. The projective clustering problem takes as inputs the matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ and integers $k > 0$ for the target dimension of each subspace and $s > 0$ for the number of subspaces, and the goal is to output s k-dimensional linear subspaces

$\mathbf{H}_1, \ldots, \mathbf{H}_s$ that minimizes $(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{H})^p)^{\frac{1}{p}}$, where $p \in \{1, 2\}$, $\mathbf{H} = \mathbf{H}_1 \cup \ldots \cup \mathbf{H}_s$, and $d(\mathbf{A}_i, \mathbf{H})$ is the distance from $\mathbf{A}_i$ to union $\mathbf{H}$ of s subspaces $\mathbf{H}_1, \ldots, \mathbf{H}_s$. A number of streaming algorithms for projective clustering [BHI02, HM04, Che09, FMSW10] are based on the notion of core-sets, which are small numbers of weighted representative points. These results require a stream of (possibly high dimensional) points, which is equivalent to the row-arrival model and thus do not extend to turnstile streams. [KR15] gives a turnstile algorithm based on random projections, but the algorithm requires space linear in the number of points. Thus our adaptive sampling procedure gives the first turnstile algorithm for projective clustering that uses space sublinear in the number of points.

Theorem 1.6 *Given $p \in \{1, 2\}$, $\epsilon > 0$ and a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set $\mathbf{S}$ of $\operatorname{poly}\left(k, s, \frac{1}{\epsilon}\right)$ rows, which includes a union $\mathbf{T}$ of s k-dimensional subspaces such that*

$$\mathbf{Pr}\left[\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T})^p\right)^{\frac{1}{p}} \leq (1+\epsilon)\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{H})^p\right)^{\frac{1}{p}}\right] \geq \frac{2}{3},$$

where $\mathbf{H}$ is the union of s k-dimensional subspaces that is the optimal solution to the projective clustering problem. The algorithm uses $\operatorname{poly}\left(d, k, s, \frac{1}{\epsilon}, \log n\right)$ bits of space. (See Theorem 4.10.)

Volume maximization. The volume maximization problem takes as inputs a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and a parameter k for the number of selected rows, and the goal is to output k rows $\mathbf{r}_1, \ldots, \mathbf{r}_k$ of $\mathbf{A}$ that maximize the volume of the parallelepiped spanned by the rows. [IMGR20, IMGR19] give core-set constructions for volume maximization that approximate the optimal solution within a factor of $\tilde{\mathcal{O}}(k)^{k/2}$ and $\mathcal{O}(k)^k$ respectively, and can be implemented in the row-arrival model. Their algorithms are based on spectral spanners and local search based on directional heights and do not immediately extend to turnstile streams. Hence our adaptive sampling procedure gives the first turnstile algorithm for volume maximization that uses space sublinear in the input size.

Theorem 1.7 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives in a turnstile data stream and an approximation factor $\alpha > 1$, there exists a one-pass algorithm that outputs a set $\mathbf{S}$ of k noisy rows of $\mathbf{A}$ such that*

$$\mathbf{Pr}\left[\alpha^k (k!) \operatorname{Vol}(\mathbf{S}) \geq \operatorname{Vol}(\mathbf{M})\right] \geq \frac{2}{3},$$

where $\operatorname{Vol}(\mathbf{S})$ is the volume of the parallelepiped spanned by $\mathbf{S}$ and $\mathbf{M}$ is a set of k rows that maximizes the volume. The algorithm uses $\tilde{\mathcal{O}}\left(\frac{ndk^2}{\alpha^2}\right)$ bits of space. (See Theorem 4.13.)

We complement Theorem 4.13 with a lower bound for the volume maximization problem on turnstile streams that is tight up to lower order terms. Additionally, we give a lower bound for volume maximization in the random order row-arrival model, which we will also show is tight up to lower order terms. Our lower bounds complement the thorough lower bounds for extent problems given by [AS15].

Theorem 1.8 *There exists a constant $C > 1$ so that any one-pass streaming algorithm that outputs a C^k approximation to the volume maximization problem with probability at least $\frac{63}{64}$ in the random order row-arrival model requires $\Omega(n)$ bits of space. Moreover for any integer $p > 0$, any p-pass turnstile streaming algorithm that gives an α^k approximation to the volume maximization problem requires $\Omega\left(\frac{n}{kp\alpha^2}\right)$ bits of space. (See Corollary 6.3 and Corollary 6.7.)*

Finally, we give a corresponding upper bound for volume maximization in the row-arrival model competitive with our lower bound.

Theorem 1.9 *Let $1 < C < (\log n)/k$. There exists a one-pass streaming algorithm in the row-arrival model that computes a subset $\mathbf{S}$ of size k of points in $\mathbb{R}^d$ such that*

$$\mathbf{Pr}\left[\mathcal{O}\left(Ck\right)^{k/2}\mathrm{Vol}(\mathbf{S}) \geq \mathrm{Vol}(\mathbf{M})\right] \geq \frac{2}{3},$$

where $\mathrm{Vol}(\mathbf{S})$ is the volume of the parallelepiped spanned by $\mathbf{S}$ and $\mathbf{M}$ is a set of k rows that maximizes the volume. The algorithm uses $\mathcal{O}\left(n^{\mathcal{O}(1/C)}d\right)$ bits of space. (See Lemma 5.12.)

1.2 Techniques

Our first observation is that in many applications, the role played by adaptive sampling is to sample rows of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ with probability proportional to either the distance or the squared distance from some subspace $\mathbf{H}_j$ that we have already chosen by step j of the sampling procedure. Adaptive sampling then imbues some randomness into the sampling procedure, which would otherwise reduce to the greedy paradigm of iteratively choosing the row of $\mathbf{A}(\mathbb{I}-\mathbf{H}_j^\dagger\mathbf{H}_j)$ with the largest squared norm. That is, we care more about the rows of $\mathbf{A}(\mathbb{I}-\mathbf{H}_j^\dagger\mathbf{H}_j)$ than the rows of $\mathbf{A}$.

Thus our first component towards our adaptive sampling algorithm is an $L_{p,2}$ sampler with $p \in \{1,2\}$, which takes turnstile updates to $\mathbf{A}$ and post-processing query access to a matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$ and outputs index $i \in [n]$ with probability roughly, i.e., within $(1\pm\epsilon)$ factor of $\frac{\|\mathbf{A}_i\mathbf{P}\|_2^p}{\|\mathbf{AP}\|_{p,2}^p}$. By setting $\mathbf{P} = \mathbb{I}-\mathbf{H}_j^\dagger\mathbf{H}_j$, the probability of sampling each row of $\mathbf{AP}$ then approximately follows the adaptive sampling distribution.

$L_{p,2}$ sampler. We first describe how to sample rows of $\mathbf{A}$ when $\mathbf{P}$ is the identity matrix, so that we output an index $i \in [n]$ with probability roughly $\frac{\|\mathbf{A}_i\|_2^p}{\|\mathbf{A}\|_{p,2}^p}$ with $p \in \{1,2\}$. Our scheme generalizes a line of work for ℓ_p sampling [MW10, SW11, AKO11, JST11, JW18], where the input is a vector $\mathbf{f}$ of n coordinates that are updated through a turnstile stream and the goal is to sample an index $i \in [n]$ with probability roughly $\frac{|f_i|^p}{\|\mathbf{f}\|_p^p}$, from coordinates of a vector input to rows of a matrix input. These prior ℓ_p sampling algorithms have essentially followed the same template of performing a linear transformation on $\mathbf{f}$ to obtain a new vector $\mathbf{z}$, using an instance of CountSketch on $\mathbf{z}$ to recover a vector $\mathbf{y}$, and then running a statistical test on $\mathbf{y}$. If the statistical test fails, then the algorithm aborts; otherwise the coordinate of $\mathbf{y}$ with the maximum magnitude is output. The algorithm is repeated a number of times to ensure a high probability of success.

Generalizing the template of ℓ_p sampling, we observe that if some scaling factor $t_i \in [0,1]$ is chosen uniformly at random, then $\mathbf{Pr}\left[\frac{\|\mathbf{A}_i\|_2}{t_i^{1/p}} \geq K^{1/p}\|\mathbf{A}\|_{p,2}\right] = \frac{\|\mathbf{A}_i\|_2^p}{K\|\mathbf{A}\|_{p,2}^p}$, where K is any parameter that we choose. Thus if $\mathbf{B}_i = \frac{1}{t_i^{1/p}}\mathbf{A}_i$ for $i \in [n]$ and we temporarily suppose that only the row x satisfies $\|\mathbf{B}_x\|_2 \geq T$, where $T = K^{1/p}\|\mathbf{A}\|_{p,2}$ is the threshold, for only the index $i = x$, then our task would reduce to identifying $\mathbf{B}_x$ in sublinear space. To this end, if $\mathbf{B}$ is the matrix whose rows are $\mathbf{B}_1,\ldots,\mathbf{B}_n$, then we hash the rows of $\mathbf{B}$ to a CountSketch data structure to recover the row with the largest norm, which must necessarily be $\mathbf{B}_x$ if the error in CountSketch is small enough.

Namely, if the error in CountSketch data structure is too large, then our algorithm will erroneously identify some scaled row as exceeding the threshold T when the scaled row does not, or vice versa. Hence our algorithm must first run a statistical test to determine whether the error in the CountSketch data structure caused by the randomness of the data structure is sufficiently small. If the CountSketch error is determined to be too large by the statistical test, the algorithm aborts; otherwise the algorithm outputs the row with the largest norm if it exceeds T.

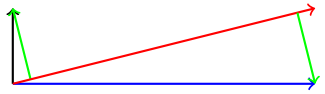
Now there can still be some error if multiple rows have norms close to or exceeding T, but it turns out that by choosing the appropriate parameters, the probability that there exists a row whose norm exceeds T is $\Omega\left(\frac{1}{K}\right)$ and the probability that the statistical test fails or that multiple rows have norms close to or exceeding T is $\mathcal{O}\left(\frac{\epsilon}{K}\right)$, which incurs a relative $(1 \pm \epsilon)$ perturbation of the sampling probabilities. Thus a single instance of the sampler outputs an index from roughly the desired distribution with probability $\Omega\left(\frac{1}{K}\right)$ and with probability $1 - \Omega(\frac{1}{K})$, it aborts and outputs nothing. Hence for $p \in \{1, 2\}$, we obtain a constant probability of success using $\text{poly}\left(\frac{1}{\epsilon}, \log n\right)$ space by setting $K = \text{poly}\left(\frac{1}{\epsilon}, \log n\right)$, repeating with $\mathcal{O}(K)$ instances, and taking the output of the first successful instance.

It remains to argue that CountSketch and norm estimation generalize to $L_{p,2}$ error for matrices, which we do through standard arguments in Section 2. In fact, the data structures maintained by the generalized matrix CountSketch and $L_{p,2}$ norm estimation procedures are linear combinations of the rows of $\mathbf{A}$, so we can right multiply the rows that are stored in the $L_{p,2}$ sampler by $\mathbf{P}$ to simulate sampling rows of $\mathbf{AP}$. In other words, if we had a stream of updates to the matrix $\mathbf{AP}$, the resulting data structure on the stream would be equivalent to maintaining the data structure on a stream of updates to the matrix $\mathbf{A}$, and then multiplying each row of the data structure by $\mathbf{P}$ in post-processing. Hence we can also sample rows of $\mathbf{AP}$ with probabilities proportional to the residual $\|\mathbf{A}_i\mathbf{P}\|_2^p$, which will be crucial for our adaptive sampler.

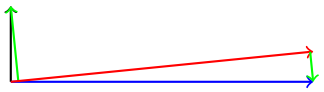
Adaptive sampler. Recall that adaptive sampling iteratively samples rows of $\mathbf{A} \in \mathbb{R}^{n \times d}$ with probability proportional to the p^{th} power of their distances from the subspace spanned by the rows that have already been sampled in previous rounds, for k rounds. Thus if $\mathbf{H}_j$ is the matrix formed by the rows sampled by step j, then we would like to sample $i \in [n]$ with probability $\left\|\mathbf{A}_i(\mathbb{I} - \mathbf{H}_j^\dagger \mathbf{H}_j)\right\|_2^p$ with the largest squared norm. Given our $L_{p,2}$ sampler, a natural approach is to run k instances of the sampler throughout the stream. Once the stream completes, we use the first instance to sample a row of $\mathbf{A}$, which forms $\mathbf{H}_2$ (recall that $\mathbf{H}_1 = \emptyset$ is "used" to sample in the first iteration). Since our $L_{p,2}$ sampler supports post-processing multiplication by a matrix $\mathbf{P}$, we subsequently use the j^{th} instance to $L_{p,2}$ sample a row of $\mathbf{A}(\mathbb{I} - \mathbf{H}_j^\dagger \mathbf{H}_j)$, which we then append to $\mathbf{H}_j$ to form $\mathbf{H}_{j+1}$. Repeating this k times, we would like to argue this simulates k steps of adaptive sampling.

The first issue with this approach is that our $L_{p,2}$ cannot return the original rows of $\mathbf{A}$, but only some noisy perturbation of the sampled row. It is easy to see that returning the noisy rows of $\mathbf{A}$ is unavoidable for sublinear space by considering a stream whose final update to some random coordinate is arbitrarily large, while the previous updates were small. Then the row containing the coordinate of the final update should be sampled with large probability, but that row can only be completely recovered if all entries of the matrix are stored. Fortunately we show that if we sample the index x, then we output a row $\mathbf{r} = \mathbf{A}_x + \mathbf{v}$, where the noisy component $\mathbf{v}$ is a linear combination of rows of $\mathbf{A}$ that satisfies $\|\mathbf{v}\|_2 \leq \epsilon \|\mathbf{A}_x\|_2$. Thus, the norms of the sampled rows are somewhat preserved.

On the other hand, sampling noisy rows of $\mathbf{A}$ rather than the original rows of $\mathbf{A}$ can drastically



(a) Troublesome distorted sampling probabilities.



(b) Actual distorted sampling probabilities.

Fig. 1: Distortion of sampling probabilities by $L_{p,2}$ sampler. Suppose we should have sampled the blue vector but instead we obtain the red vector from the noisy output. Then the sampling probabilities in the second round will be the norms of the green vectors, even though the probability of sampling the blue vector in the second round should *actually be zero* if we had sampled the actual blue vector. Thus the sampling probabilities are distorted. In particular, we are worried that in Figure 1a, the blue vector might be sampled again in the second round because the projection to the red vector has large norm. We show Figure 1a is unlikely and the actual scenario is more like Figure 1b, where the sampling probabilities are only perturbed by a small additive amount.

alter the subspace spanned by the matrix formed by the rows. This in turn can significantly alter the sampling probabilities in future rounds. Consider the following example, which is depicted in Figure 1a. Let $\mathbf{A}$ be a matrix that has $\begin{bmatrix}0 & 1\end{bmatrix}$ for half of its rows and $\begin{bmatrix}M & 0\end{bmatrix}$ for some large $M > 0$ for the other half of its rows. Then with large probability we should sample some row $\mathbf{u} = \begin{bmatrix}M & 0\end{bmatrix}$ in the first step. However, due to noise in the sampler, we will actually obtain some noisy row $\mathbf{v} = \begin{bmatrix}M' & m\end{bmatrix}$, where $M' \approx M$ and $m \neq 0$. In Figure 1a, we depict $\mathbf{u}$ with the blue vector and $\mathbf{v}$ with the red vector.

Now in the second round, if we had sampled $\mathbf{u}$ in the first round, then the only possible output of the adaptive sampler is a row $\begin{bmatrix}0 & 1\end{bmatrix}$, since all the rows $\begin{bmatrix}M & 0\end{bmatrix}$ are contained in the subspace spanned by $\mathbf{u}$. However, since we actually sampled $\mathbf{v}$ in the first round, then the distance from $\mathbf{u}$ to the subspace spanned by $\mathbf{v}$ is nonzero. Furthermore, since M is large, then it actually seems likely that we might sample a row $\begin{bmatrix}M & 0\end{bmatrix}$ rather than $\begin{bmatrix}0 & 1\end{bmatrix}$. Thus we might sample some row that we should have not sampled or worse, we might repeatedly sample the same row! Pictorially, the blue vector $\mathbf{u}$ in Figure 1a has no projection away from itself, but results in the rightmost green vector when projected to the red vector $\mathbf{v}$, and thus $\mathbf{u}$ might be sampled *again* with high probability.

Similarly, the noisy perturbations may cause us to completely avoid rows that we should have sampled with nonzero probability if we had access to the original rows. In fact, this example shows that we cannot guarantee that our adaptive sampler gives a multiplicative $(1+\epsilon)$-approximation to the true sampling probabilities of each row in any round.

Our key observation is that the noisy row $\mathbf{r}$ output by our $L_{p,2}$ sampler not only has a noisy component $\mathbf{v}$ small in norm compared to $\mathbf{A}_x$, but also the component of $\mathbf{v}$ in the space orthogonal to $\mathbf{A}_x$ must be small. That is, $\mathbf{r}$ can also be written as $\mathbf{r} = \mathbf{A}_x + \mathbf{w}$, where $\|\mathbf{wQ}\|_2 \leq \epsilon \|\mathbf{A}_x\|_2 \frac{\|\mathbf{AQ}\|_F}{\|\mathbf{A}\|_F}$ for *any* projection matrix $\mathbf{Q}$. This tighter bound in any orthogonal direction allows us to bound in subsequent rounds the *additive* error of sampling probabilities, which are based on the vector lengths in orthogonal directions. Thus, we show that with high probability, Figure 1a cannot happen and our actual situation is more like that in Figure 1b.

Namely, if we write an orthonormal basis U for the actual rows of $\mathbf{A}$ and an orthonormal basis W for the noisy rows that we sample, we can show that the norm of a row projected onto W has a small additive perturbation from when it is projected onto U. Thus we require the construction of orthonormal bases U and W from which we can easily extract the sampling probabilities of rows both with respect to the original rows and to the noisy rows. We achieve this by designing U so

that the first basis vectors of U are precisely the true sampled rows of $\mathbf{A}$, followed by the noisy perturbations for each sample. We then argue that if we design W so that the first basis vectors of W are precisely the noisy rows that we sample, then the coefficients of each row represented in terms of U and W have only a small additive difference. By summing across all rows, then we can bound the total variation distance between sampling with the noisy rows of $\mathbf{A}$ and sampling with the actual rows of $\mathbf{A}$. That is, in each adaptive sampling round where we use a noisy row obtained from our $L_{p,2}$ sampler with error parameter $\epsilon > 0$ rather than the actual row of $\mathbf{A}$, the total variation distance in the sampling distribution increases by an additive $\mathcal{O}(\epsilon)$. Now since we use the first sampled noisy row for $k-1$ additional rounds, the second sampled noisy row for $k-2$ additional rounds, and so forth, then by an inductive argument, the total variation distance between k rounds of our algorithm and k rounds of adaptive sampling is $\mathcal{O}\left(k^2\epsilon\right)$.

Applications. For many applications on turnstile streams, we show it suffices in each step to obtain a noisy row that is orthogonal to the previously selected rows, sampled with probability proportional to the p^{th} power of the distance to the subspace spanned by those rows. Thus for $p = 1$, our adaptive sampler allows us to perform residual based sampling in place of subspace embedding techniques used by previous work in various applications [KR15, SW18, LSW18]. Additionally for $p = 2$, our adaptive sampler allows us to simulate volume sampling, which has a wide range of applications [DV06, DRVW06, DV07, ÇM09].

For volume maximization on turnstile streams, we use a combination of our $L_{2,2}$ sampler and our generalized CountSketch data structure to simulate an approximation to the greedy algorithm of choosing the row with the largest residual at each step. If the largest residual found by CountSketch exceeds a certain threshold, we use that row; otherwise any row output by our adaptive sampler will be a good approximation to the row with the largest residual. Thus the volume of the parallelepiped spanned by these rows is a good approximation to the optimal solution.

Volume Maximization. We provide lower bounds for volume maximization in turnstile streams and the row-arrival model through reductions from the Gap ℓ_∞ problem and the distributional set-disjointness problem, respectively. For both cases we show that embedding the same instance across multiple columns gives hardness of approximating within a factor with exponential dependency on k. For our algorithmic results in the row-arrival model, we first note that the composable core-set techniques of [IMGR20] automatically gives a streaming algorithm for volume maximization. In fact, [IMGR20] shows the stronger guarantee that any composable core-set for the directional height of a point set suffices to give a good approximation for volume maximization. Using this idea, we give a dimensionality reduction algorithm for volume maximization in the row-arrival model competitive with our lower bounds by embedding the input into a lower dimensional space.

Recall that from Johnson-Lindenstrauss, right multiplication by a random matrix whose entries are drawn i.i.d from a Gaussian distribution suffices to preserve the directional heights of the points in an optimal set by a constant factor, say 2, and thus the volume of the largest set of k points is only distorted by a factor of 2^k. We then prove that for every other subset of k points, their volume does not increase by too much by showing that the eigenvalues of the matrix representation of the points are preserved by some factor C with very high probability. Thus taking a union bound over all subsets of k points, all volumes are preserved by a factor C^k and we obtain dimensionality reduction of the problem by applying right multiplication of the random matrix to each of the input rows.

Paper Organization. We first handle $L_{2,2}$ sampling in Section 2. Using the $L_{2,2}$ sampler, we build an adaptive sampler in Section 3 that samples rows with probability proportional to the squared distances of the subpsace spanned by previously selected rows. We show the applications of our adaptive sampler in Section 4, including projective clustering, subspace approximation, column subset selection, and volume maximization. We give lower bounds for volume maximization in Section 6, showing that our adaptive sampler gives nearly optimal algorithms in turnstile streams. Finally we give algorithms for volume maximization in the row-arrival model in Section 5, competitive with the lower bounds in Section 6. For completeness, we detail $L_{1,2}$ sampling and adaptive sampling rows with probability proportional to the distances of the subspace spanned by previously selected rows in Appendix A.

1.3 Preliminaries

For any positive integer n, we use the notation $[n]$ to represent the set $\{1, \ldots, n\}$. We use the notation $x = (1 \pm \epsilon)y$ to denote the containment $(1-\epsilon)y \leq x \leq (1+\epsilon)y$. We write $\text{poly}(n)$ to denote some fixed constant degree polynomial in n but we write $\frac{1}{\text{poly}(n)}$ to denote some arbitrary degree polynomial in n. When an event has probability $1 - \frac{1}{\text{poly}(n)}$ of occurring, we say the event occurs with high probability. We use $\tilde{\mathcal{O}}(\cdot)$ to omit lower order terms and similarly $\text{polylog}(n)$ to omit terms that are polynomial in $\log n$.

For our purposes, a turnstile stream will implicitly define a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ through a sequence of m updates. We use $\mathbf{A}_i$ to denote the i^{th} row of $\mathbf{A}$ and $A_{i,j}$ to denote the j^{th} entry of $\mathbf{A}_i$. The matrix $\mathbf{A}$ initially starts as the all zeros matrix. Each update in the stream has the form (i_t, j_t, Δ_t), where $t \in [m]$, $i_t \in [n]$, $j_t \in [d]$, and $\Delta_t \in \{-M, -M+1, \ldots, M-1, M\}$ for some large positive integer M. The update then induces the change $A_{i_t,j_t} \leftarrow A_{i_t,j_t} + \Delta_t$ in $\mathbf{A}$. We assume throughout that $m, M = \text{poly}(n)$ and $n \gg d$. We will typically only permit one pass over the stream, but for multiple passes the order of the updates remains the same in each pass.

In the row-arrival model, the stream has length n and the i^{th} update in the stream is precisely row $\mathbf{A}_i$. Again we restrict each entry $A_{i,j}$ of $\mathbf{A}$ to be in the range $\{-M, -M+1, \ldots, M-1, M\}$ for some large positive integer $M = \text{poly}(n)$. We assume that $\mathbf{A}$ can be adversarially chosen in the row-arrival model, but for the random order row-arrival model, once the entries of $\mathbf{A}$ are chosen, an arbitrary permutation of the rows of $\mathbf{A}$ is chosen uniformly at random, and the rows of that permutation constitute the stream. For the problems that we consider, the optimal solution is invariant to permutation of the rows of $\mathbf{A}$, so the random order does not impact the desired solution. Observe that algorithms for turnstile streams can be used in the row-arrival model, but not necessarily vice versa.

We use $\mathbb{I}_k$ to denote the $k \times k$ identity matrix and we drop the subscript when the dimensions are clear. We use the notation $\mathbf{A} = \mathbf{A}_1 \circ \mathbf{A}_2 \circ \ldots \circ \mathbf{A}_n$ to denote that the matrix $\mathbf{A}$ is formed by the rows $\mathbf{A}_1, \ldots, \mathbf{A}_n$ and the notation $\mathbf{A}^\top$ to denote the transpose of $\mathbf{A}$. For a matrix $\mathbf{M} \in \mathbb{R}^{k \times d}$ with linearly independent rows, we use $\mathbf{M}^\dagger \in \mathbb{R}^{d \times k}$ to denote the Moore-Penrose pseudoinverse of $\mathbf{M}$, so that $\mathbf{M}^\dagger = \mathbf{M}^\top(\mathbf{M}\mathbf{M}^\top)^{-1}$ and $\mathbf{M}\mathbf{M}^\dagger = \mathbb{I}_k$.

Definition 1.10 (Vector/matrix norms) *For a vector $\mathbf{v} \in \mathbb{R}^n$, we have the Euclidean norm $\|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^n v_i^2}$ and more generally, $\|\mathbf{v}\|_p = (\sum_{i=1}^n |v_i|^p)^{\frac{1}{p}}$. For a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$, we denote the Frobenius norm of $\mathbf{A}$ by $\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^d A_{i,j}^2}$. More generally, we write the $L_{p,q}$ norm of*

$\mathbf{A}$ *by* $\|\mathbf{A}\|_{p,q} = \left(\sum_{i=1}^{n}\left(\sum_{j=1}^{d}|A_{i,j}|^q\right)^{\frac{p}{q}}\right)^{\frac{1}{p}}$*, so that* $\|\mathbf{A}\|_F = \|\mathbf{A}\|_{2,2}$.

For $\mathbf{A} \in \mathbb{R}^{n\times d}$, we use $\mathbf{A}_{tail(b)}$ to denote $\mathbf{A}$ with the b rows of $\mathbf{A}$ with the largest Euclidean norm set to zeros.

Definition 1.11 ($L_{p,q}$ sampling) *Let* $\mathbf{A} \in \mathbb{R}^{n\times d}$*,* $0 \le \epsilon < 1$*, and* $p, q > 0$*. An* $L_{p,q}$ sampler *with* ϵ*-relative error is an algorithm that outputs an index* $i \in [n]$ *such that for each* $j \in [n]$*,*

$$\mathbf{Pr}\left[i = j\right] = \frac{\|\mathbf{A}_j\|_q^p}{\|\mathbf{A}\|_{p,q}^p}(1 \pm \epsilon) + \mathcal{O}\left(n^{-c}\right),$$

for some arbitrarily large constant $c \ge 1$*. In each case, the sampler is allowed to output fail with some probability* δ*, in which case it must output* $\perp$*. When the underlying matrix is just a vector, i.e.,* $d = 1$*, we drop the* q *term and call such an algorithm an* L_p *sampler.*

Definition 1.12 (Adaptive sampling) *Let* $\mathbf{A} \in \mathbb{R}^{n\times d}$ *be a matrix from which we wish to sample and* $\mathbf{M} \in \mathbb{R}^{m\times d}$ *be a matrix corresponding to a specific subspace. For* $p \in \{1, 2\}$*, an* adaptive sampler *is an algorithm that outputs an index* $i \in [n]$ *and the corresponding row* $\mathbf{A}_i$ *such that for each* $j \in [n]$*,*

$$\mathbf{Pr}\left[i = j\right] = \frac{\|\mathbf{A}_j\mathbf{P}\|_2^p}{\|\mathbf{A}\mathbf{P}\|_{p,2}^p},$$

where $\mathbf{P} = \mathbb{I} - \mathbf{M}^\dagger\mathbf{M}$.

In typical applications, we will wish to perform k rounds of adaptive sampling with subspaces $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_k$, where $\mathbf{M}_1$ is the all zeros matrix, and each $\mathbf{M}_i$ will consist of the rows sampled from rounds 1 to $i-1$. We will use the term adaptive sampling to refer to both a single round of sampling and multiple rounds of sampling interchangeably when the context is clear.

Note that the adaptive sampling for input matrices $\mathbf{A} \in \mathbb{R}^{n\times d}$ and $\mathbf{M} \in \mathbb{R}^{m\times d}$ can be seen as $L_{p,2}$ sampling on an input matrix $\mathbf{AP}$ with $\epsilon = 0$ and $\mathbf{P} = \mathbb{I} - \mathbf{M}^\dagger\mathbf{M}$, but returning the row $\mathbf{A}_i$ instead of $\mathbf{A}_i\mathbf{P}$.

AMS and CountSketch. We will refer to the classical AMS and CountSketch algorithms for intuition, but we require more generalized versions that we will present in Section 2. For the sake of completeness, the classical AMS algorithm [AMS99] can be formulated as taking a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ through a turnstile stream and using $\mathcal{O}\left(\frac{1}{\epsilon^2}\log^2 n\right)$ bits of space to output a $(1+\epsilon)$-approximation to $\|\mathbf{A}\|_F$ with high probability. For each entry $A_{i,j}$, the algorithm generates a random sign $h_{i,j}$ and maintains $S = \sum_{i=1}^{n}\sum_{j=1}^{d} h_{i,j}A_{i,j}$ throughout the stream. At the end of the stream, the algorithm uses S^2 as its estimator for $\|\mathbf{A}\|_F^2$. By running $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ instances of the estimator and taking the mean, the variance of the estimator decreases. By taking the median of $\mathcal{O}\left(\log n\right)$ means, the estimator succeeds with high probability. By taking $d = 1$, the AMS algorithm can also be used to approximate $\|\mathbf{v}\|_2$ for any vector $\mathbf{v} \in \mathbb{R}^n$ whose entries are updated in a turnstile stream.

The classical CountSketch algorithm [CCF04] can be used to find all entries $A_{i,j}$ of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives implicitly through a turnstile stream such that $|A_{i,j}| \ge \epsilon\|\mathbf{A}\|_F$ for a constant input parameter $\epsilon > 0$. The CountSketch data structure maintained by the algorithm is a $r \times b$ table T. For each row $k \in [r]$, a random sign $h_k(i,j)$ is generated for each $A_{i,j}$ and each

entry $A_{i,j}$ is randomly hashed to a bucket $g_k(i,j) \in [b]$ in row k. Each bucket ℓ in row k then maintains $T_{k,\ell} = \sum_{(i,j):g_k(i,j)=\ell} A_{i,j} h_k(i,j)$, which is a linear combination of the entries assigned to the bucket along with the random signs for the entries. Then for each row k, the estimator for $A_{i,j}$ is $T_{k,g_k(i,j)} h_k(i,j)$, which is the value in the bucket of row k assigned to $A_{i,j}$, rescaled by the random sign. Finally, the estimator for $A_{i,j}$ by the CountSketch data structure is the median of the estimators of $A_{i,j}$ across all rows. It can be seen that $r = \mathcal{O}(\log n)$ and $b = \mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ suffices to estimate each entry of $A_{i,j}$ within an additive $\frac{\epsilon}{4} \|\mathbf{A}\|_F$ factor with high probability. Thus if $A_{i,j} \geq \epsilon \|\mathbf{A}\|_F$, its estimated value will exceed $\frac{\epsilon}{2} \|\mathbf{A}\|_F$ and will be output by CountSketch given an accurate estimation of $\|\mathbf{A}\|_F$, such as by AMS.

We require the following definition of total variation distance to bound the difference between two probability distributions, such as the "ideal" sampling distributions compared to the distributions provided by our algorithms.

Definition 1.13 (Total variation distance) *Let μ, ν be two probability distributions on a finite domain Ω. Then the total variation distance between μ and ν is defined as $d_{\mathsf{TV}}(\mu,\nu) = \frac{1}{2}\sum_{x\in\Omega} |\mu(x) - \nu(x)|$.*

2 $L_{2,2}$ Sampler

In this section, we first describe a turnstile streaming algorithm that takes a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives as a data stream and post-processing query access to a matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$, and outputs an index $i \in [n]$ of a row of $\mathbf{AP}$ sampled with probability roughly $\frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\|\mathbf{AP}\|_F^2}$.

High level idea. First suppose we only wanted to sample a row i of $\mathbf{A} \in \mathbb{R}^{n\times d}$ with probability roughly $\frac{\|\mathbf{A}_i\|_2^2}{\|\mathbf{A}\|_F^2}$. By multiplying each row $\mathbf{A}_i$ with a random scaling factor $\frac{1}{\sqrt{t_i}}$, where $t_i \in [0,1]$ is chosen independently and uniformly at random, the probability that $\frac{1}{t_i}\|\mathbf{A}_i\|_2^2 \geq \|\mathbf{A}\|_F^2$ is precisely the probability that $t_i \leq \frac{\|\mathbf{A}_i\|_2^2}{\|\mathbf{A}\|_F^2}$, which is the desired probability of sampling row i.

Now suppose only one row $\mathbf{A}_i$ satisfies $\frac{1}{t_i}\|\mathbf{A}_i\|_2^2 \geq \|\mathbf{A}\|_F^2$, so that we would like to output $\mathbf{A}_i$. If we stored all rows of $\mathbf{A}$ as well as all scaling factors t_j, then we could identify and output this row, but the required space would be linear in the input size. Instead, we hash all scaled rows $\frac{1}{t_j}\|\mathbf{A}_j\|$ to a number of buckets in a CountSketch data structure. Observe that if $\frac{1}{t_i}\|\mathbf{A}_i\|_2^2 \geq \|\mathbf{A}\|_F^2$ for only one index i, then $\frac{1}{\sqrt{t_i}}\mathbf{A}_i$ must also be the scaled row with the largest norm. Moreover, it turns out that the mass of $\sum_{j=1}^n \frac{1}{t_j}\|\mathbf{A}_j\|_2^2$ is dominated by a small number of rows. Hence with a sufficiently large number of buckets, the scaled row i is the heavy hitter with the largest norm among all the heavy hitters of the scaled rows and so CountSketch will ideally identify the row i.

This approach can fail due to two reasons. The first potential issue is if the accuracy of CountSketch does not suffice to identify the row $\mathbf{A}_i$ due to the noise from the tail of the mass of $\sum_{j=1}^n \frac{1}{t_j}\|\mathbf{A}_j\|_2^2$. That is, if the noise of the tail due to the selection of the scaling factors t_j prevents CountSketch from successfully identifying the heavy hitters, then this approach will fail. We can run a statistical test to identify when the noise is too large and preemptively abort accordingly. Moreover, if the CountSketch data structure maintains enough buckets, then the

noise being sufficiently small happens with probability $\Omega(\epsilon)$, so we can run $\mathcal{O}\left(\frac{1}{\epsilon}\right)$ instances of the algorithm in parallel and take the first instance that does not abort.

A separate issue is resolving the assumption that only one row $\mathbf{A}_i$ satisfies $\frac{1}{t_i}\|\mathbf{A}_i\|_2^2 \geq \|\mathbf{A}\|_F^2$. As it turns out, many rows can exceed this threshold, but if we instead require $\frac{1}{t_i}\|\mathbf{A}_i\|_2^2 \geq \frac{1}{\epsilon}\|\mathbf{A}\|_F^2$, then the probability that some row exceeds this threshold is $\Theta(\epsilon)$. The probability that multiple rows exceed this higher threshold is now $\mathcal{O}\left(\epsilon^2\right)$. Our algorithm outputs the row with the largest norm when some row exceeds the threshold, so in the case where multiple rows exceed the higher threshold we attribute the output to possible sampling probability perturbation. Hence the probability that multiple rows exceed the higher threshold only slightly perturbs the sampling probability of each row by a $(1\pm\epsilon)$ factor. Thus we can again repeat $\mathcal{O}\left(\frac{1}{\epsilon}\right)$ times until some row $\mathbf{A}_i$ satisfies $\frac{1}{t_i}\|\mathbf{A}_i\|_2^2 \geq \|\mathbf{A}\|_F^2$. Similarly, if the error from CountSketch causes an inaccurate estimation of the row with the largest norm, then we might think the heaviest row does not exceed the threshold when it does in reality or vice versa. Fortunately, this only occurs when the row with the largest norm is very close to the threshold, which we again show only causes the sampling probability of each row to perturb by a $(1 \pm \epsilon)$ factor.

For technical reasons, we further increase the threshold and thus run a larger number of instances in parallel to avoid failure. We note that although CountSketch successfully identifies the row i, it can only output a noisy perturbation of $\mathbf{A}_i$. That is, it can only output some row $\mathbf{r} = \mathbf{A}_i + \mathbf{v}$, where the noisy component $\mathbf{v}$ satisfies $\|\mathbf{v}\|_2 \leq \epsilon\|\mathbf{A}_i\|_2$.

Finally, we note that these procedures are all performed through linear sketches and that each bucket stores aggregate rows of the matrix $\mathbf{A}$. Thus if we had a stream of updates to the matrix $\mathbf{AP}$, the resulting data structure would be equivalent to maintaining the data structure on a stream of updates to the matrix $\mathbf{A}$, and then multiplying each row of the data structure by $\mathbf{P}$ post-processing. Hence we can also sample rows of $\mathbf{AP}$ with probabilities proportional to $\|\mathbf{A}_i\mathbf{P}\|_2^2$.

2.1 Streaming Algorithms with Post-Processing

We require generalizations of the celebrated AMS [AMS99] and CountSketch [CCF04] algorithms to handle Frobenius norm estimation of $\mathbf{AP}$ and to output the rows of $\mathbf{AP}$ whose norm exceed a certain fraction of the total Frobenius norm, respectively. These generalizations are streaming algorithms that perform their desired function in low space even though query access to $\mathbf{P}$ is only provided after the stream ends.

Algorithm 1 Basic algorithm that estimates $\|\mathbf{AP}\|_F$, where $\mathbf{P}$ is a post-processing matrix

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, query access to matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$ after the stream ends, constant parameter $\epsilon > 0$.

Output: $(1+\epsilon)$-approximation of $\|\mathbf{AP}\|_F$.

1: Let $h_i \in \{-1, +1\}$ be 4-wise independent for $i \in [n]$.
2: Let $\mathbf{v} \in \mathbb{R}^{1\times d}$ be a vector of zeros.
3: **Streaming Stage:**
4: **for** each update Δ_t to entry $A_{i,j}$ **do**
5: Add $\Delta_t \cdot h_i$ to v_j.
6: **Processing P Stage:**
7: Output $\|\mathbf{vP}\|_2$.

We give in Algorithm 1 the generalization of the AMS [AMS99] algorithm that estimates $\|\mathbf{AP}\|_F^2$, where $\mathbf{A}$ arrives in a stream and post-processing query access to $\mathbf{P}$ is given after the stream ends. Moreover, Algorithm 1 is a linear sketch, so it can also be used to estimate $\|\mathbf{AP}-\mathbf{M}\|_F^2$ for a second arbitrary post-processing matrix $\mathbf{M} \in \mathbb{R}^{n\times d}$.

Lemma 2.1 *Given a constant $\epsilon > 0$, there exists a one-pass streaming algorithm* AMS-M *that takes updates to entries of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, as well as query access to post-processing matrices $\mathbf{P} \in \mathbb{R}^{d\times d}$ and $\mathbf{M} \in \mathbb{R}^{n\times d}$ that arrive after the stream, and outputs a quantity $\hat{F}$ such that $(1-\epsilon)\|\mathbf{AP}-\mathbf{M}\|_F \le \hat{F} \le (1+\epsilon)\|\mathbf{AP}-\mathbf{M}\|_F$. The algorithm uses $\mathcal{O}\left(\frac{d}{\epsilon^2}\log^2 n\right)$ bits of space and succeeds with high probability.*

Proof : Recall that the classic AMS estimator [AMS99] takes a vector $\mathbf{f} \in \mathbb{R}^{n\times 1}$ that arrives as a data stream and outputs an estimate $\hat{f}$ such that $(1-\epsilon)\|\mathbf{f}\|_2 \le \hat{f} \le (1+\epsilon)\|\mathbf{f}\|_2$. The algorithm generates 4-wise independent signs $s_i \in \{-1,+1\}$ for each $i \in [n]$ and maintains $\sum_{i=1}^n s_i f_i$ in the stream. At the end of the stream, $\left(\sum_{i=1}^n s_i f_i\right)^2$ is a good estimator for $\|\mathbf{f}\|_2^2$. The algorithm can then be run $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ times in parallel, taking the mean of the instances to decrease the variance and output a $(1+\epsilon)$-approximation for $\|\mathbf{f}\|_2$. Taking the median of $\mathcal{O}(\log n)$ estimators further increases the probability of success to $1-\frac{1}{\text{poly}(n)}$.

For a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives as a data stream, the Frobenius norm of $\mathbf{A}$ can then be approximated by running a classic AMS estimator for each of the d columns of $\mathbf{A}$. That is, each row of $\mathbf{A}$ can be given a random sign and the algorithm stores the sum of the signed rows in the stream. The Frobenius norm estimator is then the two norm of the stored row. Since each of the classic AMS estimator for the d columns of $\mathbf{A}$ succeeds with high probability, then by a union bound over the $d \ll n$ columns, the estimator is a $(1+\epsilon)$-approximation of the Frobenius norm of $\mathbf{A}$ with high probability. Moreover, since an entire row is stored, the algorithm requires an additional factor of d space. This is less efficient than using a Frobenius norm estimator by hashing each entry of $\mathbf{A}$ to a separate sign and simply storing the sum of the scaled entries, but it is more flexible. In particular, we can apply linear transformations to the stored row to simulate right multiplication on $\mathbf{A}$.

Let $\mathbf{P} \in \mathbb{R}^{d\times d}$ be a given post-processing matrix. To show that Algorithm 1 provides a good approximation to $\|\mathbf{AP}\|_F$, it suffices to argue that running an AMS estimator for the d columns of $\mathbf{A}$ and then multiplying by $\mathbf{P}$ afterwards is equivalent to running an AMS estimator for each of the d columns of $\mathbf{AP}$. Observe that running an AMS estimator for $\mathbf{AP}$ simply requires multiplying each row of $\mathbf{AP}$ by a random sign and adding the resulting signed rows. This is equivalent to multiplying each row of $\mathbf{A}$ by a random sign, adding the resulting signed rows of $\mathbf{A}$, and then multiplying by $\mathbf{P}$, which is exactly what Algorithm 1 does. In other words, the AMS estimator is a linear transformation that maps from $\mathbf{A}$ to $\mathbf{SA}$ for some sketching matrix $\mathbf{S}$, but seeing rows of $\mathbf{A}$ and then multiplying by $\mathbf{P}$ results in the same data structure as seeing the rows of $\mathbf{AP}$, since $\mathbf{S}(\mathbf{AP}) = (\mathbf{SA})\mathbf{P}$ by associativity. Finally, note that if we want to estimate $\|\mathbf{AP}-\mathbf{M}\|_F$ given a post-processing matrix $\mathbf{M}$, then we can compute $\mathbf{S}(\mathbf{AP}-\mathbf{M}) = \mathbf{SAP}-\mathbf{SM}$ for the AMS estimator, given $\mathbf{SAP}$ along with $\mathbf{M}$ and the sketching matrix $\mathbf{S}$.

Each estimator stores a row with d entries each using $\mathcal{O}(\log n)$ bits. The estimator is repeated $\mathcal{O}\left(\frac{1}{\epsilon^2}\log n\right)$ times to give a $(1+\epsilon)$-approximation and to obtain high probability guarantees. Thus, the algorithm requires $\mathcal{O}\left(\frac{d}{\epsilon^2}\log^2 n\right)$ bits of space in total. □

We give in Algorithm 2 the generalization of the CountSketch [CCF04] algorithm that outputs all rows i of $\mathbf{AP}$ such that $\|\mathbf{A}_i\mathbf{P}\|_2 \ge \epsilon\|\mathbf{AP}\|_F$, where $\mathbf{A}$ arrives in a stream and post-processing

query access to $\mathbf{P}$ is given after the stream ends. We call a row i a *heavy row* if $\|\mathbf{A}_i\mathbf{P}\|_2 \geq \epsilon \|\mathbf{AP}\|_F$.

Algorithm 2 Basic algorithm that outputs heavy rows of $\|\mathbf{AP}\|_F$, where $\mathbf{P}$ is a post-processing matrix

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, query access to matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$ after the stream ends, constant parameter $\epsilon > 0$.

Output: Slight perturbations of the rows $\mathbf{A}_i\mathbf{P}$ with $\|\mathbf{A}_i\mathbf{P}\|_2 \geq \epsilon \|\mathbf{AP}\|_F$.

$r \leftarrow \Theta(\log n)$ with a sufficiently large constant.
$b \leftarrow \Omega\left(\frac{1}{\epsilon^2}\right)$ with a sufficiently large constant.
Let T be an $r \times b$ table of buckets, where each bucket stores an $\mathbb{R}^{1\times d}$ row, initialized to zeros.
Let $s_{i,j} \in \{-1, +1\}$ be 4-wise independent for $i \in [n]$, $j \in [r]$.
Let $h_i : [n] \to [b]$ be 4-wise independent for $i \in [r]$.
Streaming Stage:
for each update Δ_t to entry $A_{i,j}$ **do**
    for each $k = 1$ to r **do**
        Add $\Delta_t \cdot s_{i,k}$ to entry j of the vector in bucket $h_k(i)$ of row k.
Let $\mathbf{v}_{k,\ell}$ be the vector in row k, bucket ℓ of T for $k \in [r], \ell \in [b]$.
Processing P Stage:
for $k \in [r], \ell \in [b]$ **do**
    $\mathbf{v}_{k,\ell} \leftarrow \mathbf{v}_{k,\ell}\mathbf{P}$
On query $i \in [n]$, report $\text{median}_{k\in[r]} \left\|\mathbf{v}_{k,h_k(i)}\right\|_2$.

We first show that if $\mathbf{X} = \mathbf{AP}$ and the stream updates the entries of $\mathbf{X}$ rather than the entries of $\mathbf{A}$, then we can obtain a good approximation to the heavy rows. Equivalently, the statement reads that if $\mathbf{P} = \mathbb{I}$ is the identity matrix, then Algorithm 2 finds the heavy rows of $\mathbf{AP}$. We will ultimately show Algorithm 2 finds the heavy rows of $\mathbf{AP}$ for general $\mathbf{P}$ by using the same linear sketching argument as in the proof of Lemma 2.1.

For a matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, recall that $\mathbf{X}_{tail(b)}$ denotes $\mathbf{X}$ with the b rows of $\mathbf{X}$ with the largest norm set to zeros. The following lemma shows that the $\frac{2}{\epsilon^2}$ rows with the largest norm output by Algorithm 2 forms a good estimate of $\mathbf{X}$, even with respect to the stronger Frobenius tail error.

Lemma 2.2 *For any matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, Algorithm 2 outputs an estimate $\widehat{\mathbf{X}_i}$ for each row $\mathbf{X}_i$, which together form an estimate matrix $\widehat{\mathbf{X}}$. Then with high probability, for all $i \in [n]$, there exists a vector $\mathbf{v}_i$ such that $\widehat{\mathbf{X}_i} = \mathbf{X}_i + \mathbf{v}_i$ and $\|\mathbf{v}_i\|_2 \leq \epsilon \left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$. Consequently, $\left|\|\mathbf{X}_i\|_2 - \left\|\widehat{\mathbf{X}_i}\right\|_2\right| \leq \epsilon \left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$ for all $i \in [n]$. Moreover, $\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \left\|\mathbf{X} - \tilde{\mathbf{X}}\right\|_F \leq 2\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$ with high probability, where $\tilde{\mathbf{X}} = \widehat{\mathbf{X}} - \widehat{\mathbf{X}}_{tail\left(\frac{2}{\epsilon^2}\right)}$ denotes the top $\frac{2}{\epsilon^2}$ rows of $\widehat{\mathbf{X}}$ by norm.*

Proof : For a fixed $i \in [n]$ and row k in the CountSketch table T, let $h_k(i)$ be the bucket to which $\mathbf{X}_i$ hashes. Let $\mathcal{E}_1$ be the event that the $\frac{2}{\epsilon^2}$ rows with the highest norms excluding $\mathbf{X}_i$ are not hashed to $h_k(i)$. For $b = \Omega\left(\frac{1}{\epsilon^2}\right)$ with sufficiently large constant, $\mathcal{E}_1$ occurs with probability at least $\frac{9}{10}$. Let $\mathbf{v}_i$ be the sum of the vectors hashed to $h_k(i)$, excluding $\mathbf{X}_i$, so that the vector stored in bucket $h_k(i)$ is $\widehat{\mathbf{X}_i} = \mathbf{X}_i + \mathbf{v}_i$. Conditioned on $\mathcal{E}_1$, the expected squared norm of the noise in bucket

$h_k(i)$ can be bounded by $\mathbb{E}\left[\|\mathbf{v}_i\|_2^2\right] \leq \frac{\epsilon^2}{100}\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F^2$ for sufficiently large b. Note that Jensen's inequality implies $\mathbb{E}\left[\|\mathbf{v}_i\|_2\right] \leq \frac{\epsilon}{10}\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$ and we also have $\mathrm{Var}(\|\mathbf{v}_i\|_2) \leq \frac{\epsilon^2}{100}\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F^2$. Thus by Chebyshev's inequality,

$$\mathbf{Pr}\left[\|\mathbf{v}_i\|_2 \geq \epsilon\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F\right] \leq \frac{1}{81},$$

conditioning on $\mathcal{E}_1$. Hence,

$$\mathbf{Pr}\left[\|\mathbf{v}_i\|_2 \geq \epsilon\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F\right] \leq \frac{1}{81} + \frac{1}{10}.$$

By repeating for each of the $r = \Theta(\log n)$ rows and taking the median, we have from triangle inequality that $\left|\|\mathbf{X}_i\|_2 - \left\|\widehat{\mathbf{X}_i}\right\|_2\right| \leq \epsilon\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$ for all $i \in [n]$ with high probability, thus proving the first part of the claim.

For the second part of the claim, note that $\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \left\|\mathbf{X} - \tilde{\mathbf{X}}\right\|_F$ trivially holds, since $\tilde{\mathbf{X}}$ is a matrix with at most $\frac{2}{\epsilon^2}$ nonzero rows, and $\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}$ has removed the $\frac{2}{\epsilon^2}$ rows of $\mathbf{X}$ with the largest mass from $\mathbf{X}$. Moreover, $\mathbf{X} - \tilde{\mathbf{X}}$ alters at most $\frac{2}{\epsilon^2}$ rows of $\mathbf{X}$, each by at most $\epsilon\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$. Thus,

$$\left\|\mathbf{X} - \tilde{\mathbf{X}}\right\|_F \leq \sqrt{\sum_{i=1}^{2/\epsilon^2}\left(\epsilon\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F\right)^2} = \sqrt{2}\left\|\mathbf{X}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F.$$

□

Taking $b = \Theta\left(\frac{1}{\epsilon^2}\right)$ in Lemma 2.2, we have the following guarantees of CountSketch-M.

Lemma 2.3 *Given a constant $b > 0$, there exists a one-pass streaming algorithm* CountSketch-M *that takes updates to entries of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, as well as query access to a post-processing matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$ that arrives after the stream, and outputs all indices i such that $\|\mathbf{A}_i\mathbf{P}\|_2 \geq \frac{1}{\sqrt{b}}\|\mathbf{AP}\|_F$. For each index i,* CountSketch-M *also outputs a vector $\mathbf{r}$ such that $\mathbf{r} = \mathbf{A}_i\mathbf{P} + \mathbf{v}_i$ and $\|\mathbf{v}_i\|_2 \leq \frac{1}{\sqrt{b}}\left\|(\mathbf{AP})_{tail(b)}\right\|_F$. The algorithm uses $\mathcal{O}\left(db\log^2 n\right)$ bits of space and succeeds with high probability.*

Proof : Correctness follows from Lemma 2.2 providing an accurate estimate of the norms of the heavy rows and Lemma 2.1 providing an accurate estimate of the Frobenius norm of $\mathbf{AP}$. The space complexity results from using a table with $\Theta(\log n)$ rows and b buckets in each row. Furthermore, each bucket consists of a vector of dimension d, whose entries are each represented using $\mathcal{O}(\log n)$ bits. Thus, the algorithm requires $\mathcal{O}\left(db\log^2 n\right)$ bits of space. □

2.2 $L_{2,2}$ Sampling Algorithm

In this section, we give an algorithm for $L_{2,2}$ sampling that will ultimately be used to simulate adaptive sampling on turnstile streams.

2.2.1 Algorithm Description

Given subroutines that estimate $\|\mathbf{AP}\|_F$ and the heavy rows of $\mathbf{AP}$, we implement our $L_{2,2}$ sampler in Algorithm 3. Our algorithm first takes each row $\mathbf{A}_i$ of matrix i and forms a row $\mathbf{B}_i = \frac{\mathbf{A}_i}{\sqrt{t_i}}$, where t_i is a scaling factor drawn uniformly at random from $[0, 1]$. Note that we have the following observation:

Observation 2.4 *For any value $\gamma > 0$,* $\mathbf{Pr}\left[\|\mathbf{B}_i\mathbf{P}\|_2 \geq \gamma \|\mathbf{AP}\|_F\right] = \frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\gamma^2\|\mathbf{AP}\|_F^2}$.

Proof : Since $\mathbf{B}_i = \frac{\mathbf{A}_i}{\sqrt{t_i}}$ and t_i is drawn uniformly at random from $[0, 1]$, then we have

$$\begin{aligned}\mathbf{Pr}\left[\|\mathbf{B}_i\mathbf{P}\|_2 \geq \gamma \|\mathbf{AP}\|_F\right] &= \mathbf{Pr}\left[\frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{t_i} \geq \gamma^2 \|\mathbf{AP}\|_F^2\right] \\ &= \mathbf{Pr}\left[t_i \leq \frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\gamma^2 \|\mathbf{AP}\|_F^2}\right] = \frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\gamma^2 \|\mathbf{AP}\|_F^2}.\end{aligned}$$

□

Intuitively, Observation 2.4 claims that by setting $T \propto \|\mathbf{AP}\|_F$, we can identify a row of $\mathbf{BP}$ whose norm exceeds T to effectively $L_{2,2}$ sample a row of $\mathbf{AP}$. For technical reasons, we set $T = \sqrt{\frac{C \log n}{\epsilon}} \|\mathbf{AP}\|_F$. Our algorithm then uses CountSketch-M to find heavy rows of $\mathbf{BP}$ and AMS-M to give an estimate $\widehat{F}$ of $\|\mathbf{AP}\|_F$ to determine whether there exists a row of $\mathbf{BP}$ whose norm exceeds T.

Our algorithm also uses CountSketch-M and a separate instance of AMS-M to compute $\widehat{S}$, which estimates the error in the tail of $\mathbf{BP}$ and also indicates how accurate CountSketch-M is. If $\widehat{S}$ is large, then our estimations for each row of $\mathbf{BP}$ from CountSketch-M may be inaccurate, so our algorithm must abort. Otherwise, if $\widehat{S}$ is sufficiently small, then our estimations for each row of $\mathbf{BP}$ is somewhat accurate. Thus if the row of $\mathbf{BP}$ with the largest norm exceeds $\sqrt{\frac{C \log n}{\epsilon}}\widehat{F}$, which is our estimation for T, then we output that particular row rescaled by $\sqrt{t_i}$ to recover the (noisy) original row of $\mathbf{AP}$.

2.2.2 Analysis

Conditioning on only a single row $\mathbf{B}_i\mathbf{P}$ satisfying $\|\mathbf{B}_i\mathbf{P}\|_2 \geq T = \sqrt{\frac{C \log n}{\epsilon}} \|\mathbf{AP}\|_F$, we could immediately identify this row if we had access to all rows of $\mathbf{BP}$, as well as $\|\mathbf{AP}\|_F$, but this requires too much space. Instead, we use CountSketch-M to find the heavy rows of $\mathbf{BP}$ and compare their norms to an estimate of T. However, if the error caused by CountSketch-M is high due to the randomness of the data structure, then the estimations of the row norms may be inaccurate and so our algorithm should abort. Our algorithm uses an estimator $\widehat{S}$ to compute the tail of $\mathbf{BP}$, which bounds the error caused by CountSketch-M. We first show that the event of our algorithm aborts because the tail estimator $\widehat{S}$ is too large has small probability and is independent of the index i and the value of t_i.

Lemma 2.5 *For each $j \in [n]$ and value of t_j,*

$$\mathbf{Pr}\left[\widehat{S} > \sqrt{\frac{C \log n}{\epsilon}}\widehat{F} \,\middle|\, t_j\right] = \mathcal{O}(\epsilon) + \frac{1}{\text{poly}(n)}.$$

Algorithm 3 Single $L_{2,2}$ Sampler

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives as a stream, matrix $\mathbf{P} \in \mathbb{R}^{d \times d}$ that arrives after the stream, approximation parameter $\epsilon > 0$.

Output: Noisy row $\mathbf{r}$ of $\mathbf{AP}$ sampled roughly proportional to the squared row norms of $\mathbf{AP}$.

Pre-processing Stage:
$b \leftarrow \Omega\left(\frac{1}{\epsilon^2}\right)$, $r \leftarrow \Theta(\log n)$ with sufficiently large constants
For $i \in [n]$, generate independent scaling factors $t_i \in [0, 1]$ uniformly at random.
Let $\mathbf{B}$ be the matrix consisting of rows $\mathbf{B}_i = \frac{1}{\sqrt{t_i}}\mathbf{A}_i$.
Let AMS-M_1 and AMS-M_2 track the Frobenius norms of $\mathbf{AP}$ and $\mathbf{BP}$, respectively.
Let CountSketch-M be an $r \times b$ table, where each entry is a vector in $\mathbb{R}^d$, to find the heavy hitters of $\mathbf{B}$.
Streaming Stage:
for each row $\mathbf{A}_i$ **do** ▷Presented in row-arrival model but also works for turnstile streams
    Update CountSketch-M with $\mathbf{B}_i = \frac{1}{\sqrt{t_i}}\mathbf{A}_i$.
    Update linear sketch AMS-M_1 with $\mathbf{A}_i$.
    Update linear sketch AMS-M_2 with $\mathbf{B}_i = \frac{1}{\sqrt{t_i}}\mathbf{A}_i$.
Processing P Stage:
After the stream, obtain matrix $\mathbf{P}$.
Multiply each vector $\mathbf{v}$ in each entry of the CountSketch-M table by $\mathbf{P}$: $\mathbf{v} \leftarrow \mathbf{vP}$.
Multiply each vector $\mathbf{v}$ in AMS-M_1 by $\mathbf{P}$: $\mathbf{v} \leftarrow \mathbf{vP}$.
Multiply each vector $\mathbf{v}$ in AMS-M_2 by $\mathbf{P}$: $\mathbf{v} \leftarrow \mathbf{vP}$.
Extraction Stage:
Use AMS-M_1 to compute $\widehat{F}$ with $\|\mathbf{AP}\|_F \leq \widehat{F} \leq 2\|\mathbf{AP}\|_F$.
Extract the $\frac{2}{\epsilon^2}$ (noisy) rows of $\mathbf{BP}$ that are estimated by CountSketch-M to have the largest norms.
Let $\mathbf{M} \in \mathbb{R}^{d \times d}$ be the matrix with $\frac{2}{\epsilon^2}$-nonzero rows consisting of these top (noisy) rows.
Use AMS-M_2 to compute $\widehat{S}$ with $\|\mathbf{BP} - \mathbf{M}\|_F \leq \widehat{S} \leq 2\|\mathbf{BP} - \mathbf{M}\|_F$.
Let $\mathbf{r}_i$ be the (noisy) row of $\mathbf{AP}$ in CountSketch-M with the largest norm.
Let $C > 0$ be some large constant so that the probability of failure is $\mathcal{O}\left(\frac{1}{n^{C/2}}\right)$.
if $\widehat{S} > \sqrt{\frac{C \log n}{\epsilon}}\widehat{F}$ or $\|\mathbf{r}_i\|_2 < \sqrt{\frac{C \log n}{\epsilon}}\widehat{F}$ **then**
    return FAIL.
else
    return $\mathbf{r} = \sqrt{t_i}\mathbf{r}_i$.

Proof : Let $\mathcal{E}_1$ be the event that:

(1) $\|\mathbf{AP}\|_F \leq \widehat{F} \leq 2\|\mathbf{AP}\|_F$

(2) $\|\mathbf{BP} - \mathbf{M}\|_F \leq \widehat{S} \leq 2\|\mathbf{BP} - \mathbf{M}\|_F$

(3) $\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \|\mathbf{BP} - \mathbf{M}\|_F \leq 2\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$

Fix an index $j \in [n]$ and let t_j be any fixed value $t \in [0, 1]$. Since the goal of the lemma is to show

that the probability of the failure event is independent of our choice of j and t, these variables will actually not appear in the remainder of the proof.

Observe that $\mathcal{E}_1$ holds with high probability by Lemma 2.3 and Lemma 2.1. Conditioned on $\mathcal{E}_1$, it suffices to bound the probability that $4\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F > \sqrt{\frac{C\log n}{\epsilon}}\left\|\mathbf{AP}\right\|_F$.

Let $U = \sqrt{\epsilon}\left\|\mathbf{AP}\right\|_F$ and define the indicator variable for whether row $\mathbf{B}_i\mathbf{P}$ is heavy. That is, we define $y_i = 1$ if $\left\|\mathbf{B}_i\mathbf{P}\right\|_2 > U$ and $y_i = 0$ otherwise. Define z_i as the scaled indicator variable $z_i = \frac{1}{U^2}\left\|\mathbf{B}_i\mathbf{P}\right\|_2^2(1-y_i)$ so that $z_i \in [0,1]$ represents a scaled contribution of the small rows. Define $Y = \sum_{i\neq j} y_i$ to be the total number of heavy rows, $Z = \sum_{i\neq j} z_i$ to be the total scaled contribution of the small rows, and $\mathbf{W} \in \mathbb{R}^{n\times d}$ to be the matrix of the heavy rows, i.e., $\mathbf{W}_i = \mathbf{B}_i\mathbf{P}$ if $y_i = 1$ and $\mathbf{W}_i$ is the row of all zeros otherwise. Observe that $\mathbf{W}$ contains at most $Y+1$ nonzero rows and $U^2Z = \left\|\mathbf{BP}-\mathbf{W}\right\|_F^2$. Moreover, $\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq U\sqrt{Z}$ unless $Y \geq \frac{2}{\epsilon^2}$. Hence if $\mathcal{E}_2$ denotes the event that $Y \geq \frac{2}{\epsilon^2}$ and $\mathcal{E}_3$ denotes the event that $Z \geq \frac{C\log n}{16U^2\epsilon}\left\|\mathbf{AP}\right\|_F^2$, then it suffices to bound the probability of the events $\mathcal{E}_2$ and $\mathcal{E}_3$ by $\mathcal{O}(\epsilon)$, since $\neg\mathcal{E}_2 \wedge \neg\mathcal{E}_3$ implies $4\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \sqrt{\frac{C\log n}{\epsilon}}\left\|\mathbf{AP}\right\|_F$. In other words, the probability of failure due to the tail estimator is small if the number of heavy rows is small ($\neg\mathcal{E}_2$) and the total contribution of the small rows is small ($\neg\mathcal{E}_3$).

By Observation 2.4, $\mathbb{E}[y_i] = \frac{\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2}{U^2}$ and so $\mathbb{E}[Y] \leq \frac{1}{\epsilon}$ by linearity of expectation since $U = \sqrt{\epsilon}\left\|\mathbf{AP}\right\|_F$. Hence $\mathbf{Pr}[\mathcal{E}_2] = \mathcal{O}(\epsilon)$ by Markov's inequality for sufficiently small ϵ.

To bound $\mathbf{Pr}[\mathcal{E}_3]$, observe that $z_i > 0$ only for $\left\|\mathbf{B}_i\mathbf{P}\right\|_2 \leq U$ or equivalently, $t_i \geq \frac{\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2}{\epsilon\left\|\mathbf{AP}\right\|_F^2}$. For sufficiently small ϵ, $\frac{\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2}{\epsilon\left\|\mathbf{AP}\right\|_F^2} \geq \frac{\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2}{\left\|\mathbf{AP}\right\|_F^2}$. Thus,

$$\mathbb{E}[z_i] \leq \int_{\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2/\left\|\mathbf{AP}\right\|_F^2}^1 z_i\,dt_i = \int_{\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2/\left\|\mathbf{AP}\right\|_F^2}^1 \frac{1}{t_i}\frac{1}{U^2}\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2\,dt_i.$$

Let $\mathcal{E}_4$ be the event that $t_i \geq \frac{1}{n^{C/2}}$ for all $i \in [n]$, so that $\mathbf{Pr}[\mathcal{E}_4] \geq 1 - \frac{1}{n^{C/2-1}}$. Conditioned on $\mathcal{E}_4$, we have

$$\mathbb{E}[z_i \mid \mathcal{E}_4] \leq \frac{1}{1-\frac{1}{n^{C/2}}}\int_{\frac{1}{n^{C/2}}}^1 \frac{1}{t_i}\frac{1}{U^2}\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2\,dt_i \leq \frac{C\log n}{U^2}\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2.$$

Hence, we have $\mathbb{E}[Z \mid \mathcal{E}_4] \leq \frac{C\log n}{\epsilon}$ and so the probability that $Z > \frac{C\log n}{16U^2\epsilon}\left\|\mathbf{AP}\right\|_F^2 = \frac{C\log n}{16\epsilon^2}$ for sufficiently small ϵ is bounded by $\mathcal{O}(\epsilon)$ by Markov's inequality. Since the events $\neg\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4$ each occur with probability at most $\mathcal{O}(\epsilon) + \frac{1}{\text{poly}(n)}$, then the claim follows. □

The probability of sampling each row $i \in [n]$ will still be slightly distorted due to the noise from CountSketch-M, since we do not have exact values for the norm of each row. Similarly, if multiple rows exceed the threshold, we will output the row with the largest norm, which also alters the sampling probability of each row. We now show that these events only slightly perturb the probability of sampling each index i and moreover, the output row is a small noisy perturbation of the original row.

Lemma 2.6 *Conditioned on a fixed value of $\widehat{F}$, the probability that Algorithm 3 outputs (noisy) row i is $(1\pm\mathcal{O}(\epsilon))\frac{\epsilon}{C\log n}\frac{\left\|\mathbf{A}_i\mathbf{P}\right\|_2^2}{\widehat{F}^2} + \frac{1}{\text{poly}(n)}$.*

Proof : We first define the following set of events.

- Let $\mathcal{E}$ denote the event that $t_i < \frac{\epsilon}{C\log n}\frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\widehat{F}^2}$ so that the algorithm should ideally output index i and observe that $\mathbf{Pr}\left[\mathcal{E}\right] = \frac{\epsilon}{C\log n}\frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\widehat{F}^2}$ since $t_i \in [0,1]$ is selected uniformly at random.
- Let $\mathcal{E}_1$ denote the event that one of the data structures CountSketch-M, AMS-M_1, or AMS-M_2 fails. Note that $\mathcal{E}_1$ occurs with probability $\frac{1}{\mathrm{poly}(n)}$ by Lemma 2.3 and Lemma 2.1.
- Let $\mathcal{E}_2$ denote the event that $\widehat{S} > \sqrt{\frac{C\log n}{\epsilon}}\widehat{F}$. Conditioned on $\mathcal{E}$, the probability of $\mathcal{E}_2$ is $\mathcal{O}\left(\epsilon\right)$ by Lemma 2.5.
- Let $\mathcal{E}_3$ denote the event that some other row $\mathbf{B}_j\mathbf{P}$ also either exceeds the threshold or is close enough to the threshold, thus possibly preventing $\mathbf{B}_i\mathbf{P}$ from being reported. Specifically, $\mathcal{E}_3$ can only occur if some other row j satisfies $\|\mathbf{B}_j\mathbf{P}\|_2 \geq \sqrt{\frac{C\log n}{\epsilon}}\widehat{F} - \sqrt{C\epsilon\log n}\widehat{F}$. Since $\mathbf{B}_j\mathbf{P} = \frac{1}{\sqrt{t_j}}\mathbf{A}_j\mathbf{P}$ and t_j is chosen uniformly at random from $[0,1]$, then row j exceeds this threshold with probability at most $\mathcal{O}\left(\frac{\epsilon}{C\log n}\frac{\|\mathbf{A}_j\mathbf{P}\|_2^2}{\widehat{F}^2}\right)$ by Observation 2.4. Taking a union bound over all n rows, the probability of $\mathcal{E}_3$ is $\mathcal{O}\left(\frac{\epsilon}{\log n}\right)$.
- Let $\mathcal{E}_4$ denote the event that $\|\mathbf{B}_i\mathbf{P}\|$ exceeds the threshold but is not reported due to noise in the CountSketch data structure, i.e., $\|\mathbf{r}_i\|_2 < \sqrt{\frac{C\log n}{\epsilon}}\widehat{F}$. We now analyze the probability of $\mathcal{E}_4$. Conditioning on $\neg\mathcal{E}_2$, we have $\widehat{S} \leq \sqrt{\frac{C\log n}{\epsilon}}\widehat{F}$. Conditioning on $\neg\mathcal{E}_1$, then $\|\mathbf{BP}-\mathbf{M}\|_F \leq \widehat{S}$. Thus by Lemma 2.2 (or Lemma 2.3),

$$\left|\|\mathbf{B}_i\mathbf{P}\|_2 - \left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2\right| \leq \epsilon\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \epsilon\|\mathbf{BP}-\mathbf{M}\|_F \leq \epsilon\widehat{S} \leq \sqrt{C\epsilon\log n}\widehat{F}.$$

  Hence, $\mathcal{E}_4$ can only occur for

$$\sqrt{\frac{C\log n}{\epsilon}}\widehat{F} \leq \|\mathbf{B}_i\mathbf{P}\|_2 \leq \sqrt{\frac{C\log n}{\epsilon}}\widehat{F} + \sqrt{C\epsilon\log n}\widehat{F},$$

  which occurs with probability at most $\mathcal{O}\left(\frac{\epsilon^2}{C\log n}\frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\widehat{F}^2}\right)$ over the choice of t_i.

Conditioning on $\mathcal{E}$, the sampler should return $\mathbf{A}_i\mathbf{P}$ but may fail to do so because of any of the events $\mathcal{E}_1$, $\mathcal{E}_2$, $\mathcal{E}_3$, or $\mathcal{E}_4$. Putting things together, $\mathcal{E}_1$ occurs with probability $\frac{1}{\mathrm{poly}(n)}$. Conditioning on $\mathcal{E}$, $\mathcal{E}_2$ and $\mathcal{E}_3$ each occur with probability $\mathcal{O}\left(\epsilon\right)$, so the probability of $\mathcal{E}$ and at least one of $\mathcal{E}_2$ or $\mathcal{E}_3$ occurring is $\mathcal{O}\left(\frac{\epsilon^2}{C\log n}\frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\widehat{F}^2}\right)$, which is also the probability of $\mathcal{E}_4$. Thus the sampling probability of each $\mathbf{A}_i\mathbf{P}$ follows.

Finally, we emphasize that for the index i selected, it holds by Lemma 2.2 that

$$\left|\|\mathbf{B}_i\mathbf{P}\|_2 - \left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2\right| \leq \sqrt{C\epsilon\log n}\widehat{F}$$

and $\left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2 \geq \sqrt{\frac{C\log n}{\epsilon}}\widehat{F}$. Thus, $\left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2$ is a $(1+\epsilon)$ approximation to $\|\mathbf{B}_i\mathbf{P}\|_2$ and similarly, $\sqrt{t_i}\left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2$ has norm $(1+\epsilon)$ within that of $\|\mathbf{A}_i\mathbf{P}\|_2$. □

Since each row is sampled with roughly the desired probability, we now analyze the space complexity of the resulting $L_{2,2}$ sampler.

Theorem 2.7 *Given $\epsilon > 0$, there exists a one-pass streaming algorithm that takes rows of a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ as a turnstile stream and a matrix $\mathbf{P} \in \mathbb{R}^{d \times d}$ after the stream, and outputs (noisy) row i of $\mathbf{AP}$ with probability $(1 \pm \mathcal{O}(\epsilon)) \frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\|\mathbf{AP}\|_F^2} + \frac{1}{\text{poly}(n)}$. The algorithm uses $\mathcal{O}\left(\frac{d}{\epsilon^3} \log^3 n \log \frac{1}{\delta}\right)$ space to succeed with probability $1 - \delta$.*

Proof : By Lemma 2.6 and the fact that $\|\mathbf{AP}\|_F \leq \widehat{F} \leq 2\|\mathbf{AP}\|_F$ with high probability by Lemma 2.1, each row $\mathbf{A}_i\mathbf{P}$ is output with probability $(1+\epsilon)\frac{\|\mathbf{A}_i\mathbf{P}\|_2^2}{\|\mathbf{AP}\|_F^2} + \frac{1}{\text{poly}(n)}$, conditioned on the sampler outputting *some* index rather than aborting. Recall that the sampler outputs index if the tail estimator $\widehat{S}$ is small and the estimated norm of some row exceeds the threshold. Lemma 2.5 shows that the tail estimator is small only with probability $\mathcal{O}(\epsilon)$ while a straightforward computation shows that the probability that the estimated norm of some row exceeding the threshold is $\Theta\left(\frac{\epsilon}{\log n}\right)$. Thus the sampler can be repeated $\mathcal{O}\left(\frac{1}{\epsilon} \log n \log \frac{1}{\delta}\right)$ times to obtain probability of success at least $1 - \delta$. By Lemma 2.1, each instance of AMS-M uses $\mathcal{O}\left(\frac{d}{\epsilon^2}\log^2 n\right)$ space. Moreover, each sampler uses a $\mathcal{O}\left(\frac{1}{\epsilon^2}\right) \times \mathcal{O}(\log n)$ table, and each entry in the table is a vector of d integers, the total space complexity is $\mathcal{O}\left(\frac{d}{\epsilon^3} \log^3 n \log \frac{1}{\delta}\right)$. □

Generation of Uniform Random Variables. First observe that with high probability, each of the uniform random variables t_i are least $\frac{1}{\text{poly}(ndmM)} = \frac{1}{\text{poly}(n)}$ precision, where $m = \text{poly}(n)$ is the length of the stream and $M = \text{poly}(n)$ is the largest change an update can induce in the matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$. Then we claim it suffices to generate the uniform random variables t_i up to $\mathcal{O}(\log n)$ bits of precision, for a sufficiently large constant. Indeed note that the truncation perturbs the values of the uniform random variables by an additive $\frac{1}{\text{poly}(n)}$ value, which induces a $\mathcal{O}\left(\frac{1}{\text{poly}(n)}\right)$ additive error for each CountSketch bucket. Therefore, we can incorporate the additive error induced by truncating the uniform random variables at $\mathcal{O}(\log n)$ bits of precision into the $\mathcal{O}\left(\frac{1}{\text{poly}(n)}\right)$ additive error of the $L_{2,2}$ sampler.

We say a family $\mathcal{H} = \{h : [n] \to [m]\}$ is *δ-approximate* if for all $r \leq n$, possible inputs $i_1, \ldots, i_r \in [n]$ and possible outputs $o_1, \ldots, o_1 \in [m]$,

$$\Pr_{h \in H}[h(i_1) = o_1 \wedge \ldots \wedge h(i_r) = o_r] = \frac{1}{m^r} + \delta.$$

If $\delta = 0$ for all $r \leq k$, the function is *k-wise independent.*

We observe that $\mathcal{O}(1)$-wise independent random variables t_i would suffice for justifying the low-probability failure events in Lemma 2.5 through Chebyshev's inequality. Recall that k-wise independent random variables can be generated from a polynomial of degree k over a field of size $\mathcal{O}(\text{poly}(n))$ [WC81], which can be stored using $\mathcal{O}(k \log n)$ space [WC81]. Thus for the purposes of our $L_{2,2}$ sampler, using $\mathcal{O}(1)$-wise independent random variables t_i instead of fully independent random variables gives the exact guarantees as Theorem 2.7.

However, Section 3 requires Chernoff bounds to analyze the size of specific sets, which will not naïvely work with $\mathcal{O}(1)$-wise independent random variables. Instead, we can apply the limited independence Chernoff-Hoeffding bounds in [SSS95] using $\mathcal{O}(\log n)$-wise independent random variables, thus limiting the probability of the failure events by $\mathcal{O}\left(\frac{1}{\text{poly}(n)}\right)$. Moreover, δ-approximate

k-wise hash functions can be generated using $\mathcal{O}\left(k + \log n + \log \frac{1}{\delta}\right)$ bits, e.g., by composing the generators of [ABI86] and [NN93]. Thus for $\delta = \frac{1}{\text{poly}(n)}$, the error can again be absorbed into the $\mathcal{O}\left(\frac{1}{\text{poly}(n)}\right)$ additive error of the $L_{2,2}$ sampler, while the family of hash functions requires $\mathcal{O}(\log n)$ bits to store.

3 Noisy Adaptive Squared Distance Sampling

Given a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives in a data stream, either turnstile or row-arrival, we want to simulate k rounds of adaptive sampling. That is, in the first round we want to sample some row $\mathbf{r}_1$ of $\mathbf{A}$, such that each row $\mathbf{A}_i$ is selected with probability proportional to its squared row norm $\|\mathbf{A}_i\|_2^2$. Once rows $\mathbf{r}_1, \ldots, \mathbf{r}_{j-1}$ are selected, then the j^{th} round of adaptive sampling samples each row $\mathbf{A}_i$ with probability proportional to the squared row norm of the orthogonal component to $\mathbf{R}_{j-1}$, $\left\|\mathbf{A}_i(\mathbb{I} - \mathbf{R}_{j-1}^{\dagger}\mathbf{R}_{j-1})\right\|_2^2$, where for each $j \leq k$, $\mathbf{R}_j = \mathbf{r}_1 \circ \ldots \circ \mathbf{r}_j$.

Observe that if only a single round of adaptive sampling were required, the problem would reduce to $L_{2,2}$ sampling, which we can perform in a stream through Algorithm 3. In fact, the post-processing stage of Algorithm 3 would not be necessary since the post-processing matrix would be $\mathbf{P} = \mathbb{I}$, which is the identity matrix. Moreover, the sketch of $\mathbf{A}$ of Algorithm 3 is oblivious to the choice of the post-processing matrix $\mathbf{P}$, so we would like to repeat this k times by creating k separate instances of the $L_{2,2}$ sampler of Algorithm 3 and for the j^{th} instance, multiply by the post-processing matrix $\mathbf{P}_j = \mathbb{I} - \mathbf{R}_{j-1}^{\dagger}\mathbf{R}_{j-1}$. Unfortunately, if $f(j)$ is the index of the row of $\mathbf{AP}_j$ that is selected in the j^{th} round, the row $\mathbf{r}_j$ that the $L_{2,2}$ sampler outputs is not $\mathbf{A}_{f(j)}\mathbf{P}_j$ but rather a noisy perturbation of it, which means in future rounds we are not sampling with respect to a subspace containing $\mathbf{A}_{f(j)}\mathbf{P}_j$ but rather a subspace containing $\mathbf{r}_j$. This is particularly a problem if $\mathbf{r}_j$ is parallel to another row $\mathbf{A}_i$ that is not contained in the subspace of $\mathbf{A}_{f(j)}\mathbf{P}_j$, then in future rounds the probability of sampling $\mathbf{A}_i$ is zero, when it should in fact be nonzero. Although the above example shows that the noisy perturbation does not preserve relative sampling probabilities for each row, we show that the perturbations give a good additive approximation to the sampling probabilities. That is, we bound the total variation distance between sampling with respect to the true rows of $\mathbf{A}$ and sampling with respect to the noisy rows of $\mathbf{A}$. We give our algorithm in full in Algorithm 4.

For the purpose of the analysis, we first show that if the $L_{2,2}$ sampler outputs row $\mathbf{r}_1$ that is a noisy perturbation of row $\mathbf{A}_{f(1)}\mathbf{P}$, then not only can we bound $\left\|\mathbf{A}_{f(1)}\mathbf{P} - \mathbf{r}_1\right\|_2$ as in Lemma 2.2, but also we can bound the norm of the component of $\mathbf{r}_1$ orthogonal to $\mathbf{A}_{f(1)}\mathbf{P}$. This is significant because future rounds of sampling will focus on the norms of the orthogonal components for the sampling probabilities.

Lemma 3.1 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and a matrix $\mathbf{P} \in \mathbb{R}^{d \times d}$, as defined in Line 8 and round $i \leq k$, of Algorithm 4, suppose index $j \in [n]$ is sampled (in round i). Then with high probability, the sampled (noisy) row $\mathbf{r}_i$ satisfies $\mathbf{r}_i = \mathbf{A}_j\mathbf{P} + \mathbf{v}_e$ with*

$$\|\mathbf{v}_e\mathbf{Q}\|_2 \leq \frac{\epsilon\sqrt{\epsilon}\,\|\mathbf{APQ}\|_F}{\sqrt{C \log n}\,\|\mathbf{AP}\|_F}\,\|\mathbf{A}_j\mathbf{P}\|_2\,,$$

for any projection matrix $\mathbf{Q} \in \mathbb{R}^{d \times d}$. Hence, $\mathbf{v}_e$ is orthogonal to each noisy row $\mathbf{r}_y$, where $y \in [i-1]$.

Algorithm 4 Noisy Adaptive Sampler

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives as a stream $\mathbf{A}_1, \ldots, \mathbf{A}_n \in \mathbb{R}^d$, parameter k for number of rows to be sampled, constant parameter $\epsilon > 0$.

Output: k Noisy and projected rows of $\mathbf{A}$.

1: Create instances $\mathcal{A}_1, \ldots, \mathcal{A}_k$ of the $L_{2,2}$ sampler of Algorithm 3 where the number of buckets $b = \Theta\left(\frac{\log^2 n}{\epsilon^2}\right)$ is sufficiently large.
2: Let $\mathbf{M}$ be empty $0 \times d$ matrix.
3: **Streaming Stage:**
4: **for** each row $\mathbf{A}_i$ **do**
5: Update each sketch $\mathcal{A}_1, \ldots, \mathcal{A}_k$
6: **Post-processing Stage:**
7: **for** $j = 1$ to $j = k$ **do**
8: Post-processing matrix $\mathbf{P} \leftarrow \mathbb{I} - \mathbf{M}^\dagger\mathbf{M}$.
9: Update $\mathcal{A}_j$ with post-processing matrix $\mathbf{P}$.
10: Let $\mathbf{r}_j$ be the noisy row output by $\mathcal{A}_j$.
11: Append $\mathbf{r}_j$ to $\mathbf{M}$: $\mathbf{M} \leftarrow \mathbf{M} \circ \mathbf{r}_j$.
12: **return** $\mathbf{M}$.

Proof : Let $\mathbf{Q} \in \mathbb{R}^{d\times d}$ be a projection matrix. For each $x \in [n]$, let $\mathbf{B}_x = \frac{\mathbf{A}_x}{\sqrt{t_x}}$ be the rescaled row of $\mathbf{A}_x$. Let $\mathbf{E}$ be the noise in the bucket corresponding to the selected row j, so that the output vector is $\mathbf{A}_j + \sqrt{t_j}\mathbf{E}$ and the noise is $\mathbf{v}_e = \sqrt{t_j}\mathbf{E}$. Note that $\mathbf{E}$ is a linear combination of rows of $\mathbf{AP}$ and thus $\mathbf{E}$ is orthogonal to all previous noisy rows $\mathbf{r}_y$ with $y \in [i-1]$. Let $\mathbf{B} \in \mathbb{R}^{n\times d}$ be the rescaled matrix of $\mathbf{A}$ so that row x of $\mathbf{B}$ is $\mathbf{B}_x$ for $x \in [n]$. Recall that $t_x \in [0, 1]$ is selected uniformly at random for each $x \in [n]$, so that for each integer $c \geq 0$,

$$\mathbf{Pr}\left[\frac{\|\mathbf{A}_x\mathbf{PQ}\|_2^2}{t_x} \geq \frac{\|\mathbf{APQ}\|_F^2}{2^c}\right] \leq \frac{2^c\,\|\mathbf{A}_x\mathbf{PQ}\|_2^2}{\|\mathbf{APQ}\|_F^2},$$

e.g., by Observation 2.4. Since $\mathbf{B}_x = \frac{\mathbf{A}_x}{\sqrt{t_x}}$, then by linearity of expectation over $x \in [n]$, we can bound the expected size of each of the disjoint level sets $S_c := \left\{x \in [n] \,:\, \frac{\|\mathbf{APQ}\|_F^2}{2^{c-1}} > \|\mathbf{B}_x\mathbf{PQ}\|_2^2 \geq \frac{\|\mathbf{APQ}\|_F^2}{2^c}\right\}$ by $\mathbb{E}\left[|S_c|\right] \leq \min(2^c, n)$ for each c. From the independence of the scaling factors t_x, then the Chernoff bound implies that

$$\mathbf{Pr}\left[|S_c| \leq \min(2^{c+\beta}\log n, n)\right] \geq 1 - \frac{1}{\mathrm{poly}(n)},$$

for sufficiently large constant β. Thus the Frobenius norm of $\mathbf{BPQ}$ can be roughly upper bounded by the Frobenius norm of $\mathbf{APQ}$ by a union bound over level sets S_c for $c \leq \log n$ and upper bounding the norms of each of the rows in level sets S_c with $c > \log n$ by $\frac{\|\mathbf{APQ}\|_F^2}{n}$ and thus the total mass of the level sets S_c with $c > \log n$ by $\|\mathbf{APQ}\|_F^2$. That is,

$$\mathbf{Pr}\left[\|\mathbf{BPQ}\|_F^2 \geq 2^\beta \log^2 n\,\|\mathbf{APQ}\|_F^2\right] \geq 1 - \frac{1}{\mathrm{poly}(n)}.$$

Hence the total mass $\|\mathbf{BPQ}\|_F^2$ distributed across the CountSketch table is $\mathcal{O}\left(\log^2 n\,\|\mathbf{APQ}\|_F^2\right)$ with high probability.

By Lemma 2.3 and hashing rows of $\mathbf{BPQ}$ to a CountSketch table with $b = \Theta\left(\frac{\log^2 n}{\epsilon^2}\right)$ buckets with sufficiently large constant, the bucket corresponding to $\mathbf{A}_j$ has mass at most $\epsilon \|\mathbf{APQ}\|_F$ when projected onto $\mathbf{Q}$. That is, $\|\mathbf{EQ}\|_2 \leq \epsilon \|\mathbf{APQ}\|_F$. This can also be seen from the fact that CountSketch-M is a linear sketch and considering the error in a certain subspace $\mathbf{Q}$ is equivalent to right multiplication by $\mathbf{Q}$.

Since row j was selected, it must have been true that $\|\mathbf{B}_j\mathbf{P}\|_2 \geq \sqrt{\frac{C\log n}{\epsilon}} \|\mathbf{AP}\|_F$. Because $\mathbf{B}_j = \frac{\mathbf{A}_j}{\sqrt{t_j}}$, then $\sqrt{t_j} \leq \frac{\sqrt{\epsilon}\|\mathbf{A}_j\mathbf{P}\|_2}{\sqrt{C\log n}\|\mathbf{AP}\|_F}$. Therefore,

$$\left\|\sqrt{t_j}\mathbf{EQ}\right\|_2 \leq \frac{\epsilon\sqrt{\epsilon}\,\|\mathbf{APQ}\|_F}{\sqrt{C\log n}\,\|\mathbf{AP}\|_F}\,\|\mathbf{A}_j\mathbf{P}\|_2\,.$$

In particular, since $\mathbf{v}_e\mathbf{Q} = \sqrt{t_j}\mathbf{EQ}$, the above expression also bounds the Euclidean norm of $\mathbf{v}_e\mathbf{Q}$. Intuitively, not only is the overall noise of $\mathbf{AP}$ well-partitioned among the buckets of CountSketch, but the noise in each direction $\mathbf{APQ}$ must also be well-partitioned among the buckets of CountSketch with high probability. □

Recall that our sampler only returns noisy rows $\mathbf{r}_i$, rather than the true rows of $\mathbf{AP}_i$, where $\mathbf{P}_i$ is any post-processing matrix. This is problematic for multiple rounds of sampling, since $\mathbf{r}_i$ is then used to form the next post-processing matrix $\mathbf{P}_{i+1}$, rather than the true row. We next show that the total variation distance has not been drastically altered by sampling with respect to the noisy rows rather than the true rows.

The main idea is that because the noise in each direction is proportional to the total mass in the subspace by Lemma 3.1, we can bound the total perturbation in the squared norms of each row of $\mathbf{AP}_i$. We first argue that if we obtain a noisy row in the first sampling iteration but then we obtain the true rows in the subsequent iterations, then the total variation distance between the resulting probability distribution of sampling each row is close to the ideal probability distribution of sampling each row if we had obtained the true rows over all iterations. It then follows from triangle inequality that the actual sampling distribution induced by obtaining noisy rows in each round is close to the ideal sampling distribution if we had obtained the true rows.

To bound the perturbation in the sampling probability of each row, we require a change of basis matrix from a representation of vectors in terms of the true rows of $\mathbf{A}$ to a representation of vectors in terms of the noisy rows of $\mathbf{A}$. This change of basis matrix crucially must be close to the identity matrix, in order to preserve the perturbation in the squared norms.

Lemma 3.2 *Let $f(1)$ be the index of a noisy row $\mathbf{r}_1$ sampled in the first iteration of Algorithm 4. Let $\mathcal{P}_1$ be a process that projects away from $\mathbf{A}_{f(1)}$ and iteratively selects $k-1$ additional rows of $\mathbf{A}$ through adaptive sampling (with $p=2$). Let $\mathcal{P}_2$ be a process that projects away from $\mathbf{r}_1$ and iteratively selects $k-1$ additional rows of $\mathbf{A}$ through adaptive sampling (with $p=2$). Then for $\epsilon < \frac{1}{d}$, the total variation distance between the distributions of k indices output by $\mathcal{P}_1$ and $\mathcal{P}_2$ is $\mathcal{O}(k\epsilon)$.*

Proof : Suppose $\mathcal{P}_1$ sequentially samples rows $\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(k)}$. For each $t \in [k]$, let $\mathbf{T}_t = \mathbf{A}_{f(1)} \circ \ldots \circ \mathbf{A}_{f(t)}$ and $\mathbf{Z}_t = \mathbb{I} - \mathbf{T}_t^\dagger \mathbf{T}_t$ and $\mathbf{R}_t = \mathbf{r}_1 \circ \mathbf{A}_{f(2)} \ldots \circ \mathbf{A}_{f(t)}$ and $\mathbf{Y}_t = \mathbb{I} - \mathbf{R}_t^\dagger \mathbf{R}_t$. We assume for the sake of presentation that $\mathbf{A}$ is a full-rank matrix, i.e. $\text{rank}(\mathbf{A}) = d$ and prove the claim by induction.

Base case. For $t = 2$, we first must show that the sampling distributions induced by $\mathbf{r}_1$ and $\mathbf{A}_{f(1)}$ are similar. Let $U = \{\mathbf{u}_1, \ldots, \mathbf{u}_d\}$ be the orthonormal basis for the row span of $\mathbf{A}$ so that $\mathbf{u}_1 = \frac{\mathbf{A}_{f(1)}}{\|\mathbf{A}_{f(1)}\|_2}$ points in the direction of $\mathbf{A}_{f(1)}$. Similarly, let $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_d\}$ be an orthonormal basis for the row span of $\mathbf{A}$ obtained by applying the Gram-Schmidt process to the set $\{\mathbf{w}_1, \mathbf{u}_2, \ldots, \mathbf{u}_d\}$, where $\mathbf{w}_1 = \frac{\mathbf{r}_1}{\|\mathbf{r}_1\|_2}$. We argue that the change of basis matrix $\mathbf{B}$ from U to W must be close to the identity matrix.

First, observe that from Lemma 3.1, we have $\mathbf{r}_1 = \|\mathbf{A}_{f(1)}\|_2 \left(\mathbf{u}_1 + \sum_{i=1}^{d} (\pm\tau_i)\mathbf{u}_i\right)$, where $\tau_i \leq \frac{\epsilon\sqrt{\epsilon}}{\sqrt{C \log n}} \frac{\|\mathbf{A}\mathbf{P}_i\|_F}{\|\mathbf{A}\|_F}$ with high probability and $\mathbf{P}_i = \mathbf{u}_i^\dagger \mathbf{u}_i$ is the projection matrix onto $\mathbf{u}_i$. Thus by setting $\tau^2 = \sum_{i=1}^{d} \tau_i^2$, we can write the first row $\mathbf{b}_1$ of $\mathbf{B}$ so that the first entry is at least $1 - \tau$ and entry i is at most τ_i in magnitude.

Claim 3.3 *The first entry in row $j > 1$ is at most $2\tau_j$ in magnitude, entry j in row j is at least $1 - 3\tau_j$ in magnitude, and entry i in row j is at most $5\tau_i\tau_j$ in magnitude for $i < j$ with $i \neq 1$ and at most $2\tau_i\tau_j$ in magnitude for $i > j$.*

Proof : We first consider the base case $j = 2$ and determine $\mathbf{b}_2$ through the Gram-Schmidt process. For the elementary vector $\mathbf{e}_2 \in \mathbb{R}^d$, note that $|\langle \mathbf{e}_2, \mathbf{b}_1 \rangle| \, \|\mathbf{b}_1\|_2 \leq \tau_2$. Thus we have that entry i in $\mathbf{b}_2$ for $i \neq 2$ is at most $\frac{|\langle \mathbf{e}_i, \mathbf{b}_1 \rangle| \tau_2}{1 - \tau_2}$ in magnitude. Specifically for $i = 1$, the first entry in $\mathbf{b}_2$ is bounded by $2\tau_2$ in magnitude, while for $i > 2$, entry i in $\mathbf{b}_2$ is bounded by $2\tau_2\tau_i$ in magnitude. It follows that the second entry in $\mathbf{b}_2$ is at least $1 - 2\tau_2$.

We use similar reasoning to bound the entries in row j of $\mathbf{B}$. Note that for sufficiently small $\epsilon < \frac{1}{d}$, we have

$$\sum_{i=1}^{j-1} |\langle \mathbf{e}_j, \mathbf{b}_i \rangle| \, \|\mathbf{b}_i\|_2 \leq \tau_j + \sum_{i=2}^{j-1} 5\tau_j\tau_i \leq 2\tau_j.$$

Thus from the Gram-Schmidt process, entry i in $\mathbf{b}_j$ for $i \neq j$ is at most $\frac{1}{1-2\tau_j} \left| \sum_{\ell=1}^{j-1} \langle \mathbf{e}_i, \mathbf{b}_\ell \rangle \cdot \langle \mathbf{e}_j, \mathbf{b}_\ell \rangle \right|$, which is at most

$$\frac{1}{1 - 2\tau_j} \left(\tau_j + \sum_{\ell=2}^{j-1} 4\tau_j\tau_\ell^2 \right) \leq 2\tau_j$$

in magnitude for $i = 1$ for sufficiently small $\epsilon < \frac{1}{d}$ and at most

$$\frac{1}{1 - 2\tau_j} \left(\tau_i\tau_j + 2\tau_i\tau_j + \sum_{\ell=2, \ell\neq i}^{j-1} 10\tau_j\tau_i\tau_\ell^2 \right) \leq 5\tau_i\tau_j$$

in magnitude for $i < j$ with $i \neq 1$ and at most

$$\frac{1}{1 - 2\tau_j} \left(\tau_i\tau_j + \sum_{\ell=2}^{j-1} 4\tau_j\tau_i\tau_\ell^2 \right) \leq 2\tau_i\tau_j$$

in magnitude for $i > j$. Thus it follows that entry j in row j of $\mathbf{B}$ is at least $1 - 3\tau_j$, which completes the induction. □

Therefore by Claim 3.3, we have

$$\mathbf{B} = \begin{bmatrix} 1-\mathcal{O}(\tau) & \pm\mathcal{O}(\tau_2) & \pm\mathcal{O}(\tau_3) & \pm\mathcal{O}(\tau_4) & \dots & \pm\mathcal{O}(\tau_d) \\ \pm\mathcal{O}(\tau_2) & 1-\mathcal{O}(\tau_2) & \pm\mathcal{O}(\tau_2\tau_3) & \pm\mathcal{O}(\tau_2\tau_4) & \dots & \pm\mathcal{O}(\tau_2\tau_d) \\ \pm\mathcal{O}(\tau_3) & \pm\mathcal{O}(\tau_3\tau_2) & 1-\mathcal{O}(\tau_3) & \pm\mathcal{O}(\tau_3\tau_4) & \dots & \pm\mathcal{O}(\tau_3\tau_d) \\ \pm\mathcal{O}(\tau_4) & \pm\mathcal{O}(\tau_4\tau_2) & \pm\mathcal{O}(\tau_4\tau_3) & 1-\mathcal{O}(\tau_4) & \dots & \pm\mathcal{O}(\tau_4\tau_d) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \pm\mathcal{O}(\tau_d) & \pm\mathcal{O}(\tau_d\tau_2) & \pm\mathcal{O}(\tau_d\tau_3) & \pm\mathcal{O}(\tau_d\tau_4) & \dots & 1-\mathcal{O}(\tau_d) \end{bmatrix}. \tag{$\star$}$$

for $\epsilon < \frac{1}{d}$, where the $\mathcal{O}(\cdot)$ notation in ($\star$) hides a constant that is at most 5.

We can write each row $\mathbf{A}_s$ in terms of basis U as $\mathbf{A}_s = \sum_{i=1}^d \lambda_{s,i}\mathbf{u}_i$ and in terms of basis W as $\mathbf{A}_s = \sum_{i=1}^d \zeta_{s,i}\mathbf{w}_i$. Then since we project away from $\mathbf{A}_{f(1)}$, we should have sampled $\mathbf{A}_s$ with probability $\frac{\|\mathbf{A}_s\mathbf{Z}_{t-1}\|_2^2}{\|\mathbf{A}\mathbf{Z}_{t-1}\|_F^2} = \frac{\sum_{i=2}^d \lambda_{s,i}^2}{\sum_{j=1}^n\sum_{i=2}^d \lambda_{j,i}^2}$ in the second round but instead we sample it with probability $\frac{\|\mathbf{A}_s\mathbf{Y}_{t-1}\|_2^2}{\|\mathbf{A}\mathbf{Y}_{t-1}\|_F^2} = \frac{\sum_{i=2}^d \zeta_{s,i}^2}{\sum_{j=1}^n\sum_{i=2}^d \zeta_{j,i}^2}$. From the change of basis matrix $\mathbf{B}$, we can also write

$$\zeta_{s,i} = (1-\mathcal{O}(\tau_i))\lambda_{s,i} \pm \mathcal{O}(\tau_i)\lambda_{s,1} \pm \sum_{j\notin\{i,1\}} \mathcal{O}(\tau_i\tau_j)\lambda_{s,j},$$

for $i \geq 2$. Therefore we can bound the difference

$$\begin{aligned} |\zeta_{s,i}^2 - \lambda_{s,i}^2| &\leq 25\Big(\tau_i\lambda_{s,i}^2 + \tau_i^2\lambda_{s,1}^2 + \tau_i\lambda_{s,1}\lambda_{s,i} + \sum_{j,\ell\neq\{i,1\}} \tau_i^2\tau_j\tau_\ell\lambda_{s,j}\lambda_{s,\ell} \\ &\quad + \sum_{j\neq\{i,1\}} \tau_i\tau_j\lambda_{s,i}\lambda_{s,j} + \sum_{j\neq\{i,1\}} \tau_i^2\tau_j\lambda_{s,1}\lambda_{s,j}\Big) \\ &\leq 25\left(\tau_i\lambda_{s,i}^2 + d\tau_i^2\lambda_{s,1}^2 + \tau_i\lambda_{s,1}\lambda_{s,i} + 4\sum_{j=2}^d d\tau_j^2\lambda_{s,j}^2\right), \end{aligned}$$

where the last inequality follows from AM-GM and that all values of $\tau_i, \tau_j \leq \epsilon^{3/2}$ and thus $\tau < 1$ for $\epsilon < \frac{1}{d}$. We also have $\tau_i\lambda_{s,1}\lambda_{s,i} \leq \epsilon\lambda_{s,i}^2 + \frac{\tau_i^2}{\epsilon}\lambda_{s,1}^2$. Thus,

$$\begin{aligned} \left|\sum_{i=2}^d \zeta_{s,i}^2 - \sum_{i=2}^d \lambda_{s,i}^2\right| &\leq 25\sum_{i=2}^d \left[2\left(\epsilon + 4d^2\tau_i^2\right)\lambda_{s,i}^2 + \frac{2\tau_i^2}{\epsilon}\lambda_{s,1}^2\right] \\ &\leq 25\sum_{i=2}^d \left(6\epsilon\lambda_{s,i}^2 + \frac{2\tau_i^2}{\epsilon}\lambda_{s,1}^2\right), \end{aligned}$$

since $\tau_i^2 < \epsilon^3$, and $\epsilon < \frac{1}{d}$. Moreover, $\tau_i^2 \leq \frac{\epsilon^3}{C\log n}\frac{\sum_{a=1}^n \lambda_{a,i}^2}{\sum_{a=1}^n\sum_{b=1}^d \lambda_{a,b}^2}$, thus we have $\sum_{s=1}^n \frac{\tau_i^2}{\epsilon}\lambda_{s,1}^2 \leq \frac{\epsilon^2}{C\log n}\sum_{s=1}^n \lambda_{s,i}^2$. Therefore, we have $\left|\sum_{j=1}^n\sum_{i=2}^d \zeta_{j,i}^2 - \sum_{j=1}^n\sum_{i=2}^d \lambda_{j,i}^2\right| \leq 200\epsilon\sum_{j=1}^n\sum_{i=2}^d \lambda_{j,i}^2$. In other words, $\|\mathbf{A}\mathbf{Y}_1\|_F^2$ is within a $(1+200\epsilon)$ factor of $\|\mathbf{A}\mathbf{Z}_1\|_F^2$. Moreover, $\frac{\|\mathbf{A}_s\mathbf{Y}_1\|_2^2}{\|\mathbf{A}\mathbf{Y}_1\|_F^2}$ and $\frac{\|\mathbf{A}_s\mathbf{Z}_1\|_2^2}{\|\mathbf{A}\mathbf{Z}_1\|_F^2}$ are probability distributions that each sum to 1 across all s. Thus we have the distortion in the

sampling probability of $\mathbf{A}_s$ is

$$\left|\frac{\sum_{i=2}^d \lambda_{s,i}^2}{\sum_{j=1}^n \sum_{i=2}^d \lambda_{j,i}^2} - \frac{\sum_{i=2}^d \zeta_{s,i}^2}{\sum_{j=1}^n \sum_{i=2}^d \zeta_{j,i}^2}\right| \leq \frac{2(1+200\epsilon)25}{\sum_{j=1}^n \sum_{i=2}^d \lambda_{j,i}^2} \sum_{i=2}^d \left(6\epsilon\lambda_{s,i}^2 + 2\frac{\tau_i^2}{\epsilon}\lambda_{s,1}^2\right).$$

Taking the sum over all rows $\mathbf{A}_s$ and noting $\tau_i^2 \leq \frac{\epsilon^3}{C\log n}\frac{\sum_{a=1}^n \lambda_{a,i}^2}{\sum_{a=1}^n \sum_{b=1}^d \lambda_{a,b}^2}$, we have that

$$\sum_{i=1}^n \left|\frac{\|\mathbf{A}_i\mathbf{Y}_1\|_2^2}{\|\mathbf{A}\mathbf{Y}_1\|_F^2} - \frac{\|\mathbf{A}_i\mathbf{Z}_1\|_2^2}{\|\mathbf{A}\mathbf{Z}_1\|_F^2}\right| \leq 799\epsilon,$$

for sufficiently small ϵ. Thus including the $\frac{1}{\text{poly}(n)}$ event of failure from Lemma 3.1, the total variation distance is at most 800ϵ, which completes our base case.

Inductive step. Suppose that the total variation distance between the distributions of the first $t-1$ indices sampled by $\mathcal{P}_1$ and $\mathcal{P}_2$ is at most $800(t-1)\epsilon$. We consider the difference in the probability distribution induced by linearly independent vectors $\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(t-1)}$ and the probability distribution induced by linearly independent vectors $\mathbf{r}_1, \mathbf{A}_{f(2)}, \ldots, \mathbf{A}_{f(t-1)}$. We can define $U = \{\mathbf{u}_1, \ldots, \mathbf{u}_d\}$ to be an orthonormal basis for the row span of $\mathbf{A}$ such that $\{\mathbf{u}_1, \ldots, \mathbf{u}_s\}$ is a basis for the row span of $\{\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(s)}\}$ for each $2 \leq s \leq t-1$. Similarly, let $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_d\}$ be an orthonormal basis for the row span of $\mathbf{A}$ such that $\{\mathbf{w}_1, \ldots, \mathbf{w}_s\}$ is an orthonormal basis that extends the row span of $\{\mathbf{r}_1, \mathbf{A}_{f(2)}, \ldots, \mathbf{A}_{f(s)}\}$ for each $2 \leq s \leq t-1$. We again have from Lemma 3.1 that with high probability, $\mathbf{r}_1 = \|\mathbf{A}_{f(1)}\|_2 \left(\mathbf{u}_1 + \sum_{i=1}^d (\pm\mathcal{O}(\tau_i))\mathbf{u}_i\right)$ and $\tau_i = \frac{\epsilon\sqrt{\epsilon}}{\sqrt{C\log n}}\frac{\|\mathbf{A}\mathbf{P}_i\|_F}{\|\mathbf{A}\|_F}$ with constant at most 5 hidden in the $\mathcal{O}(\cdot)$ notation and $\mathbf{P}_i = \mathbf{u}_i^\dagger\mathbf{u}_i$ is the projection matrix onto $\mathbf{u}_i$. We condition on this relationship between $\mathbf{r}_1$ and the basis U and incorporate the $\frac{1}{\text{poly}(n)}$ probability of failure into our variation distance at the end of the inductive step. Thus, we can apply the Gram-Schmidt process to obtain the change of basis matrix $\mathbf{B}$ from U to W whose entries are again bounded as in $(\star)$. We emphasize that the same bounds apply in the matrix $\mathbf{B}$ since we still receive a noisy row in the first iteration of the sampling procedure and we receive the true rows in the subsequent iterations, just as in the base case. Thus, Lemma 3.1 is only invoked in determining the bounds of the first row $\mathbf{b}_1$ and the subsequent bounds are determined using the Gram-Schmidt process, exactly as in Claim 3.3.

We again write each row $\mathbf{A}_s$ in terms of basis U as $\mathbf{A}_s = \sum_{i=1}^d \lambda_{s,i}\mathbf{u}_i$ and in terms of basis W as $\mathbf{A}_s = \sum_{i=1}^d \zeta_{s,i}\mathbf{w}_i$. Then since we project away from $\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(t-1)}$, we should have sampled $\mathbf{A}_s$ with probability $\frac{\|\mathbf{A}_s\mathbf{Z}_{t-1}\|_2^2}{\|\mathbf{A}\mathbf{Z}_{t-1}\|_F^2} = \frac{\sum_{i=t}^d \lambda_{s,i}^2}{\sum_{j=1}^n \sum_{i=t}^d \lambda_{j,i}^2}$ in round t but instead we sample it with probability $\frac{\|\mathbf{A}_s\mathbf{Y}_{t-1}\|_2^2}{\|\mathbf{A}\mathbf{Y}_{t-1}\|_F^2} = \frac{\sum_{i=t}^d \zeta_{s,i}^2}{\sum_{j=1}^n \sum_{i=t}^d \zeta_{j,i}^2}$. From the change of basis matrix $\mathbf{B}$, we again have that for $i \geq 2$:

$$\begin{aligned}|\zeta_{s,i}^2 - \lambda_{s,i}^2| \leq 25\Big(&\tau_i\lambda_{s,i}^2 + \tau_i^2\lambda_{s,1}^2 + \tau_i\lambda_{s,1}\lambda_{s,i} + \sum_{j,\ell\neq\{i,1\}} \tau_i^2\tau_j\tau_\ell\lambda_{s,j}\lambda_{s,\ell} \\ &+ \sum_{j\neq\{i,1\}} \tau_i\tau_j\lambda_{s,i}\lambda_{s,j} + \sum_{j\neq\{i,1\}} \tau_i^2\tau_j\lambda_{s,1}\lambda_{s,j}\Big). \qquad (\boxplus)\end{aligned}$$

From AM-GM, we have:

$$\tau_i\lambda_{s,1}\lambda_{s,i} \le \epsilon\lambda_{s,i}^2 + \frac{\tau_i^2}{\epsilon}\lambda_{s,1}^2$$
$$\tau_i^2\tau_j\tau_\ell\lambda_{s,j}\lambda_{s,\ell} \le \tau_i^2\tau_j^2\lambda_{s,j}^2 + \tau_i^2\tau_\ell^2\lambda_{s,\ell}^2,$$
$$\tau_i\tau_j\lambda_{s,i}\lambda_{s,j} \le \epsilon^2\lambda_{s,i}^2 + \frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2,$$
$$\tau_i^2\tau_j\lambda_{s,1}\lambda_{s,j} \le \tau_i^2\lambda_{s,1}^2 + \tau_i^2\tau_j^2\lambda_{s,j}^2.$$

Thus for $\epsilon < \frac{1}{d}$, $|\zeta_{s,i}^2 - \lambda_{s,i}^2| \le 25\left(2\epsilon\lambda_{s,i}^2 + \frac{2\tau_i^2}{\epsilon}\lambda_{s,1}^2 + 4\sum_{j=2}^d \frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2\right)$. Then

$$\left|\sum_{i=t}^d \zeta_{s,i}^2 - \sum_{i=t}^d \lambda_{s,i}^2\right| \le 25\sum_{i=t}^d\left(2\epsilon\lambda_{s,i}^2 + \frac{2\tau_i^2}{\epsilon}\lambda_{s,1}^2 + 4\sum_{j=2}^d \frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2\right).$$

Now recall that $\tau_i^2 = \frac{\epsilon^3}{C\log n}\frac{\|\mathbf{A}\mathbf{P}_i\|_F^2}{\|\mathbf{A}\|_F^2} = \frac{\epsilon^3}{C\log n}\frac{\sum_{a=1}^n \lambda_{a,i}^2}{\sum_{a=1}^n\sum_{b=1}^d \lambda_{a,b}^2}$. Therefore we get that,

$$\sum_{s=1}^n\sum_{i=t}^d\sum_{j=2}^d\left(\frac{\tau_i^2}{\epsilon^2}\right)\lambda_{s,j}^2 = \sum_{i=t}^d\left(\frac{\tau_i^2}{\epsilon^2}\right)\sum_{s=1}^n\sum_{j=2}^d\lambda_{s,j}^2 \le \sum_{i=t}^d\frac{\epsilon}{C\log n}\sum_{s=1}^n\lambda_{s,i}^2.$$

Similarly, we get that,

$$\sum_{s=1}^n\sum_{i=t}^d\left(\frac{\tau_i^2}{\epsilon}\right)\lambda_{s,1}^2 = \sum_{i=t}^d\left(\frac{\tau_i^2}{\epsilon}\right)\sum_{s=1}^n\lambda_{s,1}^2 \le \sum_{i=t}^d\frac{\epsilon^2}{C\log n}\sum_{s=1}^n\lambda_{s,i}^2.$$

Therefore, we can bound

$$\sum_{s=1}^n\left|\sum_{i=t}^d\zeta_{s,i}^2 - \sum_{i=t}^d\lambda_{s,i}^2\right| \le 200\epsilon\sum_{s=1}^n\sum_{i=t}^d\lambda_{s,i}^2$$

so that $\|\mathbf{A}\mathbf{Y}_{t-1}\|_F^2$ is once again within a $(1+200\epsilon)$ factor of $\|\mathbf{A}\mathbf{Z}_{t-1}\|_F^2$. Moreover,

$$\sum_{s=1}^n\left|\frac{\sum_{i=t}^d\zeta_{s,i}^2}{\sum_{j=1}^n\sum_{i=t}^d\lambda_{j,i}^2} - \frac{\sum_{i=t}^d\lambda_{s,i}^2}{\sum_{j=1}^n\sum_{i=t}^d\lambda_{j,i}^2}\right| \le 200\epsilon.$$

Since we consider the total variation distance across the probability distribution, then the sampling probabilities each sum to 1 and we have

$$\sum_{i=1}^n\left|\frac{\|\mathbf{A}_i\mathbf{Y}_{t-1}\|_2^2}{\|\mathbf{A}\mathbf{Y}_{t-1}\|_F^2} - \frac{\|\mathbf{A}_i\mathbf{Z}_{t-1}\|_2^2}{\|\mathbf{A}\mathbf{Z}_{t-1}\|_F^2}\right| = \sum_{s=1}^n\left|\frac{\sum_{i=t}^d\zeta_{s,i}^2}{\sum_{j=1}^n\sum_{i=t}^d\zeta_{j,i}^2} - \frac{\sum_{i=t}^d\lambda_{s,i}^2}{\sum_{j=1}^n\sum_{i=t}^d\lambda_{j,i}^2}\right| \le 2(1+200\epsilon)200\epsilon \le 799\epsilon,$$

for sufficiently small ϵ. Thus including the $\frac{1}{\text{poly}(n)}$ probability of failure from Lemma 3.1, the total variation distance between the probability distributions of the index of the sample output by $\mathcal{P}_1$ and $\mathcal{P}_2$ in round t is at most 800ϵ.

From the inductive hypothesis, the total variation distance between the probability distributions of $t-1$ indices corresponding to samples output by $\mathcal{P}_1$ and $\mathcal{P}_2$ across $t-1$ rounds is at most $800(t-1)\epsilon$. Now for any sequence of rows $\mathcal{S} = \{\mathbf{A}_{f(2)}, \ldots, \mathbf{A}_{f(t-1)}\}$, let $\mathcal{E}_{\mathcal{S},1}$ be the event that the rows of $\mathcal{S}$ are sequentially sampled, given that the first sampled row is $\mathbf{A}_{f(1)}$ and let $\mathcal{E}_{\mathcal{S},2}$ be the event that rows of $\mathcal{S}$ are sequentially sampled, given that the first sampled row is $\mathbf{r}_1$. Let BAD be the sets $\mathcal{S}$ such that at least one of $\mathbf{A}_{f(1)} \cup \mathcal{S}$ or $\mathbf{r}_1 \cup \mathcal{S}$ is not linearly independent. Observe that $P_{\mathsf{BAD}} := \sum_{\mathcal{S}\in\mathsf{BAD}} |\mathbf{Pr}[\mathcal{E}_{\mathcal{S},1}] - \mathbf{Pr}[\mathcal{E}_{\mathcal{S},2}]| \leq 800(t-1)\epsilon$, since sampling a row $\mathbf{A}_{f(i)}$ that is linearly dependent with $\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(i-1)}$ occurs with probability zero so then sampling a row $\mathbf{A}_{f(i)}$ that is linearly dependent with $\mathbf{r}_1, \ldots, \mathbf{A}_{f(i-1)}$ must be realized in the total variation distance in the first $t-1$ rounds.

Otherwise, we have that the total variation distance between the probability distributions of the index corresponding to the sample output by $\mathcal{P}_1$ and $\mathcal{P}_2$ in round t is at most 800ϵ. Let p_j be the event that $\mathbf{A}_j$ is sampled in round t. Then we have that the probability that $\mathcal{S} \cup \mathbf{A}_j$ is sequentially sampled, given that the first sampled row is $\mathbf{A}_{f(1)}$, is $\mathbf{Pr}\left[\mathcal{E}_{\mathcal{S}\cup\mathbf{A}_j,1}\right] = \mathbf{Pr}[\mathcal{E}_{\mathcal{S},1}]\,\mathbf{Pr}[p_j|\mathcal{E}_{\mathcal{S},1}]$ and the probability that $\mathcal{S} \cup \mathbf{A}_j$ is sequentially sampled, given that the first sampled row is $\mathbf{r}_1$, is $\mathbf{Pr}\left[\mathcal{E}_{\mathcal{S}\cup\mathbf{A}_j,2}\right] = \mathbf{Pr}[\mathcal{E}_{\mathcal{S},2}]\,\mathbf{Pr}[p_j|\mathcal{E}_{\mathcal{S},2}]$. We have $\sum_{\mathcal{S}\notin\mathsf{BAD}} |\mathbf{Pr}[\mathcal{E}_{\mathcal{S},1}] - \mathbf{Pr}[\mathcal{E}_{\mathcal{S},2}]| \leq 800(t-1)\epsilon - P_{\mathsf{BAD}}$. Moreover for $\mathcal{S} \notin \mathsf{BAD}$, we have $\sum_{j\in[n]} |\mathbf{Pr}[p_j|\mathcal{E}_{\mathcal{S},1}] - \mathbf{Pr}[p_j|\mathcal{E}_{\mathcal{S},2}]| \leq 800\epsilon$. Thus we have

$$\sum_{\mathcal{S}\notin\mathsf{BAD}} \sum_{j\in[n]} \left|\mathbf{Pr}\left[\mathcal{E}_{\mathcal{S}\cup\mathbf{A}_j,1}\right] - \mathbf{Pr}\left[\mathcal{E}_{\mathcal{S}\cup\mathbf{A}_j,2}\right]\right| \leq 800(t-1)\epsilon - P_{\mathsf{BAD}} + 800\epsilon.$$

Since $\sum_{\mathcal{S}\in\mathsf{BAD}} \sum_{j\in[n]} \left|\mathbf{Pr}\left[\mathcal{E}_{\mathcal{S}\cup\mathbf{A}_j,1}\right] - \mathbf{Pr}\left[\mathcal{E}_{\mathcal{S}\cup\mathbf{A}_j,2}\right]\right| \leq P_{\mathsf{BAD}}$, then we have that the total variation distance is at most $800t\epsilon$, which completes the induction. Thus the total variation distance between the probability distributions of k indices output by $\mathcal{P}_1$ and $\mathcal{P}_2$ across k rounds is at most $800k\epsilon$. □

Since the total variation distance induced by a single noisy row is small, we obtain that the total variation distance between offline adaptive sampling and our adaptive sampler is small by rescaling the error parameter. Thus we now provide the full guarantees for our adaptive sampler.

Theorem 3.4 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives in a turnstile stream, there exists a one-pass algorithm* AdaptiveStream *that outputs a set of k indices such that the probability distribution for each set of k indices has total variation distance ϵ of the probability distribution induced by adaptive sampling with respect to squared distances to the selected subspace in each iteration. The algorithm uses $\mathcal{O}\left(\frac{d^3k^6}{\epsilon^3}\log^6 n\right)$ bits of space.*

Proof : Consider a set of $k+1$ processes $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{k+1}$, where for each $i \in [k+1]$, $\mathcal{P}_i$ is a process that samples noisy rows from the $L_{2,2}$ sampler for the first $i-1$ rounds and actual rows from $\mathbf{A}$ beginning with round i, through adaptive sampling with $p=2$. Observe that $\mathcal{P}_1$ is the actual adaptive sampling process and $\mathcal{P}_{k+1}$ is the noisy process of Algorithm 4. Then Lemma 3.2 argues that the total variation distance between the output distributions of the k indices sampled by $\mathcal{P}_1$ and $\mathcal{P}_2$ is at most $\mathcal{O}(k\epsilon)$. In fact, the proof of Lemma 3.2 also shows that the total variation distance between the output distributions of the indices sampled by $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$ is at most $\mathcal{O}(k\epsilon)$ for any $i \in [k]$. This is because the sampling distributions of $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$ is identical in the first i rounds, so we can use the same argument starting at round i using the input matrix $\mathbf{AQ}$ rather than $\mathbf{A}$, where $\mathbf{Q}$ is the projection matrix away from the noisy rows sampled in the first i rounds.

Let μ_i be the probability distribution of the k indices output by $\mathcal{P}_i$. Thus from a triangle inequality argument, we have that

$$d_{\mathsf{TV}}(\mathcal{P}_1, \mathcal{P}_{k+1}) \leq \sum_{i=1}^{k} d_{\mathsf{TV}}(\mathcal{P}_i, \mathcal{P}_{i+1}) = \sum_{i=1}^{k} \mathcal{O}(k\epsilon) = \mathcal{O}\left(k^2\epsilon\right).$$

In other words, the total variation distance between the probability distribution of the k indices output by Algorithm 4 and the probability distribution of the k indices output by adaptive sampling is at most $\mathcal{O}\left(k^2\epsilon\right)$. Then we obtain total variation distance ϵ by the appropriate rescaling factor.

Lemma 3.2 requires the error parameter to be less than $\frac{1}{d}$. To analyze the space complexity, observe that with the error parameter $\mathcal{O}\left(\frac{\epsilon}{dk^2}\right)$, then Lemma 3.2 suggests that $\mathcal{O}\left(\frac{d^2k^4\log^2 n}{\epsilon^2}\right)$ buckets are necessary in each CountSketch structure in the $L_{2,2}$ sampler. Thus each CountSketch structure is a $\mathcal{O}\left(\frac{d^2k^4\log^2 n}{\epsilon^2}\right) \times \mathcal{O}(\log n)$ table. Each entry in the table is a vector of d integers that use $\mathcal{O}(d\log n)$ bits of space for each vector, and the sampler can be repeated $\mathcal{O}\left(\frac{k}{\epsilon}\log^2 n\right)$ times to obtain probability of success at least $1 - \frac{1}{\text{poly}(n)}$. This forms one $L_{2,2}$ sampler, but we need k iterations of the $L_{2,2}$ sampler to simulate k rounds of adaptive sampling. Therefore, the total space complexity is $\mathcal{O}\left(\frac{d^3k^6}{\epsilon^3}\log^6 n\right)$. □

Note that the proof of Lemma 3.2 also showed that $\|\mathbf{A}\mathbf{Y}_t\|_F^2$ is within a $(1+\mathcal{O}(\epsilon))$ factor of $\|\mathbf{A}\mathbf{Z}_t\|_F^2$. In Theorem 3.4, we have now sampled k noisy rows rather than a single noisy row followed by $k-1$ true rows, but we also rescale the error parameter down to $\mathcal{O}\left(\frac{\epsilon}{dk^2}\right)$.

Corollary 3.5 *Suppose Algorithm 4 samples noisy rows $\mathbf{r}_1, \ldots, \mathbf{r}_k$ rather than the actual rows $\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(k)}$. Let $\mathbf{T}_k = \mathbf{A}_{f(1)} \circ \ldots \circ \mathbf{A}_{f(k)}$, $\mathbf{Z}_k = \mathbb{I} - \mathbf{T}_k^\dagger \mathbf{T}_k$, $\mathbf{R}_k = \mathbf{r}_1 \circ \ldots \circ \mathbf{r}_k$ and $\mathbf{Y}_k = \mathbb{I} - \mathbf{R}_k^\dagger \mathbf{R}_k$. Then $(1-\epsilon)\|\mathbf{A}\mathbf{Y}_k\|_F^2 \leq \|\mathbf{A}\mathbf{Z}_k\|_F^2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{Y}_k\|_F^2$ with probability at least $1-\epsilon$.*

At first glance, it might seem strange that Corollary 3.5 obtains increased accuracy with higher probability, but recall that Algorithm 4 has a space dependency on poly $\left(\frac{1}{\epsilon}\right)$.

4 Applications

In this section, we give a number of data summarization applications for our adaptive sampler. In each application, the goal is to find a set S of k rows of an underlying matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ defined on a turnstile stream that optimizes a given predetermined function, which quantifies how well S represents $\mathbf{A}$. In particular among other things, we must show that for the purposes of each application, (1) it suffices to return a noisy perturbation of the orthogonal component at each sampling iteration, rather than the original row of the underlying matrix and (2) the algorithm still succeeds with an additive perturbation to sampling probabilities, rather than an ideal $(1 \pm \epsilon)$ multiplicative perturbation.

4.1 Column/Row Subset Selection

We first show that our adaptive sampling procedure can be used to give turnstile streaming algorithms for column/row subset selection. Recall that in the row (respectively column) subset selection problem, the inputs are an approximation parameter $\epsilon > 0$, a parameter k for the number of

selected rows or columns, and a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives as a data stream and the goal is to output a set $\mathbf{M}$ of k rows (respectively columns) of $\mathbf{A}$ such that $\left\|\mathbf{A}-\mathbf{A}\mathbf{M}^{\dagger}\mathbf{M}\right\|_F^2 \leq (1+\epsilon)\left\|\mathbf{A}-\mathbf{A}_k^*\right\|_F^2$ (respectively $\left\|\mathbf{A}-\mathbf{M}\mathbf{M}^{\dagger}\mathbf{A}\right\|_F^2 \leq (1+\epsilon)\left\|\mathbf{A}-\mathbf{A}_k^*\right\|_F^2$), where $\mathbf{A}_k^*$ is the best rank k approximation to $\mathbf{A}$. For the remainder of the section, we focus on row subset selection with the assumption that $n \gg d$ as continuation of the adaptive sampling scheme in previous sections, but we note that our results extend naturally to column subset selection.

Recall that volume sampling induces a probability distribution on subsets of rows rather than individual rows of $\mathbf{A}$. For a subset $\mathbf{T}$ of k rows of $\mathbf{A}$, let $\Delta(\mathbf{T})$ be the simplex defined by these rows and the origin and $\mathrm{Vol}(\mathbf{T})$ be the volume of $\Delta(\mathbf{T})$. Then the volume sampling probability distribution samples each subset $\mathbf{T}$ of k rows of $\mathbf{A}$ with probability

$$p_{\mathbf{T}} = \frac{\mathrm{Vol}(\mathbf{T})^2}{\sum_{\mathbf{S}:|\mathbf{S}|=k}\mathrm{Vol}(\mathbf{S})^2},$$

where $\mathbf{S}$ is taken across all subsets of k rows of $\mathbf{A}$.

[DRVW06] gives the following relationship between volume sampling and row subset selection.

Theorem 4.1 *[DRVW06] Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, let $\mathbf{T}$ be a subset of k rows of $\mathbf{A}$ generated from the volume sampling probability distribution. Then*

$$\mathbb{E}_{\mathbf{T}}\left[\left\|\mathbf{A}-\mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2\right] \leq (k+1)\left\|\mathbf{A}-\mathbf{A}_k^*\right\|_F^2,$$

where $\mathbf{A}_k^*$ is the best rank k approximation to $\mathbf{A}$.

For the remainder of Section 4.1, we consider the adaptive sampling scheme in Algorithm 5 to obtain a subset of k rows of $\mathbf{A}$, proportional to the squared row norms of the orthogonal projection at each step.

Algorithm 5 Offline Adaptive Sampling by Squared Row Norms of Orthogonal Projection

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, integer $k > 0$
Output: Subset of k rows of $\mathbf{A}$

1: $\mathbf{M} \leftarrow \emptyset$
2: **for** $i = 1$ to $i = k$ **do**
3: Choose $\mathbf{r}$ to be $\mathbf{A}_j$, with probability $\frac{\left\|\mathbf{A}_j(\mathbb{I}-\mathbf{M}^{\dagger}\mathbf{M})\right\|_2^2}{\left\|\mathbf{A}(\mathbb{I}-\mathbf{M}^{\dagger}\mathbf{M})\right\|_F^2}$, for $j \in [n]$.
4: $\mathbf{M} \leftarrow \mathbf{M} \circ \mathbf{r}$
5: **return** $\mathbf{M}$

[DV06] shows that the adaptive sampling probabilities can be bounded by a multiple of the volume sampling probabilities.

Lemma 4.2 *[DV06] Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, let $p_{\mathbf{T}}$ be the probability of sampling a set $\mathbf{T}$ of k rows from the volume sampling probability distribution and let $q_{\mathbf{T}}$ be the probability of sampling $\mathbf{T}$ from the adaptive sampling probability distribution, as in Algorithm 5. Then $q_{\mathbf{T}} \leq k!p_{\mathbf{T}}$.*

Thus our adaptive sampling procedure immediately gives a one-pass turnstile streaming algorithm for column/row subset selection.

Theorem 4.3 *Given a matrix* $\mathbf{A} \in \mathbb{R}^{n \times d}$ *that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set* $\mathbf{R}$ *of* k *(noisy) rows of* $\mathbf{A}$ *such that*

$$\mathbf{Pr}\left[\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2 \leq 16(k+1)! \left\|\mathbf{A} - \mathbf{A}_k^*\right\|_F^2\right] \geq \frac{2}{3}.$$

The algorithm uses $\text{poly}\,(d, k, \log n)$ *bits of space.*

Proof : Theorem 3.4 states that the indices of the rows sampled by AdaptiveStream have probability distribution roughly equivalent to the probability distribution of the indices of the rows sampled by adaptive sampling, as in Algorithm 5. We would thus like to apply Lemma 4.2 and Theorem 4.1, but we need to avoid specific failure events in our analysis. First, it may be the case that AdaptiveStream samples a set $\mathbf{R}$ so that $\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2$ is very large, but the corresponding set $\mathbf{T}$ has very little probability of being sampled by the adaptive sampling probability distribution. We absorb this failure event into the total variation distance. Second, we must analyze the difference between $\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2$ for the set $\mathbf{R}$ noisy rows compared to $\left\|\mathbf{A} - \mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2$ for the actual rows. This is handled by Corollary 3.5, which bounds the difference. We formalize these notions below.

Let $\mathcal{E}$ be the event that algorithm AdaptiveStream of Theorem 3.4 with error parameter $\epsilon = \frac{1}{12}$ sequentially samples an ordered set $\mathbf{R}$ of k noisy rows corresponding to an arbitrary ordered subset $\mathbf{T}$ of k rows of $\mathbf{A}$ and $\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2 \leq \left(1 + \frac{1}{12}\right)\left\|\mathbf{A} - \mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2$ and note that $\mathbf{Pr}\left[\mathcal{E}\right] \geq \frac{11}{12}$ by Corollary 3.5. Let $\mathcal{E}_{\mathbf{T}}$ be the event that algorithm AdaptiveStream of Theorem 3.4 with error parameter $\epsilon = \frac{1}{12}$ samples a set $\mathbf{R}$ of k noisy rows corresponding to the *specific* subset $\mathbf{T}$ of k rows of $\mathbf{A}$ and $\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2 \leq \left(1 + \frac{1}{12}\right)\left\|\mathbf{A} - \mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2$.

Let $\widehat{q_{\mathbf{T}}}$ be the probability that the indices corresponding to $\mathbf{T}$ are sampled by AdaptiveStream of Theorem 3.4. Let $q_{\mathbf{T}}$ be the probability of sampling $\mathbf{T}$ from the adaptive sampling probability distribution, as in Algorithm 5. Let $\mathcal{S}$ be the collection of k-sets of indices of rows $\mathbf{T}$ such that $\widehat{q_{\mathbf{T}}} > 2q_{\mathbf{T}}$ and note that $\sum_{\mathbf{T} \in \mathcal{S}} q_{\mathbf{T}} \leq \frac{1}{12}$ since $\{q_{\mathbf{T}}\}_{\mathbf{T}}$ and $\{\widehat{q_{\mathbf{T}}}\}_{\mathbf{T}}$ have total variation distance at most $\frac{1}{12}$ using AdaptiveStream with error parameter $\epsilon = \frac{1}{12}$ by Theorem 3.4. When the event $\mathcal{E}$ occurs, we abuse notation by saying $\mathbf{R} \notin \mathcal{S}$ if the indices corresponding to the set of sampled rows does not belong in $\mathcal{S}$. Observe that algorithmically, we do not know the indices corresponding to the set $\mathbf{R}$ of k noisy rows, but analytically each row of $\mathbf{R}$ must correspond to a certain row of $\mathbf{T}$, based on the scaling of the uniform random variables at each round of AdaptiveStream. Then

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{R} \notin \mathcal{S}}\left[\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2 \,\middle|\, \mathcal{E}\right] &= \sum_{\mathbf{T} \notin \mathcal{S}} \widehat{q_{\mathbf{T}}} \cdot \mathbb{E}\left[\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2 \,\middle|\, \mathcal{E}_{\mathbf{T}}\right] \\
&\leq \frac{1}{\mathbf{Pr}\left[\mathcal{E}_{\mathbf{T}}\right]} \sum_{\mathbf{T} \notin \mathcal{S}} \widehat{q_{\mathbf{T}}} \left(1 + \frac{1}{12}\right)\left\|\mathbf{A} - \mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2 \\
&\leq \frac{12}{11} \sum_{\mathbf{T} \notin \mathcal{S}} 2q_{\mathbf{T}} \left(1 + \frac{1}{12}\right)\left\|\mathbf{A} - \mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2 \\
&\leq \frac{13}{11} \sum_{\mathbf{T} \notin \mathcal{S}} 2q_{\mathbf{T}} \left\|\mathbf{A} - \mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2,
\end{aligned}$$

where the penultimate inequality follows from $\mathbf{T} \notin \mathcal{S}$ implying that $\widehat{q_{\mathbf{T}}} \leq 2q_{\mathbf{T}}$. By Lemma 4.2 and Theorem 4.1, it follows that

$$\mathop{\mathbb{E}}_{\mathbf{R} \notin \mathcal{S}}\left[\left\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\dagger}\mathbf{R}\right\|_F^2 \,\middle|\, \mathcal{E}\right] \leq \frac{26}{11} \sum_{\mathbf{T}:|\mathbf{T}|=k} p_{\mathbf{T}} k! \left\|\mathbf{A} - \mathbf{A}\mathbf{T}^{\dagger}\mathbf{T}\right\|_F^2$$

$$\leq \frac{26}{11}(k+1)! \left\|\mathbf{A} - \mathbf{A}_k^*\right\|_F^2,$$

where $p_{\mathbf{T}}$ is the probability of sampling a set $\mathbf{T}$ of k rows from the volume sampling probability distribution. Therefore by Markov's inequality, the correctness of the claim follows by additionally taking a union bound over $\mathbf{Pr}\left[\neg\mathcal{E}\right] \leq \frac{1}{12}$ and $\sum_{\mathbf{T}\in\mathcal{S}} q_{\mathbf{T}} \leq \frac{1}{12}$. The space of the algorithm follows from taking $\epsilon = \mathcal{O}(1)$ in Theorem 3.4. □

4.2 Subspace Approximation

We next show that our adaptive sampling procedure can be used to give turnstile streaming algorithms for the subspace approximation. Recall that in the subspace approximation problem, the inputs are a parameter $p \geq 1$, a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives as a data stream, and a parameter k for the target dimension of the subspace, and the goal is to output a k-dimensional linear subspace $\mathbf{H}$ that minimizes $(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{H})^p)^{\frac{1}{p}}$, where $d(\mathbf{A}_i, \mathbf{H}) = \left\|\mathbf{A}_i(\mathbb{I} - \mathbf{H}^\dagger\mathbf{H})\right\|_2$ is the distance from $\mathbf{A}_i$ to the subspace $\mathbf{H}$.

We consider a generalized version of the adaptive sampling scheme that appears in Algorithm 5 to obtain a subset of k rows of $\mathbf{A}$, where rows are sampled with probabilities proportional to p^{th} power of the distance to the subspace formed by the span of the sampled rows. The generalized version, which appears in Algorithm 6, corresponds to Algorithm 5 when $p = 2$.

Algorithm 6 Offline Adaptive Sampling by p^{th} Power of Distance to Subspace

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, integers $k > 0$, $p \geq 1$
Output: Subset of k rows of $\mathbf{A}$

1: $\mathbf{M} \leftarrow \emptyset$
2: **for** $i = 1$ to $i = k$ **do**
3: Choose $\mathbf{r}$ to be $\mathbf{A}_j$, with probability $\frac{d(\mathbf{A}_j,\mathbf{M})^p}{\sum_{\ell=1}^n d(\mathbf{A}_\ell,\mathbf{M})^p}$, for $j \in [n]$.
4: $\mathbf{M} \leftarrow \mathbf{M} \circ \mathbf{r}$
5: **return** $\mathbf{M}$

[DV07] shows that adaptive sampling based on the p^{th} powers of the subspace distances can be used to give a good approximation to the subspace approximation problem.

Theorem 4.4 *[DV07] Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, let $\mathbf{T}$ be a subset of k rows of $\mathbf{A}$ generated from the adaptive sampling probability distribution, as in Algorithm 6. Then*

$$\mathbb{E}_{\mathbf{T}}\left[\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^p\right] \leq ((k+1)!)^p \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{A}_k^*)^p,$$

where $\mathbf{A}_k^*$ is the best rank k solution to the subspace approximation problem.

Thus our adaptive sampling procedure immediately gives a one-pass turnstile streaming algorithm for the subspace approximation problem with $p = 2$, by a similar argument to Theorem 4.3.

Theorem 4.5 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set $\mathbf{R}$ of k (noisy) rows of $\mathbf{A}$ such that*

$$\mathbf{Pr}\left[\left(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^2\right)^{\frac{1}{2}} \leq 4(k+1)!\left(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{A}_k^*)^2\right)^{\frac{1}{2}}\right] \geq \frac{2}{3},$$

where $\mathbf{A}_k^*$ *is the best rank* k *solution to the subspace approximation problem. The algorithm uses* $\text{poly}(d, k, \log n)$ *bits of space.*

Proof : As in the proof of Theorem 4.3, let $\mathcal{E}$ be the event that algorithm AdaptiveStream of Theorem 3.4 with error parameter $\epsilon = \frac{1}{12}$ samples a set $\mathbf{R}$ of k noisy rows corresponding to an arbitrary subset $\mathbf{T}$ of rows of $\mathbf{A}$ and $\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^2 \leq \left(1 + \frac{1}{12}\right) \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^2$, so that $\mathbf{Pr}[\mathcal{E}] \geq \frac{11}{12}$ by Corollary 3.5. For a set $\mathbf{T}$ of rows of $\mathbf{A}$, let $\mathcal{E}_{\mathbf{T}}$ be the event that algorithm AdaptiveStream of Theorem 3.4 with error parameter $\epsilon = \frac{1}{12}$ samples a set $\mathbf{R}$ of k noisy rows corresponding to $\mathbf{T}$ and $\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^2 \leq \left(1 + \frac{1}{12}\right) \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^2$.

Let $\widehat{q_{\mathbf{T}}}$ be the probability that the indices corresponding to $\mathbf{T}$ are sampled by AdaptiveStream of Theorem 3.4 and $q_{\mathbf{T}}$ be the probability of sampling $\mathbf{T}$ from the adaptive sampling probability distribution, as in Algorithm 6 with $p = 2$. Let $\mathcal{S}$ be the set of rows $\mathbf{T}$ such that $\widehat{q_{\mathbf{T}}} > 2q_{\mathbf{T}}$ and note that $\sum_{\mathbf{T} \in \mathcal{S}} q_{\mathbf{T}} \leq \frac{1}{12}$ since $\{q_{\mathbf{T}}\}_{\mathbf{T}}$ and $\{\widehat{q_{\mathbf{T}}}\}_{\mathbf{T}}$ have total variation distance at most $\frac{1}{12}$ using AdaptiveStream with error parameter $\epsilon = \frac{1}{12}$ by Theorem 3.4. We again abuse notation by saying $\mathbf{R} \notin \mathcal{S}$ when the event $\mathcal{E}$ occurs if the indices corresponding to the set of sampled rows $\mathbf{R}$ does not belong in $\mathcal{S}$. Then

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{R} \notin \mathcal{S}} \left[\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^2 \,\middle|\, \mathcal{E} \right] &= \sum_{\mathbf{T} \notin \mathcal{S}} \widehat{q_{\mathbf{T}}} \cdot \mathbb{E} \left[\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^2 \,\middle|\, \mathcal{E}_{\mathbf{T}} \right] \\
&\leq \frac{1}{\mathbf{Pr}[\mathcal{E}_{\mathbf{T}}]} \sum_{\mathbf{T} \notin \mathcal{S}} \widehat{q_{\mathbf{T}}} \cdot \left(1 + \frac{1}{12}\right) \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^2 \\
&\leq \frac{12}{11} \sum_{\mathbf{T} \notin \mathcal{S}} 2q_{\mathbf{T}} \left(1 + \frac{1}{12}\right) \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^2 \\
&\leq \frac{13}{11} \sum_{\mathbf{T} \notin \mathcal{S}} 2q_{\mathbf{T}} \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^2,
\end{aligned}$$

where the penultimate inequality follows from $\mathbf{T} \notin \mathcal{S}$ implying that $\widehat{q_{\mathbf{T}}} \leq 2q_{\mathbf{T}}$. By Theorem 4.4,

$$\mathop{\mathbb{E}}_{\mathbf{R} \notin \mathcal{S}} \left[\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^2 \,\middle|\, \mathcal{E} \right] \leq \frac{26}{11}((k+1)!)^2 \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{A}_k^*)^2.$$

Therefore by Markov's inequality and taking a union bound over $\mathbf{Pr}[\neg\mathcal{E}] \leq \frac{1}{12}$ and $\sum_{\mathbf{T} \in \mathcal{S}} q_{\mathbf{T}} \leq \frac{1}{12}$, we have that

$$\mathbf{Pr}\left[\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^2 \leq 16((k+1)!)^2 \sum_{i=1}^n d(\mathbf{A}_i, \mathbf{A}_k^*)^2 \right] \geq \frac{2}{3},$$

as desired. The space of the algorithm follows from taking $\epsilon = \mathcal{O}(1)$ in Theorem 3.4. □

To simulate Algorithm 6 for $p = 1$, we need to sample a row $\mathbf{A}_i\mathbf{P}$ with probability proportional to $\|\mathbf{A}_i\mathbf{P}\|_2$ rather than $\|\mathbf{A}_i\mathbf{P}\|_2^2$. We show how to do this in Algorithm 11 in Section A.2. By applying Theorem A.7 and using the same argument as Theorem 4.5, we also obtain a one-pass turnstile streaming algorithm for the subspace approximation problem for $p = 1$.

Theorem 4.6 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set $\mathbf{T}$ of k (noisy) rows of $\mathbf{A}$ such that*

$$\mathbf{Pr}\left[\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T}) \leq 4(k+1)!\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{A}_k^*)\right)\right] \geq \frac{2}{3},$$

where $\mathbf{A}_k^*$ is the best rank k solution to the subspace approximation problem. The algorithm uses $\text{poly}\,(d, k, \log n)$ bits of space.

Proof : Let $\mathcal{E}$ be the event that Theorem A.7 with $\epsilon = \frac{1}{12}$ samples a set $\mathbf{R}$ of k noisy rows corresponding to an arbitrary subset $\mathbf{T}$ of rows of $\mathbf{A}$ and $\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{R}) \leq \left(1+\frac{1}{12}\right)\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T})$, so that $\mathbf{Pr}\left[\mathcal{E}\right] \geq \frac{11}{12}$ by Corollary A.8. For a set $\mathbf{T}$ of rows of $\mathbf{A}$, let $\mathcal{E}_{\mathbf{T}}$ be the event that Theorem A.7 with $\epsilon = \frac{1}{12}$ samples a set $\mathbf{R}$ of k noisy rows corresponding to $\mathbf{T}$ and $\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{R}) \leq \left(1+\frac{1}{12}\right)\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T})$.

Let $\widehat{q_{\mathbf{T}}}$ be the probability that the indices corresponding to $\mathbf{T}$ are sampled by Theorem A.7 and $q_{\mathbf{T}}$ be the probability of sampling $\mathbf{T}$ from the adaptive sampling probability distribution, as in Algorithm 6 with $p = 1$. Let $\mathcal{S}$ be the set of rows $\mathbf{T}$ such that $\widehat{q_{\mathbf{T}}} > 2q_{\mathbf{T}}$ and note that $\sum_{\mathbf{T}\in\mathcal{S}} q_{\mathbf{T}} \leq \frac{1}{12}$ since $\{q_{\mathbf{T}}\}_{\mathbf{T}}$ and $\{\widehat{q_{\mathbf{T}}}\}_{\mathbf{T}}$ have total variation distance at most $\frac{1}{12}$ by Theorem A.7. Then

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{R}\notin\mathcal{S}}\left[\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{R}) \,\middle|\, \mathcal{E}\right] &= \sum_{\mathbf{T}\notin\mathcal{S}} \widehat{q_{\mathbf{T}}} \cdot \mathbb{E}\left[\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{R}) \,\middle|\, \mathcal{E}_{\mathbf{T}}\right] \\
&\leq \frac{1}{\mathbf{Pr}\left[\mathcal{E}_{\mathbf{T}}\right]} \sum_{\mathbf{T}\notin\mathcal{S}} \widehat{q_{\mathbf{T}}}\left(1+\frac{1}{12}\right)\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T}) \\
&\leq \frac{13}{11}\sum_{\mathbf{T}\notin\mathcal{S}} 2q_{\mathbf{T}} \sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T}),
\end{aligned}$$

where the last inequality follows from the fact that $\mathbf{Pr}\left[\mathcal{E}\right] \geq \frac{11}{12}$ and from $\mathbf{T} \notin \mathcal{S}$ implying that $\widehat{q_{\mathbf{T}}} \leq 2q_{\mathbf{T}}$. By Theorem 4.4,

$$\mathop{\mathbb{E}}_{\mathbf{R}\notin\mathcal{S}}\left[\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{R})^2 \,\middle|\, \mathcal{E}\right] \leq \frac{26}{11}(k+1)!\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{A}_k^*).$$

By Markov's inequality and a union bound over $\mathbf{Pr}\left[\neg\mathcal{E}\right] \leq \frac{1}{12}$ and $\sum_{\mathbf{T}\in\mathcal{S}} q_{\mathbf{T}} \leq \frac{1}{12}$, we have that

$$\mathbf{Pr}\left[\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{R}) \leq 16(k+1)!\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{A}_k^*)\right] \geq \frac{2}{3},$$

as desired. The space of the algorithm follows from taking $\epsilon = \mathcal{O}\left(1\right)$ in Theorem A.7. □

[DV07] also shows that adaptive sampling can be used to give a bicriteria approximation to the subspace approximation problem. We use the notation $\tilde{\mathcal{O}}\left(\cdot\right)$ in the remainder of Section 4.2 to omit polylog $\left(k, \frac{1}{\epsilon}\right)$ factors, with degrees depending on p.

Theorem 4.7 *[DV07] Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ and a parameter $\epsilon > 0$, let $m = \mathcal{O}\left(k\right)$ and let $\mathbf{T}_1, \ldots, \mathbf{T}_m$ each be a subset of $\tilde{\mathcal{O}}\left(k^2 \cdot \left(\frac{k}{\epsilon}\right)^{p+1}\right)$ rows generated from the adaptive sampling*

Algorithm 7 Repeated Offline Adaptive Oversampling by p^{th} Power of Distance to Subspace

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$, integers $k > 0$, $p \geq 1$

Output: Subset of $\tilde{\mathcal{O}}\left(k^2 \cdot \left(\frac{k}{\epsilon}\right)^{p+1}\right)$ rows of $\mathbf{A}$

1: Let $\mathbf{S}_0$ be a set of k rows obtained through Algorithm 5.
2: $\delta \leftarrow \frac{\epsilon}{\log k}$, $\mathbf{S} \leftarrow \mathbf{S}_0$, $t \leftarrow \mathcal{O}(k \log k)$
3: **for** $j = 1$ to $j = t$ **do**
4: **for** $i = 1$ to $i = k$ **do**
5: Sample a subset of rows $\mathbf{S}_i$ to be $\mathcal{O}\left(\left(\frac{2k}{\delta}\right)^p \frac{k}{\delta} \log \frac{k}{\delta}\right)$ rows of $\mathbf{A}$, where each $\mathbf{A}_\ell$ is selected with probability $\frac{d(\mathbf{A}_\ell, \mathbf{S}_{i-1})^p}{\sum_{\ell=1}^{n} d(\mathbf{A}_\ell, \mathbf{S}_{i-1})^p}$, for $\ell \in [n]$.
6: $\mathbf{S} \leftarrow \mathbf{S} \circ \mathbf{S}_i$
7: $\mathbf{S}_1 = \ldots = \mathbf{S}_k = \emptyset$
8: **return** $\mathbf{S}$

probability distribution with respect to repeated oversampling, as in Algorithm 7. Then for $\mathbf{T} = \mathbf{T}_1 \cup \ldots \cup \mathbf{T}_m$,

$$\mathbf{Pr}\left[\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T})^p\right)^{\frac{1}{p}} \leq (1+\epsilon)\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{A}_k^*)^p\right)^{\frac{1}{p}}\right] \geq \frac{3}{4},$$

where $\mathbf{A}_k^*$ *is the best rank* k *solution to the subspace approximation problem.*

By a similar argument to Theorem 4.3 and Theorem 4.5, our adaptive sampling procedure gives a one-pass turnstile streaming algorithm that produces a bicriteria approximation to the subspace approximation problem with $p = 2$.

Theorem 4.8 *Given a matrix* $\mathbf{A} \in \mathbb{R}^{n \times d}$ *that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set* $\mathbf{T}$ *of* $\tilde{\mathcal{O}}\left(k^3 \cdot \left(\frac{k}{\epsilon}\right)^{p+1}\right)$ *(noisy) rows of* $\mathbf{A}$ *such that*

$$\mathbf{Pr}\left[\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T})^p\right)^{\frac{1}{p}} \leq (1+\epsilon)\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{A}_k^*)^p\right)^{\frac{1}{p}}\right] \geq \frac{2}{3}$$

for $p \in \{1, 2\}$*, where* $\mathbf{A}_k^*$ *is the best rank* k *solution to the subspace approximation problem. The algorithm uses* $\mathrm{poly}(d, k, \frac{1}{\epsilon}, \log n, \log \frac{k}{\epsilon})$ *bits of space.*

Proof : The proof follows the same template as Theorem 4.3 that analyzes both the total variation distance between the ideal distribution and the actual distribution, as well as the quality of the approximation by the noisy rows to the actual rows. We first consider the case $p = 2$ and observe that Theorem 4.7 requires Algorithm 7 to sample $r := \tilde{\mathcal{O}}\left(k^3 \cdot \left(\frac{k}{\epsilon}\right)^{p+1}\right)$ rows in total. We cannot quite run Algorithm 4 as stated, since it uses k instances of the $L_{2,2}$ sampler in Algorithm 3 to sequentially sample k rows.

On the other hand, by creating $\mathrm{poly}(r, \log n)$ instances of the $L_{2,2}$ sampler Algorithm 3 with sufficiently small error parameter, we can simulate each round of Algorithm 7 used to generate $\mathbf{T}_1, \ldots, \mathbf{T}_m$ in the statement of Theorem 4.7. We require $\mathrm{poly}(r, \log n)$ instances of the $L_{2,2}$ sampler

to produce r samples, since recall that each sampler has some probability of outputting FAIL. To sample a set $\mathbf{R}_j$ corresponding to $\mathbf{T}_j$, we first use Algorithm 4 to sample a set $\mathbf{Y}_0$ of k (noisy) rows of $\mathbf{A}$ with total variation distance $\mathcal{O}\left(\frac{\epsilon}{r}\right)$ from the distribution of $\mathbf{S}_0$ in Algorithm 7. Now for $\delta = \frac{\epsilon}{\log k}$, each time we should have sampled a set $\mathbf{S}_i$ of $\mathcal{O}\left(\left(\frac{2k}{\delta}\right)^p \frac{k}{\delta}\log\frac{k}{\delta}\right)$ rows of $\mathbf{A}$ after projecting away from $\mathbf{S}_{i-1}$, we instead use instances of the $L_{2,2}$ sampler Algorithm 3 to sample a set $\mathbf{Y}_i$ of $\mathcal{O}\left(\left(\frac{2k}{\delta}\right)^p \frac{k}{\delta}\log\frac{k}{\delta}\right)$ noisy rows of $\mathbf{A}$ by projecting away from $\mathbf{Y}_{i-1}$.

Since we perform r rounds of sampling in total, then using the same argument as Theorem 3.4, we can bound the total variation distance between our output distribution and the distribution of Theorem 4.7 by $\mathcal{O}(\epsilon)$. Let S be the set of r indices corresponding to sets $\mathbf{R}$ of rows sampled by repeated iterations of Algorithm 7 so that $(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^p)^{\frac{1}{p}} \leq (1+\epsilon)(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{A}_k^*)^p)^{\frac{1}{p}}$. Let $\mathcal{E}$ be the event that the set $\mathbf{R}$ of rows sampled by repeated iterations of Algorithm 7 correspond to a set of r indices from S. Thus the probability that the indices of the r samples $\mathbf{T}$ produced by the $L_{2,2}$ samplers correspond to indices of S is at least $\mathbf{Pr}[\mathcal{E}] - \epsilon$.

Moreover for the noisy rows $\mathbf{T}$ that we sample, $(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^p)^{\frac{1}{p}} \leq (1+\epsilon)(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{R})^p)^{\frac{1}{p}}$ with probability at least $1-\epsilon$ by Corollary 3.5. By Theorem 4.7, $\mathbf{Pr}[\mathcal{E}] \geq \frac{3}{4}$. Thus for sufficiently small ϵ and by a rescaling argument, the probability that $(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^p)^{\frac{1}{p}} \leq (1+\epsilon)(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{A}_k^*)^p)^{\frac{1}{p}}$ is at least $\frac{2}{3}$. Then the correctness of the claim holds and the total space required is $\text{poly}(d, k, \frac{1}{\epsilon}, \log n, \log\frac{k}{\epsilon})$ by argument in Theorem 3.4. For $p = 1$, we use the $L_{1,2}$ sampler Algorithm 10 in place of the $L_{2,2}$ sampler Algorithm 3 to sample rows $\mathbf{Y}_i$ for $i > 0$ with probability proportional to their distances from the current subspace at each iteration, rather than the squared distances. Correctness then follows from the same argument, using Theorem A.7 and Corollary A.8 for the $L_{1,2}$ samplers. □

4.3 Projective Clustering

We now show that our adaptive sampling procedure can also be used to give turnstile streaming algorithms for projective clustering, where the inputs are a parameter $p \geq 1$, a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives as a data stream and parameters k for the target dimension of each subspace and s for the number of subspaces, and the goal is to output s k-dimensional linear subspaces $\mathbf{H}_1, \ldots, \mathbf{H}_s$ that minimizes:

$$\left(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{H})^p\right)^{\frac{1}{p}},$$

where $\mathbf{H} = \mathbf{H}_1 \cup \ldots \cup \mathbf{H}_s$ and $d(\mathbf{A}_i, \mathbf{H})$ is the distance from $\mathbf{A}_i$ to union $\mathbf{H}$ of s subspaces $\mathbf{H}_1, \ldots, \mathbf{H}_s$. Again we use $\tilde{\mathcal{O}}(\cdot)$ to omit $\text{polylog}\left(k, s, \frac{1}{\epsilon}\right)$ factors, with degrees depending on p.

[DV07] also shows that adaptive sampling can be used to perform dimensionality reduction for projective clustering.

Theorem 4.9 *[DV07] Let $\mathbf{V}$ be a subspace of dimension at least k such that*

$$\left(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{V})^p\right)^{\frac{1}{p}} \leq 2\left(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{H})^p\right)^{\frac{1}{p}},$$

Algorithm 8 Dimensionality Reduction for Projective Clustering

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$, integers $k > 0$, $s > 0$, $p \geq 1$, and a subspace $\mathbf{V}$ of dimension at least k.
Output: Subset of $\tilde{\mathcal{O}}\left(\left(\frac{k^2}{\epsilon}\right)^p \frac{k^4 s}{\epsilon^2}\right)$ rows of $\mathbf{A}$

1: $\mathbf{S} \leftarrow \emptyset$
2: **for** $t = 1$ to $t = \tilde{\mathcal{O}}\left(\left(\frac{k^2}{\epsilon}\right)^p \frac{k^4 s}{\epsilon^2}\right)$ **do**
3: Sample a row $\mathbf{r}$ of $\mathbf{A}$, where each row $\mathbf{A}_i$ is selected with probability $\frac{d(\mathbf{A}_i, \mathbf{S} \cup \mathbf{V})}{\sum_{j=1}^{n} d(\mathbf{A}_j, \mathbf{S} \cup \mathbf{V})}$.
4: $\mathbf{S} \leftarrow \mathbf{S} \circ \mathbf{r}$
5: **return** $\mathbf{S}$

where $\mathbf{H}$ *is the union of* s k*-dimensional subspaces that is the optimal solution to the projective clustering problem. Then with probability at least* $\frac{3}{4}$*, Algorithm 8 outputs a set* $\mathbf{S}$ *such that* $\mathbf{V} \cup \mathbf{S}$ *contains a union* $\mathbf{T}$ *of* s k*-dimensional subspaces such that*

$$\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T})^p\right)^{\frac{1}{p}} \leq (1+\epsilon)\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{H})^p\right)^{\frac{1}{p}}.$$

Since the optimal solution to the projective clustering problem is certainly no better than the optimal ks-dimensional subspace, we use the bicriteria subspace approximation algorithm of Theorem 4.8 with input dimension ks. Thus, we obtain a one-pass turnstile streaming algorithm for the projective clustering problem.

Theorem 4.10 *Given a matrix* $\mathbf{A} \in \mathbb{R}^{n \times d}$ *that arrives in a turnstile data stream, there exists a one-pass algorithm that outputs a set* $\mathbf{S}$ *of* $\tilde{\mathcal{O}}\left((ks)^{p+4} + \frac{k^4 s}{\epsilon^2}\left(\frac{k^2}{\epsilon}\right)^p\right)$ *(noisy) rows of* $\mathbf{A}$*, which includes a union* $\mathbf{T}$ *of* s k*-dimensional subspaces such that*

$$\mathbf{Pr}\left[\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{T})^p\right)^{\frac{1}{p}} \leq (1+\epsilon)\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{H})^p\right)^{\frac{1}{p}}\right] \geq \frac{2}{3}$$

for $p = \{1, 2\}$*, where* $\mathbf{H}$ *is the union of* s k*-dimensional subspaces that is the optimal solution to the projective clustering problem. The algorithm uses* $\text{poly}\left(d, k, s, \frac{1}{\epsilon}, \log n\right)$ *bits of space.*

Proof : Note that the optimal solution $\mathbf{H}$ of s k-dimensional subspaces is no better than the solution $\mathbf{A}_{ks}^*$ of the ks subspace approximation problem:

$$\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{A}_{ks}^*)^p\right)^{\frac{1}{p}} \leq \left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{H})^p\right)^{\frac{1}{p}},$$

By setting $\epsilon = \mathcal{O}(1)$ in Theorem 4.8, we can first obtain a set $\mathbf{V}$ of $\tilde{\mathcal{O}}\left((ks)^3 \cdot (ks)^{p+1}\right)$ rows of $\mathbf{A}$ such that

$$\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{V})^p\right)^{\frac{1}{p}} \leq 2\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{A}_{ks}^*)^p\right)^{\frac{1}{p}} \leq 2\left(\sum_{i=1}^{n} d(\mathbf{A}_i, \mathbf{H})^p\right)^{\frac{1}{p}},$$

which satisfies the conditions of Theorem 4.9. This can be done with arbitrarily high constant probability by taking $\mathbf{V}$ to be a number of independent instances of the algorithm in Theorem 4.8.

We now simulate Algorithm 8 by iteratively sampling $r := \tilde{\mathcal{O}}\left(\left(\frac{k^2}{\epsilon}\right)^p \frac{k^4 s}{\epsilon^2}\right)$ rows projected away from $\mathbf{V}$. The rest of the proof uses the same template as Theorem 4.8. We first describe the case where $p = 2$. By using $\text{poly}(r, \log n)$ independent copies of Algorithm 3 with sufficiently small error parameter, we can iteratively sample rows with probability proportional to their squared distance away from the span of the previous rows and $\mathbf{V}$. By Theorem 3.4, it follows that the total variation distance of the sampled rows $\mathbf{S}$ using the $L_{2,2}$ samplers is some small constant $\mathcal{O}(\epsilon)$ from the output distribution of Theorem 4.9. By Corollary 3.5, the objective on the sampled output is within $(1 + \mathcal{O}(\epsilon))$ of the offline adaptive sampler. Hence, the existence of a union $\mathbf{T}$ of s k-dimensional subspaces that is a good approximation of the optimal solution follows from Theorem 4.9. Since the probability of failure of Theorem 4.9 is at most $\frac{1}{4}$, then for sufficiently small ϵ, it holds that $(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{T})^p)^{\frac{1}{p}} \leq (1+\epsilon)(\sum_{i=1}^n d(\mathbf{A}_i, \mathbf{H})^p)^{\frac{1}{p}}$ with probability at least $\frac{2}{3}$. By Theorem 4.8, we can obtain $\mathbf{V}$ using $\text{poly}\left(d, k, s, \frac{1}{\epsilon}, \log n\right)$ bits of space. By Theorem 3.4, we need $\text{poly}(d, k, \frac{1}{\epsilon}, \log n, \log \frac{k}{\epsilon})$ bits of space to sample the r rows of $\mathbf{A}$, so the space complexity follows.

For $p = 1$, we instead use $\text{poly}(r, \log n)$ independent copies of the $L_{1,2}$ sampler Algorithm 10. The same argument then follows using Theorem A.7 to bound the total variation distance from the output distribution of Theorem 4.9 by $\mathcal{O}(\epsilon)$ and Corollary A.8 to bound the objective on the sampled output compared to that of the offline adaptive sampler. □

4.4 Volume Maximization

We now show that our sampling procedure can also be used to give turnstile streaming algorithms for volume maximization, where the inputs are a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives as a data stream and a parameter k for the number of selected rows, and the goal is to output k rows $\mathbf{r}_1, \ldots, \mathbf{r}_k$ of $\mathbf{A}$ that maximize $\text{Vol}(\mathbf{R})$, where $\mathbf{R} = \mathbf{r}_1 \circ \ldots \circ \mathbf{r}_k$. A possible approach to the volume maximization problem in an offline model is the greedy algorithm, which repeatedly chooses the row with the largest distance from the subspace spanned by the rows that have already been selected, for k steps. [ÇM09] shows that this offline greedy algorithm gives a $k!$-approximation to the volume maximization problem. In fact, their analysis also implies that an offline *approximate* greedy algorithm gives a good approximation to the volume maximization problem.

Theorem 4.11 *[ÇM09] Given a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and an integer $k > 0$, let $\mathbf{G}$ be a set of k rows chosen by the approximate greedy algorithm that repeatedly chooses a (noisy) row whose distance from the subspace spanned by the rows that have already been chosen is within a multiplicative α factor of the largest distance of a row to the subspace. Let $\mathbf{V}$ be a set of k rows of $\mathbf{A}$ with the maximum volume. Then $\text{Vol}(\mathbf{V}) \leq (\alpha^k) k! \cdot \text{Vol}(\mathbf{G})$.*

Lemma 4.12 *Let $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\alpha > 1$ be an approximation factor. Let $\mathbf{r}_j$ be the row selected by Algorithm 9 in round j and let $\mathbf{R}_j = \mathbf{r}_1 \circ \ldots \circ \mathbf{r}_j$ for each $j \in [k]$ and $\mathbf{R}_0$ be the all zeros matrix. For each $j \in [k]$, let $\mathbf{m}_j$ be the row the maximizes $\left\|\mathbf{A}_i(\mathbb{I} - \mathbf{R}_{j-1}^\dagger \mathbf{R}_{j-1})\right\|_2$. Then $\|\mathbf{r}_j\|_2 \geq \frac{1}{2\alpha}\|\mathbf{m}_j\|_2$ with probability at least $1 - \frac{1}{4k} - \frac{1}{\text{poly}(n)}$.*

Proof : Let $\mathbf{Y}_j = \mathbb{I} - \mathbf{R}_j^\dagger \mathbf{R}_j$ for each $j \in [k]$. Suppose $\|\mathbf{m}_j\|_2^2 \geq \frac{\alpha^2}{4nk}\|\mathbf{A}\mathbf{Y}_{j-1}\|_F^2$ so that $\mathbf{m}_j$ is $\frac{\alpha}{2\sqrt{nk}}$-heavy with respect to $\mathbf{A}\mathbf{Y}_{j-1}$. Each CountSketch-M data structure in Algorithm 9 maintains

Algorithm 9 Volume Maximization

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives as a stream $\mathbf{A}_1, \ldots, \mathbf{A}_n \in \mathbb{R}^d$, parameter k for number of rows, approximation factor $\alpha > 1$.

Output: k Noisy and projected rows of $\mathbf{A}$.

Create instances $\mathcal{A}_1, \ldots, \mathcal{A}_k$, where each $\mathcal{A}_j$ for $j \in [k]$ is $\mathcal{O}(\log n \log k)$ copies of the $L_{2,2}$ sampler of Algorithm 3 with error parameter $\epsilon = \frac{1}{4}$.
Create instances CountSketch-M$_1, \ldots,$ CountSketch-M$_k$ with $\Theta\left(\frac{nk}{\alpha^2}\right)$ buckets.
Create instances AMS-M$_1, \ldots,$ AMS-M$_k$ with error parameter $\mathcal{O}(1)$.
Let $\mathbf{M}$ be empty $0 \times d$ matrix.
Streaming Stage:
for each row $\mathbf{A}_i$ **do**
    Update each sketch $\mathcal{A}_1, \ldots, \mathcal{A}_k$.
    Update each sketch AMS-M$_1, \ldots,$ AMS-M$_k$.
    Update each sketch CountSketch-M$_1, \ldots,$ CountSketch-M$_k$.
Post-processing Stage:
for $j = 1$ to $j = k$ **do**
    Post-processing matrix $\mathbf{P} \leftarrow \mathbb{I} - \mathbf{M}^\dagger \mathbf{M}$.
    Update $\mathcal{A}_j$, AMS-M$_j$, CountSketch-M$_j$ with post-processing matrix $\mathbf{P}$.
    if AMS-M$_j$ and CountSketch-M$_j$ find a (noisy) row $\mathbf{r}$ of $\mathbf{AP}$ with $\|\mathbf{r}\|_2^2 \geq \frac{\alpha^2}{4nk} \|\mathbf{AP}\|_F^2$ **then**
        Append the (noisy) row $\mathbf{r}_j$ with the largest norm to $\mathbf{M}$: $\mathbf{M} \leftarrow \mathbf{M} \circ \mathbf{r}_j$.
    else
        Let $\mathbf{r}_j$ be the (noisy) row of $\mathbf{AP}$ sampled by $\mathcal{A}_j$.
        Append $\mathbf{r}_j$ to $\mathbf{M}$: $\mathbf{M} \leftarrow \mathbf{M} \circ \mathbf{r}_j$.
return $\mathbf{M}$.

$\Theta\left(\frac{nk}{\alpha^2}\right)$ buckets and $\mathbf{m}_j$ is a heavy row for $\mathbf{AY}_{j-1}$, so Algorithm 9 will output some noisy row $\mathbf{r}_j$ with $\|\mathbf{r}_j\|_2^2 \geq \frac{1}{2} \|\mathbf{m}_j\|_2^2$ for sufficiently small $\mathcal{O}(1)$ error parameter by each AMS-M data structure.

On the other hand, suppose $\|\mathbf{m}_j\|_2^2 < \frac{\alpha^2}{4nk} \|\mathbf{AY}_{j-1}\|_F^2$. Since $\mathbf{AY}_{j-1}$ contains n rows and $\epsilon = \frac{1}{4}$, the $L_{2,2}$ sampler will select an index $s \in [n]$ such that $\|\mathbf{A}_s \mathbf{Y}_{j-1}\|_2^2 \geq \frac{1}{8nk} \|\mathbf{AY}_{j-1}\|_F^2$ and output a row $\mathbf{r}_j$ such that $\|\mathbf{r}_j\|_2 \geq \frac{1}{\sqrt{2}} \|\mathbf{A}_s \mathbf{Y}_{j-1}\|_2$, with probability at least $1 - \frac{1}{4k} - \frac{1}{\text{poly}(n)}$ Therefore,

$$\|\mathbf{r}_j\|_2^2 \geq \frac{1}{16nk} \|\mathbf{AY}_{j-1}\|_F^2 > \frac{1}{4\alpha^2} \|\mathbf{m}_j\|_2^2,$$

which suffices to imply $\|\mathbf{r}_j\|_2 \geq \frac{1}{2\alpha} \|\mathbf{m}_j\|_2$. □

Theorem 4.13 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives in a turnstile data stream and an approximation factor $\alpha > 0$, there exists a one-pass streaming algorithm that outputs a set of k noisy rows of $\mathbf{A}$ that is an $\alpha^k(k!)$-approximation to volume maximization with probability at least $\frac{2}{3}$, using $\tilde{\mathcal{O}}\left(\frac{ndk^2}{\alpha^2}\right)$ bits of space.*

Proof : Recall that a single instance of the $L_{2,2}$ sampler of Algorithm 3 succeeds with probability $\frac{1}{\log n}$ for error parameter $\epsilon = \mathcal{O}(1)$. Thus by using $\mathcal{O}(\log n \log k)$ copies of the $L_{2,2}$ sampler of

Algorithm 3, the probability that Algorithm 9 successfully acquires a sample in each round is at least $1 - \mathcal{O}\left(\frac{1}{k}\right)$. Thus by Lemma 4.12 and a union bound, Algorithm 9 repeatedly chooses rows $\mathbf{r}_1, \ldots, \mathbf{r}_k$ whose distance from the subspace spanned by the previously chosen rows is within a multiplicative factor of 2α of the largest distance, with probability at least $\frac{2}{3}$. Let $\mathbf{R}$ be the parallelepiped spanned by $\mathbf{r}_1, \ldots, \mathbf{r}_k$. Thus by Lemma 4.11, $(2\alpha)^k (k!) \operatorname{Vol}(\mathbf{R}) \geq \operatorname{Vol}(\mathbf{V})$, where $\mathbf{V}$ is a set of k rows of $\mathbf{A}$ with the maximum volume. The result then follows from rescaling α.

Each CountSketch-M data structure maintains $\Theta\left(\frac{nk}{\alpha^2}\right)$ buckets of vectors with d entries, each with $\mathcal{O}(\log n)$ bits. Each AMS-M data structure uses $\mathcal{O}\left(d \log^2 n\right)$ bits of space. Each $L_{2,2}$ sampler uses $\mathcal{O}(\log n)$ buckets of vectors with d entries, each with $\mathcal{O}(\log n)$ bits. Since Algorithm 9 requires k instances of each data structure, then the total space complexity follows. □

5 Volume Maximization in the Row-Arrival Model

In this section, we consider the volume maximization problem on row-arrival streams. As before, we are given the rows of the matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ and a parameter k, and the goal is to output k rows of the matrix whose volume is maximized. Throughout the section, we will use the equivalent view that the stream consists of n points from $\mathbb{R}^d$.

5.1 Volume Maximization via Composable Core-sets.

We first observe that we can get an approximation algorithm for volume maximization in the row-arrival model by using algorithms of [IMGR20, IMGR19] for composable core-sets for volume maximization. We recall the definition of composable core-sets, such as given in [IMMM14].

Definition 5.1 (α-composable core-set) *Let $V \subset \mathbb{R}^d$ be an input set. Then a function $c : V \to W$, where $W \subset V$, is an α-composable core-set for an optimization problem with a maximization objective with respect to a function $f : 2^{\mathbb{R}^d} \to \mathbb{R}$ if for any collection of set $V_1, \ldots, V_m \subset \mathbb{R}^d$,*

$$f(c(V_1) \cup \ldots \cup c(V_m)) \geq \frac{1}{\alpha} f(V_1 \cup \ldots \cup V_m).$$

[IMGR20] gives composable core-sets for volume maximization.

Theorem 5.2 *[IMGR20] There exists a polynomial time algorithm for computing an $\tilde{\mathcal{O}}(k)^{k/2}$-composable core-set of size $\tilde{\mathcal{O}}(k)$ for the volume maximization problem.*

We can partition the stream into consecutive blocks and apply a core-set for each block to get a streaming algorithm for volume maximization.

Corollary 5.3 *There exists a one pass streaming algorithm in the row-arrival model that computes a $\tilde{\mathcal{O}}(k)^{k/(2\epsilon)}$-approximation to the volume maximization problem, using $\tilde{\mathcal{O}}\left(\frac{1}{\epsilon} n^{\epsilon} dk\right)$ space.*

Proof : Consider a b-ary tree over the stream with n leaves that correspond to the elements of the stream in the order they arrive. Let $b = n^{\epsilon}$ so that the height of the tree is $\frac{\log n}{\log b} = \frac{1}{\epsilon}$. For each node in the tree, as soon as all the elements corresponding to its subtree arrive in the stream, we build a core-set of size $\tilde{\mathcal{O}}(k)$ for the points using the algorithm of [IMGR20] in Theorem 5.2, and pass the core-set to the parent node. More precisely, when the node receives a composable core-set

from each of its b children, it computes a composable core-set over the union of the core-sets of its children and passes the new core-set on to its parent.

Then the k points reported by the root gives an $\left(\tilde{\mathcal{O}}\left(k\right)^{k/2}\right)^{1/\epsilon} = \tilde{\mathcal{O}}\left(k\right)^{k/(2\epsilon)}$ approximation to the volume maximization problem. Moreover, at each time step during the stream arrival, there is only one path of active nodes (nodes whose corresponding leaf nodes have arrived but not finished) in the tree from the root to the leaves. Each of the nodes on this active path might need to store a composable core-set of size $\tilde{\mathcal{O}}\left(k\right)$ for each of its b children. Since each point has dimension d, then the total memory usage of the algorithm is thus at most $\left(\frac{1}{\epsilon}\right) \cdot b \cdot \tilde{\mathcal{O}}\left(dk\right) = \tilde{\mathcal{O}}\left(\frac{1}{\epsilon} n^{\epsilon} dk\right)$. □

5.2 Exponential Dependence on d

In this section, we give a streaming algorithm whose space complexity depends exponentially on the dimension d. Our main tool is the ϵ-kernels of [AHPV05] improved by [Cha06] for directional width of a point set. We first define the concept of the directional width.

Definition 5.4 (Directional width [AHPV05]) *Given a point set $P \subset \mathbb{R}^d$ and a unit direction vector $\mathbf{x} \in \mathbb{R}$, the directional width of P with respect to $\mathbf{x}$ is defined to be $\omega(\mathbf{x}, P) = \max_{\mathbf{p}\in P}\langle \mathbf{x}, \mathbf{p}\rangle - \min_{\mathbf{p}\in P}\langle \mathbf{x}, \mathbf{p}\rangle$.*

The following lemma shows the existence of core-sets with size exponential in the directional width of a point set but independent of the number of points.

Lemma 5.5 *[Cha06] For any $0 < \epsilon < 1$, there exists a one pass streaming algorithm that computes an ϵ-core-set $Q \subseteq P$, such that for any unit direction vector $\mathbf{x} \in \mathbb{R}^d$, we have $\omega(\mathbf{x}, Q) \geq (1 - \epsilon)\omega(\mathbf{x}, P)$. Moreover the algorithm uses space $\mathcal{O}\left(\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)^{d-1}\right)$ and the core-set has size $|Q| \leq \mathcal{O}\left(\frac{1}{\epsilon^{d-1}}\right)$.*

Now let us define the the directional height (which was implicitly defined in [IMGR19]), and its relation to directional width.

Definition 5.6 (Directional height) *Given a point set $P \subset \mathbb{R}^d$ and a unit direction vector $\mathbf{x} \in \mathbb{R}$, the directional height of P with respect to x is defined to be $h(\mathbf{x}, P) = \max_{\mathbf{p}\in P} |\langle \mathbf{x}, \mathbf{p}\rangle|$.*

Lemma 5.7 *For a point set $P \subset \mathbb{R}^d$, an ϵ-core-set Q for directional width is a (2ϵ)-core-set for directional height.*

Proof : Let $\mathbf{x}$ be a unit direction vector in $\mathbb{R}^d$. Let $\mathbf{p}_1 = \text{argmax}_{\mathbf{p}\in P}\langle \mathbf{x}, \mathbf{p}\rangle$ and $\mathbf{p}_2 = \text{argmin}_{\mathbf{p}\in P}\langle \mathbf{x}, \mathbf{p}\rangle$, and let $\mathbf{q}_1 = \text{argmax}_{\mathbf{p}\in Q}\langle \mathbf{x}, \mathbf{p}\rangle$ and $\mathbf{q}_2 = \text{argmin}_{\mathbf{p}\in Q}\langle \mathbf{x}, \mathbf{p}\rangle$. Now consider two cases.

- First suppose that $\langle \mathbf{x}, \mathbf{p}_2\rangle \geq 0$. In this case, we have that $h(\mathbf{x}, P) = \langle \mathbf{x}, \mathbf{p}_1\rangle$. Since Q is an ϵ-core-set for directional width, then $\langle \mathbf{x}, \mathbf{p}_1\rangle - \langle \mathbf{x}, \mathbf{p}_2\rangle \leq (1+\epsilon)\left(\langle \mathbf{x}, \mathbf{q}_1\rangle - \langle \mathbf{x}, \mathbf{q}_2\rangle\right) \leq (1+\epsilon)\left(\langle \mathbf{x}, \mathbf{q}_1\rangle - \langle \mathbf{x}, \mathbf{p}_2\rangle\right)$. Therefore, $\langle \mathbf{x}, \mathbf{q}_1\rangle \geq (1-\epsilon)\langle \mathbf{x}, \mathbf{p}_1\rangle$ so that $h(\mathbf{x}, Q) \geq (1-\epsilon)h(\mathbf{x}, P)$. The case of $\langle \mathbf{x}, \mathbf{p}_1\rangle \leq 0$ can be handled similarly.

- On the other hand, suppose $\langle \mathbf{x}, \mathbf{p}_1 \rangle > 0$ and $\langle \mathbf{x}, \mathbf{p}_2 \rangle < 0$. Assume without loss of generality that $|\langle \mathbf{x}, \mathbf{p}_1 \rangle| \geq |\langle \mathbf{x}, \mathbf{p}_2 \rangle|$. Then $h(\mathbf{x}, P) = \langle \mathbf{x}, \mathbf{p}_1 \rangle$. As Q is an ϵ-core-set for the width, then we also have $\langle \mathbf{x}, \mathbf{q}_1 \rangle \geq (1-2\epsilon)\langle \mathbf{x}, \mathbf{p}_1 \rangle = (1-2\epsilon)h(\mathbf{x}, P)$. Therefore, $h(\mathbf{x}, Q) \geq (1-2\epsilon)h(\mathbf{x}, P)$.

□

Finally we define k-directional height and observe that a core-set for directional height leads to a core-set for k-directional height.

Definition 5.8 (k-directional height [IMGR19]) *Given a point set $P \subset \mathbb{R}^d$ and a $(k-1)$-dimensional subspace $\mathcal{H}$, $\mathbf{x} \in \mathbb{R}^d$, the k-directional height of P with respect to $\mathcal{H}$ is defined to be $h_k(\mathcal{H}, P) = \max_{\mathbf{x} \in \mathcal{H}^\perp} h(\mathbf{x}, P)$, where $\mathcal{H}^\perp$ is the orthogonal complement of $\mathcal{H}$.*

Observation 5.9 *For a point set $P \subset \mathbb{R}^d$, an ϵ-core-set Q for directional height is an ϵ-core-set for k-directional height.*

In fact, a core-set for directional height is stronger than a core-set for k-directional height since it preserves the height in all directions $x \in \mathcal{H}^\perp$; thus their maximum is preserved too.

Lemma 5.10 ([IMGR19]) *For a point set $P \subset \mathbb{R}^d$, let Q be its ϵ-core-set for k-directional height. Then the solution of the k-volume maximization on Q is within a factor of $1/(1-\epsilon)^k$ of the solution of k-volume maximization over P.*

Lemma 5.11 *There exists a one pass streaming algorithm that outputs a 2^k-approximation to volume maximization, using $\mathcal{O}\left(8^d\right)$ space.*

Proof : We use the streaming algorithm of Lemma 5.5 on the the set of rows of $\mathbf{A}$ with $\epsilon = \frac{1}{4}$, which gives a $\frac{1}{4}$-core-set $Q \in \mathbb{R}^{4^d \times d}$ for the directional width using space $\mathcal{O}\left(8^d\right)$. Using Lemma 5.7 and Observation 5.9, this will be a $\frac{1}{2}$-core-set for the k-directional height of the rows of $\mathbf{A}$. Finally, using Lemma 5.10, the optimal solution of Q approximates the maximum volume over rows of $\mathbf{A}$ within a factor of 2^k. □

5.3 Dimensionality Reduction

In this section, we show how to reduce the dimension of each point to $d = \mathcal{O}(k)$. Using the result of the previous section, this will give a trade-off algorithm, improving over Corollary 5.3 in terms of the dependence on the parameter $\frac{1}{\epsilon}$. We prove the following lemma.

Lemma 5.12 *Let C be a trade-off parameter such that $1 < C < (\log n)/k$. There exists a randomized streaming algorithm that uses $\mathcal{O}\left(n^{\mathcal{O}(1/C)} d\right)$ space to computes a subset of size d whose volume maximization solution is a $\mathcal{O}(Ck)^{k/2}$ approximation to the optimal solution.*

Note that this result improves the algorithm of Corollary 5.3 for $\frac{k}{\log n} < \epsilon$: setting $C = 1/\epsilon$, this provides an algorithm with memory usage of $\mathcal{O}(n^\epsilon)$, with approximation factor of $\mathcal{O}(k/\epsilon)^{k/2}$, improving the dependence of the approximation factor on $\frac{1}{\epsilon}$ by an exponential factor.

We now continue with the proof of Lemma 5.12. Consider a random matrix $\mathbf{G} \in \mathbb{R}^{d \times r}$, for $r = \frac{\log n}{C}$, where each of its entries is an independent and identically distributed (i.i.d.) random

variable drawn from the Gaussian distribution $\mathcal{N}(0, 1/r)$. Consider the matrix $\mathbf{AG}$ and observe that its rows exist in an r dimensional space. Therefore, we can use the streaming algorithm of Lemma 5.11 to find a subset of k rows of $\mathbf{AG}$ that serves as a good estimator for the maximum volume. This approach requires $\mathcal{O}\left(2^{3r}\right) = \mathcal{O}\left(1\right)^{\mathcal{O}((\log n)/C)} = n^{\mathcal{O}(1/C)}$ memory space.

Lemma 5.13 *Let $\mathbf{G} \in \mathbb{R}^{d\times r}$, for $r = \Omega\left(\frac{\log n}{C}\right)$, have each of its entries is drawn i.i.d from the Gaussian distribution $\mathcal{N}(0, 1/r)$. With high probability, the maximum volume of the optimal k-subset of the rows of $\mathbf{AG}$ is within 2^k of the maximum volume of the optimal k-subset of the rows of $\mathbf{A}$.*

Proof : Let $P = \{\mathbf{p}_1, \cdots, \mathbf{p}_k\}$ be the subset of k points among the rows of $\mathbf{A}$ that maximizes the volume. Moreover, let $R = \{\mathbf{r}_2, \cdots, \mathbf{r}_k\}$ where r_i is the projection of $\mathbf{p}_i$ onto the subspace spanned by the points in $\{\mathbf{p}_1, \cdots, \mathbf{p}_{i-1}\}$ for each $i \in [k]$. Using the Johnson-Lindenstrauss Lemma with $\epsilon = \frac{1}{2}$ on the set of $2k$ points $P \cup R$, the lengths of each row of $\mathbf{A}$ is only distorted by a factor of at most two compared to the length of the corresponding row in $\mathbf{AG}$ as long as $r = \Omega(\log k)$, which is always the case for $C < (\log n)/k$. Hence, the maximum volume k-subset of rows of $\mathbf{A}$ does not decrease by more than a factor of 2^k with high probability. □

We now show that for every other subset S of k points from the rows of $\mathbf{A}$, their volume does not increase by much with very high probability, so that we can union bound over all such subsets. The following lemma may seem counterintuitive at first, since the parameter C appears in the approximation factor but not the probability. However, recall that the algorithm pays for the parameter C in the space of the algorithm.

Lemma 5.14 *Let S be a subset of size k from the rows of $\mathbf{A}$. Then after applying $\mathbf{G}$, its volume does not increase by more than a factor of $(\sqrt{2Ck}+2)^k = \mathcal{O}\left(Ck\right)^{k/2}$ with probability at least $1-n^{-k}$.*

Proof : Let $\mathbf{R}$ be the $k \times d$ submatrix corresponding to the rows of $\mathbf{A}$ that are in S. The volume of $\mathbf{R}$ after the embedding is equivalent to $\sqrt{\det(\mathbf{RGG}^\top\mathbf{R}^\top)}$. Now consider the singular value decomposition of $\mathbf{R} = \mathbf{U}\Sigma\mathbf{V}^\top$ where $\mathbf{U}$ and Σ are $k \times k$ (as otherwise the original volume would have been 0), and $\mathbf{V}$ is $d \times k$. Then we can rewrite this volume as $\sqrt{\det(\mathbf{U}\Sigma\mathbf{V}^\top\mathbf{GG}^\top\mathbf{V}\Sigma\mathbf{U}^\top)} = \sqrt{\det(\Sigma\mathbf{HH}^\top\Sigma)}$, where $\mathbf{H} = \mathbf{V}^\top\mathbf{G}$ is a $k \times r$ matrix with entries drawn i.i.d. from the Gaussian distribution $\mathcal{N}(0, 1/r)$, due to the rotational invariance of $\mathbf{G}$. Then $\sqrt{\det(\Sigma\mathbf{HH}^\top\Sigma)}$ is just the product of the singular values of $\mathbf{H}^\top\Sigma$.

Claim 5.15 *[Stc] The i^{th} singular value of $\mathbf{H}^\top\Sigma$ is at most $\|\mathbf{H}\|_2 \Sigma_{i,i}$.*

Proof : We include the proof for completeness. Observe that for any vector $\mathbf{x} \in \mathbb{R}^k$,

$$\frac{\langle\Sigma\mathbf{HH}^\top\Sigma\mathbf{x}, \mathbf{x}\rangle}{\langle\mathbf{x}, \mathbf{x}\rangle} = \frac{\mathbf{x}^\top\Sigma\mathbf{HH}^\top\Sigma\mathbf{x}}{\langle\mathbf{x}, \mathbf{x}\rangle} = \frac{\left\|\mathbf{H}^\top\Sigma\mathbf{x}\right\|^2}{\langle\mathbf{x}, \mathbf{x}\rangle} \leq \frac{\left\|\mathbf{H}^\top\right\|_2^2\mathbf{x}^\top\Sigma^2\mathbf{x}}{\langle\mathbf{x}, \mathbf{x}\rangle} = \left\|\mathbf{H}^\top\right\|_2^2 \frac{\langle\Sigma\mathbf{x}, \mathbf{x}\rangle}{\langle\mathbf{x}, \mathbf{x}\rangle}.$$

Note that the i^{th} singular value of $\mathbf{H}^\top\Sigma$ is the square root of the i^{th} eigenvalue of $\Sigma\mathbf{HH}^\top\Sigma$. Thus the min-max theorem for the characterization of the eigenvalues of $\Sigma\mathbf{HH}^\top\Sigma$ shows that for any i, the ith singular value of $\mathbf{H}^\top\Sigma$ can be bounded by $\|\mathbf{H}\|_2$ times the i^{th} singular value of Σ, which is $\Sigma_{i,i}$. □

Thus, it follows that the volume of $\mathbf{RG}$ is at most $\|\mathbf{H}\|_2^k \operatorname{Vol}(\mathbf{R})$, where $\operatorname{Vol}(\mathbf{R}) = \prod_{i=1}^k \Sigma_{i,i}$ is the original volume of $\mathbf{R}$. In other words, right multiplication by $\mathbf{G}$ increases the volume of $\mathbf{R}$ by at most $\|\mathbf{H}\|_2^k$ in the embedding. We now bound $\|\mathbf{H}\|_2^k$ using the following.

Lemma 5.16 (Corollary 35 of [Ver10]) *Let $\mathbf{H}$ be a matrix of size $k \times r$ whose entries are independent standard normal random variables. Then for every $t \geq 0$, it follows that $\sigma_{\max} \leq \sqrt{k}+\sqrt{r}+t$ with probability at least $1 - 2\exp(-t^2/2)$, where $\sigma_{\max}$ is the largest singular value of $\mathbf{H}$.*

By Lemma 5.16 and the fact that entries of $\mathbf{H}$ have variance $\frac{1}{r}$, we have that $\|\mathbf{H}\|_2 \leq 1 + \sqrt{k/r} + t/\sqrt{r} \leq 2 + t/\sqrt{r}$ with probability at least $1 - 2e^{-t^2/2}$. Equivalently, $\|\mathbf{H}\|_2 \leq 2 + s$ with probability at least $1 - 2^{-s^2 r/2}$. Setting $s = \sqrt{2Ck}$ and using $r = (\log n)/C$, we have that the volume of $\mathbf{R}$ increases by at most a $(\sqrt{2Ck} + 2)^k$ factor after the embedding, with probability at least $1 - 2^{-2Ck(\log n)/(2C)} = 1 - n^{-k}$. □

Thus we can union bound over all subsets of size k of the n rows of $\mathbf{A}$, to argue that with high probability, none of them will have a volume increase by more than a $(\sqrt{4Ck} + 2)^k$ factor. This completes the proof of Lemma 5.12.

6 Volume Maximization Lower Bounds

In this section, we complement our adaptive sampling based volume maximization algorithms, i.e., Theorem 4.13, with lower bounds on turnstile streams that are tight up to lower order terms. Our lower bounds hold even for multiple passes through the turnstile stream. Additionally, we give a lower bound for volume maximization in the random order row-arrival model that is competitive with the algorithms in Section 5.

6.1 Turnstile Streams

We first consider lower bounds for turnstile streams. In the Gap ℓ_∞ problem, Alice and Bob are given vectors $\mathbf{x}$ and $\mathbf{y}$ respectively with $x, y \in [0, m]^n$ for some $m > 0$ and the promise that either $|x_i - y_i| \leq 1$ for all $i \in [n]$ or there exists some $i \in [n]$ such that $|x_i - y_i| = m$. The goal is for Alice and Bob to perform some communication protocol to decide whether there exists an index $i \in [n]$ such that $|x_i - y_i| = m$, possibly over multiple rounds of communication. To succeed with probability $\frac{8}{9}$, Alice and Bob must use at least $\Omega\left(\frac{n}{m^2}\right)$ bits of total communication, even if they can communicate over multiple rounds.

Theorem 6.1 *[BJKS04] Any protocol that solves the Gap ℓ_∞ problem with probability at least $\frac{8}{9}$ requires $\Omega\left(\frac{n}{m^2}\right)$ total bits of communication.*

We first reduce an instance of the Gap ℓ_∞ problem to giving an α-approximation to the volume maximization problem when $k = d = 1$.

Theorem 6.2 *Any p-pass turnstile streaming algorithm that gives an α-approximation to the volume maximization problem requires $\Omega\left(\frac{n}{p\alpha^2}\right)$ bits of space.*

Proof : Let $\mathcal{A}$ be a turnstile streaming algorithm that provides an α-approximation to the volume maximization problem. Let $d = 1$ and $k = 1$ in the volume maximization problem and $\mathbf{M} \in \mathbb{R}^{n \times 1}$ be the underlying matrix. Given an instance of the Gap ℓ_∞ problem with $m = \alpha + 1$, suppose Alice has vector $\mathbf{x}$ and Bob has vector $\mathbf{y}$. Alice creates a stream with the coordinates of $\mathbf{x}$ so that $\mathbf{M} = \mathbf{x}$ at the end of the stream. Alice then passes the state of the algorithm to Bob, who updates the stream with the coordinates of $\mathbf{y}$ so that $\mathbf{M} = \mathbf{x} - \mathbf{y}$ at the end of the stream. Note

that if $|x_i - y_i| \leq 1$ for all $i \in [n]$, then the maximum possible volume is 1 (in this case the volume is just the norm as $k = 1$), whereas if $|x_i - y_i| = m = \alpha + 1$ for some $i \in [n]$, then the maximum volume is equal to $\alpha + 1$. Since $\mathcal{A}$ is an α-approximation, $\mathcal{A}$ can differentiate between these two cases and solve the Gap ℓ_∞ problem. Thus by Theorem 6.1, $\mathcal{A}$ uses $\Omega\left(\frac{n}{\alpha^2}\right)$ bits of space over the p passes and hence at least $\Omega\left(\frac{n}{p\alpha^2}\right)$ bits of space. □

Corollary 6.3 *Any p-pass turnstile streaming algorithm that gives an α^k-approximation to the volume maximization problem requires $\Omega\left(\frac{n}{kp\alpha^2}\right)$ bits of space.*

Proof : We generalize the above construction to the case of $k = d > 1$ for any value of $k > 1$. Consider the same instance $\mathbf{M} \in \mathbb{R}^{n/k \times 1}$ as the above lemma but with Gap ℓ_∞ problem of size $\frac{n}{k}$ instead of n. Now we construct a new instance $\mathbf{M}' \in \mathbb{R}^{n \times k}$ as follows. For each row $i \in [n/k]$ and $j \in [k]$, let $\mathbf{M}'_{(i-1)k+j,j} = \mathbf{M}_{i,1}$ and let all the other entries be equal to 0. In words, Alice and Bob embed the problem k times across the d columns for $k = d$. Thus in one case the maximum possible volume is 1, while in the other case the maximum volume is equal to $(\alpha + 1)^k$. □

6.2 Row-Arrival Model

We now present streaming lower bounds for the row-arrival model. We consider a version of the distributional set-disjointness communication problem $\mathsf{DISJ}_{n,d}$ in which Alice is given the set of vectors $U = \{u_1, \ldots, u_n\}$ and Bob is given the set of vectors $V = \{v_1, \ldots, v_n\}$. With probability $\frac{1}{4}$, U and V are chosen uniformly at random among all instances with the following properties:

- Any vector in $U \cup V$ is in $\{0,1\}^d$ and moreover its weight is exactly $\frac{d}{2}$,
- $U \cap V$ is non-empty.

This forms the NO case. Otherwise with probability $\frac{3}{4}$, U and V are chosen uniformly at random among all instances with the following properties:

- Any vector in $U \cup V$ is in $\{0,1\}^d$ and moreover its weight is exactly $\frac{d}{2}$,
- $U \cap V = \emptyset$.

This forms the YES case. The goal is for Alice and Bob to perform some communication protocol to decide whether the instance is a YES or a NO instance, i.e., whether $U \cap V = \emptyset$, possibly over multiple rounds of communication.

The following result, originally due to Razborov [Raz92] and generalized by others [KS92, WZ12], lower bounds the communication complexity of any randomized protocol that solves $\mathsf{DISJ}_{n,d}$ with probability at least $\frac{7}{8}$, even given multiple rounds of communication.

Theorem 6.4 *[Raz92, KS92, WZ12] Any protocol for $\mathsf{DISJ}_{n,d}$ that fails with probability at most $\frac{1}{6}$ requires $\Omega(n)$ bits of total communication.*

We first reduce an instance of the distributional set-disjointness problem to giving a C^k approximation to the volume maximization problem in the row-arrival model when the order of the stream can be adversarial.

Theorem 6.5 *For constant p and $C = \frac{16}{15}$, any p-pass streaming algorithm that outputs a C^k approximation to the $(2k)$-volume maximization problem in the row-arrival model with probability at least $\frac{8}{9}$ requires $\Omega(n)$ bits of space.*

Proof : Suppose Alice and Bob have an instance of $\mathsf{DISJ}_{n/2k,d/k}$, so that Alice has a set $U = \{u_1, \ldots, u_{n/2k}\}$ of vectors of $\{0,1\}^{d/k} \subset \mathbb{R}^{d/k}$ and Bob has a set $V = \{v_1, \ldots, v_{n/2k}\}$ of vectors of $\{0,1\}^{d/k} \subset \mathbb{R}^{d/k}$. Alice creates the matrix $\mathbf{A} \in \mathbb{R}^{\frac{n}{2k} \times \frac{d}{k}}$ by setting row $r \in \left[\frac{n}{2k}\right]$ of $\mathbf{A}$ to be precisely u_r. Alice then creates a $\frac{n}{2} \times d$ block diagonal matrix $\mathbf{M}_A$ to be the direct sum $\mathbf{A} \oplus \mathbf{A} \oplus \ldots \oplus \mathbf{A}$, where there are k terms in the direct sum.

For each $r \in \left[\frac{n}{2k}\right]$, Bob takes vector v_r and creates a new vector w_r by setting w_r to be the complement of v_r, so that w_r is the unique binary vector with weight $\frac{k}{2}$, but $\langle w_r, v_r \rangle = 0$. Bob then creates the matrix $\mathbf{B} \in \mathbb{R}^{\frac{n}{2k} \times \frac{d}{k}}$ by setting row $r \in \left[\frac{n}{2k}\right]$ of $\mathbf{B}$ to be precisely w_r. Bob also creates a $\frac{n}{2} \times d$ block diagonal matrix $\mathbf{M}_B$ to be the direct sum $\mathbf{B} \oplus \mathbf{B} \oplus \ldots \oplus \mathbf{B}$, where there are k terms in the sum. Finally, define $\mathbf{M} \in \mathbb{R}^{n \times d}$ to be the matrix $\mathbf{M}_A$ stacked on top of $\mathbf{M}_B$ so that

$$\mathbf{M}_A = \begin{bmatrix} \mathbf{A} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{A} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{A} \end{bmatrix}, \qquad \mathbf{M}_B = \begin{bmatrix} \mathbf{B} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{B} \end{bmatrix}, \qquad \mathbf{M} = \begin{bmatrix} \mathbf{M}_A \\ \mathbf{M}_B \end{bmatrix}.$$

Let $C = \frac{16}{15}$ and suppose there exists a p-pass streaming algorithm $\mathcal{A}$ that computes a C^k-approximation to the $2k$-volume maximization problem with probability at least $1 - \frac{1}{9}$ while using $o\left(\frac{n}{p}\right)$ space. Let $\mathcal{E}_1$ denote the event that $\mathcal{A}$ correctly computes a C^k approximation to the $2k$-volume maximization problem, which by the statement of the theorem holds with probability $8/9$. We claim that if $\mathcal{E}_1$ occurs, then Alice and Bob can use $\mathcal{A}$ to construct a p round communication protocol that solves $\mathsf{DISJ}_{n/2k,d/k}$ with high probability using $o(n)$ total communication, which contradicts the $\Omega(n)$ communication complexity of solving $\mathsf{DISJ}_{n/2k,d/k}$ for constant k.

Alice can create a stream S by inserting the rows of $\mathbf{M}_A$ into the stream, since Alice has knowledge of the rows $\{u_r\}_{r \in [n/2k]}$. Alice can run $\mathcal{A}$ on this stream S and then pass the state of the algorithm to Bob, who appends the rows of $\mathbf{M}_B$ onto the stream S and runs $\mathcal{A}$ on this portion of the stream, starting with the state passed from Alice. Bob then passes the state of the algorithm back to Alice, completing both a single communication round as well as a single pass of $\mathcal{A}$ through S. Alice and Bob can repeatedly pass the state of the algorithm between each other, to emulate passes over the stream. Thus after p rounds of communication, $\mathcal{A}$ will have completed p passes over S and output an approximation $\hat{\mathbf{D}}$ to the $2k$-volume maximization problem.

We first claim that in a NO instance of $\mathsf{DISJ}_{n/2k,d/k}$, then with high probability, the optimal solution $\mathbf{D}$ to the $2k$-volume maximization problem contains two orthogonal rows a and b whose nonzero entries are between columns $\frac{(i-1)d}{k} + 1$ and $\frac{id}{k}$ for each $i \in [k]$. In a NO instance, when Alice embeds vector v into a row a, Bob embeds the complement of v into a row b. Hence, there are $2k$ orthogonal vectors in the NO case, so the volume of the parallelpiped spanned by $2k$ vectors is maximized with the choice of the $2k$ orthogonal vectors, in which case the determinant is $\left(\frac{d}{2k}\right)^{2k}$.

To analyze the YES instance, we first let $\mathcal{E}_2$ denote the event that there exist two orthogonal rows a and b in $\mathbf{M}$ whose nonzero entries are between columns $\frac{(i-1)d}{k} + 1$ and $\frac{id}{k}$ for some $i \in [k]$ and either both $a \in \mathbf{M}_A$ and $b \in \mathbf{M}_B$ or $a \in \mathbf{M}_B$ and $b \in \mathbf{M}_A$. In other words, rows a and b were inserted by different people. For a fixed $i \in [k]$, the probability that rows a and b were both

inserted by Alice is the probability that two vectors among $\frac{n}{2k}$ vectors of $\mathbb{R}^{\frac{d}{k}}$ with weight $\frac{d}{2k}$ are orthogonal. By symmetric reasoning with Bob inserting both vectors and removing the instances where Alice and Bob have "random" orthogonal vectors, we note that

$$\mathbf{Pr}\left[\neg\mathcal{E}_2\right] \geq 1 - 2\binom{n/2k}{2}\frac{1}{\binom{d/k}{d/2k} - n} \geq 1 - \frac{n^2}{4k^2}\frac{1}{2^{d/2k} - n}.$$

Note that since $\mathbf{M}_A$ and $\mathbf{M}_B$ are each direct sums of k instances of $\mathbf{A}$ and $\mathbf{B}$, we do not need to take a union bound over all indices $i \in [k]$, though even such a union bound would still cause $\mathcal{E}_2$ to hold with low probability.

Moreover, by symmetry the volume of the spanning parallelpiped is maximized when the $2k$ rows are k direct sums of two rows[1]. Let $\mathcal{E}_3$ be the event all the rows u and v among the rows in both $\mathbf{A}$ and $\mathbf{B}$ intersect by more than $\frac{d}{8k}$ coordinates. We claim that $\mathcal{E}_3$ holds with high probability in a YES instance. If that were true, then the maximum volume in the YES case is less than

$$\left(\left(\frac{d}{2k}\right)^2 - \left(\frac{d}{8k}\right)^2\right)^k = \left(\frac{15d^2}{64k^2}\right)^k,$$

which would show a separation between the YES and NO instances, since the volume in the NO case is $\left(\frac{d^2}{4k^2}\right)^k$. Thus for $C = \frac{16}{15}$ and conditioning on $\mathcal{E}_1$, $\neg\mathcal{E}_2$ and $\mathcal{E}_3$, Alice and Bob can use any C^k approximation algorithm to the volume maximization problem differentiate between a YES instance and a NO instance of $\mathsf{DISJ}_{n/2k,d/k}$.

It remains to prove the claim that $\mathcal{E}_3$ holds with high probability in a YES instance.

Claim 6.6 *In a YES instance, all the rows given to Alice and Bob intersect by more than $\frac{d}{8k}$ coordinates with probability at least $1 - \frac{n^2}{k^2}\sqrt{\frac{\gamma^2 d}{k}}\frac{8^{d/3k}}{9^{3d/8k}}$, for some fixed constant γ. That is,*

$$\mathbf{Pr}\left[\mathcal{E}_3\right] \geq 1 - \frac{n^2}{k^2}\sqrt{\frac{\gamma^2 d}{k}}\frac{8^{d/3k}}{9^{3d/8k}}.$$

Proof : For a fixed pair of vectors a and b, the probability that a and b intersect in at most $\frac{d}{8k}$ coordinates without Alice and Bob having "random" orthogonal vectors is at most

$$\frac{d}{8k}\frac{\binom{d/2k}{d/8k}\binom{d/2k}{3d/8k}}{\binom{d/k}{d/2k} - n} > \frac{d}{16k}\frac{\binom{d/2k}{d/8k}\binom{d/2k}{3d/8k}}{\binom{d/k}{d/2k}}, \tag{1}$$

for sufficiently large d/k. By Stirling's approximation, there exists a fixed constant γ such that Equation 1 is at most

$$\frac{d}{k}\sqrt{\frac{\gamma k}{d}}\frac{(d/2k)^{d/2k}(d/2k)^{d/2k}(d/2k)^{d/2k}(d/2k)^{d/2k}}{(d/k)^{d/k}(d/8k)^{d/8k}(3d/8k)^{3d/8k}(3d/8k)^{3d/8k}(d/8k)^{d/8k}} = \sqrt{\frac{\gamma^2 d}{k}}\frac{8^{d/3k}}{9^{3d/8k}}.$$

Taking a union bound over at most $\frac{n^2}{k^2}$ pairs of vectors, the probability that there exist two rows that intersect by at most $\frac{d}{8k}$ coordinates is at most $\frac{n^2}{k^2}\sqrt{\frac{\gamma^2 d}{k}}\frac{8^{d/3k}}{9^{3d/8k}}$. □

[1]Even in the NO case, the maximizer of the determinant is the direct sum of k terms with two rows.

For $n > d$ and $d = \Theta(k \log \gamma n)$ with a sufficiently large constant, then $\mathbf{Pr}[\mathcal{E}_1] \geq \frac{8}{9}$, $\mathbf{Pr}[\neg\mathcal{E}_2] \geq 1 - \frac{1}{\text{poly}(n)}$, and $\mathbf{Pr}[\mathcal{E}_3] \geq 1 - \frac{1}{\text{poly}(n)}$. Thus Alice and Bob can use $\mathcal{A}$ to decide $\mathsf{DISJ}_{n/2k,d/k}$ with probability at least

$$\mathbf{Pr}[\mathcal{E}_1 \cap \neg\mathcal{E}_2 \cap \mathcal{E}_3] > 1 - \frac{1}{8}.$$

Thus if $\mathcal{A}$ uses $o\left(\frac{n}{p}\right)$ space per pass over p passes, then the total communication between Alice and Bob is $o(n)$, which contradicts Theorem 6.4. It follows that any C^k approximation algorithm to the volume maximization problem that succeeds with probability at least $1 - \frac{1}{9}$ requires $\Omega(n)$ space for constant k. □

Recall that for problems that are invariant to the permutation of the rows of the input matrix $\mathbf{A}$, once the entries of $\mathbf{A}$ are chosen, an arbitrary permutation of the rows of $\mathbf{A}$ is chosen uniformly at random, and the rows of that permutation constitute the stream in the random order row-arrival model.

Corollary 6.7 *For $C = \frac{16}{15}$, any one-pass streaming algorithm that outputs a C^k approximation to the $2k$-volume maximization problem in the random order row-arrival model with probability at least $\frac{63}{64}$ requires $\Omega(n)$ bits of space.*

Proof : First, observe that Alice and Bob construct matrix $\mathbf{M}_A$ and $\mathbf{M}_B$ from rows drawn uniformly at random. Thus in the NO case, the distribution of the matrix $\mathbf{M}_A$ and $\mathbf{M}_B$ follows the same distribution as when the rows of $\mathbf{M}$ arrive uniformly at random. In the YES case in the above model, the two orthogonal vectors must be in separate halves of the matrix $\mathbf{M}$. Namely, one vector is in $\mathbf{M}_A$ and one vector is in $\mathbf{M}_B$. In the random order model, the two orthogonal vectors are in separate halves with probability at least $\frac{1}{2}$. Since the YES case occurs with probability $\frac{1}{4}$, the total variation distance between the distribution of the rows in the random order model and the above distribution is $\frac{1}{8}$. Hence for a $\frac{7}{8}$ fraction of the inputs, Alice and Bob has the same distribution as that of Theorem 6.5.

In that case, any one-pass streaming algorithm $\mathcal{A}$ that outputs a C^k approximation to the $2k$-volume maximization problem with probability at least $\frac{63}{64}$ can decide between a YES instance and a NO instance of $\mathsf{DISJ}_{n/2k,d/k}$ for sufficiently large n and d with probability at least $\frac{31}{32}$ by the same argument as Theorem 6.5. Hence, the total probability of failure of the protocol is at most $\frac{1}{8} + \frac{1}{32} \leq \frac{1}{6}$ and so by Theorem 6.4, $\mathcal{A}$ requires $\Omega(n)$ space. □

Acknowledgements

D. Woodruff acknowledges support in part from the National Science Foundation under Grant No. CCF-1815840.

References

[ABI86] Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms*, 7(4):567–583, 1986. 22

[ABIW09] Alexandr Andoni, Khanh Do Ba, Piotr Indyk, and David P. Woodruff. Efficient sketches for earth-mover distance, with applications. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 324–330, 2009. 53

[AHPV05] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52:1–30, 2005. 42

[AKO11] Alexandr Andoni, Robert Krauthgamer, and Krzysztof Onak. Streaming algorithms via precision sampling. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS*, pages 363–372, 2011. 6

[AMS99] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. 11, 13, 14

[AS15] Pankaj K. Agarwal and R. Sharathkumar. Streaming algorithms for extent problems in high dimensions. *Algorithmica*, 72(1):83–98, 2015. 5

[BDM+18] Vladimir Braverman, Petros Drineas, Cameron Musco, Christopher Musco, Jalaj Upadhyay, David P. Woodruff, and Samson Zhou. Near optimal linear algebra in the online and sliding window models. *CoRR*, abs/1805.03765, 2018. 4

[BHI02] Mihai Badoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via coresets. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 250–257, 2002. 5

[BJKS04] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.*, 68(4):702–732, 2004. 45

[BMD09] Christos Boutsidis, Michael W. Mahoney, and Petros Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 968–977, 2009. 3

[CCF04] Moses Charikar, Kevin C. Chen, and Martin Farach-Colton. Finding frequent items in data streams. *Theor. Comput. Sci.*, 312(1):3–15, 2004. 11, 13, 14

[Cha06] Timothy M Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Computational Geometry*, 35(1-2):20–35, 2006. 42

[Che09] Ke Chen. On coresets for k-median and k-means clustering in metric and euclidean spaces and their applications. *SIAM J. Comput.*, 39(3):923–947, 2009. 5

[ÇM09] Ali Çivril and Malik Magdon-Ismail. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theor. Comput. Sci.*, 410(47-49):4801–4811, 2009. 9, 39

[CMM17] Michael B. Cohen, Cameron Musco, and Christopher Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017*, pages 1758–1777, 2017. 4

[CW15] Kenneth L. Clarkson and David P. Woodruff. Input sparsity and hardness for robust subspace approximation. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS*, pages 310–329, 2015. 4

[DRVW06] Amit Deshpande, Luis Rademacher, Santosh Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(12):225–247, 2006. 2, 3, 9, 31

[DV06] Amit Deshpande and Santosh Vempala. Adaptive sampling and fast low-rank matrix approximation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX and 10th International Workshop on Randomization and Computation, RANDOM, Proceedings*, pages 292–303, 2006. 2, 9, 31

[DV07] Amit Deshpande and Kasturi R. Varadarajan. Sampling-based dimension reduction for subspace approximation. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 641–650, 2007. 2, 4, 9, 33, 35, 37

[FMSW10] Dan Feldman, Morteza Monemizadeh, Christian Sohler, and David P. Woodruff. Coresets and sketches for high dimensional subspace approximation problems. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 630–649, 2010. 4, 5

[GS12] Venkatesan Guruswami and Ali Kemal Sinop. Optimal column-based low-rank matrix reconstruction. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1207–1214, 2012. 3

[HM04] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 291–300, 2004. 5

[IMGR19] Piotr Indyk, Sepideh Mahabadi, Shayan Oveis Gharan, and Alireza Rezaei. Composable core-sets for determinant maximization: A simple near-optimal algorithm. In *International Conference on Machine Learning*, pages 4254–4263, 2019. 5, 41, 42, 43

[IMGR20] Piotr Indyk, Sepideh Mahabadi, Shayan Oveis Gharan, and Alireza Rezaei. Composable core-sets for determinant maximization problems via spectral spanners. In *Proceedings of the thirty-first annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2020. 5, 9, 41

[IMMM14] Piotr Indyk, Sepideh Mahabadi, Mohammad Mahdian, and Vahab S. Mirrokni. Composable core-sets for diversity and coverage maximization. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pages 100–108, 2014. 41

[JST11] Hossein Jowhari, Mert Saglam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pages 49–58, 2011. 6

[JW18] Rajesh Jayaram and David P. Woodruff. Perfect lp sampling in a data stream. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 544–555, 2018. 6

[KR15] Michael Kerber and Sharath Raghvendra. Approximation and streaming algorithms for projective clustering via random projections. In *Proceedings of the 27th Canadian Conference on Computational Geometry, CCCG*, 2015. 5, 9

[KS92] Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Discrete Math.*, 5(4):545–557, 1992. 46

[LSW18] Roie Levin, Anish Prasad Sevekari, and David P. Woodruff. Robust subspace approximation in a stream. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems, NeurIPS*, pages 10706–10716, 2018. 4, 9

[MW10] Morteza Monemizadeh and David P. Woodruff. 1-pass relative-error l_p-sampling with applications. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1143–1160, 2010. 6

[NN93] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993. 22

[Pro17] Cecilia M. Procopiuc. Projective clustering, 2017. 4

[Raz92] Alexander A. Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992. 46

[SSS95] Jeanette P. Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff-hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math.*, 8(2):223–250, 1995. 21

[Stc] https://math.stackexchange.com/questions/183744/inequality-for-singular-values. 44

[SV12] Nariankadu D. Shyamalkumar and Kasturi R. Varadarajan. Efficient subspace approximation algorithms. *Discrete & Computational Geometry*, 47(1):44–63, 2012. 4

[SW11] Christian Sohler and David P. Woodruff. Subspace embeddings for the l_1-norm with applications. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC*, pages 755–764, 2011. 6

[SW18] Christian Sohler and David P. Woodruff. Strong coresets for k-median and subspace approximation: Goodbye dimension. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 802–813, 2018. 9

[Ver10] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010. 45

[WC81] Mark N. Wegman and Larry Carter. New hash functions and their use in authentication and set equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981. 21

[WZ12] David P. Woodruff and Qin Zhang. Tight bounds for distributed functional monitoring. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC*, pages 941–960, 2012. 46

A Noisy Distance Sampling

A.1 $L_{1,2}$ Sampler

Recall that for a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, we define the $L_{p,q}$ norm of $\mathbf{A}$ by

$$\|\mathbf{A}\|_{p,q} = \left(\sum_{i=1}^{n}\left(\sum_{j=1}^{d}|A_{i,j}|^q\right)^{\frac{p}{q}}\right)^{\frac{1}{p}}.$$

In this section, we describe an algorithm for sampling rows of a matrix $\mathbf{AP}$ with probability proportional to $\|\mathbf{AP}\|_2$, which we call $L_{1,2}$ sampling. By comparison, in Section 2 we sampled rows of $\mathbf{AP}$ with probability proportional to $\|\mathbf{AP}\|_2^2$, which can be seen as $L_{2,2}$ sampling.

Before describing our general $L_{1,2}$ sampler, we need a subroutine similar to AMS-M for estimating $\|\mathbf{AP}\|_{1,2}$, when the data stream updates entries of $\mathbf{A}$ and query access to $\mathbf{P}$ is only given in post-processing. We first describe a turnstile streaming algorithm of [ABIW09] that can be used to compute a constant factor approximation to $\|\mathbf{A}\|_{1,2}$ and then we show that it can be modified to approximate $\|\mathbf{AP}\|_{1,2}$ due its nature of being a linear sketch. For each j, define the level sets S_j by $\left\{i \in [n] : \frac{\|\mathbf{A}\|_{1,2}}{2^{j+1}} < \|\mathbf{A}_i\|_2 \leq \frac{\|\mathbf{A}\|_{1,2}}{2^j}\right\}$. The algorithm of [ABIW09] approximates the number of rows in each level set S_j by first implicitly subsampling rows at different rates. The rows that are sampled at each rate then form a *level* and the rows in a particular level are then aggregated across a number of buckets. The norms of the aggregates across each bucket are then computed and by rescaling the number of aggregates that are in each level set, we obtain an accurate estimate of the sizes of the level sets. The sizes of the level sets are then used to output a good approximation to $\|\mathbf{A}\|_{1,2}$.

Crucially, the aggregates of the rows in the algorithm of [ABIW09] is a linear combination of the rows. Hence by taking the aggregates and multiplying by $\mathbf{P}$ after the stream ends, we obtain aggregates of the rows of $\mathbf{AP}$, which can then be used to estimate the sizes of the level sets of $\|\mathbf{AP}\|_{1,2}$. The algorithm of [ABIW09] uses d polylog(n) space by storing aggregates of entire rows for each bucket across multiple levels. Thus, we have the following:

Lemma A.1 *[ABIW09] There exist a fixed constant $\xi > 1$ and a one-pass turnstile streaming algorithm* Estimator-M *that takes updates to entries of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, as well as query access to post-processing matrices $\mathbf{P} \in \mathbb{R}^{d\times d}$ and $\mathbf{M} \in \mathbb{R}^{n\times d}$ that arrive after the stream, and outputs a quantity $\hat{F}$ such that $\|\mathbf{AP}-\mathbf{M}\|_{1,2} \leq \hat{F} \leq \xi\|\mathbf{AP}-\mathbf{M}\|_{1,2}$. The algorithm uses d polylog(n) bits of space and succeeds with high probability.*

Using the $L_{1,2}$ estimator, we can develop a $L_{1,2}$ sampler similar to our ℓ_2 sampler.

We first show the probability that the tail is too large, i.e., $\widehat{S} > \frac{C\log n}{\epsilon}\widehat{F}$, is independent of the index i and the value of t_i. The proof is almost verbatim to Lemma 2.5, but the thresholds now depend on $\|\mathbf{AP}\|_{1,2}$ rather than $\|\mathbf{AP}\|_F$.

Algorithm 10 Single $L_{1,2}$ Sampler

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ that arrives as a stream $\mathbf{A}_1, \ldots, \mathbf{A}_n \in \mathbb{R}^d$, matrix $\mathbf{P} \in \mathbb{R}^{d \times d}$ that arrives after the stream, constant parameter $\epsilon > 0$.

Output: Noisy row $\mathbf{r}$ of $\mathbf{AP}$ sampled roughly proportional to the row norms of $\mathbf{AP}$.

Pre-processing Stage:
$b \leftarrow \Omega\left(\frac{1}{\epsilon^2}\right)$, $r \leftarrow \Theta(\log n)$ with sufficiently large constants
For $i \in [n]$, generate independent scaling factors $t_i \in [0, 1]$ uniformly at random.
Let $\mathbf{B}$ be the matrix consisting of rows $\mathbf{B}_i = \frac{1}{t_i}\mathbf{A}_i$.
Let ESTIMATOR-M and AMS-M track the $L_{1,2}$ norm of $\mathbf{AP}$ and Frobenius norm of $\mathbf{BP}$, respectively.
Let COUNTSKETCH-M be an $r \times b$ table, where each entry is a vector $\mathbb{R}^d$.
Streaming Stage:
for each row $\mathbf{A}_i$ **do**
    Update COUNTSKETCH-M with $\mathbf{B}_i = \frac{1}{t_i}\mathbf{A}_i$.
    Update linear sketch ESTIMATOR-M with $\mathbf{A}_i$.
    Update linear sketch AMS-M with $\mathbf{B}_i = \frac{1}{t_i}\mathbf{A}_i$.
Processing P Stage:
After the stream, obtain matrix $\mathbf{P}$.
Multiply each vector $\mathbf{v}$ in each entry of the COUNTSKETCH-M table by $\mathbf{P}$: $\mathbf{v} \leftarrow \mathbf{vP}$.
Multiply each vector $\mathbf{v}$ in AMS-M by $\mathbf{P}$: $\mathbf{v} \leftarrow \mathbf{vP}$.
Multiply each vector $\mathbf{v}$ in ESTIMATOR-M by $\mathbf{P}$: $\mathbf{v} \leftarrow \mathbf{vP}$.
Extraction Stage:
Use ESTIMATOR-M to compute $\widehat{F}$ with $\|\mathbf{AP}\|_{1,2} \le \widehat{F} \le \xi \|\mathbf{AP}\|_{1,2}$. ▷Lemma A.1
Extract the $\frac{2}{\epsilon^2}$ (noisy) rows of $\mathbf{BP}$ with the largest estimated norms by COUNTSKETCH-M.
Let $\mathbf{M}$ be the $\frac{2}{\epsilon^2}$-sparse matrix consisting of these top (noisy) rows.
Use AMS-M to compute $\widehat{S}$ with $\|\mathbf{BP} - \mathbf{M}\|_F \le \widehat{S} \le 2\|\mathbf{BP} - \mathbf{M}\|_F$.
Let $\mathbf{r}_i$ be the (noisy) row in COUNTSKETCH-M with the largest norm.
Let $C > 0$ be some large constant so that the probability of failure is $\mathcal{O}\left(\frac{1}{n^{C/2}}\right)$.
if $\widehat{S} > \frac{C \log n}{\epsilon}\widehat{F}$ or $\|\mathbf{r}_i\|_2 < \frac{C \log n}{\epsilon^2}\widehat{F}$ **then**
    return FAIL.
else
    return $\mathbf{r} = t_i \mathbf{r}_i$.

Lemma A.2 *For each $j \in [n]$ and value of t_j,*

$$\mathbf{Pr}\left[\widehat{S} > \frac{C \log n}{\epsilon}\widehat{F}\right] = \mathcal{O}(\epsilon) + \frac{1}{\text{poly}(n)}.$$

Proof : Let ξ be defined as in Lemma A.1. We first define the event $\mathcal{E}_1$ as when the following three inequalities hold:

(1) $\|\mathbf{AP}\|_{1,2} \le \widehat{F} \le \xi \|\mathbf{AP}\|_{1,2}$

(2) $\|\mathbf{BP} - \mathbf{M}\|_F \le \widehat{S} \le 2\|\mathbf{BP} - \mathbf{M}\|_F$

(3) $\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \|\mathbf{BP}-\mathbf{M}\|_F \leq 2\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F$

Let $j \in [n]$ be a fixed index and $t_j = t$ be a fixed uniform random scaling variable. $\mathcal{E}_1$ holds with high probability by Lemma 2.3 and Lemma A.1. We bound the probability that $2\xi\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F > \frac{C\log n}{\epsilon}\|\mathbf{AP}\|_{1,2}$, which must hold if $\widehat{S} > \frac{C\log n}{\epsilon}\widehat{F}$.

Let $U = \|\mathbf{AP}\|_{1,2}$ and for each $i \in [n]$, define the indicator variable $y_i = 1$ if $\|\mathbf{B}_i\mathbf{P}\|_2 > U$ and $y_i = 0$ otherwise. For each $i \in [n]$, define the scaled indicator variable $z_i = \frac{1}{U}\|\mathbf{B}_i\mathbf{P}\|_2(1-y_i)$. Observe that $z_i \in [0,1]$ represents a scaled contribution of the rows that are not heavy. Let $Y = \sum_{i\neq j} y_i$ and $Z = \sum_{i\neq j} z_i$. Define the matrix $\mathbf{W} \in \mathbb{R}^{n\times d}$ so that for each $i \in [n]$, its row i satisfies $\mathbf{W}_i = \mathbf{B}_i\mathbf{P}$ if $y_i = 1$ and otherwise if $y_i = 0$, then $\mathbf{W}_i$ is the row of all zeros. Hence, $\mathbf{W}$ has at most $Y+1$ nonzero rows and $UZ = \|\mathbf{BP}-\mathbf{W}\|_{1,2}$.

If there are not too many heavy rows in $\mathbf{W}$, i.e., $Y < \frac{2}{\epsilon^2}$, then $\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq UZ$ since the rows of $\mathbf{W}$ that are all zeros contain the tail of $\mathbf{BP}$ and the Frobenius norm is at most the $L_{1,2}$ norm. Let $\mathcal{E}_2$ denote the event that $Y \geq \frac{2}{\epsilon^2}$ and $\mathcal{E}_3$ denote the event that $Z \geq \frac{C\log n}{2\xi U\epsilon}\|\mathbf{AP}\|_{1,2} = \frac{C\log n}{2\xi\epsilon}$. If we bound the probability of the events $\mathcal{E}_2$ and $\mathcal{E}_3$ by $\mathcal{O}(\epsilon)$, then $2\xi\left\|(\mathbf{BP})_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \frac{C\log n}{\epsilon}\|\mathbf{AP}\|_{1,2}$ with probability at least $1-\mathcal{O}(\epsilon)$, conditioned on $\mathcal{E}_1$.

To bound $\mathcal{E}_2$, observe that $\mathbb{E}[y_i] = \frac{\|\mathbf{A}_i\mathbf{P}\|_2}{U}$ so that $\mathbb{E}[Y] \leq 1$ since $U = \|\mathbf{AP}\|_{1,2}$. Thus for sufficiently large n, $\mathbf{Pr}[\mathcal{E}_2] = \mathcal{O}(\epsilon)$, by Markov's inequality.

To bound $\mathbf{Pr}[\mathcal{E}_3]$, observe that $z_i = \frac{1}{U}\|\mathbf{B}_i\mathbf{P}\|_2(1-y_i)$ implies $z_i > 0$ only if $y_i = 0$, i.e., $\|\mathbf{B}_i\mathbf{P}\|_2 \leq U$. Since $\mathbf{B}_i\mathbf{P} = \frac{\mathbf{A}_i\mathbf{P}}{t_i}$, then $z_i > 0$ only for $t_i \geq \frac{\|\mathbf{A}_i\mathbf{P}\|_2}{\|\mathbf{AP}\|_{1,2}}$. Therefore,

$$\mathbb{E}[z_i] \leq \int_{\|\mathbf{A}_i\mathbf{P}\|_2/\|\mathbf{AP}\|_{1,2}}^{1} z_i\,dt_i = \int_{\|\mathbf{A}_i\mathbf{P}\|_2/\|\mathbf{AP}\|_{1,2}}^{1} \frac{1}{t_i}\frac{1}{U}\|\mathbf{A}_i\mathbf{P}\|_2\,dt_i \leq C\log n\frac{\|\mathbf{A}_i\mathbf{P}\|_2}{\|\mathbf{AP}\|_{1,2}},$$

conditioned on $t_i \geq \frac{1}{\mathrm{poly}(n)}$. Hence $\mathbb{E}[Z] \leq C\log n$ so $\mathbf{Pr}[\mathcal{E}_3] = \mathbf{Pr}\left[Z > \frac{C\log n}{2\xi\epsilon}\right] = \mathcal{O}(\epsilon)$ by Markov's inequality. Therefore, the failure events $\neg\mathcal{E}_1 \vee \mathcal{E}_2 \vee \mathcal{E}_3$ occur with probability $\mathcal{O}(\epsilon) + \frac{1}{\mathrm{poly}(n)}$, and the claim follows. □

Lemma A.3 *Conditioned on a fixed value of $\widehat{F}$, the probability that Algorithm 10 outputs (noisy) row i is $(1\pm\mathcal{O}(\epsilon))\frac{\|\mathbf{A}_i\mathbf{P}\|_2}{\widehat{F}} + \frac{1}{\mathrm{poly}(n)}$.*

Proof : We first define $\mathcal{E}$ to be the event that $t_i < \frac{\epsilon^2\|\mathbf{A}_i\mathbf{P}\|_2}{(C\log n)\widehat{F}}$. Note that $\mathbf{Pr}[\mathcal{E}] = \frac{\epsilon^2\|\mathbf{A}_i\mathbf{P}\|_2}{(C\log n)\widehat{F}}$. Next, we define $\mathcal{E}_1$ to be the event that CountSketch-M, AMS-M, or Estimator-M fails and note that $\mathbf{Pr}[\mathcal{E}_1] = \frac{1}{\mathrm{poly}(n)}$ by Lemma 2.3, Lemma 2.1, and Lemma A.1. We then define $\mathcal{E}_2$ to be the event that $\widehat{S} > \frac{C\log n}{\epsilon}\widehat{F}$ and note that $\mathbf{Pr}[\mathcal{E}_2] = \mathcal{O}(\epsilon)$ by Lemma A.2. Finally, we let $\mathcal{E}_3$ be the event that the CountSketch data structure observes multiple rows $\mathbf{B}_j\mathbf{P}$ exceeding the threshold and $\mathcal{E}_4$ be the event that $\|\mathbf{B}_i\mathbf{P}\|_2$ exceeds the threshold but is not reported due to noise in the CountSketch data structure.

Observe that a row j is close enough to the threshold if $\|\mathbf{B}_j\mathbf{P}\|_2 \geq \frac{C\log n}{\epsilon^2}\widehat{F} - (C\log n)\widehat{F}$, which occurs with probability at most $\mathcal{O}\left(\frac{\epsilon\|\mathbf{A}_j\mathbf{P}\|_2}{\widehat{F}}\right)$. Taking a union bound over all n rows, we have $\mathbf{Pr}\left[\mathcal{E}_3\right] = \mathcal{O}\left(\epsilon\right)$.

To analyze the probability of $\mathcal{E}_4$, we first condition on $\neg\mathcal{E}_1$ and $\neg\mathcal{E}_2$, so that we have $\|\mathbf{BP}-\mathbf{M}\|_F \leq \widehat{S}$ and $\widehat{S} \leq \frac{C\log n}{\epsilon}\widehat{F}$. Thus by Lemma 2.3,

$$\left|\|\mathbf{B}_i\mathbf{P}\|_2 - \left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2\right| \leq \epsilon\left\|\mathbf{BP}_{tail\left(\frac{2}{\epsilon^2}\right)}\right\|_F \leq \epsilon\|\mathbf{BP}-\mathbf{M}\|_F \leq \epsilon\widehat{S} \leq (C\log n)\widehat{F}.$$

Hence, $\mathcal{E}_4$ can only occur for

$$\frac{C\log n}{\epsilon^2}\widehat{F} \leq \|\mathbf{B}_i\mathbf{P}\|_2 \leq \frac{C\log n}{\epsilon^2}\widehat{F} + (C\log n)\widehat{F},$$

which occurs with probability at most $\frac{\epsilon^4\|\mathbf{A}_i\mathbf{P}\|_2}{(C\log n)\widehat{F}}$.

In summary if $\mathcal{E}$ occurs, then the sampler should output (noisy) row $\mathbf{A}_i\mathbf{P}$ but may fail to do so because of any of the events $\mathcal{E}_1$, $\mathcal{E}_2$, $\mathcal{E}_3$, or $\mathcal{E}_4$. We have $\mathbf{Pr}\left[\mathcal{E}_2 \vee \mathcal{E}_3 \,|\, \mathcal{E}\right] = \mathcal{O}\left(\epsilon\right)$ and $\mathbf{Pr}\left[\mathcal{E}_4\right] = \frac{\epsilon^4\|\mathbf{A}_i\mathbf{P}\|_2}{(C\log n)\widehat{F}}$ so that $\mathbf{Pr}\left[\mathcal{E}_4 \,|\, \mathcal{E}\right] = \mathcal{O}\left(\epsilon^2\right)$. Since $\mathbf{Pr}\left[\mathcal{E}_1\right] = \frac{1}{\text{poly}(n)}$, then each $\mathbf{A}_i\mathbf{P}$ is output with probability $(1+\mathcal{O}\left(\epsilon\right))\frac{\|\mathbf{A}_i\mathbf{P}\|_2}{\widehat{F}}$.

Moreover by Lemma 2.2, we have that $\left|\|\mathbf{B}_i\mathbf{P}\|_2 - \left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2\right| \leq (C\log n)\widehat{F}$ and $\left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2 \geq \frac{C\log n}{\epsilon^2}\widehat{F}$. Thus, $\left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2$ is a $(1+\mathcal{O}\left(\epsilon\right))$ approximation to $\|\mathbf{B}_i\mathbf{P}\|_2$ and similarly, $t_i\left\|\widehat{\mathbf{B}_i\mathbf{P}}\right\|_2$ is within $(1+\mathcal{O}\left(\epsilon\right))$ of $\|\mathbf{A}_i\mathbf{P}\|_2$. □

We now provide the full guarantees of the $L_{1,2}$ sampler.

Theorem A.4 *Given $\epsilon > 0$, there exists a one-pass streaming algorithm that takes rows of a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ as a data stream and a matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$ after the stream, and outputs (noisy) row i of $\mathbf{AP}$ with probability $(1\pm\mathcal{O}\left(\epsilon\right))\frac{\|\mathbf{A}_i\mathbf{P}\|_2}{\|\mathbf{AP}\|_{1,2}} + \frac{1}{\text{poly}(n)}$. The algorithm uses $\mathcal{O}\left(d\ \text{poly}\left(\frac{1}{\epsilon}, \log n\right)\right)$ bits of space and succeeds with high probability.*

Proof : From Lemma A.3 and the fact that $\|\mathbf{AP}\|_{1,2} \leq \widehat{F} \leq \xi\|\mathbf{AP}\|_{1,2}$ with high probability by Lemma A.1, then it follows that each row $\mathbf{A}_i\mathbf{P}$ is sampled with probability $(1+\epsilon)\frac{\|\mathbf{A}_i\mathbf{P}\|_2}{\|\mathbf{AP}\|_{1,2}} + \frac{1}{\text{poly}(n)}$, conditioned on the sampler succeeding. The probability of the sampler succeeds is $\Theta\left(\frac{\epsilon^2}{C\log n}\right)$, then the sampler can be repeated $\text{poly}\left(\frac{1}{\epsilon}, \log n\right)$ times to obtain probability of success at least $1-\frac{1}{\text{poly}(n)}$. Since each instance of AMS-M, ESTIMATOR-M, and COUNTSKETCH-M uses $\mathcal{O}\left(d\ \text{poly}\left(\frac{1}{\epsilon}, \log n\right)\right)$ bits of space, then the total space complexity follows. □

A.2 Noisy Adaptive Distance Sampling

Our algorithm for noisy adaptive distance sampling, given in Algorithm 11, is similar to Section 3, except it uses the $L_{1,2}$ sampling primitive of Theorem A.4 instead of the $L_{2,2}$ sampler.

We first bound the norm of the perturbation of the sampled row at each instance.

Algorithm 11 Noisy Adaptive Sampler

Input: Matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives as a stream $\mathbf{A}_1, \ldots, \mathbf{A}_n \in \mathbb{R}^d$, parameter k for number of sampled rows, constant parameter $\epsilon > 0$.

Output: k Noisy and projected rows of $\mathbf{A}$.

1: Create instances $\mathcal{A}_1, \ldots, \mathcal{A}_k$ of the $L_{1,2}$ sampler of Algorithm 10 where the number of buckets $b = \Theta\left(\frac{\log^4 n}{\epsilon^2}\right)$ is sufficiently large.
2: Let $\mathbf{M}$ be empty $0 \times d$ matrix.
3: **Streaming Stage:**
4: **for** each row $\mathbf{A}_i$ **do**
5: Update each sketch $\mathcal{A}_1, \ldots, \mathcal{A}_k$
6: **Post-processing Stage:**
7: **for** $j = 1$ to $j = k$ **do**
8: Post-processing matrix $\mathbf{P} \leftarrow \mathbb{I} - \mathbf{M}^\dagger\mathbf{M}$.
9: Update $\mathcal{A}_j$ with post-processing matrix $\mathbf{P}$.
10: Let $\mathbf{r}_j$ be the noisy row output by $\mathcal{A}_j$.
11: Append $\mathbf{r}_j$ to $\mathbf{M}$: $\mathbf{M} \leftarrow \mathbf{M} \circ \mathbf{r}_j$.
12: **return** $\mathbf{M}$.

Lemma A.5 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ and a matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$, as defined in Line 8 and round $i \le k$, of Algorithm 10, suppose index $j \in [n]$ is sampled (in round i). Then with high probability, the sampled (noisy) row $\mathbf{r}_i$ satisfies $\mathbf{r}_i = \mathbf{A}_j\mathbf{P} + \mathbf{v}_e$ with*

$$\|\mathbf{v}_e\mathbf{Q}\|_2 \le \frac{\epsilon^3}{C\log n}\frac{\|\mathbf{APQ}\|_{1,2}}{\|\mathbf{AP}\|_{1,2}}\|\mathbf{A}_j\mathbf{P}\|_2,$$

for any projection matrix $\mathbf{Q} \in \mathbb{R}^{d\times d}$. Hence, $\mathbf{v}_e$ is orthogonal to each noisy row $\mathbf{r}_y$, where $y \in [i-1]$.

Proof : Let $\mathbf{Q} \in \mathbb{R}^{d\times d}$ be a projection matrix, $\mathbf{B}_x = \frac{\mathbf{A}_x}{t_x}$ be the rescaled row of $\mathbf{A}_x$ for each $x \in [n]$, and $\mathbf{B} \in \mathbb{R}^{n\times d}$ be the rescaled matrix of $\mathbf{A}$ so that row x of $\mathbf{B}$ is $\mathbf{B}_x$ for $x \in [n]$. Let $\mathbf{E}$ be the noise in the bucket corresponding to the selected row j, so that the output vector is $\mathbf{A}_j + t_j\mathbf{E}$. Since $t_x \in [0,1]$ is selected uniformly at random for each $x \in [n]$, then for each integer $c \ge 0$,

$$\mathbf{Pr}\left[\frac{\|\mathbf{A}_x\mathbf{PQ}\|_2}{t_x} \ge \frac{\|\mathbf{APQ}\|_{1,2}}{2^c}\right] \le \frac{2^c\|\mathbf{A}_x\mathbf{PQ}\|_2}{\|\mathbf{APQ}\|_{1,2}}.$$

Because $\mathbf{B}_x = \frac{\mathbf{A}_x}{t_x}$, then by linearity of expectation over $x \in [n]$, we can bound the expected size of each of the level sets $S_c := \left\{x \in [n] : \frac{\|\mathbf{APQ}\|_{1,2}}{2^{c-1}} > \|\mathbf{B}_x\mathbf{PQ}\|_2 \ge \frac{\|\mathbf{APQ}\|_{1,2}}{2^c}\right\}$ by $\mathbb{E}[|S_c|] \le \min(2^c, n)$. Thus $\mathbf{Pr}\left[|S_c| \le \min(2^{c+C}\log n, n)\right] \ge 1 - \frac{1}{\mathrm{poly}(n)}$ by standard Chernoff bounds for appropriate constant C.

We can now roughly bound the $L_{1,2}$ norm of $\mathbf{BPQ}$ by the norm of $\mathbf{APQ}$ using a union bound over level sets S_c for $0 \le c \le \log n$ and upper bounding the norms of all the rows in level sets S_c with $c > \log n$ by $\frac{\|\mathbf{APQ}\|_{1,2}}{n}$. That is,

$$\mathbf{Pr}\left[\|\mathbf{BPQ}\|_{1,2} \ge 2^C \cdot 4\log^2 n\,\|\mathbf{APQ}\|_{1,2}\right] \le \frac{1}{\mathrm{poly}(n)}.$$

Hence with high probability, the total mass $\|\mathbf{BPQ}\|_{1,2}$ distributed across the CountSketch table is $\mathcal{O}\left(\log^2 n \, \|\mathbf{APQ}\|_{1,2}\right)$.

Using a CountSketch table with $b = \Theta\left(\frac{\log^4 n}{\epsilon^2}\right)$ buckets with sufficiently large constant to hash the rows of $\mathbf{BPQ}$, then Lemma 2.3 implies that the bucket corresponding to $\mathbf{A}_j$ has mass at most $\epsilon \|\mathbf{APQ}\|_F \leq \epsilon \|\mathbf{APQ}\|_{1,2}$ in the subspace to which $\mathbf{Q}$ projects, i.e., $\|\mathbf{EQ}\|_2 \leq \epsilon \|\mathbf{APQ}\|_{1,2}$. Since row j was sampled by Algorithm 10, then $\|\mathbf{B}_j\mathbf{P}\|_2 \geq \frac{C \log n}{\epsilon^2} \|\mathbf{AP}\|_{1,2}$. Thus $t_j \leq \frac{\epsilon^2}{C \log n} \frac{\|\mathbf{A}_j\mathbf{P}\|_2}{\|\mathbf{AP}\|_{1,2}}$ since $\mathbf{B}_j = \frac{\mathbf{A}_j\mathbf{P}}{t_j}$, and moreover with high probability,

$$\|t_j\mathbf{EQ}\|_2 \leq \frac{\epsilon^3}{C \log n} \frac{\|\mathbf{APQ}\|_{1,2}}{\|\mathbf{AP}\|_{1,2}} \|\mathbf{A}_j\mathbf{P}\|_2 .$$

Since $\mathbf{v}_e\mathbf{Q} = t_j\mathbf{EQ}$, then the claim follows. □

We now bound the total variation distance between the distribution of sampled rows and the distribution of adaptive sampling with respect to distances to selected subspace. The proof is almost verbatim to Lemma 3.2, except we now consider the probabilities with respect to the distances to the previous subspace, rather than the squared distances. We can still use the change of basis matrix in $(\star)$ to denote the perturbation in each round, where we set $\tau_i = \frac{\epsilon^3 \sum_{a=1}^n \lambda_{a,i}}{\sum_{a=1}^n \sqrt{\sum_{b=1}^d \lambda_{a,b}^2}}$ though Lemma A.5 implies we could actually even set the scaling factor to $\frac{\epsilon^3}{C \log n}$ rather than just ϵ^2. We can then bound $\left|\sqrt{\sum_{i=2}^d \zeta_{s,i}^2} - \sqrt{\sum_{i=2}^d \lambda_{s,i}^2}\right|$ from a bound on $\left|\sum_{i=2}^d \zeta_{s,i}^2 - \sum_{i=2}^d \lambda_{s,i}^2\right|$.

Lemma A.6 *Let $f(1)$ be the index of a noisy row $\mathbf{r}_1$ sampled in the first iteration of Algorithm 11. Let $\mathcal{P}_1$ be a process that projects away from $\mathbf{A}_{f(1)}$ and iteratively selects $k-1$ additional rows of $\mathbf{A}$ through adaptive sampling (with $p = 1$). Let $\mathcal{P}_2$ be a process that projects away from $\mathbf{r}_1$ and iteratively selects $k-1$ additional rows of $\mathbf{A}$ through adaptive sampling (with $p = 1$). Then for $\epsilon < \frac{1}{d}$, the total variation distance between the distributions of the k indices output by $\mathcal{P}_1$ and $\mathcal{P}_2$ is $\mathcal{O}(k\epsilon)$.*

Proof : As in the proof of Lemma 3.2, we consider the probability distributions induced by linearly independent vectors $\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(t-1)}$ and by linearly independent vectors $\mathbf{r}_1, \mathbf{A}_{f(2)}, \ldots, \mathbf{A}_{f(t-1)}$. Let $U = \{\mathbf{u}_1, \ldots, \mathbf{u}_d\}$ be an orthonormal basis for the row span of $\mathbf{A}$ such that $\{\mathbf{u}_1, \ldots, \mathbf{u}_s\}$ is a basis for the row span of $\{\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(s)}\}$ for each $2 \leq s \leq t-1$ and let $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_d\}$ be an orthonormal basis for the row span of $\mathbf{A}$ such that $\{\mathbf{w}_1, \ldots, \mathbf{w}_s\}$ is an orthonormal basis that extends the row span of $\{\mathbf{r}_1, \mathbf{A}_{f(2)}, \ldots, \mathbf{A}_{f(s)}\}$ for each $2 \leq s \leq t-1$.

Lemma A.5 then implies that $\mathbf{r}_1 = \left\|\mathbf{A}_{f(1)}\right\|_2 \left(\mathbf{u}_1 + \sum_{i=1}^d (\pm\mathcal{O}(\tau_i))\mathbf{u}_i\right)$, where $\tau_i = \frac{\epsilon^3 \|\mathbf{AP}_i\|_{1,2}}{\|\mathbf{A}\|_{1,2}}$ (in fact we could set τ_i to $\frac{\epsilon^3}{C \log n} \frac{\|\mathbf{AP}_i\|_{1,2}}{\|\mathbf{A}\|_{1,2}}$ but we do not need this smaller value) with high probability, and $\mathbf{P}_i = \mathbf{u}_i^\dagger \mathbf{u}_i$ is the projection matrix onto $\mathbf{u}_i$. From the Gram-Schmidt process, the change of basis matrix $\mathbf{B}$ from U to W has the form $(\star)$.

We can write each row $\mathbf{A}_s$ in terms of basis U as $\mathbf{A}_s = \sum_{i=1}^d \lambda_{s,i}\mathbf{u}_i$ and in terms of basis W as $\mathbf{A}_s = \sum_{i=1}^d \zeta_{s,i}\mathbf{w}_i$. Since we project away from $\mathbf{A}_{f(1)}$, we should have sampled $\mathbf{A}_s$ with probability $\frac{\|\mathbf{A}_s\mathbf{Z}_{t-1}\|_2}{\|\mathbf{AZ}_{t-1}\|_{1,2}} = \frac{\sqrt{\sum_{i=2}^d \lambda_{s,i}^2}}{\sum_{j=1}^n \sqrt{\sum_{i=2}^d \lambda_{j,i}^2}}$ in round t but instead we sample it with probability

$\frac{\|\mathbf{A}_s\mathbf{Y}_{t-1}\|_2}{\|\mathbf{A}\mathbf{Y}_{t-1}\|_{1,2}} = \frac{\sqrt{\sum_{i=2}^d \zeta_{s,i}^2}}{\sum_{j=1}^n \sqrt{\sum_{i=2}^d \zeta_{j,i}^2}}$. From the change of basis matrix $\mathbf{B}$, $\zeta_{s,i} = (1-\mathcal{O}(\tau_i))\lambda_{s,i} \pm \mathcal{O}(\tau_i)\,\lambda_{s,1} \pm \sum_{j\notin\{i,1\}} \mathcal{O}(\tau_i\tau_j)\,\lambda_{s,j}$, for $i \geq 2$. Note that $\zeta_{s,i}$ has exactly the same form as the proof of Lemma 3.2, since $\zeta_{s,i}$ is derived from the same change of basis matrix $\mathbf{B}$, albeit with different values of τ_i. It then follows by the same reasoning as (⊞) in the proof of Lemma 3.2 that for

$$|\zeta_{s,i}^2 - \lambda_{s,i}^2| \leq 25\Big(\tau_i\lambda_{s,i}^2 + \tau_i^2\lambda_{s,1}^2 + \tau_i\lambda_{s,1}\lambda_{s,i} + \sum_{j,\ell\neq\{i,1\}} \tau_i^2\tau_j\tau_\ell\lambda_{s,j}\lambda_{s,\ell} + \sum_{j\neq\{i,1\}} \tau_i\tau_j\lambda_{s,i}\lambda_{s,j} + \sum_{j\neq\{i,1\}} \tau_i^2\tau_j\lambda_{s,1}\lambda_{s,j}\Big).$$

Thus from AM-GM, we have

$$\begin{aligned}
\tau_i\lambda_{s,1}\lambda_{s,i} &\leq \epsilon^2\lambda_{s,i}^2 + \frac{\tau_i^2}{\epsilon^2}\lambda_{s,1}^2\\
\tau_i^2\tau_j\tau_\ell\lambda_{s,j}\lambda_{s,\ell} &\leq \tau_i^2\tau_j^2\lambda_{s,j}^2 + \tau_i^2\tau_\ell^2\lambda_{s,\ell}^2,\\
\tau_i\tau_j\lambda_{s,i}\lambda_{s,j} &\leq \epsilon^2\lambda_{s,i}^2 + \frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2,\\
\tau_i^2\tau_j\lambda_{s,1}\lambda_{s,j} &\leq \tau_i^2\lambda_{s,1}^2 + \tau_i^2\tau_j^2\lambda_{s,j}^2
\end{aligned}$$

so that for $\epsilon < \frac{1}{d}$, we have $|\zeta_{s,i}^2 - \lambda_{s,i}^2| \leq 25\left(2\epsilon^2\lambda_{s,i}^2 + \frac{2\tau_i^2}{\epsilon^2}\lambda_{s,1}^2 + 4\sum_{j=2}^d \frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2\right)$. Note that in comparison to Lemma 3.2, we compare the $\tau_i\lambda_{s,1}\lambda_{s,i}$ term with $\epsilon^2\lambda_{s,i}^2$ rather than $\epsilon\lambda_{s,i}^2$. Therefore,

$$\begin{aligned}
\left|\sqrt{\sum_{i=t}^d \zeta_{s,i}^2} - \sqrt{\sum_{i=t}^d \lambda_{s,i}^2}\right| &\leq \sqrt{\left|\sum_{i=t}^d \zeta_{s,i}^2 - \sum_{i=t}^d \lambda_{s,i}^2\right|}\\
&\leq \sqrt{25\sum_{i=t}^d\left(2\epsilon^2\lambda_{s,i}^2 + \frac{2\tau_i^2}{\epsilon^2}\lambda_{s,1}^2 + 4\sum_{j=2}^d \frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2\right)}\\
&\leq 5\sqrt{\sum_{i=t}^d 2\epsilon^2\lambda_{s,i}^2} + 5\sqrt{\sum_{i=t}^d \frac{2\tau_i^2}{\epsilon^2}\lambda_{s,1}^2} + 5\sqrt{4\sum_{i=t}^d\sum_{j=2}^d \frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2}.
\end{aligned}$$

Since $\tau_i = \frac{\epsilon^3\|\mathbf{A}\mathbf{P}_i\|_{1,2}}{\|\mathbf{A}\|_{1,2}} = \frac{\epsilon^3\sum_{a=1}^n \lambda_{a,i}}{\sum_{a=1}^n\sqrt{\sum_{b=1}^d \lambda_{a,b}^2}}$ and $\epsilon < \frac{1}{d}$, then we have

$$\begin{aligned}
\sum_{s=1}^n \sqrt{\sum_{i=t}^d \frac{\tau_i^2}{\epsilon^2}\lambda_{s,1}^2} &\leq \left(\sum_{s=1}^n \lambda_{s,1}\right)\sqrt{\sum_{i=t}^d \frac{\tau_i^2}{\epsilon^2}} \leq \epsilon^2\left(\sum_{s=1}^n \lambda_{s,1}\right)\frac{\sum_{i=t}^d\sum_{a=1}^n \lambda_{a,i}}{\sum_{a=1}^n\sqrt{\sum_{b=1}^d \lambda_{a,b}^2}}\\
&\leq \epsilon^2\sum_{s=1}^n\sum_{i=t}^d \lambda_{s,i} \leq \epsilon\sum_{s=1}^n\sqrt{\sum_{i=t}^d \lambda_{s,i}^2}.
\end{aligned}$$

Similarly, since $\tau_j < \frac{1}{d}$ for each integer $j \in [2, d]$, we have

$$\begin{aligned}\sum_{s=1}^{n}\sqrt{\sum_{i=t}^{d}\sum_{j=2}^{d}\frac{\tau_i^2\tau_j^2}{\epsilon^2}\lambda_{s,j}^2} &\leq \sum_{s=1}^{n}\max_{j\in[2,d]}\sqrt{\sum_{i=t}^{d}\frac{\tau_i^2}{\epsilon^2}\lambda_{s,j}^2} \\ &\leq \epsilon^2\left(\sum_{s=1}^{n}\max_{j\in[2,d]}\lambda_{s,j}\right)\frac{\sum_{i=t}^{d}\sum_{a=1}^{n}\lambda_{a,i}}{\sum_{a=1}^{n}\sqrt{\sum_{b=1}^{d}\lambda_{a,b}^2}} \\ &\leq \epsilon^2\sum_{s=1}^{n}\sum_{i=t}^{d}\lambda_{s,i} \leq \epsilon\sum_{s=1}^{n}\sqrt{\sum_{i=t}^{d}\lambda_{s,i}^2}.\end{aligned}$$

Hence, we have

$$\sum_{s=1}^{n}\left|\sqrt{\sum_{i=t}^{d}\zeta_{s,i}^2}-\sqrt{\sum_{i=t}^{d}\lambda_{s,i}^2}\right| \leq 200\epsilon\sum_{s=1}^{n}\sqrt{\sum_{i=t}^{d}\lambda_{s,i}^2},$$

so that $\|\mathbf{A}\mathbf{Y}_{t-1}\|_{1,2}$ is once again within a $(1+200\epsilon)$ factor of $\|\mathbf{A}\mathbf{Z}_{t-1}\|_{1,2}$, from which we can bound the total variation distance by 800ϵ, including the $\frac{1}{\text{poly}(n)}$ event of failure from Lemma A.5. It follows from induction that the total variation distance across k rounds is at most $800k\epsilon$. □

Thus we can also approximately simulate adaptive sampling in a stream with respect to the distances to the subspace spanned by the previously sampled rows.

Theorem A.7 *Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ that arrives in a turnstile data stream, there exists a one-pass streaming algorithm* AdaptDistStream *that outputs a set of k indices such that the probability distribution for each set of k indices has total variation distance ϵ of the probability distribution induced by adaptive sampling with respect to the distances to the subspace in each iteration. The algorithm uses* $\text{poly}\left(d, k, \frac{1}{\epsilon}, \log n\right)$ *bits of space.*

Proof : Like the proof of Theorem 3.4, we again consider a set of processes $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{k+1}$, where for each $i \in [k+1]$, $\mathcal{P}_i$ is a process that samples noisy rows from the $L_{1,2}$ sampler for the first $i-1$ rounds and actual rows from $\mathbf{A}$ beginning with round i, through adaptive sampling with $p = 1$. Then $\mathcal{P}_1$ is the actual adaptive sampling process, while $\mathcal{P}_{k+1}$ is the noisy process of Algorithm 11. Then Lemma A.6 argues that the total variation distance between the output distributions of the k indices sampled by $\mathcal{P}_1$ and $\mathcal{P}_2$ is at most $\mathcal{O}(k\epsilon)$. Moreover, the total variation distance between the output distributions of the indices sampled by $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$ is at most $\mathcal{O}(k\epsilon)$ for any $i \in [k]$ since the sampling distributions of $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$ is identical in the first i rounds, so we can use the same argument starting at round i using the input matrix $\mathbf{AQ}$ rather than $\mathbf{A}$, where $\mathbf{Q}$ is the projection matrix away from the noisy rows sampled in the first i rounds. Now by a triangle inequality argument over the $k+1$ processes $\mathcal{P}_1, \ldots, \mathcal{P}_{k+1}$, the total variation distance between the probability distribution of the k indices output by Algorithm 11 and the probability distribution of the k indices output by adaptive sampling is at most $\mathcal{O}\left(k^2\epsilon\right)$. Hence the total variation distance is at most ϵ after the appropriate rescaling factor.

For the space complexity, observe that we run k instances of Algorithm 10, each using $b = \text{poly}\left(k, \frac{1}{\epsilon}, \log n\right)$ buckets due to the error parameter $\frac{\epsilon}{dk^2}$. Algorithm 10 uses $\text{poly}\left(k, \frac{1}{\epsilon}, \log n\right)$ CountSketch data structures that each use $\text{poly}\left(d, k, \frac{1}{\epsilon}, \log n\right)$ bits of space. Moreover by Lemma A.1,

each instance of ESTIMATOR-M uses d polylog(n) bits of space. Hence, the total space complexity is poly $\left(d, k, \frac{1}{\epsilon}, \log n\right)$. □

Finally, by the same argument as Corollary 3.5, we have the following:

Corollary A.8 *Suppose Algorithm 11 samples noisy rows $\mathbf{r}_1, \ldots, \mathbf{r}_k$ rather than the actual rows $\mathbf{A}_{f(1)}, \ldots, \mathbf{A}_{f(k)}$. Let $\mathbf{T}_k = \mathbf{A}_{f(1)} \circ \ldots \circ \mathbf{A}_{f(k)}$, $\mathbf{Z}_k = \mathbb{I} - \mathbf{T}_k^\dagger \mathbf{T}_k$, $\mathbf{R}_k = \mathbf{r}_1 \circ \ldots \circ \mathbf{r}_k$ and $\mathbf{Y}_k = \mathbb{I} - \mathbf{R}_k^\dagger \mathbf{R}_k$. Then $(1-\epsilon)\left\|\mathbf{A}\mathbf{Y}_k\right\|_{1,2} \leq \left\|\mathbf{A}\mathbf{Z}_k\right\|_{1,2} \leq (1+\epsilon)\left\|\mathbf{A}\mathbf{Y}_k\right\|_{1,2}$ with probability at least $1-\epsilon$.*