Robust Subspace Approximation in a Stream

Roie Levin\textsuperscript{1} Anish Sevekari\textsuperscript{2}
roiel@cs.cmu.edu asevekar@andrew.cmu.edu

David P. Woodruff\textsuperscript{1}
dwoodruf@cs.cmu.edu

\textsuperscript{1} Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213
\textsuperscript{2} Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA 15213

December 4, 2018

Abstract

We study robust subspace estimation in the streaming and distributed settings. Given a set of \( n \) data points \( \{a_i\}_{i=1}^n \) in \( \mathbb{R}^d \) and an integer \( k \), we wish to find a linear subspace \( S \) of dimension \( k \) for which \( \sum_i M(\text{dist}(S, a_i)) \) is minimized, where \( \text{dist}(S, x) := \min_{y \in S} \|x - y\|_2 \), and \( M(\cdot) \) is some loss function. When \( M(\cdot) = (\cdot)^2 \), this is the well-studied least squares subspace approximation problem. The minimizer in this case can be computed exactly by computing the truncated SVD of the data matrix. Otherwise \( M \) is often chosen from \((\cdot)^p\) for some \( p \geq 0 \), or from a class of functions called \( M \)-estimators, with the goal of providing a more robust estimate than least squares in the face of outliers. Indeed, for \( p < 2 \), since one is not squaring the distances to the subspace, one is placing less emphasis on outliers and therefore capturing more of the remaining data points. For example, when \( M \) is the identity function, we are finding a subspace so as to minimize the sum of distances to it, which could arguably be more natural than finding a subspace so as to minimize the sum of squared distances. We can write this problem in the following form:

\[
\min_{S \text{ dim } k} \sum_i \text{dist}(S, a_i) = \min_{X \text{ rank } k} \sum_i \|(A - AX)_{i*}\|_2
\]

where \( A \) is the matrix in which the \( i \)-th row is the vector \( a_i \). This is the form of robust subspace approximation that we study in this work. We will be interested in the approximate version of the problem for which

1 Introduction

A fundamental problem in large-scale machine learning is that of subspace approximation. Given a set of \( n \) data points \( \{a_i\}_{i=1}^n \) in \( \mathbb{R}^d \) and an integer \( k \), we wish to find a linear subspace \( S \) of dimension \( k \) for which \( \sum_i M(\text{dist}(S, a_i)) \) is minimized, where \( \text{dist}(S, x) := \min_{y \in S} \|x - y\|_2 \), and \( M(\cdot) \) is some loss function. When \( M(\cdot) = (\cdot)^2 \), this is the well-studied least squares subspace approximation problem. The minimizer in this case can be computed exactly by computing the truncated SVD of the data matrix. Otherwise \( M \) is often chosen from \((\cdot)^p\) for some \( p \geq 0 \), or from a class of functions called \( M \)-estimators, with the goal of providing a more robust estimate than least squares in the face of outliers. Indeed, for \( p < 2 \), since one is not squaring the distances to the subspace, one is placing less emphasis on outliers and therefore capturing more of the remaining data points. For example, when \( M \) is the identity function, we are finding a subspace so as to minimize the sum of distances to it, which could arguably be more natural than finding a subspace so as to minimize the sum of squared distances. We can write this problem in the following form:

\[
\min_{S \text{ dim } k} \sum_i \text{dist}(S, a_i) = \min_{X \text{ rank } k} \sum_i \|(A - AX)_{i*}\|_2
\]

where \( A \) is the matrix in which the \( i \)-th row is the vector \( a_i \). This is the form of robust subspace approximation that we study in this work. We will be interested in the approximate version of the problem for which
the goal is to output a \(k\)-dimensional subspace \(S'\) for which with high probability,
\[
\sum_i \text{dist}(S', a_i) \leq (1 + \epsilon) \sum_i \text{dist}(S, a_i)
\]  
(1)
The particular form with \(M\) equal to the identity was introduced to the machine learning community by Ding et al. [11], though these authors employed heuristic solutions. The series of work in [13, 16] and [9, 13, 21, 6] shows that if \(M(\cdot) = | \cdot |^p\) for \(p \neq 2\), there is no algorithm that outputs a \((1 + 1/poly(d))\) approximation to this problem unless \(P = \text{NP}\). However, [6] also show that for any \(p\) there is an algorithm that runs in \(O(\text{nnz}(A) + (n + d) \cdot \text{poly}(k/e) + \text{exp}(\text{poly}(k/e)))\) time and outputs a \(k\)-dimensional subspace whose cost is within a \((1 + \epsilon)\) factor of the optimal solution cost. This provides a considerable computational savings since in most applications \(k \ll d \ll n\). Their work builds upon techniques developed in [13] and [12] which give \(O(nd \cdot \text{poly}(k/e) + \text{exp}((k/e)^{O(p)}))\) time algorithms for the \(p \geq 1\) case. These in turn build on the weak coreset construction of [10]. In other related work [7] give algorithms for performing regression with a variety of \(M\)-estimator loss functions.

Our Contributions. We give the first sketching-based solution to this problem. Namely, we show it suffices to compute \(Z \cdot A\), where \(Z\) is a \(\text{poly}(\log(nd)k/e^{-1}) \times n\) random matrix with entries chosen obliviously to the entries of \(A\). The matrix \(Z\) is a block matrix with blocks consisting of independent Gaussian entries, while other blocks consist of independent Cauchy random variables, and yet other blocks are sparse matrices whose cost is within a \((1 + \epsilon)\) factor of the optimal solution cost. This provides a considerable computational savings since in most applications \(k \ll d \ll n\). Their work builds upon techniques developed in [13] and [12] which give \(O(nd \cdot \text{poly}(k/e) + \text{exp}((k/e)^{O(p)}))\) time algorithms for the \(p \geq 1\) case. These in turn build on the weak coreset construction of [10]. In other related work [7] give algorithms for performing regression with a variety of \(M\)-estimator loss functions.

Streaming Model. Since \(Z\) is linear and oblivious, one can maintain \(Z \cdot A\) in the presence of insertions and deletions to the entries of \(A\). Indeed, given the update \(A_{i,j} \leftarrow A_{i,j} + \Delta\) for some \(\Delta \in \mathbb{R}\), we simply update the \(j\)-th column \(ZA_j\) in our sketch to \(ZA_j + \Delta \cdot Z \cdot e_i\), where \(e_i\) is the \(i\)-th standard unit vector. Also, the entries of \(Z\) can be represented with limited independence, and so \(Z\) can be stored with a short random seed. Consequently, we obtain the first algorithm with \(d \cdot \text{poly}(\log(nd)k/e^{-1})\) memory for this problem in the standard turnstile data stream model [20]. In this model, \(A \in \mathbb{R}^{n \times d}\) is initially the zero matrix, and we receive a stream of updates to \(A\) where the \(i\)-th update is of the form \((x_i, y_i, c_i)\), which means that \(A_{x_i, y_i}\) should be incremented by \(c_i\). We are allowed one pass over the stream, and should output a rank-\(k\) matrix \(X^t\) which is a \((1 + \epsilon)\) approximation to the robust subspace estimation problem, namely \(\sum_i \| (A - AX^t)_i \|_2 \leq (1 + \epsilon) \cdot \text{rank } k \sum_i \| (A - AX)_i \|_2\). The space complexity of the algorithm is the total number of words required to store this information during the stream. Here, each word is \(O(\log(nd))\) bits. Our algorithm achieves \(d \cdot \text{poly}(\log(nd)k/e^{-1})\) memory, and so only logarithmically depends on \(n\). This is comparable to the memory of streaming algorithms when \(M(\cdot) = \cdot^2\) [3, 15], which is the only prior case for which streaming algorithms were known.

Distributed Model. Since our algorithm maintains \(Z \cdot A\) for an oblivious linear sketch \(Z\), it is parallelizable, and can be used to solve the problem in the distributed setting in which there are \(s\) machines holding \(A^1, A^2, \ldots, A^s\), respectively, and \(A = \sum_i A^i\). This is called the arbitrary partition model [18]. In this model, we can solve the problem in one round with \(s \cdot d \cdot \text{poly}(\log(nd)k/e^{-1})\) communication by having each machine agree upon (a short seed describing) \(Z\), and sending \(ZA^i\) to a central coordinator who computes and runs our algorithm on \(Z \cdot A = \sum_i ZA^i\). The arbitrary partition model is stronger than the so-called
row partition model, in which the points (rows of $A$) are partitioned across machines. For example, if each machine corresponds to a shop, the rows of $A$ correspond to customers, the columns of $A$ correspond to items, and $A^i_{c,d}$ indicates how many times customer $c$ purchased item $d$ at shop $i$, then the row partition model requires customers to make purchases at a single shop. In contrast, in the arbitrary partition model, customers can purchase items at multiple shops.

2 Notation and Terminology

For a matrix $A$, let $A_{i,*}$ denote the $i$-th row of $A$, and $A_{*,j}$ denote the $j$-th column of $A$.

**Definition 2.1 ($\|\cdot\|_{2,1}$, $\|\cdot\|_{1,2}$, $\|\cdot\|_{1,1}$, $\|\cdot\|_{\text{med},1}$, $\|\cdot\|_F$).** For a matrix $A \in \mathbb{R}^{n \times m}$, let:

$$
\|A\|_{2,1} \equiv \sum_i \|A_{i,*}\|_2 \quad \|A\|_{1,2} \equiv \|A^T\|_{2,1} = \sum_j \|A_{*,j}\|_2 \\
\|A\|_F = \sqrt{\sum_i \|A_{i,*}\|_2^2} \quad \|A\|_{1,1} \equiv \sum_i \|A_{i,*}\|_1 \quad \|A\|_{\text{med},1} \equiv \sum_j \|A_{*,j}\|_{\text{med}}
$$

where $\|\cdot\|_{\text{med}}$ denotes the function that takes the median of absolute values.

**Definition 2.2 ($X^*$, $\Delta^*$).** Let:

$$
\Delta^* \equiv \min_{X \ \text{rank} \ k} \|A - AX\|_{2,1} \quad X^* \equiv \arg\min_{X \ \text{rank} \ k} \|A - AX\|_{2,1}
$$

**Definition 2.3 ($\alpha$, $\beta$-coreset).** For a matrix $A \in \mathbb{R}^{n \times d}$ and a target rank $k$, $W$ is an $\alpha$, $\beta$-coreset if its row space is an $\alpha$-dimensional subspace of $\mathbb{R}^d$ that contains a $\beta$-approximation to $X^*$. Formally:

$$
\arg\min_{X \ \text{rank} \ k} \|A - AXW\|_{2,1} \leq \beta \Delta^*
$$

**Definition 2.4 (Count-Sketch Matrix).** A random matrix $S \in \mathbb{R}^{r \times t}$ is a Count-Sketch matrix if it is constructed via the following procedure. For each of the $t$ columns $S_{*,i}$, we first independently choose a uniformly random row $h(i) \in \{1, 2, \ldots, r\}$. Then, we choose a uniformly random element of $\{-1, 1\}$ denoted $\sigma(i)$. We set $S_{h(i),i} = \sigma(i)$ and set $S_{j,i} = 0$ for all $j \neq i$.

For the applications of Count-Sketch matrices in this paper, it suffices to use $O(1)$-wise instead of full independence for the hash and sign functions. Thus these can be stored in $O(1)$ space, and multiplication $SA$ can be computed in $\text{nnz}(A)$ time. For more background on such sketching matrices, we refer the reader to the monograph [25].

We also use the following notation: $[n]$ denotes the set $\{1, 2, 3, \cdots, n\}$. $[E]$ denotes the indicator function for event $E$. $\text{nnz}(A)$ denotes the number of non-zero entries of $A$. $A^-$ denotes the pseudoinverse of $A$. $I$ denotes the identity matrix.

3 Algorithm Overview

At a high level we follow the framework put forth in [6] which gives the first input sparsity time algorithm for the robust subspace approximation problem. In their work Clarkson and Woodruff first find a crude $(\text{poly}(k), K)$-coreset for the problem. They then use a non-adaptive implementation of a residual
We present our results as two algorithms for the robust subspace approximation problem. The first runs in time which is no longer fixed parameter tractable as desired. Our key insight is that finding the best rank-$k$ approximation than the constant factor approximation of $\|\cdot\|_{2,1}$ norm.

Second, the residual sampling step requires sampling rows of $A$ with respect to probabilities proportional to their distance to the crude approximation (in our case $TA$). This is challenging because one does not know $TA$ until the end of the stream, much less the distances of rows of $A$ to $TA$. We handle this in Section 4.2 using a row-sampling data structure of [11, 22, 24], which for a matrix $B$ maintains a sketch $HB$ in a stream from which one can extract samples of rows of $B$ according to probabilities given by their norms. By linearity, it suffices to maintain $HA$ and $TA$ in parallel in the stream, and apply the sample extraction procedure to $HA \cdot (I - P_{TA})$, where $P_{TA} = (TA)^\top (TA(TA)^\top)^{-1}TA$ is the projection onto the rowspace of $TA$. Unfortunately, the extraction procedure only returns noisy perturbations of the original rows which majorly invalidates the analysis in [6] of the residual sampling. In Section 4.2 we give an analysis of non-adaptive noisy residual sampling which we name BootstrapCoreset. This gives a procedure for transforming our poly$(k)$-dimensional space containing a poly$(k)\log(d)$ approximation into a poly$(k)\log(d)$-dimensional space containing a $3/2$ factor approximation.

Third, requiring the initial crude approximation to be oblivious yields a coarser $\log(d)\poly(k)$ initial approximation than the constant factor approximation of [6]. Thus the dimension of the subspace after residual sampling is poly$(k)\log(d)$. Applying dimension reduction techniques reduces the problem to an instance with poly$(k)$ rows and $\log(d)\poly(k)$ columns. Here the black box algorithm of [3] would take time $d^{poly(k)}$ which is no longer fixed parameter tractable as desired. Our key insight is that finding the best rank-$k$ matrix under the Frobenius norm, which can be done efficiently, is a $\sqrt{\log d (\log \log d)}\poly(k)$ approximation to the $\|\cdot\|_{2,1}$ norm minimizer. From here we can repeat the residual sampling argument which this time yields a small instance with poly$(k)$ rows by $\sqrt{\log d (\log \log d)}\poly(k/e)$ columns. Sublogarithmic in $d$ makes all the difference and now enumerating can be done in time $(n + d)\poly(k/e) + \exp(poly(k/e))$. All this is done in parallel in a single pass of the stream.

Lastly, the sketching techniques applied after the residual sampling are not oblivious in [6]. We instead use an obvious median based embedding in Section 5.1 and show that we can still use the black box algorithm of [3] to find the minimizer under the $\|\cdot\|_{\text{med},1}$ norm in Section 5.2.

We present our results as two algorithms for the robust subspace approximation problem. The first runs in fully polynomial time but gives a coarse approximation guarantee, which corresponds to stopping before repeating the residual sampling a second time. The second algorithm captures the entire procedure, and uses the first as a subroutine.
Algorithm 1 \textsc{CoarseApprox}

\begin{algorithm}
\textbf{Input:} $A \in \mathbb{R}^{n \times d}$ as a stream  \\
\textbf{Output:} $X \in \mathbb{R}^{d \times d}$ such that $\|A - AX\|_{2,1} \leq \sqrt{\log d \log \log d} \cdot \text{poly}(k) \Delta^*$
\begin{algorithmic}[1]
\State $T \in \mathbb{R}^{\text{poly}(k) \times n} \leftarrow$ Sparse Cauchy matrix // as in Thm. 4.1
\State $C_1 \in \mathbb{R}^{\text{poly}(k) \times n} \leftarrow$ Sparse Cauchy matrix // as in Thm. 4.4
\State $S_1 \in \mathbb{R}^{\log d \cdot \text{poly}(k) \times d} \leftarrow$ Count Sketch composed with Gaussian // as in Thm. 4.3
\State $R_1 \in \mathbb{R}^{\text{poly}(k) \times d} \leftarrow$ Count Sketch matrix // as in Thm. 4.3
\State $G_1 \in \mathbb{R}^{\log d \cdot \text{poly}(k) \times \text{log} d \cdot \text{poly}(k)} \leftarrow$ Gaussian matrix // as in Thm. 4.3
\State Compute $TA$ online
\State Compute $C_1 A$ online
\State $U_1^\top \in \mathbb{R}^{d \cdot \text{poly}(k) \times d} \leftarrow \textsc{BootstrapCoreset}(A, TA, 1/2)$ // as in Alg. 3
\State $\hat{X} \in \mathbb{R}^{\text{poly}(k) \times \text{log} d \cdot \text{poly}(k)} \leftarrow \arg\min_{X} \text{rank } k \|C_1 (A - AR_1^\top \hat{X} U_1^\top) S_1^\top G_1\|_F$ // as in Fact 4.3
\State \textbf{return} $R_1^\top \hat{X} U_1^\top$
\end{algorithmic}
\end{algorithm}

Theorem 3.1 (Coarse Approximation in Polynomial Time). Given a matrix $A \in \mathbb{R}^{n \times d}$, Algorithm 1 with constant probability computes a rank $k$ matrix $X \in \mathbb{R}^{d \times d}$ such that:

$$\|A - AX\|_{2,1} \leq \sqrt{\log d \log \log d} \cdot \text{poly}(k) \cdot \|A - AX^*\|_{2,1}$$

that runs in time $O(nnz(A)) + d \cdot \text{poly}(k \log(n d))$. Furthermore, it can be implemented as a one-pass streaming algorithm with space $O(d \cdot \text{poly}(k \log(n d)))$ and time per update $O(\text{poly}(\log(n d) k))$.

\textbf{Proof Sketch} We show the following are true in subsequent sections:

1. The row span of $TA$ is a $(\text{poly}(k), \log d \cdot \text{poly}(k))$-coreset for $A$ (Section 4.1) with probability 24/25.
2. $\textsc{BootstrapCoreset}(A, TA, 1/2)$ is a $(\log d \cdot \text{poly}(k), 3/2)$-coreset with probability 49/50 (Section 4.2).
3. If:

$$\hat{X} = \arg\min_{X} \text{rank } k \|C_1 A S_1^\top G_1 - C_1 A R_1^\top \hat{X} U_1^\top S_1^\top G_1\|_F$$

then with probability 47/50:

$$\|A - AR_1^\top \hat{X} U_1^\top\|_{2,1} \leq \text{poly}(k) \sqrt{\log d \log \log d} \cdot \Delta^*$$

(Sections 4.3 and 4.4, with $\epsilon = 1/2$).

By a union bound, with probability 88/100 all the statements above hold, and the theorem is proved. $\textsc{BootstrapCoreset}$ requires $d \cdot \text{poly}(k \log(n d))$ space and time. Left matrix multiplications by Sparse Cauchy matrices $TA$ and $C_1 A$ can be done in $O(nnz(A))$ time (see Section J of [23] for a full description of Sparse Cauchy matrices). Computing remaining matrix products and $\hat{X}$ requires time $d \cdot \text{poly}(k \log d)$.
Algorithm 2 \((1 + \epsilon)\)-APPROX

**Input:** \(A \in \mathbb{R}^{n \times d}\) as a stream 

**Output:** \(X \in \mathbb{R}^{d \times d}\) such that \(\|A - AX\|_{2,1} \leq (1 + \epsilon)\Delta^*\)

1: \(\hat{X} \in \mathbb{R}^{\text{poly}(k) \times \log d \times \text{poly}(k)} \leftarrow \text{COARSEAPPROX}(A)\) \(\llbracket\) as in Thm. 3.1
2: \(C_2 \in \mathbb{R}^{\sqrt{\log \delta} \times \log d \times \text{poly}(k/\epsilon) \times d}\) \(\leftarrow\) Cauchy matrix \(\llbracket\) as in Thm. 5.1
3: \(S_2 \in \mathbb{R}^{\sqrt{\log \delta} \times \log d \times \text{poly}(k/\epsilon) \times d}\) \(\leftarrow\) Count Sketch composed with Gaussian \(\llbracket\) as in Thm. 5.2
4: \(R_2 \in \mathbb{R}^{\sqrt{\log \delta} \times d \times \text{poly}(k/\epsilon) \times d}\) \(\leftarrow\) Count Sketch composed with Gaussian \(\llbracket\) as in Thm. 5.1
5: \(G_2 \in \mathbb{R}^{\sqrt{\log \delta} \times \log d \times \text{poly}(k/\epsilon) \times \sqrt{\log \delta} \times \log d \times \text{poly}(k/\epsilon)}\) \(\leftarrow\) Gaussian matrix \(\llbracket\) as in Thm. 5.1
6: Compute \(AR_2^\top\) online
7: Compute \(AS_2^\top\) online
8: Let \(V \in \mathbb{R}^{\sqrt{\log d \times \text{poly}(k)} \times k}\) be such that \(\hat{X} = WV^\top\) is the rank-\(k\) decomposition of \(\hat{X}\)
9: \(U_2^\top \in \mathbb{R}^{\text{poly}(k/\epsilon) \times \sqrt{\log d \times \log d} \times d}\) \(\leftarrow\) \text{BOOTSTRAPCORESET}(A, V^\top U_1^\top, \epsilon') \(\llbracket\) as in Alg. 3, \(U_1\) as computed during \text{COARSEAPPROX} in line 1.
10: \(\hat{X}' \in \mathbb{R}^{\text{poly}(k/\epsilon) \times \sqrt{\log d \times \log d}}\) \(\leftarrow\) \(\text{argmin}_X \text{rank} \ k \|C_2(A - AR_2^\top XU_2^\top)S_2^\top G_2\|_{\text{med},1}\) \(\llbracket\) as in Thm. 5.2
11: \text{return} \(R_2^\top \hat{X}' U_2^\top\)

**Theorem 3.2** \(((1 + \epsilon)\)-Approximation). Given a matrix \(A \in \mathbb{R}^{n \times d}\), Algorithm 2 with constant probability computes a rank \(k\) matrix \(X \in \mathbb{R}^{d \times d}\) such that:

\[\|A - AX\|_{2,1} \leq (1 + \epsilon)\|A - AX^*\|_{2,1}\]

that runs in time

\[O(mn(A)) + (n + d) \text{poly}\left(\frac{k \log(n)}{\epsilon}\right) + \exp\left(\text{poly}\left(\frac{k}{\epsilon}\right)\right)\]

Furthermore, it can be implemented as a one-pass streaming algorithm with space \(O\left(d \text{poly}\left(\frac{k \log(n)}{\epsilon}\right)\right)\)

and time per update \(O(\text{poly}(\log(nd)k/\epsilon))\).

**Proof Sketch** We show the following are true in subsequent sections:

1. If \(V\) is such that \(\hat{X} = WV^\top\), then \(V^\top\) is a \((\text{poly}(k), \text{poly}(k)\sqrt{\log d \log d})\)-coreset with probability \(88/100\) (Theorem 5.1).
2. \(\text{BOOTSTRAPCORESET}(A, V^\top U_1^\top, \epsilon')\) is a \((\text{poly}(k/\epsilon'), \sqrt{\log d \log d}, (1 + \epsilon'))\)-coreset with probability \(49/50\) (Reusing Section 4.2).
3. If:

\[\hat{X}' \leftarrow \text{argmin}_X \|C_2(A - AR_2^\top XU_2^\top)S_2^\top G_2\|_{\text{med},1}\]

then with probability \(19/20:\)

\[\|A - AR_2^\top \hat{X}' U_2^\top\|_{2,1} \leq (1 + O(\epsilon'))\Delta^*\]

(Reusing Section 4.3 and Section 5.1).
4. A black box algorithm of [3] computes \(\hat{X}'\) to within \((1 + O(\epsilon'))\) (Section 5.2).

By a union bound, with probability \(81/100\) all the statements above hold. Setting \(\epsilon'\) appropriately small as a function of \(\epsilon\), the theorem is proved.
We construct a (poly(\(k \log(nd)/\epsilon\)) space and \(O(\text{nnz}(A)) + d \cdot poly(k \log(nd)/\epsilon)\) time. Right multiplication by the sketching matrices \(AS_x^2\) and \(AR_y^2\) can be done in time \(\text{nnz}(A)\). Computing remaining matrix products and \(\tilde{X}'\) requires time \((n + d) \cdot \text{poly}(\log(d)/\epsilon) + \exp(poly(k/\epsilon))\) (See end of Section 5.2 for details on this last bound).

We give further proofs and details of these theorems in subsequent sections.

4 Coarse Approximation

4.1 Initial Coreset Construction

We construct a \((\text{poly}(k), \log d \cdot \text{poly}(k))\)-coreset which will serve as our starting point.

**Theorem 4.1.** If \(T \in \mathbb{R}^{\text{poly}(k) \times n}\) is a sparse Cauchy matrix, then the row space of \(TA\) contains a \(k\) dimensional subspace with corresponding projection matrix \(X'\) such that with probability \(24/25\):

\[
\|A - AX'\|_{2,1} \leq \log d \cdot \text{poly}(k) \cdot \min_{X \text{ rank } k} \|A - AX\|_{2,1} = \log d \cdot \text{poly}(k) \cdot \Delta^*
\]

**Proof.** In order to deal with the awkward \(\|\cdot\|_{2,1}\) norm, we make use of a well known theorem due to Dvoretzky to convert it into an entrywise 1-norm.

**Fact 4.1 (Dvoretzky’s Theorem (Special Case), Section 3.3 of \([17]\)).** There exists an appropriately scaled Gaussian Matrix \(G \in \mathbb{R}^{d \times \frac{d \log(1/\epsilon)}{\epsilon^2}}\) such that w.h.p. the following holds for all \(y \in \mathbb{R}^d\) simultaneously

\[
\|y^T G\|_1 \in (1 \pm \epsilon) \|y^T\|_2
\]

Applying this to all rows at once: \(\|AX - A\|_{2,1} \in (1 \pm \epsilon) \|AXG - AG\|_{1,1}\).

We also use some existing machinery for input sparsity time \(\ell_1\) subspace embeddings.

**Fact 4.1.1 (Theorem 4 from \([12]\)).** For any given \(D \in \mathbb{R}^{s \times t}\), let \(\Pi \in \mathbb{R}^{r \times s}\) be a random Sparse Cauchy matrix with \(r = O(t^5 \log^5 t)\) defined as follows: \(\Pi = SC\) where \(S \in \mathbb{R}^{r \times s}\) has each column uniformly and independently chosen from the \(r\) standard basis vectors in \(\mathbb{R}^r\), and where \(C \in \mathbb{R}^{s \times s}\) is a diagonal matrix with diagonal entries chosen independently from the standard Cauchy distribution. Then with probability 99/100 simultaneously for all \(x \in \mathbb{R}^t\):

\[
\frac{1}{O(t^2 \log^2 t)} \cdot \|Dx\|_1 \leq \|\Pi Dx\|_1 \leq O(t \log t) \cdot \|Dx\|_1
\]

**Fact 4.1.2 (Lemma D.25 from \([23]\)).** If \(\Pi \in \mathbb{R}^{r \times s}\) is a Sparse Cauchy matrix as defined above, and \(B \in \mathbb{R}^{s \times t}\) is a fixed matrix, then with probability at least 99/100:

\[
\|\Pi B\|_1 \leq O(\log(rt)) \cdot \|B\|_1
\]

Finally, we also need a couple of structural lemmas which we state here without proof:

**Lemma 4.1.1 (Lemma 29 from \([6]\)).** For a fixed \((B, D)\) pair such that \(B \in \mathbb{R}^{r \times s}, D \in \mathbb{R}^{r \times t}\), if \(S \in \mathbb{R}^{s/poly(\epsilon) \times r}\) is a CountSketch Matrix composed with a matrix of i.i.d. Gaussians (for background on such sketching matrices, we refer the reader to the monograph \([25]\)), then with probability 99/100 both of the properties below hold:
1. \(\|S(BX - D)\|_{1,2} \geq (1 - \epsilon) \|BX - D\|_{1,2}\) for any \(X\).
2. If \(X^* = \text{argmin}_X \text{rank}_k \|BX - D\|_{1,2}\), then \(\|S(BX^* - D)\|_{1,2} \leq (1 + \epsilon) \|BX^* - D\|_{1,2}\).

Clarkson and Woodruff [6] call such an \(S\) a lopsided embedding for \((B, D)\) with respect to the \((1, 2)\)-norm.

**Lemma 4.1.2** (Lemma 31 from [6]). If \(R\) is a lopsided embedding for \((A^T, A^T)\), then:

\[
\min_{X \text{ rank } k} \|AR^T X - A\|_{2,1} \leq (1 + 3\epsilon^*)
\]

Let \(X' = \text{argmin}_X \|TAR^T X - TA\|_{2,1}, R^T \in \mathbb{R}^{d \times \text{poly}(k)}\) as in the lemma above and \(\epsilon = O(1)\).

Define \(E_1\) to be the event that the condition in Dvoretzky’s theorem is satisfied, \(E_2\) to be the event that Fact 41.1 holds for \(D = AR^T\), \(E_3\) to be the event that Fact 41.2 holds for \(B = AR^T X^* G - AG\), and \(E_4\) to be the event that \(R\) satisfies Lemma 4.1.2.

\(E_1\) holds w.h.p., \(E_2, E_3, E_4\) each separately hold with probability 99/100 (for a suitable choice of \(K\)). By a union bound, they all hold simultaneously with probability at least 24/25. Conditioned on this happening:

\[
\begin{align*}
\|AR^T X' - A\|_{2,1} &\leq \|AR^T X^* - A\|_{2,1} + \|AR^T (X^* - X')\|_{2,1} \quad (1) \\
&\leq \|AR^T X^* - A\|_{2,1} + \text{poly}(k) \cdot \|TAR^T (X^* - X')G\|_{1,1} \quad (2) \\
&\leq \text{poly}(k) \left( \|AR^T X^* - A\|_{2,1} + \|T(AR^T X^* - A)G\|_{1,1} + \|T(AR^T X' - A)G\|_{1,1} \right) \quad (3) \\
&\leq \text{poly}(k) \left( \|AR^T X^* - A\|_{2,1} + 2 \|T(AR^T X^* - A)G\|_{1,1} \right) \quad (4) \\
&\leq \text{poly}(k) \left( \|AR^T X^* - A\|_{2,1} + O(\log d) \| (AR^T X^* - A)G\|_{1,1} \right) \quad (5) \\
&\leq \log d \cdot \text{poly}(k) \|AR^T X^* - A\|_{2,1} \quad (6)
\end{align*}
\]

(1) and (3) hold by the triangle inequality, (2) since \(E_1\) and \(E_2\) hold, (4) by \(E_1\) again and since \(X'\) is the minimizer of the expression \(\|TAR^T X - TA\|_{2,1}\), (5) since \(E_3\) holds, and (6) by \(E_1\) again.

\(X'\) lies in the rowspace of \(TA\), since otherwise there is a rank-\(k\) projection \(Z\) onto the rows of \(TA\) with \(\|TAX' Z - TA Z\|_{2,1} = \|TAX' Z - TA\|_{2,1}\) smaller than \(\|TAX' - TA\|_{2,1}\). Since \(E_4\) holds,

\[
\|AR^T X^* - A\|_{2,1} \leq O(1) \Delta^*
\]

and thus the rowspace of \(TA\) contains a \(\log d \cdot \text{poly}(k)\) approximation. \(\square\)

Thus the rowspace of \(TA\) with \(T\) as in Theorem 4.1 above is a \((\text{poly}(k), \log d \cdot \text{poly}(k))\)-coreset for \(A\).

**4.2 Bootstrapping a Coreset**

Given a poor coreset \(Q\) for \(A\), we now show how to leverage known results about residual sampling from [10] and [6] to obtain a better coreset of slightly larger dimension.

**Theorem 4.2.** Given \(Q\), an \((\alpha, \beta)\)-coreset for \(A\), with probability 49/50 \texttt{BOOTSTRAP-CORESET} returns an \((\alpha + \beta \text{ poly}(k/\epsilon), (1 + \epsilon))\)-coreset for \(A\). Furthermore \texttt{BOOTSTRAP-CORESET} runs in space and time \(O(\text{poly}(\beta \log(nd)/k/\epsilon))\), with \(\text{poly}(\beta \log(nd)/k/\epsilon)\) time per update in the streaming setting.
Algorithm 3 BootstrapCoreset

Input: $A \in \mathbb{R}^{n \times d}, Q \in \mathbb{R}^{\alpha \times d}$ ($\alpha$, $\beta$)-coreset, $\epsilon \in (0, 1)$
Output: $U \in \mathbb{R}^{(\alpha + \beta \text{poly}(k/\epsilon)) \times d}$ ($\alpha + \beta \text{poly}(k/\epsilon), (1 + \epsilon)$)-coresets

1: Compute $HA$ online $//$ as in Lem. 4.2.2
2: $P \leftarrow \beta \text{poly}(k/\epsilon)$ samples of rows of $A(I - Q)$ according to $\mathcal{P}(HA(I - Q))$ $//$ as in Lem. 4.2.2
3: $U^T \leftarrow$ Orthonormal basis for $\text{RowSpan} \left( \begin{bmatrix} Q & P \end{bmatrix} \right)$
4: return $U^T$

Proof. Consider the following idealized noisy sampling process that samples rows of a matrix $B$. Sample a row $B_i$ of $B$ with probability $\frac{\|B_i\|_2}{\|B\|_2 1,2}$ and add an arbitrary noise vector $E_i$ such that $\|E_i\|_2 \leq \nu \|B_i\|_2$, where we fix the parameter $\nu = \frac{\epsilon}{100k^2}$. Supposing we had such a process $\mathcal{P}^*(B)$, we can prove the following lemma.

Lemma 4.2.1. Suppose $Q$ is an ($\alpha$, $\beta$)-coreset for $A$, and $P$ is a noisy subset of rows of the residual $A(I - Q)$ of size $\beta(\text{poly}(k/\epsilon))$ each sampled according to $\mathcal{P}^*(A(I - Q))$. Then with probability $99/100$, $\text{RowSpan}(Q) \cup \text{RowSpan}(P)$ is an ($\alpha + \beta \text{poly}(k/\epsilon)$) dimensional subspace containing a $k$-dimensional subspace with corresponding projection matrix $X'$ such that:

$$\|A - AX'\|_{2,1} \leq (1 + \epsilon)\Delta^*$$

Proof. Our theorem is identical to Theorem 45 from [10], which is in turn an adaptation of Theorem 9 from [10], except that our sampling procedure produces noisy samples instead of actual rows of $A(I - Q)$. We highlight the difference between our proof and the originals, and refer the reader to the sources for a full description.

Let $H_{\ell}$ denote the span of the rows of $Q$ adjoined with $\ell$ samples from $\mathcal{P}^*(A(I - Q))$. The analysis considers $k + 1$ phases during the construction of $H_{\ell}$, where phase $j$ is defined such that there exists a subspace $X_j$ with:

(i) the dimension of $\text{RowSpan}(X_j) \cap H_{\ell} \geq j$.

(ii) and letting $\delta = \epsilon/2k$ we have: $\|A(I - X_j)\|_{2,1} \leq (1 + \delta) \min_{\text{rank } k} \|A - AX\|_{2,1}$

In other words, the cost of the solution $X_j$ slowly gets worse with $j$, but $H_\ell$ recovers more of it. Note that in phase $k$, $\|A(I - X_k)\|_{2,1} \leq (1 + \epsilon) \min_{\text{rank } k} \|A - AX\|_{2,1}$, and furthermore $X_k \subseteq H_\ell$.

Let $Y_{\ell}$ denote the rank-$k$ projection whose row space is that of $X_j$, but rotated about the intersection $\text{RowSpan}(X_j) \cap H_{\ell}$ such that it also contains the vector in $H_{\ell}$ realizing the smallest nonzero principle angle with $X_j$. Note that $Y_{\ell}$ satisfies condition (i) for some $j' > j$, so it remains to show that with high probability, with a small number of new samples, condition (ii) is also satisfied. In particular, we show that if condition (ii) is violated, and thus if:

$$\|A(I - Y_{\ell})\|_{2,1} > (1 + \delta) \|A(I - X_j)\|_{2,1}$$

then with probability greater than $\delta/5K$ we sample a witness noisy-row $\hat{A}_{\ell'}$ with the property:

$$\|\hat{A}_{\ell'}(I - Y_{\ell})\|_2 \geq (1 + \delta/2) \|\hat{A}_{\ell'}(I - X_j)\|_2$$

(1)

(2)
By the Angle Drop Lemma (Lemma 13 of [10]), this witness implies that the smallest nonzero principle angle between $X_j$ and $H_{i}$ (and so $Y_{i}$) decreases. By the analysis of Theorem 9 of [10], once the angle is small enough, $Y_{i}$ will satisfy (ii). We now prove this fact.

By the assumption on $P^*$, $E_{i}'$ satisfies $\|E_{i}'\|_2 \leq \nu \|A_{i}(I - Q)\|_2$. Recall we set the noise parameter $\nu = \frac{\epsilon}{100k\beta} = \frac{\delta}{50\beta}$.

Let $W$ denote the set of indices of witness noisy rows, in other words the set of all $i$ such that $A_{i}$ satisfies (2). It suffices to show that:

$$\sum_{i \in W} \|A_{i}(I - Q)\|_2 \geq \frac{\delta}{5\beta} \|A(I - Q)\|_{2,1} \tag{3}$$

Suppose that (3) is false. Let $\tilde{X}_{i}$ be the matrix projecting onto $H_{i}$.

$$\|\tilde{A}_{i}(I - Y_{i})\|_2 \leq \|\tilde{A}_{i}(I - \tilde{X}_{i})\|_2 + \|\tilde{A}_{i}\tilde{X}_{i}(I - Y_{i})\|_2 \leq (1 + \delta/2) \|\tilde{A}_{i}(I - X_{i})\|_2$$

(4) and (6) follow from the triangle inequality, (5) since the definitions of $X_{i}$, $Y_{i}$ and $H_{i}$ imply that all elements of $H_{i}$ are closer to $\text{RowSpan}(Y_{i})$ than to $\text{RowSpan}(X_{j})$, and (7) since $\text{RowSpan}(Q) \subseteq H_{i}$.

For $i \notin W$, by definition $\|\tilde{A}_{i}(I - Y_{i})\|_2 < (1 + \delta/2) \|\tilde{A}_{i}(I - X_{i})\|_2$. Combining both the bounds we have that for all $i$;

$$\|\tilde{A}_{i}(I - Y_{i})\|_2 \leq (1 + \delta/2) \|\tilde{A}_{i}(I - X_{i})\|_2 + [i \in W] \cdot 2 \|\tilde{A}_{i}(I - Q)\|_2$$

Summing over all $i$,

$$\|\tilde{A}(I - Y_{i})\|_{2,1} \leq (1 + \delta/2) \|\tilde{A}(I - X_{i})\|_{2,1} + 2 \|\tilde{A}_{W}(I - Q)\|_2$$

By triangle inequality:

$$\|A(I - Y_{i})\|_{2,1} - \|E(I - Y_{i})\|_{2,1} \leq \left[ (1 + \delta/2) \|A(I - X_{i})\|_{2,1} + 2 \|A_{W}(I - Q)\|_{2,1} \right] + (1 + \delta/2) \|E(I - X_{i})\|_{2,1} + 2 \|E(I - Q)\|_{2,1}$$

Finally, rearranging:

$$\|A(I - Y_{i})\|_{2,1} \leq \left( 1 + \frac{\delta}{2} \right) \|A(I - X_{i})\|_{2,1} + \frac{2\delta}{5\beta} \cdot \beta \|A(I - X_{j})\|_{2,1} + 5 \|E\|_{2,1} \tag{8}$$

$$\leq \left( 1 + \frac{9\delta}{10} + 5\nu\beta \right) \|A(I - X_{j})\|_{2,1} \tag{9}$$

$$\leq (1 + \delta) \|A(I - X_{j})\|_{2,1}$$
Which contradicts our assumption that (1) held. (8) follows from the assumption that (3) is false and the fact that \( \|A(I - Q)\|_{2,1} \leq \beta \|A(I - X_j)\|_{2,1} \) and (9) since \( \|E\|_{2,1} \leq \nu \|A(I - Q)\|_{2,1} \).

Note that this proof goes through for any error matrix \( E \) satisfying \( \|E_i\| \leq \nu \|A_i\| \) for all \( i \). Also, as written in [6], the proof guarantees success with constant probability. We can repeat the sampling a constant number of times, keep all samples, and guarantee success with probability 99/100.

It remains to show that we can sample from \( \mathcal{P}^* \) in a stream.

**Lemma 4.2.2.** Let \( B \in \mathbb{R}^{n \times d} \) be a matrix, and let \( \delta, \nu \in (0, 1) \) be given. Also let \( s \) be a given integer. Then there is an oblivious sketching matrix \( H \in \mathbb{R}^{\text{poly}(s/(\delta \nu)) \times n} \) and a sampling process \( \mathcal{P} \), such that \( \mathcal{P}(HB) \) returns a collection of \( s' = O(s) \) distinct row indices \( i_1, \ldots, i_{s'} \in [n] \) and approximations \( \hat{B}_{ij} = B_{ij} + E_{ij} \) with \( \|E_{ij}\|_2 \leq \nu \|B_{ij}\|_2 \) for \( j = 1, \ldots, s \). With probability \( 1 - \delta \) over the choice of \( H \), the probability an index \( i \) appears in the sampled set \( \{i_1, \ldots, i_{s'}\} \) is at least the probability that \( i \) appears in a set of \( s \) samples without replacement from the distribution \( \left( \frac{\|B_{i1}\|_2}{\|B\|_{2,1}}, \ldots, \frac{\|B_{in}\|_2}{\|B\|_{2,1}} \right) \). Furthermore the multiplication \( HB \) and sampling process \( \mathcal{P} \) can be done in \( \text{nnz}(B) + d \cdot \text{poly}(s/(\delta \nu)) \) time, and can be implemented in the streaming model with \( d \cdot \text{poly}(s/(\delta \nu)) \) bits of space.

The theorem builds on the work of [11], [22] and [23].

**Proof.** We will show that with probability \( O(1) \cdot \delta' \), we produce a set of \( 1 - O(1) \cdot s \) samples such that the probability a noisy row \( \hat{B}_i = B_i + E_i \) with \( \|E_i\|_2 \leq \nu \|B_i\|_2 \) appears in this set is at least \( \frac{\|B_i\|_2}{\|B\|_{2,1}} \). Fixing \( \delta' = \delta/O(1) \) will give the claim.

**Algorithm 4 H-SKETCH**

**Input:** \( B \in \mathbb{R}^{n \times d} \)

**Output:** \( HB \in \mathbb{R}^{d \cdot \text{poly}(s/(\delta \nu)) \times d} \)

1: for level \( j \in [\ell] \)

2: \( H_{j,*} \leftarrow \) new hash table with \( w = O\left(\left(\frac{s \log n}{\delta \nu}\right)^{15}\right) \) buckets and independent hash function \( h_j : [n] \rightarrow [w] \) (each bucket stores a \( d \) dimensional vector).

3: Sample a set \( J_j \subset [n] \) where each \( i \in [n] \) is included with probability \( p_j = \frac{1}{2^j} \).

4: for \( v \in [w] \)

5: \( H_{j,v} = \sum_{i \in J_j} [h_j(i) = v] \cdot \varepsilon_j(i) \cdot B_{i,v} \) where \( \varepsilon_j(i) \) are 2-wise ind. uniform \( \pm 1 \) random variables.

6: end for

7: end for

8: return \[
\begin{bmatrix}
H^{(1)} \\
H^{(2)} \\
\vdots \\
H^{(\ell)}
\end{bmatrix}
\]

Before describing the algorithm we define a number of parameters.

- \( M \) is an estimate for \( \|B\|_{2,1} \) such that \( \|B\|_{2,1} \leq M \leq O(1) \|B\|_{2,1} \) (we show in Appendix A that we can calculate such an \( M \) with high probability).

- Setting \( T_j = M/2^j \), define \( S_j = \{ i \in [n] \mid \|B_i\|_2 \in (T_j, 2T_j) \} \) to be the \( j \)th level set of \( B \).
- Define \( s_j \) to be the number of rows in level \( j \).
- For convenience, we also use the notation \( S_{\geq j} = \bigcup_{j' \geq j} S_{j'} \) and \( S_{\leq j} = \bigcup_{j' \leq j} S_{j'} \).
- Let \( \ell = 4 \log(n/\delta) \) be the set of levels we consider in our sketch.
- Define a level \( j \in [\ell] \) to be important if \( s_j \geq \frac{\delta/2}{\epsilon n} \). Informally, \( j \) is an important level if the set of rows in level \( j \) contribute a significant fraction of \( \| B \|_{2,1} \).
- Let \( \mathcal{J} \subset [\ell] \) denote the set of all important levels.

Observe that for any level \( j \) we have \( s_j \leq 2^j \). It will suffice to consider only levels \( j \in [\ell] \), since \( s \leq n \) implies that these rows necessarily capture a \((1 - \delta'/s)\) fraction of the mass of \( B \). By definition, the idealized process sampling from the distribution \( \left( \frac{\| B_{1,s} \|_2}{\| B \|_2}, \ldots, \frac{\| B_{n,s} \|_2}{\| B \|_2} \right) \) will sample a level \( j \in [\ell] \) with probability \( 1 - \delta'/s \), and by a union bound all \( s \) samples come from such a level with probability \( 1 - \delta' \).

Similarly, the idealized process will take a single sample from an important level with probability \( (1 - \sum_{j \in \mathcal{J}} T_j s_j / \| B \|_{2,1}) \geq 1 - \delta'/s \), meaning it only every samples from important levels also with probability \( 1 - \delta' \).

The main idea of the sketch is the following. For every level \( j \), we subsample every row of \( B \) independently at random with probability proportional to \( 2^{-j} \), and then hash the subsampled rows independently at random into buckets (each bucket is a vector that is the sum of the vectors assigned to it). Doing so guarantees that with high enough probability, for every important level \( j \), there is a nearby level \( k \) such that with high probability at least one row from \( j \) is sampled in level \( k \). Furthermore, all sufficiently heavy rows in level \( k \) hash to different buckets, and all light rows contribute at most \( \| B \|_{2,1} \). In particular, this means that if any bucket in important level \( k \) has norm in the range \( ((1 - \nu)T_j, (2 + \nu)T_j) \), that bucket is of the form \( \tilde{B}_i = B_i + E_i \) where \( B_i \) is a row of \( B \) and \( E_i \) has small norm. We defer formal descriptions of these guarantees to Appendix A.4.

Next we argue that we can use the sketch from Algorithm 4 to produce samples from the idealized process with high enough probability. The general idea of the sampling algorithm \textsc{SAMPLER} is the following. Partition the rows of \( B \) by assigning each row to one of \( t = 100s^3 \) pieces uniformly at random: \( B^{(1)}, B^{(2)}, \ldots, B^{(t)} \). We can bound the probability that any two out of \( s \) samples from the idealized process come from the same piece by \( \left( \frac{s}{2} \right) \cdot \frac{\delta'}{1000s^3} \leq \delta' \) so we can condition on this being the case. Sketch each piece using \textsc{H-SKETCH} to obtain: \( H^{(1)}, H^{(2)}, \ldots, H^{(t)} \). Let \( \tilde{s}_j^{(p)}, s_j^{(p)}, M^{(p)}, T_j^{(p)} \) denote the quantities \( \ell, s_j, M \) and \( T_j \) respectively for piece \( B^{(p)} \). Using Lemma A.3.1, with constant probability we can calculate simultaneously for all \( p \) an \( O(1) \) estimate \( \tilde{b}_p \) for \( \| B^{(p)} \|_{2,1} \). Using Lemma A.3, we can also calculate simultaneously for all \( j \in [\ell^{(p)}] \) a \( O(1) \) estimate \( s_j^{(p)} \) for \( s_j^{(p)} \). Now repeat the following until we have generated \( s' \) samples.

Sample a piece with probability proportional to \( \tilde{s}_j T_j \). Examine level \( j \) of the output of Algorithm 4. If at least one bucket of this level has a norm that is in the target range \( (T_j, 2T_j) \), then output a uniform random choice of such a bucket. We show that this process generates samples with probabilities sufficiently close to those of the idealized process.

Let \( C_i \) be the event that noisy row \( \tilde{B}_i = B_i + E_i \) is extracted on line 7. Let \( G_p \) be the event that piece \( p \) is sampled on line 9. Let \( D_j \) be the event that level \( j \) is sampled on line 10. Finally let \( E \) denote the event that any noisy row of the form \( B_i + E_i \) with \( \| E_i \|_2 \leq \nu \| B_i \|_2 \) is extracted at all in iteration \( z \). We wish to understand the probability of \( C_i \):

\[
P[C_i] = P[C_i \mid E \land D_j \land G_p] \cdot P[E \mid D_j \land G_p] \cdot P[D_j \mid G_p] \cdot P[G_p]
\]
We have straightforward bounds on the last two probabilities:

\[ \mathbb{P}[G_p] = \frac{\tilde{b}_{p'}}{\sum_p \tilde{b}_p} = \Theta(1) \frac{\|B(p)\|_{2,1}}{\|B\|_{2,1}} \]

\[ \mathbb{P}[D_{j'} \mid G_p] = \frac{s_{j'}(p') T_{j'}(p)}{\sum_j s_{j'}(p') T_{j'}(p)} = \Theta(1) \cdot s_{j'}(p) \|B_1\|_2 / \|B(p)\|_{2,1} \]

Now we can also lower bound \( \mathbb{P}[E \mid D_{j'} \land G_p] \). \( E \) will not hold if either:

(i) a noisy row \( \tilde{B} \) is sampled but \( \tilde{B} \) cannot be written \( B_i + E_i \) with \( \|E_i\|_2 \leq \nu \).

(ii) no row is sampled at all.

If Lemmas \( \text{A.1} \) and \( \text{A.2} \) hold (i) will not occur. If Lemmas \( \text{A.1} \) and Corollary \( \text{A.1} \) hold (ii) will not occur. All these hold individually with probability at least \( 1 - O(1)/\log n \), so \( E \) holds with probability at least \( 1 - O(1)/\log n \). Finally, since conditioned on \( E \land D_{j'} \land G_p \), we pick any row in level \( j' \) from piece \( p' \) with the same probability i.e. \( 1/s_{j'}(p) \). Putting all of this together, we get that:

\[ \mathbb{P}[C_i] = \Theta(1) \cdot \frac{1}{s_{j'}(p)} \cdot \frac{s_{j'}(p) \|B_1\|_2}{\|B(p)\|_{2,1}} \cdot \frac{\|B(p)\|_{2,1}}{\|B\|_{2,1}} = \Theta(1) \frac{\|B_i\|_2}{\|B\|_{2,1}} \]
We show how to reduce the right dimension of our problem. This result is used in both Algorithm 4.2.1 and a union bound, the projection matrix of \( \text{RowSpan}(Q) \cup \text{RowSpan}(P) \) is an \((\alpha + \beta \log(k/e), 1 + e)\)-coreset for \( A \) with probability 49/50. \( \text{BOOTSTRAPCORESCET} \) takes total time \( O(\text{nnz}(A)) + O(d \cdot \text{poly}(\beta \log(nd)k/e)) \) and space \( O(d \cdot \text{poly}(\beta \log(nd)k/e)) \).

Finally note that \( \text{H-SKETCH} \), and the \( \|\cdot\|_{2,1} \)-norm estimation procedure of Lemma A.3.1, can be implemented as oblivious linear sketches. Since no two distinct pieces share any rows in common, all matrix multiplications can be done in \( \text{nnz}(B) + d \cdot \text{poly}(s/\delta) \) time. Furthermore they can be implemented in the streaming model with \( d \cdot \text{poly}(s/\delta) \) bits of space.

Setting \( b = \log(nd), \delta = 1/100, \nu = \frac{\epsilon}{100k \beta} \) and \( s = \beta \text{poly}(k/e) \), it follows that \( P \) contains \( \beta \text{poly}(k/e) \) samples from \( P^*(A(I - Q)) \) with probability 99/100. By Lemma 4.3.1 and a union bound, the projection matrix of \( \text{RowSpan}(Q) \cup \text{RowSpan}(P) \) is an \((\alpha + \beta \text{poly}(k/e), (1 + e))\)-coreset for \( A \) with probability 49/50.

Note that in our main algorithm we cannot compute the projection \( A(I - Q) \) until the after the stream is finished. Fortunately, since \( H \) is oblivious, we can right multiply \( HA \) by \( (I - Q) \) once \( Q \) is available, and only then perform the sampling procedure \( \mathcal{P} \).

### 4.3 Right Dimension Reduction

We show how to reduce the right dimension of our problem. This result is used in both Algorithm 4.2.1 and Algorithm 4.3.2.

**Theorem 4.3.** If \( U^\top \) is an \((\alpha, \beta)\)-coreset, \( S \in \mathbb{R}^{\alpha \cdot \text{poly}(k/e) \times d} \) is a CountSketch matrix composed with a matrix of i.i.d. Gaussians, and \( R \in \mathbb{R}^{d \times \text{poly}(k/e)} \) is a CountSketch matrix, then with probability 49/50, if \( X' = \arg\min_X \|AS^\top - AR^\top XU^\top S^\top\|_{2,1} \) then:

\[
\|A - AR^\top X'U^\top\|_{2,1} \leq (1 + O(\epsilon)) \min_{X \text{ rank } k} \|A - AXU^\top\|_{2,1}
\]

**Proof.** Here we apply reasoning similar to that at the bottom of page 32 from [6]. We need a couple of lemmas from [6].

**Lemma 4.3.1** (Lemma 30 from [6]). If \( S \) is a lopsided embedding for \((B, D)\), then if \( X'' \) has the property that \( \|SBX'' - SD\|_{1,2} \leq \kappa \min_X \|SBX - SD\|_{1,2} \), for some \( \kappa \), then:

\[
\|BX'' - D\|_{1,2} \leq \kappa(1 + 3\epsilon) \min_X \|BX - D\|_{1,2}
\]

**Lemma 4.3.2.** If \( U \in \mathbb{R}^{d \times \alpha} \) and \( R \in \mathbb{R}^{\text{poly}(k/e) \times d} \) is a CountSketch matrix, then with probability 99/100:

\[
\min_{X \text{ rank } k} \|A - AR^\top XU^\top\|_{2,1} \leq (1 + 3\epsilon) \min_{X \text{ rank } k} \|A - AXU^\top\|_{2,1}
\]

**Proof.** Let \( V^* = \arg\min_Y \|UV - AT\|_{1,2} \) and let \( V = V_1V_2 \) be its rank factorization. Applying Lemmas 4.1.1 and 4.3.1, \( R \) is a lopsided embedding for \((UV_1, A^\top)\) with probability 99/100. If \( Y = \arg\min_Y \|R(UV_1 Y - A^\top)\|_{1,2} \) then:

\[
\|UV_1 Y - A^\top\|_{2,1} \leq (1 + 3\epsilon) \|UV^* - A^\top\|_{1,2} \leq (1 + 3\epsilon) \min_{X \text{ rank } k} \|A - AXU^\top\|_{2,1}
\]
But $Y = (RUV_1)^{-1}RA^T$, and taking transposes this means that:

$$\min_{X \text{ rank } k} \| A - AR^T U X U^T \|_{2,1} \leq \| A - AR^T ((RUV_1)^{-1})^T V_1 U^T \|_{2,1} \leq (1 + 3\epsilon) \min_{X \text{ rank } k} \| A - AXU^T \|_{2,1}$$

From the last lemma, a solution to $\min_{X \text{ rank } k} \| A - AR^T U X U^T \|_{2,1}$ will yield a $(1 + O(\epsilon))$-approximate solution to the problem $\min_{X \text{ rank } k} \| A - AXU^T \|_{2,1}$. Lemma 4.3.2 holds with probability $99/100$. Applying Lemma 4.4 with probability $99/100$, $S \in \mathbb{R}^{d \times \alpha \cdot \text{poly}(k/\epsilon)}$ CountSketch composed with a Gaussian is a lopsided embedding for $(U, A^T)$. Union bounding over these events, and applying Lemma 4.3.1 with $C$ as the set of matrices in RowSpan$(RA^T)$ proves the claim with probability $49/50$.

### 4.4 Left Dimension Reduction

We show how to reduce the left dimension of our problem. Together with results from Section 4.3, this preserves the solution to $X^*$ to within a coarse \( \sqrt{\log d \log \log d \cdot \text{poly}(k/\epsilon)} \) factor.

**Theorem 4.4.** Suppose the matrices $S_1$, $R_1$ and $U_1$ are as in Algorithm 4. If $C_1 \in \mathbb{R}^{\text{poly}(k/\epsilon) \times n}$ is a Sparse Cauchy matrix, and $G_1 \in \mathbb{R}^{\log d \cdot \text{poly}(k/\epsilon) \times \log d \cdot \text{poly}(k/\epsilon)}$ is a matrix of appropriately scaled i.i.d. Gaussians (as in Fact 4.4), and

$$\hat{X} = \arg \min_{X \text{ rank } k} \| C_1 A S_1^T G_1 - C_1 A R_1^T X U_1^T S_1^T G_1 \|_F$$

then with probability $24/25$:

$$\| A S_1^T - A R_1^T \hat{X} U_1^T S_1^T \|_{2,1} \leq \sqrt{\log d \log \log d \cdot \text{poly}(k/\epsilon) \cdot \Delta^*}$$

**Proof.** Define $E_1$ to be the event that the condition in Dvoretzky’s theorem is satisfied, $E_2$ to be the event that Fact 4.1.1 holds for $D = AR_1^T$, and $E_3$ to be the event that Fact 4.1.2 holds for $B = (A S_1^T - A R_1^T X^* U_1^T S_1^T) G_1$. $E_1$ holds w.h.p., $E_2$, $E_3$ each separately hold with probability $99/100$ (for a suitable choice of $K$). By a union bound, they all hold simultaneously with probability at least $24/25$. Conditioned on this happening:

$$\| A S_1^T - A R_1^T \hat{X} U_1^T S_1^T \|_{2,1} \leq \| A S_1^T - A R_1^T X^* U_1^T S_1^T \|_{2,1} + \| A R(X^* - \hat{X}) U_1^T S_1^T \|_{2,1}$$

(1)

$$\leq \| A S_1^T - A R_1^T X^* U_1^T S_1^T \|_{2,1} + \text{poly}(k/\epsilon) \| C A R(X^* - \hat{X}) U_1^T S_1^T G_1 \|_{1,1}$$

(2)

$$\leq \text{poly}(k/\epsilon) \left[ \| A S_1^T - A R_1^T X^* U_1^T S_1^T \|_{2,1} + \| C(A - A R_1^T X^* U_1^T) S_1^T G_1 \|_{1,1} \right]$$

(3)

$$\leq \text{poly}(k/\epsilon) \left[ \| A S_1^T - A R_1^T X^* U_1^T S_1^T \|_{2,1} + \sqrt{\log 7} \| C(A - A R_1^T \hat{X} U_1^T) S_1^T G_1 \|_F \right]$$

(4)

$$\leq \text{poly}(k/\epsilon) \left[ \| A S_1^T - A R_1^T X^* U_1^T S_1^T \|_{2,1} + \sqrt{\log d \log \log d} \| C(A S_1^T - A R_1^T X^* U_1^T S_1^T) G_1 \|_{1,1} \right]$$

(5)

$$\leq \text{poly}(k/\epsilon) \left[ \| A S_1^T - A R_1^T X^* U_1^T S_1^T \|_{2,1} + \sqrt{\log d \log \log d} \| (A S_1^T - A R_1^T X^* U_1^T S_1^T) G_1 \|_{1,1} \right]$$

(6)

$$\leq \sqrt{\log d \log \log d \cdot \text{poly}(k/\epsilon)} \| A S_1^T - A R_1^T X^* U_1^T S_1^T \|_{2,1}$$

(7)
(1) and (3) hold by triangle inequality, (2) since $E_1$ and $E_2$ hold, (4) comes from the relationship between the 1-norm and 2-norm, (5) since $\hat{X}$ is the minimizer of the expression $\|C_1(A - C_1AR_1^TXU_1^T)S_1^T\|_F$ and $p$-norms decrease with $p$, (6) since $E_3$ holds, and (7) by $E_1$ again.

The rank constrained Frobenius norm minimization problem above has a closed form solution.

**Fact 4.2.** For a matrix $M$, let $U_M\Sigma_MV_M^T$ be the SVD of $M$. Then:

$$\text{argmin}_{X \text{ rank } k} \|Y - ZXW\|_F = Z^{-1}[U_ZU_2^TYV_WV_W^T]kW$$

### 5 (1 + $\epsilon$)-Approximation

#### 5.1 Left Dimension Reduction

The following median based embedding allows us to reduce the left dimension of our problem. Together with results from Section 4.2, this preserves the solution to $X^*$ to within a $(1 + O(\epsilon))$ factor.

**Theorem 5.1.** Suppose $S_2$, $R_2$ and $U_2$ are as in Algorithm 4. If $C_2 \in \mathbb{R}^{\log d \log d \log d \text{poly}(k/\epsilon) \times n}$ is a Cauchy matrix, and $G_2 \in \mathbb{R}^{\log d \log d \log d \text{poly}(k/\epsilon) \times \log d \log d \text{poly}(k/\epsilon)}$ is a matrix of appropriately scaled i.i.d. Gaussians (as in Fact 4.1), and:

$$\hat{X}' = \text{argmin}_{X \text{ rank } k} \|C_2AS_2^TG_2 - C_2AR_2^TXU_2^TS_2^TG_2\|_{\text{med},1}$$

then with probability 99/100:

$$\|AS_2^TG_2 - AR_2^T\hat{X}'U_2^TS_2^TG_2\|_{1,1} \leq (1 + \epsilon) \min_{X \text{ rank } k} \|AS_2^TG_2 - AR_2^TXU_2^TS_2^TG_2\|_{1,1}$$

**Proof.** The following fact is known:

**Fact 5.1** (Lemma F.1 from [2]). Let $L$ be a $t$ dimensional subspace of $\mathbb{R}^n$. Let $C \in \mathbb{R}^{m \times s}$ be a matrix with $m = O\left(\frac{1}{\epsilon^2} t \log \frac{1}{\epsilon}\right)$ and i.i.d. standard Cauchy entries. With probability 99/100, for all $x \in L$ we have

$$(1 - \epsilon) \|x\|_1 \leq \|Cx\|_{\text{med},1} \leq (1 + \epsilon) \|x\|_1$$

The theorem statement is simply the lemma applied to $L = \text{ColSpan}([AS_2^T | AR_2^T])$.

#### 5.2 Solving Small Instances

Given problems of the form $\hat{X} = \text{argmin}_{X \text{ rank } k} \|Y - ZXW\|_{\text{med},1}$, we leverage an algorithm for checking the feasibility of a system of polynomial inequalities as a black box.

**Lemma 5.1.** [3] Given a set $K = \{\beta_1, \cdots, \beta_s\}$ of polynomials of degree $d$ in $k$ variables with coefficients in $\mathbb{R}$, the problem of deciding whether there exist $X_1, \cdots, X_k \in \mathbb{R}$ for which $\beta_i(X_1, \cdots, X_k) \geq 0$ for all $i \in [s]$ can be solved deterministically with $(sd)^{O(k)}$ arithmetic operations over $\mathbb{R}$.

**Theorem 5.2.** Fix any $\epsilon \in (0, 1)$ and $k \in [0, \min(m_1, m_2)]$. Let $Y \in \mathbb{R}^{n \times m''}$, $Z \in \mathbb{R}^{n \times m_1}$, and $W \in \mathbb{R}^{m_2 \times m''}$ be any matrices. Let $C \in \mathbb{R}^{m' \times n}$ be a matrix of i.i.d. Cauchy random variables, and $G \in \mathbb{R}^{m'' \times m}$ be a matrix of scaled i.i.d. Gaussian random variables. Then conditioned on $C$ satisfying Fact 5.1 for the adjoint matrix $[Y, Z]$ and $G$ satisfying the condition of Fact 4.1, a rank-$k$ projection matrix $X$ can be found that minimizes $\|C(Y - ZXW)G\|_{\text{med},1}$ up to a $(1 + \epsilon)$-factor in time $\text{poly}(m'm''/\epsilon)^{O(mk)}(m'' + m')\text{poly}(1/\epsilon)$, where $m = \max(m_1, m_2)$. 

16
Proof. We write $X = PQ$, where $P$ is $m_1 \times k$ and $Q$ is $k \times m_2$, to ensure that $X$ is rank $\leq k$.

Guess a permutation $\pi_j$ for each column $j$ of $C(ZXW - Y)G$ and define constraints enforcing the permutation. Since the $(i, j)$-th entry of the matrix is $\sum_{k, \ell} (CZ)_{ik}X_{k\ell}(WG)_{\ell j} - (CY)_{ij}$ these constraints are of the form $((C(ZXW - Y)G)_{\pi_j(i)j}^2 \leq ((C(ZXW - Y)G)_{\pi_j(i+1)j})^2$. Then define the median of the $j$-th column to be:

$$M_j = \left( |(C(ZXW - Y)G)_{\pi_j(m\nu/2)j}| + |(C(ZXW - Y)G)_{\pi_j(m\nu/2)j}| \right)/2$$

which can be expressed via polynomial constraints. Thus we have $O(mk + m'' \text{poly}(1/\epsilon))$ variables in our polynomial inequality system, $O(mk)$ variables to describe $P$ and $Q$, and $m'' \text{poly}(1/\epsilon)$ variables to describe the column medians $M_j$. We have poly$(m'm''/\epsilon)$ constraints, each involving polynomials of $O(1)$ degree. By Lemma 5.1, checking the feasibility of this system takes time $\text{poly}(m'm''/\epsilon)^O(mk + m'' \text{poly}(1/\epsilon))$.

We can minimize the objective $\sum_j M_j$ using binary search. This requires a lower bound on the objective value, which we can get by noting from Fact 5.1 that:

$$\min_X \|CZXWG - CY\|_{med,1} \geq (1 - \epsilon) \min_X \|ZXW - Y\|_{1,1} \geq (1 - \epsilon) \min_X \|ZXW - Y\|_{2,1}$$

As in the proof of Theorem 51 in [8], when the solution is constrained to be rank $k$, the right hand side is lower bounded by $\frac{1}{\text{poly}(d)}(\sigma_{k+1}(Y))^{1/2}$ (where $\sigma_{k+1}(Y)$ is the $k+1$st singular value of $Y$), which itself is lower bounded by $\left(\frac{1}{\text{exp(poly}(m'm''))}\right)^k$. Thus we can do binary search in $\text{poly}(m'm''/\epsilon)$ steps.

Finally, since there are $m'' \cdot m'$ possible permutation guesses, the entire procedure takes time $\text{poly}(m'm''/\epsilon)^O(mk + (m'' + m') \text{poly}(1/\epsilon))$.

We remark that if, as we do in our algorithm, we set the all the parameters $m$, $m'$ and $m''$ to be $\log \log d \sqrt{\log d} \cdot \text{poly}(k/\epsilon)$, we can write the runtime of this step (Line 9 of Algorithm 2) as $(n+d) \text{poly}(k/\epsilon) + \text{exp(poly}(k/\epsilon))$. If $\text{poly}(k/\epsilon) \leq \sqrt{\log d}/(\log \log d)^2$, then this step is captured in the $(n + d) \text{poly}(k/\epsilon)$ term. Otherwise this step is captured in the $\text{exp(poly}(k/\epsilon))$ term.

6 Experiments

In this section we empirically demonstrate the effectiveness of the COARSE APPROX compared to the truncated SVD. We experiment on synthetic and real world data sets. Since the algorithm is randomized, we run it 20 times and take the best performing run. For a fair comparison, we use an input sparsity time approximate SVD as in [5].

For the synthetic data, we use two example matrices all of dimension $1000 \times 100$. In Figure 13 we use a Rank-3 matrix with additional large outlier noise. First we sample $U$ random $100 \times 3$ matrix and $V$ random $3 \times 10$ matrix. Then we create a random sparse matrix $W$ with each entry nonzero with probability 0.9999 and then scaled by a uniform random variable between 0 and 10000. We use $10 \cdot UV + W$. In Figure 13 we create a simple Rank-2 matrix with a large outlier. The first row is $n$ followed by all zeros. All subsequent rows are 0 followed by all ones.
While the approximation guarantee of COARSEAPPROX is weak, we find that it performs well against the SVD baseline in practice on our examples, namely when the data has large outliers rows. The second example in particular serves as a good demonstration of the robustness of the (2,1)-norm to outliers in comparison to the Frobenius norm. When $k = 1$, the truncated SVD which is the Frobenius norm minimizer recovers the first row of large magnitude, whereas our algorithm recovers the subsequent rows. Note that both our algorithm and the SVD recover the matrix exactly when $k$ is greater than or equal to rank.

We have additionally compared our algorithm against the SVD on two real world datasets from the UCI Machine Learning Repository: Glass is a $214 \times 9$ matrix representing attributes of glass samples, and E.Coli is a $336 \times 7$ matrix representing attributes of various proteins. For this set of experiments, we use a heuristic extension of our algorithm that performs well in practice. After running COARSEAPPROX, we iterate solving $Y_t = \min_Y \| CAS^t G - Y Z_{t-1} \|_{1,1}$ and $Z_t = \min_Z \| CAS^t G - Y_t Z \|_{1,1}$ (via Iteratively Reweighted Least Squares for speed). Finally we output the rank $k$ Frobenius minimizer constrained to RowSpace$(Y_t Z_t)$. In Figure 1c we consistently outperform the SVD by between 5% and 15% for nearly all $k$, and nearly match the SVD otherwise. In Figure 1d we are worse than the SVD by no more than 5% for $k = 1$ to 4, and beat the SVD by up to 50% for $k = 5$ and 6. We have additionally implemented a greedy column selection algorithm which performs worse than the SVD on all of our datasets.
Acknowledgements: We would like to thank Ainesh Bakshi for many helpful discussions. D. Woodruff thanks partial support from the National Science Foundation under Grant No. CCF-1815840. Part of this work was also done while D. Woodruff was visiting the Simons Institute for the Theory of Computing.

References


A Elided Proofs

A.1 **H-SKETCH** Guarantees

Note that in the proofs that follow, we require that our sketching matrices have small representations. For each level \( j \) of each **H-SKETCH**, we can store the matrix as three different \( O(1) \)-wise independent hashes:

1. \( P_j : [n] \rightarrow \{0, 1\} \) determines whether the row is subsampled
2. \( h_j : [n] \rightarrow [w] \) determines the bucket
3. \( \varepsilon_j : [n] \rightarrow \pm 1 \) determines the sign.

Since our sketching matrices are stored with small seeds and do not guarantee full independence of the entries of the matrix, we cannot easily use Chernoff-Hoeffding bounds.

For ease of notation below, we define a parameter \( C = O(\frac{s \log n}{\delta^3}) \) with a sufficiently large constant.
For any level $j$, we separate the rows of $B$ into “light” and “heavy” rows with respect to $j$:

$$S^j_L = S_{\geq j + \log(C^6)+1}$$
$$S^j_H = S_{\leq j + \log(C^6)}$$

We first analyze the contribution of light and heavy elements to level $j$ of the sketch.

**Lemma A.1** (Noise from light elements is small). For any $j \in [\ell]$, with probability $1 - O(1)/\log n$:

$$\max_v \sum_{i \in I_j} [i \in I_j] [h_j(i) = v] \cdot \|B_i\|_2 \leq \frac{0.1 \cdot \nu \cdot T_j}{C^2}$$

**Proof.** Define the variables $Z_i^v = [i \in J_j] \cdot [h_j(i) = v] \cdot \varepsilon_j(i) \cdot \|B_i\|_2$, where $i \in S^j_L$. For a fixed $v$, we first study $\sigma^2$, the second moment of $\sum_{i \in S^j_L} Z_i^v$.

$$\sigma^2 = \mathbb{E} \left( \left( \sum_i [i \in J_j] [h_j(i) = v] \cdot \varepsilon_j(i) \cdot \|B_i\|_2 \right)^2 \right)$$

$$= \sum_{i,i' \in S^j_L} \mathbb{E} ([i \in J_j] [i' \in J_j] [h_j(i) = v] [h_j(i') = v]) \cdot \mathbb{E} [\varepsilon_j(i) \varepsilon_j(i')] \cdot \|B_i\|_2 \cdot \|B_{i'}\|_2$$

$$= \sum_{i \in S^j_L} \frac{P_j}{w} \cdot \|B_i\|_2^2$$

$$\leq \frac{P_j}{w} \max_i (\|B_i\|_2) \cdot \sum_i \|B_i\|_2$$

$$\leq \frac{T_j^2}{wC^6}$$

In step (2) we used the fact that the two variables $[i \in J_j] [i' \in J_j] [h_j(i) = v] [h_j(i') = v]$ and $\varepsilon_j(i) \varepsilon_j(i')$ are independent. In step (3), we used the 2-wise independence of $\varepsilon_j$ and the fact that $\varepsilon_j(i) \varepsilon_j(i') = 1$ if $i = i'$ and 0 otherwise. In step (5) we used the fact that $\frac{P_j}{w} \ell \leq \frac{M_j}{wC} = \frac{T_j}{w}$ and:

$$\|B_i\|_2 \leq T_j + \log(C^6) = T_j/2^{\log(C^6)} = \frac{T_j}{C^6}$$

By Chebyshev’s inequality:

$$\mathbb{P} \left( \sum_{i \in S^j_L} Z_i \geq 0.1 \cdot \nu \cdot T_j \right) \leq \frac{O(1)}{w \log n}$$

The desired bound follows by a union bound over the $w$ buckets. □

**Lemma A.2** (Heavy Elements do not collide). For any level $j \in [\ell]$, with probability at least $1 - \frac{O(1)}{\log n}$ no two elements from $S^j_H$ hash to the same bucket.

**Proof.** We can bound the expected number of samples:

$$\mathbb{E} \left( \sum_{i \in S^j_H} [i \in I_j] \right) \leq \left( \sum_{j' = 1}^{j + \log(C^6)} s_{j'} \right) \cdot P_j \leq 2^{j + \log(C^6) + 1} \cdot 2^{-j} = 2C^6$$

21
Thus, by Markov’s bound,
\[
P \left[ \sum_{i \in S^j_k} [i \in J_j] > C^7 \right] \leq \frac{O(1)}{\log n} \]
Thus, no more than \( C^7 \) heavy elements are subsampled with high probability. Conditioned on this happening, we can bound the probability that any two of them hash into the same bucket by:
\[
w^{-1} \cdot \left( \frac{C^7}{2} \right) \leq \frac{C^{14}}{w} \leq \frac{1}{\log n}
\]
since we chose \( w = O(C^{15}) \). The claim follows by a union bound over the two \( 1/\log n \) probability events.

Lemma A.3 (Level estimates). For any important level \( j \in J \), let \( \hat{s}_j \) be the number of buckets in \( H^{(k)} \) with norm in the interval \( [(1 - \nu)T_j, (2 + \nu)T_j] \), where \( k = \max \left( 0, j - \log C^2 \right) \). Let \( \hat{s}_j = 2\hat{s}_j p_k^{-1} \). Then with probability \( 1 - O(1)/\log n \) it holds that \( \hat{s}_j \in [s_j, 4s_j] \).

Proof. By Lemma A.2, with probability \( 1 - O(1)/\log n \) the rows in \( S^j_k \) do not collide, and by Lemma A.1, with probability \( 1 - O(1)/\log n \) the contribution of elements in \( S^j_k \) is less than \( \frac{0.1\nu T_j}{C^2} = 0.1\nu T_j \) (due to the relationship between \( j \) and \( k \)). Conditioned on this holding, \( \hat{s}_j = J^j_k \), where \( J^j_k \) is the number of elements of \( S_j \) subsampled in level \( k \).

If \( k = 0 \), all rows of \( S_j \) are subsampled, \( p_k = 1 \) and the claim is proved. Otherwise, we use a second moment method.
\[
\mathbb{E} \left[ J^j_k \right] = s_j p_k
\]
and additionally:
\[
\mathbb{E} \left[ J^j_k \right]^2 = \mathbb{E} \left[ \left( \sum_{i \in S_j} [i \in J^j_k] \right)^2 \right]
\]
\[
= \sum_{i,i' \in S_j} \mathbb{E} \left[ [i \in J^j_k] [i' \in J^j_k] \right]
\]
\[
= \sum_{i \in S_j} \mathbb{E} \left[ [i \in J^j_k] \right] + \sum_{i \neq i' \in S_j} \mathbb{E} \left[ [i \in J^j_k] [i' \in J^j_k] \right]
\]
\[
\leq s_j p_k + s_j^2 p_k^2
\]
Note that we only use the 2-wise independence of the subsampling function \( P_k \). Thus \( |J^j_k| \) has variance \( \sigma^2 \leq s_j p_k \). Using Chebyshev’s inequality:
\[
P \left[ \left| J^j_k \right| - s_j p_k \geq \frac{s_j p_k}{2} \right] \leq \frac{2}{s_j p_k}
\]
Since \( s_j p_k \geq \log n \), the claim follows from a union bounding guaranteeing that Lemmas A.1 and A.2 hold together with the last event that the number of subsampled elements from \( S_j \) is within a factor of two of its expectation.

Corollary A.1 (At least one element is sampled). For any important level \( j \in J \), with probability at least \( 1 - O(1)/\log n \), at least one element of \( S_j \) is subsampled in \( J_k \) when \( k = \max \left( 0, j - \log C^2 \right) \).
A.2 High Probability \( \| \cdot \|_{2,1} \) Estimation

Lemma A.4. Given a matrix \( B \in \mathbb{R}^{n \times d} \), there is an algorithm that with probability \( 1 - 1/(50t) \) outputs an estimate \( M \) such that \( \| B \|_{2,1} \leq M \leq 20010 \| B \|_{2,1} \). Furthermore this algorithm runs in time \( O(mz(B) + O(n + d)(\text{poly log}(ndt))) \) and can be implemented in the streaming model with \( d \cdot \text{poly}(\log(ndt)) \) bits of space.

Proof. First we prove an intermediate result:

Lemma A.4.1. If \( S \) is a Count Sketch matrix with \( O(t^2) \) rows, then with probability \( 1 - \frac{1}{1000t} \), it holds that \( \| B \|_{2,1} / 2 \leq \| BS^T \|_{2,1} \leq 2 \| B \|_{2,1} \).

Proof. For any fixed row \( i \), if \( S \) has \( O(1/\delta) \) rows, then with probability at least \( 1 - \delta \) it holds that \( \| B_i^T S \|_2 \leq (1 \pm 0.5) \| B_i \|_2 \). For a proof of this, see e.g. Theorem 2.6 of [25] which shows that with probability \( 1 - \delta \), \( S \) is a \((1 \pm 0.5)\) subspace embedding for \( B_i^T \).

Let \( T \) be the set of rows \( i \) for which \( \| B_i S_i^T \|_2 \leq (1 \pm 0.5) \| B_i \|_{2,1} \). Then:

\[
\| BS^T \|_{2,1} = \sum_{i \in T} \| B_i S_i^T \|_2 + \sum_{i \notin T} \| B_i S_i^T \|_2 \\
\leq \sum_{i \in T} \| B_i S_i^T \|_2 + \sum_{i \notin T} \frac{3}{2} \| B_i \|_2
\]

Since \( \mathbb{E} \left[ \sum_{i \in T} \| B_i S_i^T \|_2 \right] \leq \delta \| B \|_{2,1} \), by a Markov Bound:

\[
\mathbb{P} \left[ \sum_{i \in T} \| B_i S_i^T \|_{2,1} \geq (3/2 + \gamma \delta) \| B \|_{2,1} \right] \leq \frac{1}{\gamma}
\]

For a lower bound, let \( y_i = \begin{cases} 0 & \text{if } i \in T \\ \| B_i \|_2 / 2 & \text{if } i \notin T \end{cases} \) and let \( z_i = \| B_i \|_2 / 2 - y_i \). Then \( \mathbb{E} \left[ \sum_i z_i \right] \leq \frac{\delta \| B \|_{2,1}}{2} \).

Note that \( \| BS^T \|_{2,1} \geq \sum_{i \notin T} \| B_i \|_2 = \sum_i y_i \). Again by a Markov Bound:

\[
\mathbb{P} \left[ \| BS^T \|_{2,1} \leq (1 - \gamma \delta) \| B \|_{2,1} \right] \leq \mathbb{P} \left[ \sum_i y_i \leq (1 - \gamma \delta) \| B \|_{2,1} \right] \\
= \mathbb{P} \left[ \sum_i z_i \geq \gamma \delta \| B \|_{2,1} \right] \\
\leq \frac{1}{\gamma}
\]

Setting \( \gamma = 100t \) and \( \delta = 1/10000t^2 \), the claim is proved.

Calculating \( BS^T \) in a stream normally requires \( \Omega(n) \) bits of space which exceeds our desired space bound. However we can remedy this with some lemmas from [24].

By Sections 4.1 and 4.2 of [24], for any matrix \( B' \) with probability at least 0.9 it holds that:

\[
\frac{2 \| B' \|_{2,1}}{5} \leq \| \text{F-SKETCH}(B') \|_{2,1} \leq 2001 \| B' \|_{2,1}
\]
for sufficiently large $\|B'\|_{2,1}$. Consequently, if we repeat this $O(\log t)$ times and let $M_0$ be the median of these trials, then $M_0$ achieves the same guarantee but with probability $1 - 1/(100t)$.

Letting $B' = B$ would naively require time $O(\text{nnz}(B') \cdot \log t)$, which exceeds our desired time bounds. However, using Lemma 4.1, if we set $B' = BS^T$ where $S$ is a Count Sketch matrix with $O(t^2)$ rows, we get a similar constant factor guarantee. By a union bound over the event that $M_0$ is a good estimator for $\|B'\|_{2,1} = \|BS^T\|_{2,1}$ and the event that $\|BS^T\|_{2,1}$ is itself a good estimator for $\|B\|_{2,1}$, with probability $1 - 1/(50t)$:

$$\frac{1}{5} \|B\|_{2,1} \leq M_0 \leq 4002 \|B\|_{2,1}$$

Note that we can afford to store $O(\log t)$ copies of $\text{H-SKETCH}(BS^T)$ in the stream. Outputting $M = 5 \cdot M_0$ yields the claim. \qed