Universal Sketches for the Frequency Negative Moments and Other Decreasing Streaming Sums

Vladimir Braverman[*1] **and Stephen R. Chestnut**[†2]

1 **Department of Computer Science, Johns Hopkins University**
3400 N. Charles St, Baltimore MD, USA
vova@cs.jhu.edu

2 **Department of Mathematics, ETH Zürich**
Rämistrasse 101, Zürich, Switzerland
stephen.chestnut@ifor.math.ethz.ch

Abstract

Given a stream with frequencies f_d, for $d \in [n]$, we characterize the space necessary for approximating the frequency negative moments $F_p = \sum |f_d|^p$, where $p < 0$ and the sum is taken over all items $d \in [n]$ with nonzero frequency, in terms of n, ϵ, and $m = \sum |f_d|$. To accomplish this, we actually prove a much more general result. Given any nonnegative and nonincreasing function g, we characterize the space necessary for any streaming algorithm that outputs a $(1\pm\epsilon)$-approximation to $\sum g(|f_d|)$, where again the sum is over items with nonzero frequency. The storage required is expressed in the form of the solution to a relatively simple nonlinear optimization problem, and the algorithm is universal for $(1\pm\epsilon)$-approximations to any such sum where the applied function is nonnegative, nonincreasing, and has the same or smaller space complexity as g. This partially answers an open question of Nelson (IITK Workshop Kanpur, 2009).

1998 ACM Subject Classification F.1.2 Models of Computation, F.2.1 Numerical Algorithms and Problems

Keywords and phrases data streams, frequency moments, negative moments

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2015.591

1 Introduction

A *stream* is a sequence $S = ((d_1, \delta_1), (d_2, \delta_2), \ldots, (d_N, \delta_N))$, where $d_i \in [n]$ are called the items or elements in the stream and $\delta_i \in \mathbb{Z}$ is an update to the d_ith coordinate of an implicitly defined n-dimensional vector. Specifically, the *frequency* of $d \in [n]$ after $k \leq N$ updates is

$$f_d^{(k)} = \sum \{\delta_j | j \leq k, d_j = d\},$$

and the implicitly defined vector $f := f^{(N)}$ is commonly referred to as the *frequency vector* of the stream S. Let $M := \max\{n, |f_d^{(k)}| : d \in [n], 0 \leq k \leq N\}$ and $m = \sum_d |f_d|$; thus, it requires $O(\log M)$ bits to exactly determine the frequency of a single item. This model is commonly known as the *turnstile* streaming model, as opposed to the *insertion-only*

* This material is based upon work supported in part by the National Science Foundation under Grant No. 1447639, by the Google Faculty Award and by DARPA grant N660001-1-2-4014. Its contents are solely the responsibility of the authors and do not represent the official view of DARPA or the Department of Defense.

† This material is based upon work supported in part by the National Science Foundation under Grant No. 1447639 and completed while the author was at Johns Hopkins University.

18th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'15) / 19th Int'l Workshop on Randomization and Computation (RANDOM'15).
Editors: Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim; pp. 591–605

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

model which has $\delta_i = 1$, for all i, but is the same otherwise. In an insertion-only stream $N = m \geq M$.

Streams model computing scenarios where the processor has very limited access to the input. The processor reads the updates one-at-a-time, without control of their order, and is tasked to compute a function on the frequency vector. The processor can perform its computation exactly if it stores the entire vector f, but this may be undesirable or even impossible when the dimension of f is large. Thus, the goal is to complete the computation using as little storage as possible. Typically, exact computation requires storage linear in n, so we seek approximations.

Given a stream with frequencies f_d, for $d \in [n]$, we consider the problem of approximating the frequency negative moments, specifically $F_p = \sum |f_d|^p$ where $p < 0$ and the sum is taken over all items $d \in [n]$ with nonzero frequency. We characterize, up to factors of $O(\epsilon^{-1} \log^2 n \log M)$ in the turnstile model and $O(\epsilon^{-1} \log M)$ in the insertion-only model, the space necessary to produce a $(1 \pm \epsilon)$-approximation to F_p, for $p < 0$, in terms of the accuracy ϵ, the dimension n, and the L_1 length m of f.

Negative moments, also known as "inverse moments", of a probability distribution have found several applications in statistics. Early on, they were studied in application to sampling and estimation problems where the sample size is random [35, 18] as well as in life-testing problems [31]. More recently, they appear in the design of multi-center clinical trials [24] and in the running time analysis of a quantum adiabatic algorithm for 3-SAT [37, 38]. F_0/F_{-1} is the harmonic mean of the (nonzero) frequencies in the insertion-only model, and more generally, the value $(F_p/F_0)^{1/p}$ is known as the pth power mean [10]. The harmonic mean is the truest average for some types of data, for example speeds, parallel resistances, and P/E ratios [34].

To our knowledge this is the first paper to consider streaming computation of the frequency negative moments and the first to determine the precise dependence of the space complexity of streaming computations on m. In fact, in the process of characterizing the storage necessary to approximate the frequency negative moments, we actually characterize the space complexity of a much larger class of streaming sum problems. Specifically, given any nonnegative, nonincreasing function $g : \mathbb{N} \to \mathbb{R}$ we determine to within a factor of $O(\epsilon^{-1} \log^2 n \log M)$ the space necessary to approximate

$$g(f) := \sum_{d \in \mathrm{supp}(f)} g(|f_d|),$$

where $\mathrm{supp}(f) := \{d \in [n] : f_d \neq 0\}$ is the support of f. Furthermore, the sketch providing a $(1 \pm \epsilon)$-approximation for $g(f)$ is universal for a $(1 \pm \epsilon)$-approximation for any nonnegative nonincreasing function with the same or smaller space complexity as g. This partially answers a question of Nelson [33] – which families of functions admit universal sketches?

The attention on m is warranted; in fact, the complexity in question depends delicately on this parameter. If we forget about m for a moment, then a standard reduction from the communication problem Index implies that computing a $(1 \pm \frac{1}{2})$-approximation to F_p, for $p < 0$, requires $\Omega(n)$ bits of storage – nearly enough to store the entire vector f. However, the reduction requires $m = \Omega(n^{1-1/p})$, recall that $p < 0$. If $m = o(n^{1-1/p})$ then, as we show, one can often get away with $o(n)$ bits of memory.

The next two sections outline our approach to the decreasing streaming sum problem and state our main results. Section 1.3 reviews previous work on streaming sum problems. In Section 2 we show how our results solve the frequency negative moments problem. Sections 3 and 5 prove the main results. Section 4 and Section 6 describe the implementation details for the streaming setting.

1.1 Preliminaries

Let $\mathcal{F} = \{f \in \mathbb{N}^n : \sum f_d \leq m\}$ and let $\mathcal{T}$ and $\mathcal{I}$ denote the sets of turnstile streams and insertion-only streams, respectively, that have their frequency vector f satisfying $|f| \in \mathcal{F}$. The set $\mathcal{F}$ is the set of all nonnegative frequency vectors with L_1 norm at most m. Clearly, $\mathcal{F}$ is the image under coordinate-wise absolute value of the set of all frequency vectors with L_1 norm at most m. We assume $n \leq m$.

In order to address the frequency negative moments problem we will address the following more general problem. Given a nonnegative, nonincreasing function $g : \mathbb{N} \to \mathbb{R}$, how much storage is needed by a streaming algorithm that $(1 \pm \epsilon)$-approximates $g(f)$, for the frequency vector f of any stream $S \in \mathcal{T}$ or $S \in \mathcal{I}$? Equivalently, we can assume that $g(0) = 0$, g is nonnegative and nonincreasing on the interval $[1, \infty)$, and extend the domain of g to $\mathbb{Z}$ by requiring it to be symmetric, i.e., $g(-x) = g(x)$. Therefore, $g(f) = \sum_{i=1}^{n} g(f_d)$. For simplicity, we call such functions "decreasing functions".

A randomized algorithm $\mathcal{A}$ is a *turnstile streaming* $(1 \pm \epsilon)$*-approximation algorithm* for $g(f)$ if

$$P\left\{(1-\epsilon)g(f) \leq \mathcal{A}(S) \leq (1+\epsilon)g(f)\right\} \geq \frac{2}{3}$$

holds for every stream $S \in \mathcal{T}$, and insertion only algorithms are defined analogously. For brevity, we just call such algorithms "approximation algorithms" when g, ϵ, and the streaming model are clear from the context. We consider the maximum number of bits of storage used by the algorithm $\mathcal{A}$ with worst case randomness on any valid stream.

A sketch is a, typically randomized, data structure that functions as a compressed version of the stream. Let $\mathcal{G} \subseteq \mathbb{R}^{\mathbb{N}} \times (0, 1/2]$. We say that a sketch is *universal* for a class $\mathcal{G}$ if for every $(g, \epsilon) \in \mathcal{G}$ there is an algorithm that, with probability at least $2/3$, extracts from the sketch a $(1 \pm \epsilon)$-approximation to $g(f)$. The probability here is taken over the sketch as well as the extraction algorithm.

Our algorithms assume a priori knowledge of the parameters m and n, where $m = \|f\|_1$ and n is the dimension of f. In practice, one chooses n to be an upper bound on the number of distinct items in the stream. Our algorithm remains correct if one instead only knows $m \geq \|f\|_1$, however if $m \gg \|f\|_1$ the storage used by the algorithm may not be optimal. We assume that our algorithm has access to an oracle that computes g on any valid input. In particular, the final step of our algorithms is to submit a list of frequencies, i.e., a sketch, as inputs for g. We do not count the storage required to evaluate g or to store its value.

1.2 Our results

Our lower bound is proved by a reduction from the communication complexity of disjointness wherein we parameterize the reduction with the coordinates of $|f|$, the absolute value of a frequency vector. The parameterization has the effect of giving a whole collection of lower bounds, one for each frequency vector among a set of many. Specifically, if $f \in \mathcal{F}$ and $g(f) \leq \epsilon^{-1} g(1)$ then we find an $\Omega(|\operatorname{supp}(f)|)$ lower bound on on the number of bits used by any approximation algorithm. This naturally leads us to the following nonlinear optimization problem

$$\sigma(\epsilon, g, m, n) := \max\left\{|\operatorname{supp}(f)| : f \in \mathcal{F}, g(f) \leq \epsilon^{-1} g(1)\right\}, \tag{1}$$

which gives us the "best" lower bound. We will use $\sigma = \sigma(\epsilon, g, m, n)$ when ϵ, g, m, and n are clear from the context. Our main lower bound result is the following.

▶ **Theorem 1.** *Let g be a decreasing function, then any k-pass insertion-only streaming $(1 \pm \epsilon)$-approximation algorithm requires $\Omega(\sigma/k)$ bits of space.*

Before we consider approximation algorithms, let us consider a special case. Suppose there is an item d^* in the stream that satisfies $g(f_{d^*}) \geq \epsilon g(f)$. An item such as d^* is called an ϵ-heavy element. If there is an ϵ-heavy element in the stream, then $g(1) \geq g(f_{d^*}) \geq \epsilon g(f)$ which implies $|\operatorname{supp}(f)| \leq \sigma$, by the definition of σ. Of course, in this case it is possible compute $g(f)$ with $O(\sigma \log M)$ bits in one pass in the insertion-only model and with not much additional space in the turnstile model simply by storing a counter for each element of $\operatorname{supp}(f)$. Considering the $\Omega(\sigma)$ lower bound, this is nearly optimal. However, it only works when f contains an ϵ-heavy element.

Our approximation algorithm is presented next. It gives a uniform approach for handling all frequency vectors, not just those with ϵ-heavy elements.

Algorithm 1 $(1 \pm \epsilon)$-approximation algorithm for $g(f)$.

1: Compute $\sigma = \sigma(\epsilon, g, m, n)$ and let

$$q \geq \min\left\{1, \frac{9\sigma}{\epsilon|\operatorname{supp}(f)|}\right\}. \tag{2}$$

2: Sample pairwise independent random variables $X_d \sim \text{Bernoulli}(q)$, for $d \in [n]$, and let $W = \{d \in \operatorname{supp}(f) : X_d = 1\}$.
3: Compute f_d, for each $d \in W$.
4: Output $q^{-1} \sum_{d \in W} g(f_d)$.

Algorithm 1 outlines the important components of our streaming algorithm and suppresses the details needed to implement it on a stream. In particular, $|\operatorname{supp}(f)|$ is not known ahead of time, and in fact, any streaming algorithm that computes it exactly requires $\Omega(n)$ bits of storage. This and the remaining details can be handled with existing streaming technology as described in Section 6.

Algorithm 1 simply samples each element of $\operatorname{supp}(f)$ pairwise independently with probability q. The expected sample size is $q|\operatorname{supp}(f)|$, so in order to achieve optimal space we should take equality in Equation 2. The choice yields, in expectation, $q|\operatorname{supp}(f)| = O(\sigma/\epsilon)$ samples. Section 4 explains how to compute σ quickly and with small storage, and the correctness of Algorithm 1 is established by the following theorem. It is proved in Section 3.

▶ **Theorem 2.** *There is a turnstile streaming algorithm that, with probability at least $2/3$, outputs a $(1 \pm \epsilon)$-approximation to $g(f)$ and uses $O(\epsilon^{-1}\sigma \log^2(n) \log(M))$ bits of space. The algorithm can be implemented in the insertion-only model with $O(\epsilon^{-1}\sigma \log(M) + \log^2 n)$ bits of space.*

It is worth mentioning that the suppressed constants in the asymptotic bounds of Theorems 1 and 2 are independent of g, ϵ, m, and n.

The optimization problem (1) reappears in the proof of Theorem 2. The key step is the observation mentioned above. Namely, for the particular frequency vector f that is our input, if there is an item d satisfying $g(|f_d|) \geq \epsilon g(f)$ then $|\operatorname{supp}(f)| \leq \sigma$.

Let us now emphasize a particular feature of this algorithm. Previously, we commented that choosing equality in (2) is optimal in terms of the space required. However, Algorithm 1 is still correct when the inequality is strict. Notice that the sketch is just a (pairwise independent) random sample of $\operatorname{supp}(f)$ and its only dependence on g and ϵ is through the

parameter σ/ϵ. Let g' and ϵ' be another decreasing function and error parameter satisfying $\frac{\sigma(\epsilon',g',m,n)}{\epsilon'} \leq \frac{\sigma(\epsilon,g,m,n)}{\epsilon}$, then

$$q' = \min\left\{1, \frac{9\sigma'}{\epsilon'|\operatorname{supp}(f)|}\right\} \leq q = \min\left\{1, \frac{9\sigma}{\epsilon|\operatorname{supp}(f)|}\right\}.$$

In particular, this means that the sketch that produces an $(1 \pm \epsilon)$-approximation to $g(f)$ also suffices for an $(1 \pm \epsilon')$-approximation to g'. For example, if one takes $g' \geq g$, pointwise with $g'(1) = g(1)$, then $\sigma(\epsilon, g', m, n) \leq \sigma(\epsilon, g, m, n)$ so one can extract from the sketch $(1 \pm \epsilon)$-approximations to $g(f)$ and $g'(f)$, each being separately correct with probability $2/3$. Thus, the sketch is universal for any decreasing function g' and accuracy ϵ' where $\frac{\sigma(\epsilon',g',m,n)}{\epsilon'} \leq \frac{\sigma(\epsilon,g,m,n)}{\epsilon}$. In the context of the frequency negative moments, this implies that the sketch yielding a $(1 \pm \epsilon)$-approximation to F_p, for $p < 0$, is universal for $(1 \pm \epsilon)$-approximations of $F_{p'}$, for all $p \leq p' < 0$.

Computing the sketch requires a priori knowledge of σ. If one over-estimates σ the algorithm remains correct, but the storage used increases. To know σ requires knowledge of m, or at least an good upper bound on m. This is a limitation, but there are several ways to mitigate it. If one does not know m but is willing to accept a second pass through the stream, then using the algorithm of [26] one can find a $(1 \pm \frac{1}{2})$-approximation to m with $O(\log M)$ bits of storage in the first pass and approximate $g(f)$ on the second pass. A $(1 \pm \frac{1}{2})$-approximation to m is good enough to determine σ to within a constant, which is sufficient for the sketch. Alternatively, one can decide first on the space used by the algorithm and, in parallel within one pass, run the algorithm and approximate m. After reading the stream one can determine for which decreasing functions g and with what accuracy ϵ does the approximation guarantee hold.

1.3 Background

Much of the effort dedicated to understanding streaming computation, so far, has been directed at the frequency moments $F_p = \sum |f_i|^p$, for $0 < p < \infty$, as well as F_0 and F_∞, the number of distinct elements and the maximum frequency respectively. In the turnstile model, F_0 is distinguished from $L_0 = |\operatorname{supp}(f)|$, the number of elements with a nonzero frequency.

The interest in the frequency moments began with the seminal paper of Alon, Matias, and Szegedy [1], who present upper and lower bounds of $O(\epsilon^{-2}n^{1-1/p})$ and $\Omega(n^{1-5/p})$, respectively, on the space needed to find a $(1 \pm \epsilon)$-approximation to F_p, and a separate $O(\epsilon^{-2}\log m)$ space algorithm for F_2. Since then, many researchers have worked to push the upper and lower bounds closer together. We discuss only a few of the papers in this line of research, see [36] an the references therein for a more extensive history of the frequency moments problem.

To approximate F_p, Alon, Matias, and Szegedy inject randomness into the stream and then craft an estimator for F_p on the randomized stream. A similar approach, known as stable random projections, is described by Indyk [22] for F_p, when $0 < p \leq 2$ (also referred to as ℓ_p approximation). Kane, Nelson, and Woodruff [26] show that Indyk's approach, with a more careful derandomization, is optimal. Using the method of stable random projections, Li [29] defined the so-called *harmonic mean estimator* for F_p, when $0 < p < 2$, which improves upon the sample complexity of previous methods. We stress that this is not an estimator for the harmonic mean of the frequencies in a data stream, rather it is an estimator for F_p that takes the form of the harmonic mean of a collection of values.

For $p > 2$, the AMS approach was improved upon [15, 16] until a major shift in the design of streaming algorithms began with the algorithm of Indyk and Woodruff [23] that solves the

frequency moments problem with, nearly optimal, $n^{1-2/p}(\frac{1}{\epsilon}\log n)^{O(1)}$ bits. Their algorithm introduced a recursive subsampling technique that was subsequently used to further reduce space complexity [5, 7], which now stands at $O(\epsilon^{-2}n^{1-2/p}\log n)$ in the turnstile model [17] with small ϵ and $O(n^{1-2/p})$ in the insertion-only model with $\epsilon = \Omega(1)$ [6].

Recently, there has been a return to interest in AMS-type algorithms motivated by the difficulty of analyzing algorithms that use recursive subsampling. "Precision Sampling" of Andoni, Krauthgamer, and Onak [2] is one such algorithm that accomplishes nearly optimal space complexity without recursive subsampling. Along these lines, it turns out that one can approximate $g(f)$ by sampling elements $d \in [n]$ with probability roughly $q_d \approx g(f_d)/\epsilon^2 g(f)$, or larger, and then averaging and scaling appropriately, see Proposition 4. Algorithm 1 takes this approach, and also fits in the category of AMS-type algorithms. However, it is far from clear how to accomplish this sampling optimally in the streaming model for a completely arbitrary function g.

A similar sampling problem has been considered before. Monemizadeh and Woodruff [32] formalized the problem of sampling with probability $q_d = g(f_d)/g(f)$ and then go on to focus on L_p sampling, specifically $g(x) = |x|^p$, for $0 \le p \le 2$. In follow-up work, Jowhari, Săglam, and Tardos offer L_p sampling algorithms with better space complexity [25].

As far as the frequency moments lower bounds go, there is a long line of research following AMS [4, 13, 19, 3] that has led to a lower bound matching the best known turnstile algorithm of Ganguly [17] to within a constant [30], at least for some settings of m and ϵ. The insertion-only algorithm of Braverman et al. [6] matches the earlier lower bound of Chakrabarti, Khot, and Sun [13].

For a general function g not much is known about the space-complexity of approximating $g(f)$. Most research has focused on specific functions. Chakrabarti, Do Ba, and Muthukrishnan [12] and Chakrabarti, Cormode, and Muthukrishnan [11] sketch the Shannon Entropy. Harvey, Nelson, and Onak [21] approximate Renyi $\log(\|f\|_\alpha^\alpha)/(1-\alpha)$, Tsallis $(1-\|x\|_\alpha^\alpha)/(\alpha-1)$, and Shannon entropies. Braverman, Ostrovsky, and Roytman [8, 9] characterized nonnegative, nondecreasing functions that have polylogarithmic-space approximation algorithms and they present a universal sketching algorithm for this class of functions. Their algorithm is based on the subsampling technique. Guha, Indyk, McGregor [20] study the problem of sketching common information divergences between the streams, i.e., statistical distances between the probability distributions with p.m.f.s $e/\|e\|_1$ and $f/\|f\|_1$.

2 The frequency negative moments

Before proving Theorems 1 and 2, let us deploy them to determine the streaming space complexity of the frequency negative moments. It will nicely illustrate the trade-off between the length of the stream and the space complexity of the approximation.

The first step is to calculate $\sigma(\epsilon, g, m, n)$, where $g(x) = |x|^p$, for $x \neq 0$ and $p < 0$, and $g(0) = 0$. There is a maximizer of (1) with L_1 length m because g is decreasing. The convexity of g on $[0, \infty)$ implies that $\sigma \le \max\{s \in \mathbb{R} : s(m/s)^p \le \epsilon^{-1}\}$, and σ is at least the minimum of n and $\max\{s \in \mathbb{N} : s(m/s)^p \le \epsilon^{-1}\}$ by definition. Thus, we can take $\sigma = \min\left\{n, \theta\left(\epsilon^{\frac{-1}{1-p}} m^{\frac{-p}{1-p}}\right)\right\}$. This gives us the following corollary to Theorems 1 and 2.

▶ **Corollary 3.** *For any $p < 0$ and $\epsilon > 0$, any algorithm that outputs a $(1 \pm \epsilon)$-approximation to F_p requires $\Omega(\min\{n, \epsilon^{\frac{-1}{1-p}} m^{\frac{-p}{1-p}}\})$ bits of space. Such an approximation can be found with $O(\epsilon^{-\frac{2-p}{1-p}} m^{\frac{-p}{1-p}} \log^2 n \log M)$ bits in a turnstile stream and $O(\epsilon^{-\frac{2-p}{1-p}} m^{\frac{-p}{1-p}} \log M)$ bits in an insertion-only stream.*

For example, taking $p = -1$, which is what we need to estimate the harmonic mean, we find that the complexity is approximately $\frac{\sigma}{\epsilon} = \min\{n, \theta(\epsilon^{-3/2}m^{1/2})\}$. This is also the space complexity of approximating the harmonic mean of the nonzero frequencies. It is apparent from the formula that the relationship between m and n is important for the complexity.

3 Correctness of the algorithm

This section presents the proof that our approximation algorithm is correct. Algorithm 1 describes the basic procedure, and Section 6 describes how it can be implemented in the streaming setting. The correctness relies on our ability to perform the sampling and the following simple proposition.

▶ **Proposition 4.** *Let g be a nonnegative function and let $X_d \sim$ Bernoulli(p_d) be pairwise independent random variables with $p_d \geq \min\left\{1, \frac{9g(f_d)}{\epsilon^2 g(f)}\right\}$, for all $d \in [n]$. Let $\hat{G} = \sum_{d=1}^{n} p_d^{-1} X_d g(f_d)$, then $P(|\hat{G} - g(f)| \leq \epsilon g(f)) \geq \frac{8}{9}$.*

Proof. We have $E\hat{G} = g(f)$ and $Var(\hat{G}) \leq \sum_d p_d^{-1} g(f_d)^2 = \frac{1}{9}(\epsilon g(f))^2$, by pairwise independence. The proposition now follows from Chebyshev's inequality. ◀

The algorithm samples each element of $\operatorname{supp}(f)$ with probability about $\sigma/\epsilon \operatorname{supp}(f)$. In order to show that this sampling probability is large enough for Proposition 4 we will need one lemma. It gives us some control on $\sigma(\epsilon, g, m, n)$ as ϵ varies.

▶ **Lemma 5.** *If $\alpha < \epsilon$, then $\epsilon(1 + \sigma(\epsilon, g, m, n)) \geq \alpha\sigma(\alpha, g, m, n)$.*

Proof. Let $\sigma_\epsilon = \sigma(\epsilon, g, m, n)$ and define σ_α similarly. Let $f \in \mathcal{F}$ such that $\sigma_\alpha = |\operatorname{supp}(f)|$ and $g(f) \leq \alpha^{-1} g(1)$, without loss of generality the coordinates are ordered such that $f_1 \geq f_2 \geq \cdots \geq f_{\sigma_\alpha} > 0$. Let $s' = \frac{\alpha}{\epsilon}\sigma_\alpha$, and let f' be the vector that takes the first $\lfloor s' \rfloor$ coordinates from f and is 0 thereafter. The choice is made so that $f' \in \mathcal{F}$ and

$$g(f') \leq \frac{\alpha}{\epsilon} g(f) \leq \epsilon^{-1} g(1).$$

Then, by definition of σ_ϵ, we have

$$\sigma_\epsilon \geq |\operatorname{supp}(f')| = \left\lfloor \frac{\alpha}{\epsilon}\sigma_\alpha \right\rfloor \geq \frac{\alpha}{\epsilon}\sigma_\alpha - 1.$$ ◀

For brevity, we only state here the correctness of the streaming model sampling algorithm, which uses standard techniques. The details of the algorithm are given in the Section 6.

▶ **Lemma 6.** *Given $s \leq n$, there is an algorithm using $O(s \log^2 n \log M)$ bits of space in the turnstile model and $O(s \log M + \log^2 n)$ bits in the insertion-only model that samples each item of $\operatorname{supp}(f)$ pairwise-independently with probability at least $\min\{1, s/|\operatorname{supp}(f)|\}$ and, with probability at least $7/9$, correctly reports the frequency of every sampled item and the sampling probability.*

Finally, we prove the correctness of our approximation algorithm. Here is where we will again use the optimality of σ in its definition (1). In regards to the lower bound of Theorem 1, this upper bound leaves gaps of $O(\epsilon^{-1} \log^2 n \log M)$ and $O(\epsilon^{-1} \log M)$ in the turnstile and insertion-only models, respectively.

Proof of Theorem 2. We use the algorithm of Lemma 6 to sample with probability at least $q = \min\{1, 9(\sigma+1)/\epsilon|\operatorname{supp}(f)|\}$. Let us first assume that $q \geq \min\{1, 9g(f_d)/\epsilon^2 g(f)\}$, for all d, so that the hypothesis for Proposition 4 is satisfied. The algorithm creates samples W_i, for $i = 0, 1, \ldots, O(\log n)$, where each item is sampled in W_i with probability $q_i = 2^{-i}$. For each i such that $q_i \geq q$, Proposition 4 guarantees that $\hat{G}_i = q_i^{-1} \sum_{d \in W_i} g(f_d)$ is a $(1 \pm \epsilon)$-approximation with probability at least $8/9$. With probability at least $7/9$, the algorithm returns one of these samples correctly, and then the approximation guarantee holds. Thus, the approximation guarantee holds with probability at least $2/3$.

It remains to show that $q \geq \min\{1, 9g(f_d)/\epsilon g(f)\}$, for all $d \in [n]$. Let $\alpha = g(1)/g(f)$ then define $\sigma_\epsilon = \sigma(\epsilon, g, m, n)$ and $\sigma_\alpha = \sigma(\alpha, g, m, n)$. By definition $|\operatorname{supp}(f)| \leq \sigma_\alpha$, thus if $\alpha \geq \epsilon$ then $|\operatorname{supp}(f)| \leq \sigma_\alpha \leq \sigma_\epsilon$, so the sampling probability is 1 and the claim holds.

Suppose that $\alpha < \epsilon$. For all $d \in [n]$, we have

$$\frac{g(f_d)}{g(f)} \leq \frac{g(1)}{g(f)} = \alpha \leq \frac{\epsilon(1+\sigma_\epsilon)}{\sigma_\alpha} \leq \frac{\epsilon(1+\sigma_\epsilon)}{|\operatorname{supp}(f)|},$$

where the second inequality comes from Lemma 5 and the third from the definition of σ_α as a maximum. In particular, this implies that

$$\frac{9\epsilon^{-1}(\sigma+1)}{|\operatorname{supp}(f)|} \geq \frac{9g(f_d)}{\epsilon^2 g(f)},$$

which completes the proof. ◀

4 Computing σ

The value σ is a parameter that is needed for Algorithm 1. That means we need a way to compute it for any decreasing function. As we mentioned before, the only penalty for overestimating σ is inflation of the storage used by the algorithm so to over-estimate σ by a constant factor is acceptable. This section shows that one can find σ' such that $\sigma \leq \sigma' \leq 4\sigma$ quickly, with $O(\log n)$ bits of storage, and by evaluating g at just $O(\log m)$ points.

Because g is decreasing, the maximum of (1) will be achieved by a vector f of length m. This is regardless of whether $m \leq n$ or $m > n$. Lemma 7 says that we might as well take all of the other frequencies to be equal, so we can find a near maximizer by enumerating the single value of those frequencies. Specifically,

$$s(y) = \min\left\{\frac{m}{y}, \frac{g(1)}{\epsilon g(y)}\right\}$$

is the maximum bound we can achieve using y as the single frequency. The value of σ is at most twice $\max\{s(y) : (m/n) \leq y \leq m\}$, by Lemma 7.

But we do not need to check every $y = 1, 2, \ldots, m$ to get a pretty good maximizer. It suffices to check only values where y is a power of two. Indeed, suppose that y^* maximizes $s(y)$ and let $y^* \leq y' \leq 2y^*$. We will show that $s(y') \geq s(y^*)/2$, and since there is a power of two between y^* and $2y^*$ this implies that its s value is at least $s(y^*)/2 \geq \sigma/4$.

Since y^* is a maximizer we have $s(y') \leq s(y^*)$, and because $y' \geq y^*$ and g is decreasing we have $g(y') \leq g(y^*)$. This gives us

$$\frac{g(1)}{\epsilon g(y')} \geq \frac{g(1)}{\epsilon g(y^*)} \geq s(y^*).$$

We also have

$$\frac{m}{y'} \geq \frac{m}{2y^*} \geq \frac{1}{2}s(y^*).$$

Combining these two we have $s(y') \geq s(y^*)/2$.

Thus, one can get by with enumerating at most $\lg m$ values to approximate the value of the parameter σ. Take the largest of the $\lg m$ values tried and quadruple it to get the approximation to σ.

5 Lower bounds for decreasing streaming sums

It bears repeating that if $g(x)$ decreases to 0 as $x \to \infty$ then one can always prove an $\Omega(n)$ lower bound on the space complexity of approximating $g(f)$. However, the stream needed for the reduction may be very long (as a function of n). Given only the streams in $\mathcal{T}$ or $\mathcal{I}$, those with L_1-length m or less, a weaker lower bound may be the best available. The present section proves this "best" lower bound, establishing Theorem 1.

The proof uses a reduction from the communication complexity of disjointness, see the book of Kushilevitz and Nisan [28] for background on communication complexity. The proof strategy is to parameterize the lower bound reduction in terms of the frequencies f. Optimizing the parameterized bound over $f \in \mathcal{F}$ gives the best possible bound from this reduction.

The proof of Theorem 1 is broken up with a two lemmas. The first lemma is used in the reduction from $\textsc{Disj}(s)$, the s-element disjointness communication problem. It will show up again later when we discuss a fast scheme for computing σ for general functions.

▶ **Lemma 7.** *Let $y_i \in \mathbb{R}_{\geq 0}$, for $i \in [s]$, and let $v : \mathbb{R} \to \mathbb{R}_{\geq 0}$. If $\sum y_i \leq Y$ and $\sum v(y_i) \leq V$, then there exists i such that $\frac{s}{2} y_i \leq Y$ and $\frac{s}{2} v(y_i) \leq V$.*

Proof. Without loss of generality $y_1 \leq y_2 \leq \cdots \leq y_s$. Let i_j, $j \in [\sigma]$, order the sequence such that $v(y_{i_1}) \leq v(y_{i_2}) \leq \cdots \leq v(y_{i_s})$ and let $I = \{i_j | j \leq \lfloor s/2 \rfloor + 1\}$. By the Pigeon Hole Principle, there exists $i \in I$ such that $i \leq \lfloor s/2 \rfloor + 1$. Thus $\frac{s}{2} y_i \leq \sum_{j=\lfloor s/2 \rfloor + 1}^{s} y_{i_j} \leq Y$ and $\frac{s}{2} v(y_i) \leq \sum_{j=\lfloor s/2 \rfloor + 1}^{s} v(y_j) \leq V$. ◀

▶ **Lemma 8.** *Let g be decreasing and $\epsilon > 0$. If $f = (y, y, \ldots, y, 0, \ldots, 0) \in \mathcal{F}$ and $g(f) \leq \epsilon^{-1} g(1)$, then any k-pass $(1 \pm \epsilon)$-approximation algorithm requires $\Omega(|\operatorname{supp}(f)|/k)$ bits of storage.*

Proof. Let $s = \lfloor |\operatorname{supp}(f)|/2 \rfloor$ and let $\mathcal{A}$ be an $(1 \pm \epsilon)$-approximation algorithm. The reduction is from $\textsc{Disj}(s, 2)$ where Alice receives $A \subseteq [s]$ and Bob receives $B \subseteq [s]$. Their goal is to jointly determine whether $A \cap B = \emptyset$ or not. Our protocol will answer the equivalent question: is $B \subseteq A^c$ or not? Alice and Bob will answer the question by jointly creating a notional stream, running $\mathcal{A}$ on it, and thresholding the outcome.

For each $d \in A^c$, Alice puts $(d, 1)$ in the stream y times. She then runs $\mathcal{A}$ on her portion of the stream and sends the contents its memory to Bob. For each $d \in B$, Bob adds $(d, 1)$ to the stream. Bob runs $\mathcal{A}$ on his portion of the stream and sends the memory back to Alice. She recreates her portion of the stream, advances $\mathcal{A}$, sends the memory to Bob, etc., until each player has acted k times. In addition to the algorithm's memory, on each pass Alice sends at most $\lceil k^{-1} \lg |A| \rceil$ binary digits of $|A|$ so that Bob knows $|A|$ at the end of the protocol.

The stream is a member of $\mathcal{I}$ by construction; let f' be its frequency vector. At the end, Bob finishes computing $\mathcal{A}(f')$. All of the frequencies are y, $y+1$, or 1. If

$$\mathcal{A}(f') \leq (1+\epsilon)[|B| g(y+1) + (s - |A| - |B|) g(y)],$$

then Bob declares $B \subseteq A^c$ and otherwise $B \not\subseteq A^c$.

The exact value of $g(f')$ is

$$|A \cap B|g(1) + |B \setminus A|g(y+1) + (s - |A| - |B| + |A \cap B|)g(y).$$

If $B \subseteq A^c$ this value is

$$V_0 := |B|g(y+1) + (s - |A| - |B|)g(y),$$

and otherwise, because g is decreasing, it is at least

$$V_1 := g(1) + (|B| - 1)g(y+1) + (s - |B| - |A| + 1)g(y).$$

We find

$$V_1 - V_0 \geq g(1) \geq \epsilon g(f) \geq 2\epsilon s g(y) \geq 2\epsilon V_0$$

Hence, if $\mathcal{A}(f')$ is a $(1 \pm \epsilon)$-approximation to $g(f')$, then Bob's decision is correct. The protocol with solves $\textsc{Disj}(s)$ which requires, in the worst case, $\Omega(s)$ bits of communication including $O(k^{-1} \lg s)$ bits to send $|A|$ and $\Omega(s) = \Omega(|\operatorname{supp}(f)|)$ bits for $(2k-1)$ transmissions of the memory of $\mathcal{A}$. Thus, in the worst case, at least one transmission has size $\Omega(|\operatorname{supp}(f)|/k)$. ◀

Proof of Theorem 1. Let $f \in \mathcal{F}$ be a maximizer of (1) and apply Lemma 7 to the positive elements of f. From this we find that there exists y such that $ys' \leq \|f\|_1$ and $g(1) \geq \epsilon s' g(y)$, for $s' = \sigma/2$. Therefore, $f' = (y, y, \ldots, y, 0, \ldots, 0) \in \mathcal{F}$ with $\lfloor s' \rfloor$ coordinates equal to y. Applying Lemma 8 to f' implies the desired bound. ◀

With Lemma 7 in mind, one may ask: why not restrict the maximization problem in (1), the definition of σ, to streams that have all frequencies equal and still get the same order lower bound? This is valid alternative definition. In fact, doing so does appreciably affect the effort needed to compute σ, it is one of the main steps used by our algorithm to approximate σ in Section 4. However, it makes reasoning about σ a bit messier. For example, in Section 1.2 we comment that if the frequency vector f contains an ϵ-heavy element then $|\operatorname{supp}(f)| \leq \sigma$. This comes directly from the fact that $\{f' \in \mathcal{F} : g(f') \leq \epsilon^{-1} g(1)\}$ is the feasible set for (1). If we restrict the feasible set, then we cannot so directly draw the conclusion. Rather, we must compare $g(f)$ to points in the restricted feasible set by again invoking Lemma 7.

6 Details of the algorithm

The streaming implementation in the turnstile model will make use of the Count Sketch algorithm of Charikar, Chen, and Farach-Colton [14]. It is easy to adapt their algorithm for the purpose of finding $\operatorname{supp}(f)$. This gives us the following theorem.

▶ **Theorem 9** (Charikar, Chen, Farach-Colton [14]). *Suppose that S is a stream with at most s items of nonzero frequency. There is a turnstile streaming algorithm* $\textsc{Count Sketch}(S, s, \delta)$ *using $O(s \log \frac{n}{\delta} \log M)$ bits that, with probability at least $1 - \delta$, returns all of the nonzero frequencies in S.*

The sampling algorithm follows. Since we do not know $|\operatorname{supp}(f)|$ at the start of the stream, we guess $O(\log n)$ possible values for it and try each one. After parsing the entire stream, we can use an estimate of $L_0 = |\operatorname{supp}(f)|$ in order to determine which guess is correct. We use $\widehat{L_0}(S^{(i)}, \epsilon, \delta)$ to denote the output of an algorithm that produces a $(1 \pm \frac{1}{8})$-approximation to L_0 with probability at least $1 - \delta$, for example the algorithm of Kane, Nelson, and Woodruff [27]. After the formal presentation of the algorithm we prove its correctness and the claimed storage bounds.

Algorithm 2 Pairwise independent sampling with probability $q \geq s/|\operatorname{supp}(f)|$.

procedure SKETCH(Stream S, $s > 0$)
  $\ell \leftarrow \lceil \lg(n/s) \rceil$
  for $0 \leq i \leq \ell$ **do**
    Sample pairwise independent r.v.s $X_{i,d} \sim \text{Bernoulli}(2^{-i})$, for $d \in [n]$
    Let $S^{(i)}$ be the substream of S with items $\{d : X_{i,d} = 1\}$
    $U^{(i)} \leftarrow$ COUNT SKETCH$(S^{(i)}, 96s, 1/48)$
  end for
  $L \leftarrow \widehat{L_0}(S^{(i)}, 1/8, 1/18)$
  $i^* \leftarrow \max\left\{0, \left\lfloor \lg \frac{L}{18s} \right\rfloor\right\}$
  return $U^{(i^*)}, q = 2^{-i^*}$
end procedure

▶ **Theorem 10.** *With probability at least* $7/9$*, Algorithm 2 samples each item in* $\operatorname{supp}(f)$ *with probability* $q \geq s/|\operatorname{supp}(f)|$ *and the resulting sample of size* $O(s)$*. The algorithm can be implemented with* $O(s \log(M) \log^2(n))$ *bits of space.*

Proof. Let

$$k = \left\lfloor \lg \frac{|\operatorname{supp}(f)|}{16s} \right\rfloor.$$

If $i^* \in \{k-1, k\}$, the streams $S^{(k-1)}$ and $S^{(k)}$ both have small enough support, and the two outputs $U^{(k-1)}$ and $U^{(k)}$ of COUNT SKETCH are correct, then the output is correct. We show that the intersection of these events occurs with probability at least $7/9$.

First, with probability at least $17/18$ L is $(1 \pm 1/8)$-approximation to $|\operatorname{supp}(S)|$. A direct calculations then shows that $i^* \in \{k-1, k\}$.

The following two inequalities arise from the definition of k

$$\frac{64s}{|\operatorname{supp}(f)|} \geq 2^{-(k-1)} \geq 2^{-k} \geq \frac{16s}{|\operatorname{supp}(f)|}. \tag{3}$$

The first inequality implies that the expected support sizes of $S^{(k-1)}$ and $S^{(k)}$ and their variances are all at most $64s$. Chebyshev's inequality implies that each of these values exceeds $96s$ with probability no larger than $64/32^2 = 1/16$. So long as they don't, both streams are valid inputs to COUNT SKETCH. The last inequality of (3), with Theorem 9, implies that the sampling probability is correct.

Putting it together, the total probability of failure is no larger than

$$\frac{1}{18} + \frac{2}{16} + \frac{2}{48} = \frac{2}{9}, \tag{4}$$

where the terms come from the $|\operatorname{supp}(f)|$ estimation, the support sizes of substreams $k-1$ and k, and COUNT SKETCH.

The space bound for turnstile streams follows from Theorem 9. Approximating the support size of the stream with $\widehat{L_0}$ can accomplished with $O(\log n \log\log nM)$ bits using the algorithm of Kane, Nelson, and Woodruff [27]. ◀

Because of deletions in the turnstile model, we need to wait until the end of the stream to rule out any of the guesses of $|\operatorname{supp}(f)|$. This is not the case in the insertion only model. As soon as the number of nonzero counters grows too large we can infer that the sampling

probability is too large and discard the sample. It turns out that doing so is enough to cut a $\log n$ factor from the space complexity of Algorithm 2. A further $\log n$ factor can be saved because COUNT SKETCH is not needed in the insertion-only model.

▶ **Corollary 11.** *Algorithm 2 can be implemented with $O(s \log M + \log^2 n)$ bits of storage for insertion-only streams.*

Proof. Define ℓ independent collections of pairwise independent random variables $Y_{i,d} \sim$ Bernoulli(1/2), for $d \in [n]$, and choose the random variables in the algorithm to be

$$X_{i,d} = \prod_{j=1}^{i} Y_{i,d}.$$

One easily checks that each collection $\{X_{i,d}\}_{d\in[n]}$ is pairwise independent and that $P(X_{i,d} = 1) = 2^{-i}$, for all i and d. Storing the seeds for the collection $Y_{i,d}$ requires $O(\log^2 n)$ bits.

We can first save a $\log n$ factor by bypassing COUNT SKETCH and instead simply storing counters for each element that appears in each of the ℓ substreams. The counters should be stored in a hash table or other data structure with no space overhead and a small look-up time. Let us label the maximum number of counters to be stored for each substream as t. We choose $t = \max\{96s, \ell\}$. If the set of counters for each substream is discarded as soon as the number of nonzero counters exceeds the limit of $O(t)$, then the total storage cannot grow to large.

According to Lemma 12, the algorithm uses more than $12t$ counters with probability at most $1/6\ell$, at any given instant.

For each $0 \le i \le \ell$ let $T^{(i)}$ be the longest prefix of stream $S^{(i)}$ such that $|\operatorname{supp}(T^{(i)})| \le s$ and let $k^{(i)}$ denote the number of updates in $T^{(i)}$. Now, notice that the number of counters stored locally maximum at each $k^{(i)}$ and increasing for updates between $k^{(i)}$ and $k^{(i+1)}$. Thus, it is sufficient to bound the storage used by the algorithm at these points.

By a union bound, the probability that the number of counters used by the algorithm at any point $k^{(1)}, k^{(2)}, \ldots, k^{(\ell)}$ is more than $12t$ is at most $\ell \cdot 1/6\ell = 1/6$. Finally, adapting the final union bound of (4) in the previous proof we have that the probability of error is at most $(1/18) + (1/6) = 2/9$. ◀

▶ **Lemma 12.** *Let $v \in \{0,1\}^n$, define ℓ independent collections of pairwise independent random variables $Y_{i,d} \sim$ Bernoulli(1/2), for $s \in [n]$ and $i \in [\ell]$, and set*

$$X_{i,d} = \prod_{j=1}^{i} Y_{i,d}.$$

For a given $s \in \mathbb{N}$, set $k = 0$ if $\sum_d v_d \le s$ or $k = \max\{i : v^T X_i > s\}$ otherwise, where $X_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,n}) \in \{0,1\}^n$. Then

$$P\Big(\sum_{i=k+1}^{\ell} v^T X_i > 4s\Big) \le \frac{1}{2s}.$$

Proof. The sum is clearly monotonically increasing, so without loss of generality assume $\ell = \infty$. Notice that if $k > 0$, the sum is unchanged (i.e., it remains the same random variable) upon replacing v with the coordinate-wise product of v and X_k. Thus we may also assume that $k = 0$, i.e., $|\operatorname{supp}(v)| \le s$.

For each $d \in \text{supp}(v)$, let $Z_d = \sup\{i : X_{i,d} = 1\}$. Notice that $\{Z_d\}_{d\in\text{supp}(v)}$ is a pairwise independent collection of Geometric(1/2) random variables and let $Z = \sum_{d\in\text{supp}(v)} Z_d$. We have that

$$Z = \sum_{i=0}^{\infty} v^T X_i,$$

because $X_{i,d} = 0$ implies $X_{j,d} = 0$ for all $j > i$.

Pairwise independence implies $EZ = Var(Z) = 2|\text{supp}(v)| \leq 2s$, and by Chebyshev's inequality

$$P(|Z - 2s| > 2s) \leq \frac{Var(Z)}{4s^2} \leq \frac{1}{2s}.$$ ◀

7 Conclusion

It may be possible to apply our methods in order to parameterize according to lengths other than L_1, for example L_∞, or in terms of more general constraints on the set of feasible streams. One challenge with such an adaptation is to ensure that the reduction "preserves" these constraints. For example, if one replaces the L_1 length constraint defining $\mathcal{F}$ with the constraint that the L_∞ length, i.e., maximum frequency, is at most M, then she must take care to ensure that the reduction from Disj never produces a stream with maximum frequency larger than M. It is immediate from the structure of the reduction that it preserves upper bounds on the L_1 size of the frequencies, but our reduction may not preserve an upper bound on the maximum frequency.

Recall that we require an upper bound on the L_1 length of the frequency vector at the start of the algorithm (in order to compute σ). Here the L_1 length has an additional practical advantage over other lengths because it can be computed exactly for insertion-only and strict-turnstile streams by a one pass algorithm with at most $O(\log M)$ bits of memory, and a $(1 \pm \epsilon)$-approximation requires only $O(\epsilon^{-2} \log M)$ bits in the turnstile model [26]. Thus, one can determine after the fact whether the supposed upper bound actually holds.

Acknowledgements. The authors would like to thank several anonymous reviewers for their careful reading of an earlier draft of this paper and their helpful comments.

References

1 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *Symposium on the Theory of Computing*, pages 20–29, 1996.

2 Alexandr Andoni, Robert Krauthgamer, and Krzysztof Onak. Streaming algorithms via precision sampling. In *IEEE Foundations of Computer Science*, pages 363–372, 2011.

3 Alexandr Andoni, Huy L. Nguyễn, Yury Polyanskiy, and Yihong Wu. Tight lower bound for linear sketches of moments. In *Automata, languages, and programming*, volume 7965 of *Lec. Notes in Comput. Sci.*, pages 25–32. Springer, Heidelberg, 2013.

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

5 Lakshminath Bhuvanagiri, Sumit Ganguly, Deepanjan Kesh, and Chandan Saha. Simpler algorithm for estimating frequency moments of data streams. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 708–713, 2006.

6 Vladimir Braverman, Jonathan Katzman, Charles Seidell, and Gregory Vorsanger. Approximating large frequency moments with $O(n^{1-2/k})$ bits. *arXiv preprint arXiv:1401.1763*, 2014.

7 Vladimir Braverman and Rafail Ostrovsky. Recursive sketching for frequency moments. *arXiv preprint arXiv:1011.2571*, 2010.

8 Vladimir Braverman and Rafail Ostrovsky. Zero-one frequency laws. In *ACM Symposium on the Theory of Computing*, pages 281–290, 2010.

9 Vladimir Braverman, Rafail Ostrovsky, and Alan Roytman. Universal streaming. *arXiv preprint arXiv:1408.2604*, 2014.

10 Peter S. Bullen. *Handbook of means and their inequalities*. Springer Science & Business Media, 2003.

11 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. A near-optimal algorithm for computing the entropy of a stream. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 328–335. Society for Industrial and Applied Mathematics, 2007.

12 Amit Chakrabarti, Khanh Do Ba, and S. Muthukrishnan. Estimating entropy and entropy norm on data streams. *Internet Math.*, 3(1):63–78, 2006.

13 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *IEEE Conference on Computational Complexity*, pages 107–117, 2003.

14 Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *Automata, languages and programming*, volume 2380 of *Lec. Notes in Comput. Sci.*, pages 693–703. Springer, Berlin, 2002.

15 Don Coppersmith and Ravi Kumar. An improved data stream algorithm for frequency moments. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 151–156, 2004.

16 Sumit Ganguly. Estimating frequency moments of data streams using random linear combinations. In *Approximation, Randomization, and Combinatorial Optimization*, pages 369–380. Springer, 2004.

17 Sumit Ganguly. Polynomial estimators for high frequency moments. *arXiv preprint arXiv:1104.4552*, 2011.

18 Edwin L Grab and I Richard Savage. Tables of the expected value of 1/X for positive bernoulli and poisson variables. *J. Am. Stat. Assoc.*, 49(265):169–177, 1954.

19 André Gronemeier. Asymptotically optimal lower bounds on the NIH-multi-party information complexity of the AND-function and disjointness. In *Symposium on Theoretical Aspects of Computer Science*, 2009.

20 Sudipto Guha, Piotr Indyk, and Andrew McGregor. Sketching information divergences. In *Learning theory*, volume 4539 of *Lec. Notes in Comput. Sci.*, pages 424–438. Springer, Berlin, 2007.

21 Nicholas JA Harvey, Jelani Nelson, and Krzysztof Onak. Sketching and streaming entropy via approximation theory. In *IEEE Symposium on Foundations of Computer Science*, pages 489–498, 2008.

22 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. of the ACM*, 53(3):307–323, 2006.

23 Piotr Indyk and David Woodruff. Optimal approximations of the frequency moments of data streams. In *ACM Symposium on the Theory of Computing*, pages 202–208, 2005.

24 C.M̃atthew Jones and Anatoly A. Zhigljavsky. Approximating the negative moments of the Poisson distribution. *Statistics & Probability letters*, 66(2):171–181, 2004.

25 Hossein Jowhari, Mert Sağlam, and Gábor Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *ACM Symposium on Principles of Database Systems*, pages 49–58, 2011.

26 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1161–1178, 2010.

27 Daniel M Kane, Jelani Nelson, and David P Woodruff. An optimal algorithm for the distinct elements problem. In *ACM Symposium on Principles of Database Systems*, pages 41–52, 2010.

28 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, Cambridge, 1997.

29 Ping Li. Estimators and tail bounds for dimension reduction in l_α $(0 < \alpha \leq 2)$ using stable random projections. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 10–19, 2008.

30 Yi Li and David P Woodruff. A tight lower bound for high frequency moment estimation with small error. In *Approximation, Randomization, and Combinatorial Optimization*, pages 623–638. Springer, 2013.

31 W. Mendenhall and H. Lehman, E.˙ An approximation to the negative moments of the positive binomial useful in life testing. *Technometrics*, 2(2):227–242, 1960.

32 Morteza Monemizadeh and David P Woodruff. 1-pass relative-error l_p-sampling with applications. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1143–1160, 2010.

33 Jelani Nelson. List of open problems in sublinear algorithms: Problem 30. `http://sublinear.info/30`.

34 Robert F Reilly and Robert P Schweihs. *The handbook of business valuation and intellectual property analysis*. McGraw Hill, 2004.

35 Frederick F Stephan. The expected value and variance of the reciprocal and other negative powers of a positive Bernoullian variate. *Ann. Math. Stat.*, 16(1):50–61, 1945.

36 David P Woodruff. Data streams and applications in computer science. *Bulletin of EATCS*, 3(114), 2014.

37 Marko Znidaric. Asymptotic expansion for inverse moments of binomial and Poisson distributions. *arXiv preprint math/0511226*, 2005.

38 Marko Žnidarič and Martin Horvat. Exponential complexity of an adiabatic algorithm for an NP-complete problem. *Phys. Rev. A*, 73(2), 2006.