A New Information Complexity Measure for Multi-pass Streaming with Applications*

Mark Braverman
Princeton University
Princeton, NJ, USA
mbraverm@cs.princeton.edu

Sumegha Garg
Rutgers University
Piscataway, NJ, USA
sumegha.garg@rutgers.edu

Qian Li
Shenzhen lnternational Center For Industrial and Applied Mathematics, Shenzhen Research Institute of Big Data
Shenzhen, China
liqian.ict@gmail.com

Shuo Wang
Shanghai Jiao Tong University
Shanghai, China
s.wangg2002@gmail.com

David P. Woodruff
Carnegie Mellon University
Pittsburgh, PA, USA
dwoodruf@andrew.cmu.edu

Jiapeng Zhang
University of Southern California
Los Angeles, California, USA
jiapengz@usc.edu

ABSTRACT

We introduce a new notion of information complexity for multi-pass streaming problems and use it to resolve several important questions in data streams:

(1) In the coin problem, one sees a stream of n i.i.d. uniformly random bits and one would like to compute the majority with constant advantage. We show that any constant-pass algorithm must use $\Omega(\log n)$ bits of memory, significantly extending an earlier $\Omega(\log n)$ bit lower bound for single-pass algorithms of Braverman-Garg-Woodruff (FOCS, 2020). This also gives the first $\Omega(\log n)$ bit lower bound for the problem of approximating a counter up to a constant factor in worst-case turnstile streams for more than one pass.

(2) In the needle problem, one either sees a stream of n i.i.d. uniform samples from a domain $[t]$, or there is a randomly chosen "needle" $\alpha \in [t]$ for which each item independently is chosen to equal α with probability p, and is otherwise uniformly random in $[t]$. The problem of distinguishing these two cases is central to understanding the space complexity of the frequency moment estimation problem in random order streams. We show tight multi-pass space bounds for this problem for every $p < 1/\sqrt{n \log^3 n}$, resolving an open question of Lovett and Zhang (FOCS, 2023); even for 1-pass our bounds are new. To show optimality, we improve both lower and upper bounds from existing results.

Our information complexity framework significantly extends the toolkit for proving multi-pass streaming lower bounds, and we give a wide number of additional streaming applications of our lower bound techniques, including multi-pass lower bounds for ℓ_p-norm estimation, ℓ_p-point query and heavy hitters, and compressed sensing problems.

CCS CONCEPTS

• **Theory of computation → Lower bounds and information complexity**; **Streaming models**.

KEYWORDS

Information Complexity, Streaming Algorithms

ACM Reference Format:
Mark Braverman, Sumegha Garg, Qian Li, Shuo Wang, David P. Woodruff, and Jiapeng Zhang. 2024. A New Information Complexity Measure for Multi-pass Streaming with Applications. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC '24), June 24–28, 2024, Vancouver, BC, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3618260.3649672

1 INTRODUCTION

Streaming problems with stochastic inputs have been popularly studied in the streaming community [6, 9, 13, 14, 16, 20, 27, 28, 37], which have applications to diverse areas including learning theory [12, 17, 47, 48] and cryptography [18, 19, 31, 50]. In this setting, one sees a stream of i.i.d. samples from some underlying distribution. As the samples are i.i.d., this is a special case of the well-studied random order streaming model. In this paper, we will consider streaming problems (with stochastic inputs) that are allowed multiple passes over their input. Surprisingly, even the most basic problems in data streams are not resolved in the stochastic setting. We discuss two such problems below.

Coin Problem. If the stream is $X_1, X_2, \ldots, X_n$, with each X_i being independently and uniformly drawn from $\{-1, 1\}$, then the *coin problem* is to compute the sum of these bits up to additive error $O(\sqrt{n})$, which gives a non-trivial advantage for estimating the majority. This can be solved trivially in $O(\log n)$ bits of memory by storing the sum of input bits, and was recently shown to require $\Omega(\log n)$ bits of memory in [9]. However, if we allow two or more

*The full version of this paper is available at https://arxiv.org/abs/2403.20283.

STOC '24, June 24–28, 2024, Vancouver, BC, Canada

ACM ISBN 979-8-4007-0383-6/24/06
https://doi.org/10.1145/3618260.3649672

passes over the stream, the only known lower bound is a trivial $\Omega(1)$ bits.

Naturally, the coin problem is closely related to the fundamental question of counting the number of elements in a data stream – maintain a counter C in a stream under a sequence of updates of the form $C \leftarrow C+1$ or $C \leftarrow C-1$, which is arguably the most basic question you could ask. More generally, one would like to approximate C up to a constant multiplicative factor. One can solve this exactly using $\lceil \log_2 n \rceil$ bits of memory, where n is the length of the stream. This bound is tight for deterministic algorithms even if you allow a large poly(n) approximation factor [3]. It is also tight for constant-factor-approximation randomized 1-pass algorithms, via a reduction from the augmented indexing problem [36]. In fact, the lower bound of [9] for the coin problem implies that the randomized 1-pass lower bound holds even when the algorithm succeeds with high probability over an input of *i.i.d.* coin flips. Despite our understanding of 1-pass algorithms, we are not aware of an $\Omega(\log n)$ lower bound for this basic problem of approximate counting for more than one pass, even for worst-case streams, once two or more passes are allowed. For k-pass streaming algorithms, a weaker lower bound of $\Omega((\log n)^{1/k})$ can be derived using communication complexity lower bounds for the Greater-Than function[1] [40, 51]. There is work on approximate counting lower bounds of [2], which analyzes a single linear sketch and therefore can only apply to single pass streaming algorithms. It also requires at least exponentially long streams in the value n of the final count, despite the count being $O(n)$ in absolute value at any time during the stream.

Needle problem. Here the goal is to distinguish between streams $X_1, \cdots, X_n$ sampled from two possible underlying distributions: let $t = \Omega(n)$,

- *Uniform distribution* $\boldsymbol{D}_0$*:* each X_i is picked independently and uniformly at random from the domain $[t]$, and
- *Needle distribution* $\boldsymbol{D}_1$*:* the distribution first uniformly samples an element α from $[t]$ (we call it the *needle*). Then, each item X_i independently with probability p equals α, and otherwise is sampled uniformly from $[t]$.

Lovett and Zhang [37] showed a lower bound of $\Omega\left(\frac{1}{p^2 n \log n}\right)$ bits for any constant number of passes to distinguish $\boldsymbol{D}_0$ from $\boldsymbol{D}_1$, and left it as an open problem to improve this bound. Using an algorithm for frequency moment estimation, the work of Braverman et al. [11] shows that for $p = o\left(\frac{1}{n^{2/3}}\right)$, there is a single-pass upper bound of $O\left(\frac{1}{p^2 n}\right)$ bits. For larger p, the best known algorithm is to store the previous $O\left(\frac{1}{p^2 n}\right)$ items in a stream, using $O\left(\frac{\log n}{p^2 n}\right)$ bits, and check for a collision. Thus, for every p there is a gap of at least $\log n$ in known upper and lower bounds, and for the important case of $p > \Omega\left(\frac{1}{n^{2/3}}\right)$, the gap is $\Theta(\log^2 n)$. We note that the work of Lovett and Zhang also requires $t = \Omega(n^2)$ in its lower bounds[2], which limits its applications to frequency moment estimation.

The needle and coin problems are related to each other if $p = \Theta\left(\frac{1}{\sqrt{n}}\right)$ and when the algorithm has access to a random string that is not counted towards the space complexity[3]. One can randomly hash each universe element in a stream in the needle problem to $\{-1, 1\}$ – under the needle distribution $\boldsymbol{D}_1$, the absolute value of the sum of bits is likely to be an additive $\Theta(\sqrt{n})$ larger than in the uniform distribution $\boldsymbol{D}_0$, and thus the coin problem is at least as hard as the needle problem. However, the needle problem for $p = \Theta\left(\frac{1}{\sqrt{n}}\right)$ could be strictly easier than the coin problem.

We stress that the coin and needle problems are arguably two of the most fundamental problems in data streams. Indeed, a vast body of work has considered estimating the q-th frequency moment $F_q = \sum_{i=1}^{t} |x_i|^q$, for an underlying t-dimensional vector x undergoing positive and negative updates to its coordinates. If $t = 1$, this problem is at least as hard as the coin problem. The lower bound of [9] thus gave the first $\Omega(\log n)$ lower bound for single pass F_2-estimation in the bounded deletion data stream model [32], even when one does not charge the streaming algorithm for storing its randomness. As the lower bound of [9] was actually an information cost lower bound, it gave rise to the first direct sum theorem for solving multiple copies of the F_2-estimation problem, showing that solving r copies each with constant probability requires $\Omega(r \log n)$ bits in this data stream model. Lower bounds for the coin problem were shown to imply additional lower bounds in random order and bounded deletion models for problems such as point query, and heavy hitters; see [9] for details. For $q > 2$, if we set $p = \Theta\left(\frac{1}{n^{1-1/q}}\right)$ in the needle problem, then it is not hard to see that F_q differs by a constant factor for distributions $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$ with large probability. In this case, the lower bound of [37] gives an $\Omega\left(\frac{n^{1-2/q}}{\log n}\right)$ lower bound for frequency moment estimation in the random order insertion-only data stream model for any constant number of passes, provided $t = \Omega(n^2)$. There is also a long sequence of prior work on this problem [6, 13, 16, 39], which obtains polynomially worse lower bounds (though does not require $t = \Omega(n^2)$). The single-pass arbitrary order stream $O\left(n^{1-2/q}\right)$ upper bound of [11] for $q > 3$ matches this up to a logarithmic factor, and a central question in data streams is to remove this logarithmic factor, as well as the requirement that $t = \Omega(n^2)$.

1.1 Our Contributions

We give a new multi-pass notion of information complexity that gives a unified approach to obtain lower bounds for the coin and the needle problem for any number k of passes. We note that the measure of information we use is a generalization of the notion of information complexity for $k = 1$ pass given in [9]. Namely, we

[1] Briefly, given inputs $x, y \in [n^{0.1}]$ to the two-player CC problem for Greater-than function, Alice adds $x \cdot n^{0.9}$ 1s to the stream and Bob adds $y \cdot n^{0.9}$ number of -1s. Determining the sign of $x - y$, or estimating $x - y$ up to additive error $n^{0.9}$ requires $\Omega((\log n)^{1/k})$ randomized k-round communication since it solves Greater-Than.

[2] Here we choose the domaint to be $[t]$ and n the number of samples to be consistent with our notation for the coin problem. Unfortunately, this is exactly the opposite notation used in [37]

[3] This is referred to as the public coin model in communication complexity, and we may naturally view this problem as an n-player communication game.

define the k-pass information complexity notion by

$$MIC(\mathsf{M},\mu) := \sum_{i=1}^{k}\sum_{j=1}^{n}\sum_{\ell=1}^{i} \mathrm{I}\left(\mathsf{M}_{(i,j)}; X_\ell \mid \mathsf{M}_{(\le i,\ell-1)}, \mathsf{M}_{(\le i-1,j)}\right) + \sum_{i=1}^{k}\sum_{j=1}^{n}\sum_{\ell=j+1}^{n} \mathrm{I}\left(\mathsf{M}_{(i,j)}; X_\ell \mid \mathsf{M}_{(\le i-1,\ell-1)}, \mathsf{M}_{(\le i-1,j)}\right),$$

where $(X_1, \cdots, X_n) \sim \mu$ and $\mathsf{M}_{(i,j)}$ represents the jth memory state in the ith pass. We will set μ to be the uniform distribution over $\{-1,1\}^n$ in the coin problem and the uniform distribution $\boldsymbol{D}_0$ in the needle problem. When μ is clear from the context, we will drop it from the notation and write $MIC(\mathsf{M})$. The primary challenge in establishing lower bounds for multi-pass streaming algorithms arises from the fact that the streaming data loses its independence when multiple passes are used. To mitigate this, the idea of $MIC(\mathsf{M})$ is to capture some residual independence by carefully fixing some memory states in the previous pass. To make this notion useful, we show that $MIC(\mathsf{M})$ is upper bounded by $2ksn$.

Note that this multi-pass information complexity notion applies to any streaming problem as long as it is defined on a product distribution. As we will see in the paper, this notion is useful for proving multi-pass streaming lower bounds via various approaches, such as round elimination as well as reductions to communication problems.

Just as the measure of information in [9] was crucial for 1-pass applications, such as the amortized complexity of approximate counting in insertion streams [1], we will show our notions have a number of important applications and can be used to obtain multi-pass lower bounds for both the coin and needle problems. We will define and motivate our information complexity notion more below, but we first describe its applications.

1.1.1 The Coin Problem. We give tight lower bounds on the information complexity of the coin problem, significantly extending the results of [9] for the 1-pass setting to the multi-pass setting. We then give a new multi-pass direct sum theorem for solving multiple copies, and use it for streaming applications.

Multi-Pass Coin Problem. We give the first non-trivial multi-pass lower bound for the coin problem.

Theorem 1.1 (k-Pass Coin Problem). *Given a stream of n i.i.d. uniformly random bits, any k-pass streaming algorithm which outputs the majority of these bits with probability $1-\gamma$ for a small enough constant $\gamma > 0$, requires $\Omega(\frac{\log n}{k})$ bits of memory.*

Theorem 1.1 is a significant strengthening of the main result in [9] which held only for $k = 1$ pass. Although the work of [10] allows for a larger bias on its coins, it also held only for $k = 1$ pass.

As discussed before, we can interpret the coins as updates in $\{-1,1\}$ to a counter C initialized to 0. Adjoining a prefix of $\alpha\sqrt{n}$ 1s to the stream for a large enough constant $\alpha > 0$, we have that by bounds on the maximum deviation for a 1-dimensional random walk that C will be non-negative at all points during the stream, which corresponds to the *strict turnstile streaming model* – where one can only delete previously inserted items. We also have that the final value of C will deviate from its expectation $\alpha\sqrt{n}$ by an additive $\Omega(\sqrt{n})$ with constant probability, that is, it is anti-concentrated. Consequently, Theorem 1.1 implies the following.

Theorem 1.2 (k-Pass Counter in Strict Turnstile Streams). *Any k-pass strict turnstile streaming algorithm which counts the number of insertions minus deletions in a stream of length n up to a small enough constant multiplicative factor and with probability at least $1-\gamma$ for a small enough constant $\gamma > 0$, requires $\Omega(\frac{\log n}{k})$ bits of memory.*

By constant multiplicative factor, we mean to output a number C' for which $(1-\epsilon)C \le C' \le (1+\epsilon)C$ for a constant $\epsilon > 0$. For insertion-only streams where no deletions of items are allowed, non-trivial algorithms based on Morris counters achieve $O(\log\log n)$ bits [21, 25, 44, 49]. We rule out any non-trivial algorithm for strict turnstile streams for any constant number of passes.

To obtain further applications, we first show a direct sum theorem for multi-pass coin problems.

Theorem 1.3 (Direct Sum for k-Pass Counter). *Suppose $k < \log n$ and $t < n^c$ for a sufficiently small constant $c > 0$. Given t independent streams each of n i.i.d. uniformly random bits, any k-pass streaming algorithm which outputs a data structure such that, with probability $1-\gamma$ for a small enough constant $\gamma > 0$ over the input, the algorithm's randomness, and over a uniformly random $j \in [t]$, outputs the majority bit of the j-th stream, requires $\Omega\left(\frac{t\log n}{k}\right)$ bits of memory.*

As an example application of this theorem, in [41] the following problem was studied for a real number $p \in [0,2]$: given vectors $v_1, \ldots, v_t \in \{-\mathrm{poly}(n), \ldots, \mathrm{poly}(n)\}^d$, estimate a constant fraction of the $\|v_1\|_p, \ldots, \|v_t\|_p$ up to a sufficiently small constant multiplicative factor with constant probability, where for a d-dimensional vector y, the p-norm[4] $\|y\|_p = \left(\sum_{i=1}^{d} |y_i|^p\right)^{1/p}$. We will refer to this problem as Multi-ℓ_p-Estimation. The best upper bound is $O(t \cdot \log n)$, which follows just by solving each instance independently with constant probability and using $O(\log n)$ bits [36]. An $\Omega(t\log\log n + \log n)$ randomized lower bound follows for any $O(1)$-pass streaming algorithm by standard arguments[5]. We note that if we do not charge the streaming algorithm for its randomness, then the $O(1)$ pass lower bound for Multi-ℓ_p-Estimation is an even weaker $\Omega(t\log\log n)$.

By using Theorem 1.3 and having each vector v_i in the Multi-ℓ_p-Estimation problem correspond to a single counter, we can show the following:

Theorem 1.4 (k-Pass Multi-ℓ_p-Estimation). *Suppose $k < \log n$ and $t < n^c$ for a sufficiently small constant $c > 0$. Any k-pass streaming algorithm which solves the* Multi-ℓ_p-Estimation *Problem on t instances of a stream of n updates for each vector, solving each ℓ_p-norm estimation problem up to a small enough constant factor with probability $1-\gamma$ for a sufficiently small constant γ , requires $\Omega\left(\frac{t\log n}{k}\right)$ bits of memory.*

[4] For $p < 1$ the quantity $\|v\|_p$ is not a norm, but it is still a well-defined quantity. With a standard abuse of notation, we will refer to it as a p-norm.
[5] The $\Omega(t\log\log n)$ bound follows just to record the output, while the $\Omega(\log n)$ lower bound follows by a reduction from the Equality problem, as in [4].

Another important streaming question we consider is the ℓ_2-Point Query Problem: given an underlying d-dimensional vector $x \in \{-\text{poly}(n), \ldots, \text{poly}(n)\}^d$ that undergoes a sequence of positive and negative additive updates to its coordinates, for each $j \in \{1, \ldots, d\}$, one should output x_j up to an additive error $\epsilon\|x\|_2$ with constant probability. Related to this question is the ℓ_2-Heavy Hitters Problem which asks to output a set S which (1) contains all indices i for which $x_i^2 \geq \epsilon\|x\|_2^2$, and (2) does not contain any index i for which $x_i^2 \leq \frac{\epsilon^2}{2}\|x\|_2^2$. Further, for all $i \in S$, one should output an estimate $\widehat{x}_i$ with $|\widehat{x}_i - x_i| \leq \epsilon\|x\|_2$. In [9], for both of these problems an $\Omega(\epsilon^{-2}\log n)$ memory lower bound was shown for single-pass algorithms on length-n streams, which improved the previous best known $\Omega(\epsilon^{-2}\log d)$ lower bounds when the stream length n is much larger than the dimension d of the input vectors. Notably, the lower bounds in [9] only hold for single pass algorithms.

By having each coordinate of an underlying vector x correspond to a counter, we can also use Theorem 1.3 to solve the ℓ_2-Point Query Problem and the ℓ_2-Heavy Hitters Problem. Here we also use that the Euclidean norm of the underlying vector is concentrated.

Theorem 1.5 (k-Pass Point Query and Heavy Hitters). *Suppose $k < \log n$ and $\epsilon^{-2} < n^c$ for a sufficiently small constant $c > 0$. Any k-pass streaming algorithm which, with probability $1-\gamma$ for a sufficiently small constant $\gamma > 0$, solves the ℓ_2-*Point Query Problem *or the ℓ_2-*Heavy Hitters Problem *on a vector $x \in \{-poly(n), \ldots, poly(n)\}^d$ in a stream of n updates, requires at least $\Omega\left(\frac{\epsilon^{-2}\log n}{k}\right)$ bits of memory.*

Our $\Omega(\epsilon^{-2}\log n)$ bit lower bound for the ℓ_2-Heavy Hitters Problem can be applied to the Sparse Recovery Problem in compressed sensing (see, e.g., [24, 45]), which involves an input vector $x \in \{-\text{poly}(n), \ldots, \text{poly}(n)\}^d$ in a stream, and asks to output an r-sparse vector $\widehat{x}$ for which

$$\|\widehat{x} - x\|_2 \leq \Delta \cdot \|x - x_r\|_2, \tag{1}$$

where x_r is x with all but the top r coordinates set to 0. Here $\Delta > 1$ is any fixed constant.

A standard parameter of sparse recovery is the Signal-to-Noise Ratio (SNR), which is defined to be $\frac{\|x_r\|_2^2}{\|x\|_2^2}$. The SNR is at most 1, and if it is 1, there is a trivial $\Omega(r(\log n + \log d))$ bit lower bound. Indeed, since the guarantee of (1) has multiplicative error, we must have $\widehat{x} = x = x_r$ in this case, and it takes $\Omega(r(\log n + \log d))$ bits to encode, for each of the r non-zero locations in x, its location and its value. However, when the SNR is a constant bounded away from 1, this encoding argument no longer applies. Indeed, while one can show an $\Omega(r\log d)$ bits lower bound to encode the identities of r locations, each of their values can now be approximated up to a small multiplicative constant, and so encoding their values requires only $\Omega(r\log\log n)$ bits.

While an $\Omega(r + \log\log n)$ *measurement* lower bound is known for multi-pass streaming algorithms [46] for constant SNR bounded away from 1, perhaps surprisingly in the data stream model, an $\Omega(r\log n)$ bit lower bound for streams of length n and SNR bounded away from 1 was unknown. As our lower bound for the ℓ_2-Heavy Hitters Problem only requires recovering a large constant fraction of the ℓ_2-heavy hitters, all of which are comparable in magnitude in our hard instance, and the Euclidean norm is concentrated, we in fact can obtain a lower bound for the Sparse Recovery Problem even if the SNR is a constant bounded away from 1. We note that there is an $O(\log\log n)$-pass streaming algorithm which uses $O(r\log n\log\log n)$ bits of memory to solve the sparse recovery problem for any SNR, see [43] which builds upon [29] (see the text after the proof of Theorem 3.7 in [29] on how to obtain an exactly r-sparse output). Our lower bound is thus tight up to $\text{poly}(\log\log n)$ factors.

Theorem 1.6 (Bit Complexity of Sparse Recovery). *Suppose $k < \log n$ and $r < n^c$ for a sufficiently small constant $c > 0$. Any k-pass streaming algorithm which, with probability $1-\gamma$ for $\gamma > 0$ a small constant, solves the* Sparse Recovery Problem *for constant SNR in $(0,1)$, requires $\Omega(\frac{r\log n}{k})$ bits of memory.*

1.1.2 The Needle Problem. Lovett and Zhang [37] recently showed the following lower bound for the needle problem.

Theorem 1.7 ([37]). *Any k-pass streaming algorithm M which distinguishes between the uniform and needle distributions with high probability, where p denotes the needle probability, n the stream length, and s the space, satisfies $ksp^2n\log(n) = \Omega(1)$, provided the domain size $t = \Omega(n^2)$.*

While this lower bound is nearly tight, it was conjectured by [6, 13, 16, 37] that the additional $\log(n)$ term can be removed, and it also was plausible that the $t = \Omega(n^2)$ restriction could be removed. This conjecture is for good reason, as for $n = \Theta(t)$ and $p \ll \frac{1}{n^{2/3}}$ and $k = 1$, an upper bound for estimating frequency moments of [11] shows that $sp^2n = O(1)$. Indeed, the upper bound of [11] shows how to estimate $F_q = \sum_{i=1}^t f_i^q$ up to an arbitrarily small but fixed constant factor in $O(t^{1-2/q})$ bits of memory and a single pass, for any $q > 3$. Notice that in distribution $\boldsymbol{D}_0$, we could choose a proper $n = \Theta(t)$ such that $F_q = n + o(n)$ with high probability. On the other hand, for distribution $\boldsymbol{D}_1$, we have that $F_q > (p\cdot n)^q$, and so if $p = \Theta(1/n^{1-1/q})$, these two distributions can be distinguished by the algorithm of [11]. In this case the conjecture would say $s = \Omega(1/(np^2)) = \Omega(n^{1-2/q}) = \Omega(t^{1-2/q})$, which matches the space upper bound of [11].

We resolve this conjecture. As a consequence, several other streaming lower bounds mentioned by [16, 37, 38] can be improved automatically. Also, our results also imply that the frequency estimation problem for $q > 2$ is as hard in the random order model as in the arbitrary order model (both are $\Omega(t^{1-2/q})$).

Theorem 1.8 (k-pass needle problem). *Any k-pass streaming algorithm M with space s that distinguishes $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$ with high probability satisfies $ksp^2n = \Omega(1)$, where p denotes the needle probability and $n \leq t/100$ denotes the number of samples.*

If we use the algorithm of [11] to the needle problem by the reduction discussed above, we can conclude that our lower bound for the needle problem is tight when $p \ll \frac{1}{n^{2/3}}$. However, we further improve the upper bound by giving a new algorithm and show that:

Theorem 1.9 (Improved upper bound). *There exists a one-pass streaming algorithm that distinguishes $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$ with high probability and uses $O(\frac{1}{p^2n})$ bits of space when $p \leq \frac{1}{\sqrt{n\log^3 n}}$.*

Our upper bound improves upon [11], and shows that our lower bound for the needle problem is indeed tight for any $p \leq \frac{1}{\sqrt{n \log^3 n}}$. The remaining gap only exists in the range of $p > \frac{1}{\sqrt{n \log^3 n}}$.

In fact, for $p \geq \frac{1}{\sqrt{n}}$, for the n-player communication problem, where each player has a stream item and the players speak one at a time from left to right in the message passing model (see, e.g., [33] for upper bounds for a number of problems in this model), we show that the problem can be solved by having each player send at most $O((\log\log n)(\log\log\log n))$ bits to the next player. It is not quite a streaming algorithm, as the players need to know their identity, but would be a streaming algorithm if we also allow for a clock, so that we know the number i for the i-th stream update, for each i.

Theorem 1.10 (upper bound for communication game). *There exists an n-player one-round communication protocol that distinguishes $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$ with high probability and each player uses at most $O((\log\log n)(\log\log\log n))$ bits of space for any $p \geq \frac{1}{\sqrt{n}}$.*

Theorem 1.10 shows that for $p = \frac{1}{\sqrt{n}}$, the needle problem is strictly easier than the coin problem, and thus the abovementioned algorithm for the needle problem, by first reducing to the coin problem, is suboptimal. Indeed, in the same communication model or for streaming algorithms with a clock, our $\Omega\left(\frac{\log n}{k}\right)$ lower bound in Theorem 1.1 applies. Thus, for a constant number k of passes, the coin problem requires $\Omega(\log n)$ bits of memory whereas the needle problem with $p = \frac{1}{\sqrt{n}}$ can be solved with $O((\log\log n)(\log\log\log n))$ bits of memory, showing that there exists a separation between the two problems in the n-player communication model.

Remark. Note that our algorithm not only applies to the needle problem, but also could be adapted to a more general setting where the needle is randomly ordered while non-needle items could be in an arbitrary order with some constraints.

Our improved lower bound for the needle problem can be used to obtain optimal lower bounds in the random order model for arguably the most studied problem in the data stream literature, namely, that of approximating the frequency moments. Starting with the work of Alon, Matias, and Szegedy [4], there has been a huge body of works on approximating the frequency moments in arbitrary order streams, see, e.g., [5, 7, 8, 15, 22, 23, 30, 34, 42], and references therein. As mentioned above, Braverman et al. [11] gave an upper bound of $O(t^{1-2/q})$ for constant approximation for all $q > 3$, which is optimal for arbitrary order insertion streams.

A number of works have also studied the frequency moment estimation problem in *randomly ordered* streams. While the $O(t^{1-2/q})$ bit upper bound of [11] still holds, we did not have a matching lower bound. Chakrabarti, Cormode, and McGregor [13] gave the first non-trivial $\Omega(t^{1-3/q})$ lower bound. A follow-up paper by Andoni et al. [6] improved this lower bound to $\Omega(t^{1-2.5/q})$. Recently, a lower bound of $\Omega(n^{1-2/q}/\log n)$ was shown by Lovett and Zhang [37] provided $t = \Omega(n^2)$. Since a stream of i.i.d. samples is automatically randomly ordered, our Theorem 1.8 resolves this long line of work, giving an $\Omega(t^{1-2/q})$ lower bound. We thus improve the lower bound of [37] by a logarithmic factor, and also remove the requirement that $t = \Omega(n^2)$. The application to frequency moments follows by applying our theorem with $t = \Theta(n)$ and $p = 1/n^{1-1/q}$ and arguing that the needle problem gives rise to a constant factor gap in the value of the q-th frequency moment in the two cases. We note that the work of [26] claimed to obtain an $\Omega(t^{1-2/q})$ lower bound for frequency moment estimation in a random order, but was later retracted due to an error which has been pointed out in multiple places, e.g., [39] retracts its lower bounds and points out the error[6] in [26].

There are other related problems to frequency moment estimation that we also obtain improved lower bounds for, such as frequency moment estimation of sub-sampled streams. McGregor et al. [38] studied streaming problems in a model where the stream comes in so rapidly that you only can see each element independently with a certain probability. Our Theorem 1.8 gives an optimal lower bound for this problem as well, via the reduction in [38]. Another example concerns stochastic streaming problems such as collision probability estimation studied by Crouch et al. [16]. They provided several lower bounds based on the needle lower bound of [6]. Our Theorem 1.8 automatically improves their lower bounds via the same reductions.

2 TECHNICAL OVERVIEW

In this section, we give a brief overview of our proofs for both the coin problem and needle problem.

2.1 Properties of New Multi-Pass IC Notion

In this section, we will show some important properties of our IC notion. In addition to the *MIC*, we get an another natural expression for the an information measure for k-pass algorithms M as follows:

Definition 2.1. Let M be a k-pass streaming algorithm, its input is denoted by $X_1, \ldots, X_n$ following a product distribution μ, then we define the following information complexity notion:

$$MIC_{cond}(\mathsf{M}, \mu) := \sum_{j=1}^{n} \sum_{\ell=1}^{j} \mathrm{I}\left(\mathsf{M}_{(\leq k,j)}; X_\ell | \mathsf{M}_{<k}, \mathsf{M}_{(\leq k,\ell-1)}\right).$$

One might see that the notion above shares some similarities with *MIC*, while the difference is *MIC* further divides the information costs into smaller components. In our paper, we show the following properties, which provide an upper bound for both notions and show some relations. Particularly, we show that, when $X_1, \ldots, X_n$ are drawn from a product distribution μ, the following holds:

- $MIC(\mathsf{M}, \mu) \leq 2ksn$;
- $MIC(\mathsf{M}, \mu) \geq MIC_{cond}(\mathsf{M}, \mu)$.

Note that our information complexity notions could be applied to any other problems as long as they are defined on product distributions.

2.2 Multi-Pass Lower Bound for the Coin Problem

In this section, we assume that $X_1, \ldots, X_n$ are drawn from the uniform distribution over $\{-1, 1\}^n$; we will drop μ for the rest of the

[6] This error has also been confirmed with the authors of [26] in 2016, and no fix with their techniques seems possible.

section. We show an $\Omega(n \log n)$ lower bound on $MIC_{cond}(\mathsf{M}, \mu)$ for any k-pass algorithm M that solves the coin problem (or computes majority of input bits with large enough constant advantage).

We will prove our k-pass lower bound for computing majority (or approximating sum) on the uniform distribution by reducing it to the one-pass lower bound proven by [9], which is stated as follows:

Theorem 2.2 ([9], Corollary 14). *Given a stream of n uniform $\{-1, 1\}$ bits $X_1, \ldots, X_n$, let O be a one-pass algorithm that uses private randomness. For all $\epsilon > c_0 n^{-\frac{1}{20}}$, there exists $\delta \geq c_1 \epsilon^5$ (for small enough constant $c_1 > 0$ and large enough constant $c_0 > 0$), such that if*

$$IC(\mathsf{O}) = \sum_{j=1}^{n} \sum_{\ell=1}^{j} \mathrm{I}(\mathsf{O}_i; X_\ell | \mathsf{O}_{\ell-1}) \leq \delta n \log n, \tag{2}$$

then

$$\mathbb{E}_{\mathsf{O}_n} \left[\left(\mathbb{E} \left[\sum_{j=1}^{n} X_j \,\middle|\, \mathsf{O}_n = o_n \right] \right)^2 \right] \leq \epsilon n. \tag{3}$$

Here, O_j represents the memory state of the one-pass algorithm O after reading j input elements.

In other words, if the output of algorithm O reduces the variance of the sum[7], then it needs to have high information cost.

Our main theorem for the k-pass coin problem is stated as follows:

Theorem 2.3. *Let M be a k-pass algorithm on a stream of n i.i.d. uniform $\{-1, 1\}$ bits, $X_1, \ldots, X_n$. For all constants $\varepsilon > 0$ and n greater than a sufficiently large constant, there exists constants $\delta, \lambda > 0$, such that if $k < n^\lambda$,*

$$MIC_{cond}(\mathsf{M}) \leq \delta n \log n \quad \text{and} \quad \forall i \in \{0, \ldots, k\},\ \mathrm{H}(\mathsf{M}_i) \leq n^\lambda, \tag{4}$$

then

$$\mathbb{E}_{\mathsf{M}_{(k,n)}} \left(\mathbb{E} \left[\sum_{j=1}^{n} X_j \,\middle|\, \mathsf{M}_{(k,n)} \right] \right)^2 \leq \varepsilon n. \tag{5}$$

Theorem 2.3 and the fact that $MIC_{cond} \leq 2ksn$, along with ([9], Claim 6)– which proved an $\Omega(n)$ lower bound on the L.H.S. of Equation (5) for any algorithm whose output computes majority with 0.999 advantage, give us the $\Omega((\log n)/k)$ space lower bound.

Construction of O. To prove Theorem 2.3, we develop a new simulation technique to prove our multi-pass streaming lower bounds for the coin problem. Given a stream X of n i.i.d. uniform bits, let M be a k-pass algorithm that goes over X twice in order and computes the majority – that is, the expected variance of the sum of input bits conditioned on the output of the second pass is a constant factor less than that of the maximum. Informally, for ease of discussion, we refer to the variance reduction as M approximating $\sum_j X_j$ up to an additive error[8] of $\epsilon\sqrt{n}$. Using M, we construct a one-pass algorithm O, which given a stream of n i.i.d. uniform bits Y, also approximates $\sum_j Y_j$ up to an additive error of $\sim \epsilon\sqrt{n}$. Let M_i represent the random variable for the output (or memory state) at the end of the i-th pass of M ($i \in [k]$). O executes k passes of M in parallel. Before reading the input bits $Y_1, \ldots, Y_n$, O samples memory states at the end of first $k-1$ passes from the joint distribution on $(\mathsf{M}_0, \ldots, \mathsf{M}_{k-1})$. O then modifies the given input Y to X' such that the parallel execution of the $k-1$ passes of the algorithm M on $X'_1, \ldots, X'_n$ end in the sampled memory states. O also maintains an approximation for the modification, that is of $\sum_{j=1}^{n}(X'_j - Y_j)$; this helps O to compute $\sum Y_j$ as long as M computes $\sum X'_j$ after k passes. As we want O to have comparable information cost to that of M, the approximation of the modification should take low memory[9]. The key observation that makes such an approximation possible is: since the KL divergence of the distribution X, conditioned on reaching memory states $\mathsf{M}_0, \ldots, \mathsf{M}_{k-1}$, from the uniform distribution is bounded by the entropy of $(\mathsf{M}_0, \ldots, \mathsf{M}_{k-1})$ (which we assume to be $<< n$), algorithm O does not need to drastically modify Y (which has a uniform distribution). Still, we cannot afford to store the modification exactly; however, a cruder approximation suffices, which can be computed using low memory.

As described above, algorithm O has two components, 1) imitate k passes of M simultaneously, and 2) maintain an approximation for modifying input Y to a valid input X' for the first $k-1$ passes of M. To formally describe algorithm O (in Section 2.2.3), we first state these two components separately as algorithms Im (in Section 2.2.1) and Apr (in Section 2.2.2) respectively.

2.2.1 Single-Pass Algorithm Im *Imitating k Passes of* M. Recall that M is a k-pass algorithm that runs on a stream of n i.i.d. uniform $\{-1, 1\}$ bits, $X_1, X_2, \ldots, X_n$. We describe algorithm Im in Algorithm 1. Let Im_j (where $j \in [n]$) represent the random variable for the memory state of Algorithm Im after reading j inputs bits, and Im_0 be a random variable for the starting memory state for the algorithm. The input Y to the algorithm Im is drawn from the uniform distribution on $\{-1, 1\}^n$.

Let M'_i denote the random variable associated with value m'_i ($i \in \{0, 1, \ldots, k-1\}$). The distribution of $\mathsf{M}'_{<k}$ is defined at Step 1 of Algorithm 1. Let $\{\mathsf{M}'_{(i,j)}\}_{i\in[k], j\in\{0,\ldots,n\}}$ denote the random variables associated with values $\{m'_{(i,j)}\}_{i\in[k], j\in\{0,\ldots,n\}}$. The distribution of $\mathsf{M}'_{(i,j)}$ ($j \in [n]$) is defined at Step 19 of Algorithm 1, and of $\mathsf{M}'_{(i,0)}$ is defined at Step 3. Let $\{X'_j\}_{j\in[n]}$ denote the random variable for value x'_j in Step 5 of Algorithm 1. These distributions depend on the joint distribution on $(X, \mathsf{M}_{\leq k}, \mathsf{M}_{(\leq k,[1,n])})$ and the uniform distribution of Y. Together with a data processing inequality on $Y \to X' \to$ Algorithm 1, we have

$$\begin{aligned} IC(\mathsf{Im}) &= \sum_{j=1}^{n} \sum_{\ell=1}^{j} \mathrm{I}\left(\mathsf{Im}_j; Y_\ell | \mathsf{Im}_{\ell-1}\right) \\ &\leq \sum_{j=1}^{n} \sum_{\ell=1}^{j} \mathrm{I}\left(\mathsf{M}_{(\leq k,j)}; X_\ell | \mathsf{M}_{<k}, \mathsf{M}_{(\leq k,\ell-1)}\right) = MIC_{cond}(\mathsf{M}). \end{aligned}$$

2.2.2 Low Information Approximation Algorithm Apr. We develop Apr for the general problem of approximating the sum of n elements, each in $\{-1, 0, 1\}$. The problem is as follows: given parameters $\gamma > 0$,

[7] We want some mathematical quantity to measure how much information the output of algorithm O has about the sum/count of the input bits, and variance turns out to be the ideal measure for [9] as well as for our paper.

[8] This is actually crucial as our proof technique does not work with large multiplicative errors, which is a bottleneck for generalizing single pass memory lower bounds for the coin problem with larger biases [10] to multiple passes.

[9] Note that it takes $\log n$ bits of memory to store $\sum_{j=1}^{n}(X'_j - Y_j)$ exactly.

Algorithm 1: Single pass algorithm Im imitating k passes of M

Input: a stream of n bits $y_1, \ldots, y_n$, drawn from uniform distribution on $\{-1,1\}^n$

1: Sample $m'_0, m'_1, \ldots, m'_{k-1} \sim (\mathsf{M}_0, \mathsf{M}_1, \ldots, \mathsf{M}_{k-1})$ {Im samples memory states for the end of first $k-1$ passes}
2: $im_0 \leftarrow (m'_0, m'_1, \ldots, m'_{k-1})$ {Im stores these memory states for the entire algorithm}
3: $\forall i \in [k]$, $m'_{(i,0)} \leftarrow m'_{(i-1)}$ {Starting memory states for the k passes of M}
4: **for** $j = 1$ to n **do**
5: $\beta_j \leftarrow \left(\Pr\left[X_j = 1 \mid \mathsf{M}_{(\le k,j-1)} = m'_{(\le k,j-1)}, \mathsf{M}_{<k} = m'_{<k}\right] - \frac{1}{2}\right)$ {Can be calculated using im_{j-1}}
6: **if** $\beta_j > 0$ **then**
7: **if** $y_j = 1$ **then**
8: $x'_j \leftarrow y_j$
9: **else**
10: $x'_j \leftarrow y_j$ with probability $1 - 2\beta_j$, and $x'_j \leftarrow 1$ otherwise
11: **end if**
12: **else if** $\beta_j \le 0$ **then**
13: **if** $y_j = 1$ **then**
14: $x'_j \leftarrow y_j$ with probability $1 + 2\beta_j$, and $x'_j \leftarrow -1$ otherwise
15: **else**
16: $x'_j \leftarrow y_j$
17: **end if**
18: **end if**
19: Sample $(m'_{(1,j)}, m'_{(2,j)}, \ldots, m'_{(k,j)})$ from the joint distribution of $\left((\mathsf{M}_{(1,j)}, \mathsf{M}_{(2,j)}, \ldots, \mathsf{M}_{(k,j)})\right)$ condition on $\left(\mathsf{M}_{(\le k,j-1)} = m'_{(\le k,j-1)},\ \mathsf{M}_{<k} = m'_{<k},\ X_j = x'_j\right)$ {Given $im_{(j-1)}$, Im executes jth time-step for all passes of M}
20: $im_j \leftarrow (m'_{(1,j)}, m'_{(2,j)}, \ldots, m'_{(k,j)}, m'_0, m'_1, \ldots, m'_{k-1})$ {At the jth time-step, Im stores these memory states}
21: **end for**

$B > \gamma\sqrt{n}$, and a stream of n elements $a_1, \ldots, a_n \in \{-1, 0, 1\}$ jointly drawn from a distribution $\mathcal{D}$ (such that $\mathbb{E}_{a\sim\mathcal{D}}\left[\sum_{j=1}^n \mathbf{1}_{a_j \neq 0}\right] \le B$), the aim is to output $\left(\sum_{j=1}^n a_j\right)$ up to an additive error of $\gamma\sqrt{n}$. Let $R^{\mathsf{Apr}} = (R_1^{\mathsf{Apr}}, \ldots, R_n^{\mathsf{Apr}})$, where R_j^{Apr} denotes the random variable for private randomness used by algorithm Apr at the jth time-step; we formalize the error guarantee as

$$\mathbb{E}_{a\sim\mathcal{D}, r\sim R^{\mathsf{Apr}}}\left[\left|\mathsf{Apr}(a,r) - \sum_{j=1}^n a_j\right|\right] \le \gamma\sqrt{n}.$$

Additionally, we establish that the streaming algorithm Apr (described in Algorithm 2) has low information cost – the memory state at each time-step has low entropy, that is, $\forall i \in [n]$, $\mathrm{H}(\mathsf{Apr}_i) \sim 2\log\left(\frac{B}{\gamma\sqrt{n}}\right)$. Note that, the exact computation of $\sum_j a_j$ requires $\log n$ memory.

Informally, Apr samples each a_j with probability $p \sim \frac{B}{\gamma^2 n}$ and maintains their sum using a counter Δ. It is easy to see that Δ/p is an approximation of $\sum_j a_j$ (with additive error $\gamma\sqrt{n}$) as long as $\sum_j \mathbf{1}_{a_j\neq 0}$ is bounded by $\sim B$. As B is an upper bound only on the expectation of $\sum_j \mathbf{1}_{a_j\neq 0}$, the algorithm Apr needs to find another way to approximate the sum whenever $\sum_j \mathbf{1}_{a_j\neq 0} >> B$. For this, Apr maintains two more counters ζ and Γ, where ζ counts the number of elements a_j sampled in the sum Δ, and Γ stores $\sum_j a_j$ exactly whenever counter ζ becomes $>> pB$. Apr is formally described in Algorithm 2.

Algorithm 2: Algorithm Apr for approximate sum

Input stream: $a_1, \ldots, a_n$, drawn from joint distribution $\mathcal{D}$ on $\{-1,0,1\}^n$
Given parameters: $\gamma > 0$, $B > \gamma\sqrt{n}$
Goal: $\forall a \in \{-1,0,1\}^n$, $\left|\mathsf{Apr}(a,r) - \sum_{j=1}^n a_j\right| \le \frac{\gamma}{2}\sqrt{n}$ with probability at least $1 - \frac{1}{n^3}$ over the private randomness $r \sim R^{\mathsf{Apr}}$

1: Let $p = \min\left\{6000\log^2 n \cdot \left(\frac{B}{\gamma^2 n}\right), 1\right\}$ {probability of sampling}
2: $\Delta \leftarrow 0$ {Δ maintains an approximation for $p \cdot \left(\sum_j a_j\right)$}
3: $\zeta \leftarrow 0$ {ζ approximates $p \cdot \left(\sum_j \mathbf{1}_{a_j\neq 0}\right)$}
4: $\Gamma \leftarrow 0$ {Γ computes $\left(\sum_j a_j\right)$ exactly when Δ/p is not a good approximation}
5: **for** $j = 1$ to n **do**
6: **if** $\zeta < 20\log n \cdot pB$ **then**
7: Let r_j be 1 with probability p and 0 otherwise
8: **if** $r_j = 1$ **then**
9: $\Delta \leftarrow \Delta + a_j$
10: $\zeta \leftarrow \zeta + \mathbf{1}_{a_j\neq 0}$ {Sample a_j and update the counters with probability p}
11: **end if**
12: **else if** $\zeta \ge 20\log n \cdot pB$ **then**
13: $\Gamma \leftarrow \Gamma + a_j$
14: **end if**
15: **end for**
16: **return** $\max\{\min\{\Delta/p + \Gamma, n\}, -n\}$ {Project $\Delta/p + \Gamma$ within $[-n, n]$}

In Theorems 2.4 and 2.5, we establish the approximation and information cost guarantees for the algorithm Apr. Before, we note that Apr uses private randomness at Step 7 of the algorithm and define R_j^{Apr} to be a $\mathrm{Ber}(p)$ random variable for all $j \in [n]$.

Theorem 2.4. *$\forall \gamma > \frac{4}{\sqrt{n}}, B > \gamma\sqrt{n}$, the output of Algorithm 2 (Apr) on every input stream $a \in \{-1,0,1\}^n$ satisfies the following with probability at least $1 - \frac{1}{n^3}$ over the private randomness $r \sim R^{\mathsf{Apr}}$,*

$$\left|\mathsf{Apr}(a,r) - \sum_{j=1}^n a_j\right| \le \frac{\gamma}{2}\sqrt{n},$$

which further implies that $\forall$ *distribution* $\mathcal{D}$ *on* $\{-1,0,1\}^n$,

$$\mathbb{E}_{a\sim\mathcal{D},r\sim R^{\mathsf{Apr}}}\left[\left(\mathsf{Apr}(a,r)-\sum_{j=1}^{n}a_j\right)^2\right]\le\gamma^2 n.$$

Theorem 2.5. *$\forall\gamma>\frac{4}{\sqrt{n}}, B>\gamma\sqrt{n}$, distributions $\mathcal{D}$ on $\{-1,0,1\}^n$ such that $\mathbb{E}_{a\sim\mathcal{D}}\left[\sum_{j=1}^{n}\mathbf{1}_{a_j\neq 0}\right]\le B$, memory states of Algorithm 2 (Apr) satisfies the following:*

$$\forall j\in\{0,\dots,n\},\mathrm{H}(\mathsf{Apr}_j)\le 40+6\log\log n+2\log\left(\frac{B}{\gamma\sqrt{n}}\right)$$

Here, Apr_j denotes the random variable for Apr's memory state after reading j input elements, and it depends on input a, as well as the private randomness $r\sim\mathcal{R}^{\mathsf{Apr}}$ used by the algorithm.

2.2.3 Single-Pass Algorithm O *for Computing Majority Using k-Pass Algorithm* M. After introducing the two components, we now describe the one-pass algorithm O that approximates the sum almost as well as the k-pass algorithm M, while having similar information cost to M. O runs Algorithm 1 (Im) to imitate the k passes of M – Im modifies input bit y_j at the j-th time-step to bit x'_j. In parallel, O runs Algorithm 2 (Apr) on the modification – the j-th input element to Apr is $(y_j-x'_j)\in\{-1,0,1\}$. After reading y_j, O runs jth time-steps of algorithms Im and Apr (the input to Apr is generated on the fly), and stores the jth memory states of both the algorithms. While describing O formally in Algorithm 3 (where parameters γ and B would be decided later), we will restate algorithm Im and use Apr as a black-box. As used in Subsection 2.2.2, $\mathcal{R}_j^{\mathsf{Apr}}$ represents the private randomness used by algorithm Apr at the jth time-step and Apr_j represents the random variable for jth memory state (r_j and apr_j represent their instantiations). The input to Apr is denoted by a. Let f_j^{Apr} ($j\in[n]$) represent the jth transition function for algorithm Apr, that is, $apr_j=f_j^{\mathsf{Apr}}(apr_{j-1},a_j,r_j)$. Let $\mathsf{Apr}_j(apr_{j-1},a_j)$ denote the random variable for the jth memory state, when the jth input element is a_j, $(j-1)$th memory state is apr_{j-1} and private randomness r_j is drawn from $\mathcal{R}_j^{\mathsf{Apr}}$.

The random variables Y,X' are as defined in Subsection 2.2.1, where Y is drawn from uniform distribution on $\{-1,1\}^n$ and X'_j corresponds to value x'_j as in Algorithm 3. Let $\mathcal{D}$ be the joint distribution generated by Algorithm 3 on inputs to Apr that is, the joint distribution on $(Y_1-X'_1,Y_2-X'_2,\dots,Y_n-X'_n)$. Let A_j be the random variable for the jth input element to Apr, that is, $A_j=Y_j-X'_j$. Let $\{\mathsf{M}'_{(i,j)}\}_{i\in[k],j\in\{0,\dots,n\}}$ and $\mathsf{M}'_{<k}$ be random variables as defined in Subsection 2.2.1 (these are random variables for corresponding values that appear in Algorithm 3).
For the approximation algorithm Apr, we show

$$\mathbb{E}_{a\sim\mathcal{D}}\left[\sum_{j=1}^{n}\mathbf{1}_{a_j\neq 0}\right]=\mathbb{E}\left[\sum_{j=1}^{n}\mathbf{1}_{Y_j\neq X'_j}\right]\le\sqrt{n\cdot\mathrm{H}(\mathsf{M}_{<k})}.$$

Informally, we relate the probability of modification at step j to the information that end memory states have about X_j, conditioned on the previous memory states. The claim follows from the fact that the sum of this information over j, is bounded by the entropy of the end states. Note that the above claim is tight if an end state

Algorithm 3: Single pass algorithm O using k-pass algorithm M for computing majority

Input: a stream of n *i.i.d* uniform $\{-1,1\}$ bits $Y_1,\dots,Y_n$
Given parameters: $\gamma>0, B>\gamma\sqrt{n}$
Goal: approximate $\sum_{j=1}^{n}Y_j$.

Sample $m'_0,m'_1,\dots,m'_{k-1}\sim(\mathsf{M}_0,\mathsf{M}_1,\dots,\mathsf{M}_{k-1})$
$im_0\leftarrow(m'_0,m'_1,\dots,m'_{k-1})$
$\forall i\in[k], m'_{(i,0)}\leftarrow m'_{(i-1)}$
Initialize Algorithm 2 (Apr) with parameters γ and B
Sample $apr_0\sim\mathsf{Apr}_0$ {For Algorithm 2, the starting state is deterministic}
for $j=1$ to n **do**
  $\beta_j\leftarrow\left(\Pr\left[X_j=1\mid\mathsf{M}_{(\le k,j-1)}=m'_{(\le k,j-1)},\mathsf{M}_{<k}=m'_{<k}\right]-\frac{1}{2}\right)$ {Can be calculated using im_{j-1}}
  if $\beta_j\ge 0$ **then**
    if $y_j=1$ **then**
      $x'_j\leftarrow y_j$
    else
      $x'_j\leftarrow y_j$ with probability $1-2\beta_j$, and $x'_j\leftarrow 1$ otherwise
    end if
  else if $\beta_j\le 0$ **then**
    if $y_j=1$ **then**
      $x'_j\leftarrow y_j$ with probability $1+2\beta_j$, and $x'_j\leftarrow -1$ otherwise
    else
      $x'_j\leftarrow y_j$
    end if
  end if
  $a_j\leftarrow(y_j-x'_j)$ {Setting jth input element to Apr}
  Sample $apr_j\sim\mathsf{Apr}_j(apr_{j-1},a_j)$
  Sample $(m'_{(1,j)},m'_{(2,j)},\dots,m'_{(k,j)})$ from the joint distribution on $\left((\mathsf{M}_{(1,j)},\mathsf{M}_{(2,j)},\dots,\mathsf{M}_{(k,j)})\right)$ condition on $\left(\mathsf{M}_{(\le k,j-1)}=m'_{(\le k,j-1)},\ \mathsf{M}_{<k}=m'_{<k},\ X_j=x'_j\right)$
  $im_j\leftarrow(m'_{(1,j)},m'_{(2,j)},\dots,m'_{(k,j)},m'_0,m'_1,\dots,m'_{k-1})$
  $o_j\leftarrow(im_j,apr_j)$
end for
Output: $o_n=(im_n,apr_n)=(m'_{(\le k,n)},m'_{<k},apr_n)$

computes the majority bit. Recall that, M is a k-pass algorithm such that $\mathrm{H}(\mathsf{M}_i)\le n^\lambda$ for all $i\in\{0,\dots,k\}$. We immediately have the following corollary.

Corollary 2.6. *As $\mathrm{H}(\mathsf{M}_i)\le n^\lambda,\forall i\in\{0,\dots,k\}$,*

$$\mathbb{E}_{a\sim\mathcal{D}}\left[\sum_{j=1}^{n}\mathbf{1}_{a_j\neq 0}\right]=\mathbb{E}\left[\sum_{j=1}^{n}\mathbf{1}_{Y_j\neq X'_j}\right]\le\sqrt{n}\cdot\sqrt{kn^\lambda}.$$

Corollary 2.6 suggests a value for parameter B that Algorithm 3 should run Algorithm Apr on, so as to use approximation guarantees from Claim 2.4. Let O be Algorithm 3 with parameters $\gamma=\frac{\varepsilon}{10}$

and $B = \sqrt{k \cdot n^{1+\lambda}}$. We prove the following lemmas regarding information cost and output of algorithm O. See the full version for detailed proofs of these lemmas.

Lemma 2.7. *For all* $\varepsilon > \frac{100}{\sqrt{n}}$, $\lambda > 0$, $IC(\mathsf{O}) \le MIC_{cond}(\mathsf{M}) + n \cdot \left(50 + 6\log\log n + \log\left(\frac{k \cdot n^{\lambda}}{\varepsilon^2}\right)\right)$.

Once we have $IC(\mathsf{Im}) \le MIC_{cond}(\mathsf{M})$ and Proposition 2.5 in place, Lemma 2.7 follows from careful disentanglement of the information costs for subroutines Im and Apr used in Algorithm O.

Lemma 2.8. *For all* $\varepsilon > \frac{100}{\sqrt{n}}$,

$$\mathbb{E}_{\mathsf{M}_{(k,n)}}\left(\mathbb{E}\left[\sum_{j=1}^{n} X_j \middle| \mathsf{M}_{(k,n)} = m_{(k,n)}\right]\right)^2 \ge \varepsilon n$$

$$\implies \mathbb{E}_{\mathsf{O}_n}\left(\mathbb{E}\left[\sum_{j=1}^{n} Y_j \middle| \mathsf{O}_n = o_n\right]\right)^2 \ge \frac{\varepsilon}{2} n.$$

Intuitively, Lemma 2.8 shows that if output of the k-pass algorithm M gives information about $\sum_j X_j$ (measured by reduction in the variance), then the output of one-pass algorithm O also gives information about $\sum_j Y_j$ – sum of the input stream to O. The former guarantee implies the output of O contains information about $\sum_j X'_j$ (the modified input); as O stores an approximation for $\sum_j (Y_j - X'_j)$, this implies that it also has information about $\sum_j Y_j$. All that remains to show is that the approximation for modification has an additive error of at most $O(\varepsilon\sqrt{n})$, with high probability. For this, we use $\mathbb{E}_{a\sim\mathcal{D}}\left[\sum_{j=1}^{n} \mathbf{1}_{a_j \neq 0}\right] \le \sqrt{n \cdot \mathrm{H}(\mathsf{M}_{<k})}$ and ℓ_2 approximation guarantee for Apr from Theorem 2.4.

2.2.4 Solving Multiple Instances of the Coin Problem. We generalize our multi-pass streaming lower bounds to solving multiple instances of the coin problem simultaneously. Informally, given t interleaved input streams generated by n *i.i.d.* uniform bits each, the goal of a multi-pass streaming algorithm is to output the majority of an arbitrary stream at the end of k passes. We show that any k-pass streaming algorithm that solves the t-Coins Problem requires $\Omega(\frac{t \log n}{k})$ bits of memory (for $t < n^{\delta}$). As for the single coin case, we reduce the multiple coin case to the analogous result for one-pass streaming algorithms proven by [9]. We simulate the multi-pass algorithm for the t-Coins Problem using a one-pass algorithm that maintains t approximations for modifying each input stream to a valid stream for the k-pass algorithm. For the generalization, we utilize the fact that the single coin simulation works even when the output of the first pass has poly(n) entropy; for the t-Coins Problem problem, we work with memories as large as $t \log n$.

2.3 Multi-pass Streaming Lower Bound for the Needle Problem

Since we have shown that $MIC(\mathsf{M}, \boldsymbol{D}_0)$ is upper bounded by $2ksn$, it suffices to give an $\Omega(1/p^2)$ lower bound for $MIC(\mathsf{M}, \boldsymbol{D}_0)$ as we formally present in Lemma 2.9. In the following, we give the intuition behind Lemma 2.9.

In the needle problem, we use the notion $MIC(\mathsf{M}, \boldsymbol{D}_0)$ *as we defined before, where* $\boldsymbol{D}_0$ *stands for the uniform distribution.* For simplicity, we write $MIC(\mathsf{M})$ in the needle problem, and it could be easily distinguished from the notion $MIC_{cond}(\mathsf{M})$ used in the coin problem.

Algorithm 4: Communication Protocol For MostlyEq

Input: $z \in [t]^m$
Output: ans $\in \{0, 1\}$
Recall: $S = \{p_1, \ldots, p_m\}$
for player j from 1 to m **do**
  let $X_{p_j} = z_j$
  uniformly sample $X_{p_j+1}, \cdots, X_{p_{j+1}-1}$ from $[t]$
end for
Player m simulates $\mathsf{M}_{(1,p_1-1)} = \mathsf{M}(X_1, \ldots, X_{p_1-1})$ and sends $\mathsf{M}_{(1,p_1-1)}$ to Player 1
for i from 1 to $k-1$ **do**
  for j from 1 to m **do**
    Player j simulates $\mathsf{M}_{(i,p_{j+1}-1)} = \mathsf{M}(\mathsf{M}_{i,p_j-1}, X_{p_j}, \ldots, X_{p_{j+1}-1})$
    Player j sends $\mathsf{M}_{(i,p_{j+1}-1)}$ to Player $j+1$ (send to Player 1 when $j = m$)
  end for
end for
return the output of Player m

Lemma 2.9. *In the needle problem, if a* k*-pass streaming algorithm* M *distinguishes between* $\boldsymbol{D}_0$ *and* $\boldsymbol{D}_1$ *with high probability, then we have* $MIC(\mathsf{M}) = \Omega(1/p^2)$.

Let us first consider the special case when $p = 1/2$. A useful observation is that the needle problem with $p = 1/2$ is very similar to the MostlyDISJ communication problem [35]. Viewing the needle problem with $p = 1/2$ as a multiparty communication problem (we name it MostlyEq), we have the following definition:

Definition 2.10 (m-party MostlyEq problem). There are m parties in the communication problem, where the i-th party holds an integer $z_i \in [t]$. We promise that $(z_1, \ldots, z_m)$ are sampled from either of the following distributions :

(1) Uniform distribution (denoted by $\boldsymbol{P}_U$): each z_i is sampled from $[t]$ independently and uniformly.
(2) Mostly equal distribution (denoted by $\boldsymbol{P}_{Eq}$): first uniformly sample an element α (needle) from $[t]$. Then each z_i independently with probability $1/2$ equals α, and uniform otherwise.

The goal of the players is to distinguish which case it is.

The MostlyEq problem and the needle problem with $p = 1/2$ are closely related. For the MostlyEq problem, we prove a information complexity lower bound, formalized by the following theorem:

Theorem 2.11. *For any communication protocol* Π *that solves the* m*-party, where* $m \le t/100$*, MostlyEq problem with failure probability smaller than* 0.1*, we have that,*

$$\mathrm{I}\left(\Pi(\boldsymbol{P}_U); \boldsymbol{P}_U\right) = \Omega(1),$$

In other words, the information complexity of Π *is* $\Omega(1)$.

Here, the failure probability for a protocol Π is defined by

$$\Pr[\Pi(\boldsymbol{P}_U) = 1] + \Pr[\Pi(\boldsymbol{P}_{Eq}) = 0].$$

By a standard reduction to MostlyEq (constructing a communication protocol by simulation streaming algorithm), we know the mutual information $\mathrm{I}(\mathsf{M}; X)$ between $\mathsf{M} = (\mathsf{M}_{(i,j)})_{i\in[k],j\in[n]}$ and input $X = (X_1, \cdots, X_n)$ is also $\Omega(1)$. Then, we can prove that $MIC(\mathsf{M}) \geq \mathrm{I}(\mathsf{M}, X) \geq \Omega(1)$ with information theory calculations.

Now, let us consider general $p \leq 1/2$. We first use decompose the needle problem into many *local needle problems* by redefining the sampling process of $\boldsymbol{D}_1$ as follows:

(1) Sample a set $S \subseteq [n]$ with each element $j \in [n]$ contained in S independently with probability $2p$.
(2) Uniformly sample a needle $\alpha \in [t]$.
(3) For each $j \notin S$, the jth streaming sample is uniformly random.
(4) For each $j \in S$, the jth streaming sample equals to α with probability $1/2$ and uniformly random otherwise.

It is easy to see that the data stream sampled by the process above follows $\boldsymbol{D}_1$. Thus, solving the needle problem with general p is equivalent to solving the needle problem with $p = 1/2$ hiding in a secret location S. Then, we define *local needle distribution* $\boldsymbol{D}^S$ as the distribution $\boldsymbol{D}_1$ condition on that the set sampled in Step (1) equals S, and define *local needle problem* as distinguishing between $\boldsymbol{D}^S$ and $\boldsymbol{D}_0$ within a small error. Since the streaming algorithm M does not know S, if M solves the needle problem for general p, M must distinguish between $\boldsymbol{D}^S$ and $\boldsymbol{D}_0$ for at least a constant faction of S.

If M solves the local needle problem with $S = \{p_1, p_2, \cdots, p_m\}$, then by a reduction shown in Algorithm 4 and Theorem 2.11, it holds that

$$\mathrm{I}\left(\{\mathsf{M}_{(i,p_j-1)}\}; \{X_{p_\ell}\}\right) = \Omega(1).$$

This comes from viewing the state of the streaming algorithms as the transcripts of the communication protocols. In addition, further information theory calculations show that

$$\sum_{i=1}^{k-1}\sum_{j=1}^{m}\sum_{\ell=1}^{i} \mathrm{I}(\mathsf{M}_{(i,p_{j+1}-1)}; X_{p_\ell} \mid \mathsf{M}_{(\leq i,p_\ell-1)}, \mathsf{M}_{(<i,p_{j+1}-1)}) \geq \Omega(1).$$

where we define p_{m+1} as p_1. Finally, by taking an expectation over S, the L.H.S. of the inequality above is about $MIC(\mathsf{M})/O(p^2)$ since each term

$$\mathrm{I}(\mathsf{M}_{(i,j)}; X_\ell \mid \mathsf{M}_{(\leq i,\ell-1)}, \mathsf{M}_{(\leq i-1,j)})$$

or

$$\mathrm{I}(\mathsf{M}_{(i,j)}; X_\ell \mid \mathsf{M}_{(\leq i-1,\ell-1)}, \mathsf{M}_{(\leq i-1,j)})$$

of $MIC(\mathsf{M})$ appears in the L.H.S. for an $O(p^2)$ fraction of S. Consequently, we have $MIC(\mathsf{M}) = \Omega(1/p^2)$ as desired.

2.4 Algorithms for the Needle Problem

We introduce the idea behind the following two algorithms in this section: (1) M_1, which solves the needle problem with $p \geq 1/\sqrt{n}$ in $O(\log\log n(\log\log\log n))$ space in the communication model; (2) M_2, which solves the needle problem with $p \leq 1/\sqrt{n\log^3 n}$ in $O(1/(p^2 t))$ space in the streaming model. As with previous work [9, 11, 52] for finding ℓ_2-heavy hitters we partition the stream items into contiguous groups (in previous work, these groups contain $\Theta(\sqrt{n})$ stream items). These works sample $O(1)$ items in each group and track them over a small number of future groups - this is sometimes called *pick and drop sampling*. A major difference between our algorithms and these is that we cannot afford to store the identity of an item and track it, as that would require messages of length at least $\log n$ bits. A natural idea is to instead track a small hash of an item but there will be a huge number of collisions throughout the stream if we use fewer than $\log n$ bits.

We start by choosing each group to be of size $\Theta(1/p)$, so each group has one occurrence of the needle with constant probability under distribution $\boldsymbol{D}_1$. We discuss the algorithm M_1 for $p = 1/\sqrt{n}$ first, so the group size is $\sqrt{n}$. This can be generalized to any $p \geq 1/\sqrt{n}$ (see the full version). For universe $[t]$, we randomly sample a hash function projecting $[t]$ to $[C_2]$, where C_2 is a constant. For each group, we randomly sample a subset of $[t]$ with size $C_1 t/\sqrt{n}$. Then, M_1 runs in $\sqrt{n}$ rounds, and processes one data group in each round. In round i, we set C_2 new counters: for each $j \in [C_2]$, we set a counter that tracks j; and when processing the following groups, we check if each element x exists in i's random subset; if it is in, we update its corresponding counter (the counter of group i with the hash value of x). After processing each group, we check each counter to see if it is at least a constant times the number of groups processed after it was initialized. If not, we drop it. The intuition is that (1) a counter that does not track the needle survives r rounds with probability less than e^{-cr}, and (2) a counter tracking the needle has constant probability to survive. We may accumulate many counters and the space may hit our $O((\log\log n)(\log\log\log n))$ bound - if so, we throw away all counters and start over. We show that at the critical time when we start processing the needle we will not throw it away.

For the second algorithm M_2, the idea is to divide the domain $[t]$ into $1/(p^2 n)$ *blocks* and simultaneously run $1/(p^2 n)$ algorithms similar to M_1 on each block, checking if the needle exists. We show we only need $O(1/(p^2 n))$ space in total when $p \leq 1/\sqrt{n\log^3 n}$ holds. Note that this algorithm is in the standard streaming model since we can afford an additive $\log n$ bit counter to track the index of current group.

3 FUTURE DIRECTIONS

Using the single-pass notion of information complexity in [9], that we extend to multiple passes, Brown, Bun and Smith [12] showed single-pass streaming lower bounds for several learning problems. A natural question is if our multi-pass techniques can be useful for learning problems. Another potential application is that of Dinur [18], who shows streaming lower bounds for distinguishing random functions from random permutations. Also, Kamath et al. [35] study the heavy hitters problem for $O(1)$ pass algorithms. These results have a logarithmic factor gap and use more classical notions of information complexity. Can our techniques apply here?

ACKNOWLEDGMENTS

The authors thank three anonymous STOC reviewers for their helpful suggestions on this paper.

Mark Braverman is supported in part by the NSF Alan T. Waterman Award, Grant No. 1933331, a Packard Fellowship in Science and Engineering, and the Simons Collaboration on Algorithms and Geometry.

Qian Li is supported by Hetao Shenzhen-Hong Kong Science and Technology Innovation Cooperation Zone Project (No.HZQSWS-KCCYB-2024016).

David P. Woodruff's research is supported in part by a Simons Investigator Award, and part of this work was done while visiting the Simons Institute for the Theory of Computing.

Jiapeng Zhang's research is supported by NSF CAREER award 2141536.

REFERENCES

[1] Ishaq Aden-Ali, Yanjun Han, Jelani Nelson, and Huacheng Yu. 2022. On the amortized complexity of approximate counting. *CoRR* abs/2211.03917 (2022).

[2] Yuqing Ai, Wei Hu, Yi Li, and David P. Woodruff. 2016. New Characterizations in Turnstile Streams with Applications. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan (LIPIcs, Vol. 50)*, Ran Raz (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 20:1–20:22.

[3] Miklós Ajtai, Vladimir Braverman, T. S. Jayram, Sandeep Silwal, Alec Sun, David P. Woodruff, and Samson Zhou. 2022. The White-Box Adversarial Data Stream Model. In *PODS '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Leonid Libkin and Pablo Barceló (Eds.). ACM, 15–27. https://doi.org/10.1145/3517804.3526228

[4] Noga Alon, Yossi Matias, and Mario Szegedy. 1999. The Space Complexity of Approximating the Frequency Moments. *J. Comput. System Sci.* 58, 1 (1999), 137–147. https://doi.org/10.1006/jcss.1997.1545

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

[6] Alexandr Andoni, Andrew McGregor, Krzysztof Onak, and Rina Panigrahy. 2008. Better Bounds for Frequency Moments in Random-Order Streams. *CoRR* abs/0808.2222 (2008). arXiv:0808.2222 http://arxiv.org/abs/0808.2222

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

[8] Lakshminath Bhuvanagiri, Sumit Ganguly, Deepanjan Kesh, and Chandan Saha. 2006. Simpler algorithm for estimating frequency moments of data streams. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. 708–713.

[9] Mark Braverman, Sumegha Garg, and David P Woodruff. 2020. The coin problem with applications to data streams. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science*. IEEE, 318–329.

[10] Mark Braverman, Sumegha Garg, and Or Zamir. 2021. Tight Space Complexity of the Coin Problem. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 1068–1079. https://doi.org/10.1109/FOCS52979.2021.00106

[11] Vladimir Braverman, Jonathan Katzman, Charles Seidell, and Gregory Vorsanger. 2014. An optimal algorithm for large frequency moments using $O(n^{1-2/k})$ bits. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[12] Gavin Brown, Mark Bun, and Adam Smith. 2022. Strong memory lower bounds for learning natural models. In *Conference on Learning Theory*. PMLR, 4989–5029.

[13] Amit Chakrabarti, Graham Cormode, and Andrew McGregor. 2008. Robust Lower Bounds for Communication and Stream Computation. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing* (Victoria, British Columbia, Canada) *(STOC '08)*. Association for Computing Machinery, New York, NY, USA, 641–650. https://doi.org/10.1145/1374376.1374470

[14] Amit Chakrabarti, T. S. Jayram, and Mihai Pundefinedtraşcu. 2008. Tight Lower Bounds for Selection in Randomly Ordered Streams. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms* (San Francisco, California) *(SODA '08)*. Society for Industrial and Applied Mathematics, USA, 720–729.

[15] A. Chakrabarti, S. Khot, and Xiaodong Sun. 2003. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *18th IEEE Annual Conference on Computational Complexity, 2003. Proceedings.* 107–117. https://doi.org/10.1109/CCC.2003.1214414

[16] Michael S. Crouch, Andrew McGregor, Gregory Valiant, and David P. Woodruff. 2016. Stochastic Streams: Sample Complexity vs. Space Complexity. In *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark (LIPIcs, Vol. 57)*, Piotr Sankowski and Christos D. Zaroliagis (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 32:1–32:15. https://doi.org/10.4230/LIPIcs.ESA.2016.32

[17] Ilias Diakonikolas, Themis Gouleakis, Daniel M Kane, and Sankeerth Rao. 2019. Communication and memory efficient testing of discrete distributions. In *Conference on Learning Theory*. PMLR, 1070–1106.

[18] Itai Dinur. 2020. On the Streaming Indistinguishability of a Random Permutation and a Random Function. In *Advances in Cryptology – EUROCRYPT 2020*, Anne Canteaut and Yuval Ishai (Eds.). Springer International Publishing, Cham, 433–460.

[19] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. 2016. Memory-efficient algorithms for finding needles in haystacks. In *Advances in Cryptology–CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*. Springer, 185–206.

[20] Alireza Farhadi, MohammadTaghi Hajiaghayi, Tung Mai, Anup Rao, and Ryan A. Rossi. 2020. Approximate Maximum Matching in Random Streams. In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms* (Salt Lake City, Utah) *(SODA '20)*. Society for Industrial and Applied Mathematics, USA, 1773–1785.

[21] Philippe Flajolet. 1985. Approximate Counting: A Detailed Analysis. *BIT* 25, 1 (1985), 113–134. https://doi.org/10.1007/BF01934993

[22] Sumit Ganguly. 2011. Polynomial estimators for high frequency moments. *arXiv preprint arXiv:1104.4552* (2011).

[23] Sumit Ganguly and David P. Woodruff. 2018. High Probability Frequency Moment Sketches. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 107)*, Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella (Eds.). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 58:1–58:15. https://doi.org/10.4230/LIPIcs.ICALP.2018.58

[24] Anna C. Gilbert, Yi Li, Ely Porat, and Martin J. Strauss. 2017. For-All Sparse Recovery in Near-Optimal Time. *ACM Trans. Algorithms* 13, 3 (2017), 32:1–32:26. https://doi.org/10.1145/3039872

[25] André Gronemeier. 2010. *Information Complexity and Data Stream Algorithms for Basic Problems*. Ph. D. Dissertation. Technical University Dortmund, Germany. https://hdl.handle.net/2003/27529

[26] Sudipto Guha and Zhiyi Huang. 2009. Revisiting the Direct Sum Theorem and Space Lower Bounds in Random Order Streams. In *Automata, Languages and Programming*, Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris Nikoletseas, and Wolfgang Thomas (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 513–524.

[27] Sudipto Guha and Andrew McGregor. 2007. Space-efficient sampling. In *Artificial Intelligence and Statistics*. PMLR, 171–178.

[28] Sudipto Guha and Andrew McGregor. 2009. Stream Order and Order Statistics: Quantile Estimation in Random-Order Streams. *SIAM J. Comput.* 38, 5 (2009), 2044–2059. https://doi.org/10.1137/07069328X arXiv:https://doi.org/10.1137/07069328X

[29] Piotr Indyk, Eric Price, and David P. Woodruff. 2011. On the Power of Adaptivity in Sparse Recovery. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, Rafail Ostrovsky (Ed.). IEEE Computer Society, 285–294.

[30] Piotr Indyk and David Woodruff. 2005. Optimal Approximations of the Frequency Moments of Data Streams. In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing* (Baltimore, MD, USA) *(STOC '05)*. Association for Computing Machinery, New York, NY, USA, 202–208. https://doi.org/10.1145/1060590.1060621

[31] Joseph Jaeger and Stefano Tessaro. 2019. Tight Time-Memory Trade-Offs for Symmetric Encryption. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11476)*, Yuval Ishai and Vincent Rijmen (Eds.). Springer, 467–497. https://doi.org/10.1007/978-3-030-17653-2_16

[32] Rajesh Jayaram and David P. Woodruff. 2018. Data Streams with Bounded Deletions. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, June 10-15, 2018*, Jan Van den Bussche and Marcelo Arenas (Eds.). ACM, 341–354.

[33] Rajesh Jayaram and David P. Woodruff. 2023. Towards Optimal Moment Estimation in Streaming and Distributed Models. *ACM Trans. Algorithms* 19, 3 (2023), 27:1–27:35.

[34] T. S. Jayram. 2009. Hellinger Strikes Back: A Note on the Multi-party Information Complexity of AND. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, Irit Dinur, Klaus Jansen, Joseph Naor, and José Rolim (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 562–573.

[35] Akshay Kamath, Eric Price, and David P. Woodruff. 2021. A Simple Proof of a New Set Disjointness with Applications to Data Streams. In *Proceedings of the 36th Computational Complexity Conference* (Virtual Event, Canada) *(CCC '21)*. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, DEU, Article lipics. https://doi.org/10.4230/LIPIcs.CCC.2021.37

[36] Daniel M. Kane, Jelani Nelson, and David P. Woodruff. 2010. On the Exact Space Complexity of Sketching and Streaming Small Norms. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, Moses Charikar (Ed.). SIAM, 1161–1178. https://doi.org/10.1137/1.9781611973075.93

[37] Shachar Lovett and Jiapeng Zhang. 2023. Streaming Lower Bounds and Asymmetric Set-Disjointness. In *2023 IEEE 64th Annual Symposium on Foundations*

of Computer Science (FOCS). IEEE Computer Society, Los Alamitos, CA, USA, 871–882. https://doi.org/10.1109/FOCS57990.2023.00056

[38] Andrew McGregor, A. Pavan, Srikanta Tirthapura, and David Woodruff. 2012. Space-Efficient Estimation of Statistics over Sub-Sampled Streams. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems* (Scottsdale, Arizona, USA) *(PODS '12)*. Association for Computing Machinery, New York, NY, USA, 273–282. https://doi.org/10.1145/2213556.2213594

[39] Andrew McGregor, A. Pavan, Srikanta Tirthapura, and David P. Woodruff. 2016. Space-Efficient Estimation of Statistics Over Sub-Sampled Streams. *Algorithmica* 74, 2 (2016), 787–811. https://doi.org/10.1007/s00453-015-9974-0

[40] Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. 1995. On data structures and asymmetric communication complexity. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*. 103–111.

[41] Marco Molinaro, David P. Woodruff, and Grigory Yaroslavtsev. 2013. Beating the Direct Sum Theorem in Communication Complexity with Implications for Sketching. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, Sanjeev Khanna (Ed.). SIAM, 1738–1756. https://doi.org/10.1137/1.9781611973105.125

[42] Morteza Monemizadeh and David P. Woodruff. 2010. 1-Pass Relative-Error Lp-Sampling with Applications. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms* (Austin, Texas) *(SODA '10)*. Society for Industrial and Applied Mathematics, USA, 1143–1160.

[43] Vasileios Nakos, Xiaofei Shi, David P. Woodruff, and Hongyang Zhang. 2018. Improved Algorithms for Adaptive Compressed Sensing. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic (LIPIcs, Vol. 107)*, Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 90:1–90:14.

[44] Jelani Nelson and Huacheng Yu. 2022. Optimal Bounds for Approximate Counting. In *PODS '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Leonid Libkin and Pablo Barceló (Eds.). ACM, 119–127. https://doi.org/10.1145/3517804.3526225

[45] Eric Price and David P. Woodruff. 2011. (1 + eps)-Approximate Sparse Recovery. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, Rafail Ostrovsky (Ed.). IEEE Computer Society, 295–304.

[46] Eric Price and David P. Woodruff. 2013. Lower Bounds for Adaptive Sparse Recovery. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, Sanjeev Khanna (Ed.). SIAM, 652–663.

[47] Ran Raz. 2016. Fast Learning Requires Good Memory: A Time-Space Lower Bound for Parity Learning. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, Irit Dinur (Ed.). IEEE Computer Society, 266–275. https://doi.org/10.1109/FOCS.2016.36

[48] Vatsal Sharan, Aaron Sidford, and Gregory Valiant. 2019. Memory-Sample Tradeoffs for Linear Regression with Small Error. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (Phoenix, AZ, USA) *(STOC 2019)*. Association for Computing Machinery, New York, NY, USA, 890–901. https://doi.org/10.1145/3313276.3316403

[49] Robert H. Morris Sr. 1978. Counting Large Numbers of Events in Small Registers. *Commun. ACM* 21, 10 (1978), 840–842. https://doi.org/10.1145/359619.359627

[50] Stefano Tessaro and Aishwarya Thiruvengadam. 2018. Provable Time-Memory Trade-Offs: Symmetric Cryptography Against Memory-Bounded Adversaries. In *Theory of Cryptography (Theory of Cryptography, Vol. 11239)*. Springer, 3–32. https://doi.org/10.1007/978-3-030-03807-6_1

[51] Emanuele Viola. 2015. The communication complexity of addition. *Combinatorica* 35 (2015), 703–747.

[52] David P. Woodruff and Samson Zhou. 2021. Separations for Estimating Large Frequency Moments on Data Streams. In *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference) (LIPIcs, Vol. 198)*, Nikhil Bansal, Emanuela Merelli, and James Worrell (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 112:1–112:21.

Received 10-NOV-2023; accepted 2024-02-11