Non-Mergeable Sketching for Cardinality Estimation

Seth Pettie ✉
University of Michigan, Ann Arbor, MI, USA

Dingyu Wang ✉
University of Michigan, Ann Arbor, MI, USA

Longhui Yin ✉
Tsinghua University, Beijing, China

Abstract

Cardinality estimation is perhaps the simplest non-trivial statistical problem that can be solved via sketching. Industrially-deployed sketches like HyperLogLog, MinHash, and PCSA are *mergeable*, which means that large data sets can be sketched in a distributed environment, and then merged into a single sketch of the whole data set. In the last decade a variety of sketches have been developed that are *non-mergeable*, but attractive for other reasons. They are *simpler*, their cardinality estimates are *strictly unbiased*, and they have substantially *lower variance*.

We evaluate sketching schemes on a reasonably level playing field, in terms of their *memory-variance product* (MVP). E.g., a sketch that occupies $5m$ bits and whose relative variance is $2/m$ (standard error $\sqrt{2/m}$) has an MVP of 10. Our contributions are as follows.

- Cohen [14] and Ting [35] independently discovered what we call the *Martingale transform* for converting a mergeable sketch into a non-mergeable sketch. We present a simpler way to analyze the limiting MVP of Martingale-type sketches.
- Pettie and Wang proved that the Fishmonger sketch [31] has the best MVP, $H_0/I_0 \approx 1.98$, among a class of mergeable sketches called "linearizable" sketches. (H_0 and I_0 are precisely defined constants.) We prove that the Martingale transform is optimal in the non-mergeable world, and that Martingale Fishmonger in particular is optimal among linearizable sketches, with an MVP of $H_0/2 \approx 1.63$. E.g., this is circumstantial evidence that to achieve 1% standard error, we cannot do better than a 2 kilobyte sketch.
- Martingale Fishmonger is neither simple nor practical. We develop a new mergeable sketch called Curtain that strikes a nice balance between simplicity and efficiency, and prove that Martingale Curtain has limiting MVP ≈ 2.31. It can be updated with $O(1)$ memory accesses and it has lower empirical variance than Martingale LogLog, a practical non-mergeable version of HyperLogLog.

2012 ACM Subject Classification Theory of computation → Sketching and sampling

Keywords and phrases Cardinality Estimation, Sketching

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.104

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2008.08739`

Funding This work was supported by NSF grants CCF-1637546 and CCF-1815316.

1 Introduction

Cardinality estimation[1] is a fundamental problem in streaming and sketching with diverse applications in databases [12, 21], network monitoring [5, 8, 39, 11], nearest neighbor search [33], caching [37], and genomics [30, 17, 38, 2]. In the *sequential* setting of this problem, we receive the elements of a multiset $\mathcal{A} = \{a_1, a_2, \ldots, a_N\}$ one at a time. We

[1] (aka F_0 estimation or *Distinct Elements*)

48th International Colloquium on Automata, Languages, and Programming (ICALP 2021).
Editors: Nikhil Bansal, Emanuela Merelli, and James Worrell; Article No. 104; pp. 104:1–104:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

maintain a small *sketch* S of the elements seen so far, such that the true cardinality $\lambda = |\mathcal{A}|$ is estimated by some $\hat{\lambda}(S)$. The ***distributed*** setting is similar, except that $\mathcal{A}$ is partitioned arbitrarily among several machines, the shares being sketched separately and combined into a sketch of $\mathcal{A}$. Only ***mergeable*** sketches are deployed in distributed settings; see Definition 2 below.

▶ **Definition 1.** *In the* RANDOM ORACLE MODEL $\mathcal{A} \subseteq [U]$ *and we have oracle access to a uniformly random permutation* $h : [U] \to [U]$ *(or a uniformly random hash function* $h : [U] \to [0, 1]$*). In the* STANDARD MODEL *we can generate random bits as necessary, but must explicitly store any hash functions in the sketch.*

▶ **Definition 2.** *Suppose* $\mathcal{A}^{(1)}, \mathcal{A}^{(2)}$ *are multisets such that* $\mathcal{A} = \mathcal{A}^{(1)} \cup \mathcal{A}^{(2)}$*. A sketching scheme is* **mergeable** *if, whenever,* $\mathcal{A}^{(1)}, \mathcal{A}^{(2)}$ *are sketched as* $S^{(1)}, S^{(2)}$ *(using the same* RANDOM ORACLE h *or the same source of random bits in the* STANDARD MODEL*), the sketch* S *of* $\mathcal{A}$ *can be computed from* $S^{(1)}, S^{(2)}$ *alone.*

STANDARD MODEL sketches [1, 3, 4, 6, 22, 27] usually make an (ϵ, δ)-guarantee, i.e.,

$$\Pr\left(\hat{\lambda} \notin [(1-\epsilon)\lambda, (1+\epsilon)\lambda]\right) < \delta.$$

The state-of-the-art STANDARD MODEL sketch [6, 27] uses $O(\epsilon^{-2} \log \delta^{-1} + \log U)$ bits, which is optimal at this level of specificity, as it meets the space lower bounds of $\Omega(\log U)$, $\Omega(\epsilon^{-2})$ (when $\delta = \Theta(1)$), and $\Omega(\epsilon^{-2} \log \delta^{-1})$ [1, 25, 26]. However, the leading constants hidden by [6, 27] are quite large.

In the RANDOM ORACLE MODEL the cardinality estimate $\hat{\lambda}$ typically has negligible bias, and errors are expressed in terms of the *relative variance* $\lambda^{-2} \cdot \mathrm{Var}(\hat{\lambda} \mid \lambda)$ or *relative standard deviation* $\lambda^{-1}\sqrt{\mathrm{Var}(\hat{\lambda} \mid \lambda)}$, also called the *standard error*. Sketches that use $\Omega(m)$ bits typically have relative variances of $O(1/m)$. Thus, the most natural way to measure the quality of the *sketching scheme* itself is to look at its limiting *memory-variance product* (MVP), i.e., the product of its memory and variance as $m \to \infty$.

Until about a decade ago, all STANDARD/RANDOM ORACLE sketches were mergeable, and suitable to both distributed and sequential applications. For reasons that are not clear to us, the idea of ***non-mergeable*** sketching was discovered independently by multiple groups [10, 24, 14, 35] at about the same time, and quite *late* in the 40-year history of cardinality estimation. Chen, Cao, Shepp, and Nguyen [10] invented the S-Bitmap in 2011, followed by Helmi, Lumbroso, Martínez, and Viola's[24] Recordinality in 2012. In 2014 Cohen [14] and Ting [35] independently invented what we call the *Martingale transform*, which is a simple, mechanical way to transform any mergeable sketch into a (better) non-mergeable sketch.[2]

In a companion paper [31], we analyzed the MVPs of *mergeable* sketches under the assumption that the sketch was compressed to its entropy bound. Fishmonger (an entropy compressed variant of PCSA with a different estimator function) was shown to have MVP $= H_0/I_0 \approx 1.98$, where

$$H_0 = (\ln 2)^{-1} + \sum_{k=1}^{\infty} k^{-1} \log_2(1 + 1/k) \qquad \text{and} \quad I_0 = \zeta(2) = \pi^2/6.$$

[2] Cohen [14] called these *Historical Inverse Probability* (HIP) sketches and Ting [35] applied the prefix *Streaming* to emphasize that they can be used in the single-stream setting, not the distributed setting.

Furthermore, H_0/I_0 was shown to be the minimum MVP among *linearizable* sketches, a subset of *mergeable* sketches that includes all the popular sketches (HyperLogLog, PCSA, MinHash, etc.).

Our aim in *this* paper is to build a useful framework for designing and analyzing *non-mergeable* sketching schemes, and, following [31], to develop a theory of space-variance optimality in the non-mergeable world. We work in the RANDOM ORACLE MODEL. Our results are as follows.

- Although the *Martingale transform* itself is simple, analyzing the variance of these sketches is not. For example, Cohen [14] and Ting [35] estimated the standard error of Martingale LogLog to be about $\approx \sqrt{3/(4m)} \approx 0.866/\sqrt{m}$ and about $\approx 1/(2\alpha_m m)$, respectively, where the latter tends to $\sqrt{\ln 2/m} \approx 0.8326/\sqrt{m}$ as $m \to \infty$.[3] We give a general method for determining the limiting relative variance of Martingale sketches that is strongly influenced by Ting's perspective.
- What is the most efficient (smallest MVP) non-mergeable sketch for cardinality estimation? The best Martingale sketches perform better than the *ad hoc* non-mergeable S-Bitmap and Recordinality, but perhaps there is a completely different, better way to systematically build non-mergeable sketches. We prove that up to some natural assumptions[4] the best non-mergeable sketch is a Martingale X sketch, for some X. Furthermore, we prove that Martingale Fishmonger, having MVP of $H_0/2 \approx 1.63$, is optimal among all Martingale X sketches, where X is *linearizable*. This provides some circumstantial evidence that Martingale Fishmonger is optimal, and that if we want, say, 1% standard error, we need to use a $H_0/2 \cdot (0.01)^{-2}$-bit sketch, ≈ 2 kilobytes.
- Martingale Fishmonger has an attractive MVP, but it is slow and cumbersome to implement. We propose a new mergeable sketch called Curtain that is "naturally" space efficient and easy to update in $O(1)$ memory accesses, and prove that Martingale Curtain has a limiting MVP ≈ 2.31.

1.1 Prior Work: Mergeable Sketches

Let S_i be the state of the sketch after processing $(a_1, \ldots, a_i)$.

The state of the PCSA sketch [20] is a 2D matrix $S \in \{0,1\}^{m \times \log U}$ and the hash function $h : [U] \to [m] \times \mathbb{Z}^+$ produces two indices: $h(a) = (j,k)$ with probability $m^{-1}2^{-k}$. $S_i(j,k) = 1$ iff $\exists i' \in [i].h(a_{i'}) = (j,k)$. Flajolet and Martin [20] proved that a certain estimator has standard error $0.78/\sqrt{m}$, making the MVP around $(0.78)^2 \log U \approx 0.6 \log U$.

Durand and Flajolet's LogLog sketch [16] consists of m counters. It interprets h exactly as in PCSA, and sets $S_i(j) = k$ iff k is maximum such that $\exists i' \in [i].h(a_{i'}) = (j,k)$. Durand and Flajolet's estimator is of the form $\hat{\lambda}(S) \propto m2^{m^{-1}\sum_j S(j)}$ and has standard error $\approx 1.3/\sqrt{m}$. Flajolet, Fusy, Gandouet, and Meunier's HyperLogLog [19] is the same sketch but with the estimator $\hat{\lambda}(S) \propto m^2(\sum_j 2^{-S(j)})^{-1}$. They proved that it has standard error tending to $\approx 1.04/\sqrt{m}$. As the space is $m \log\log U$ bits, the MVP is $\approx 1.08 \log\log U$.

The MinCount sketch (aka MinHash or Bottom-m [13, 15, 7]) stores the smallest m hash values, which we assume requires $\log U$ bits each. Using an appropriate estimator [23, 9, 29], the standard error is $1/\sqrt{m}$ and MVP $= \log U$.

[3] Here $\alpha_m = \left(m \int_0^\infty \left(\log_2\left(\frac{2+u}{1+u}\right)\right)^m du\right)^{-1}$ is the coefficient of Flajolet et al.'s HyperLogLog estimator.

[4] (the sketch is insensitive to duplicates, and the estimator is unbiased)

It is straightforward to see that the entropy of PCSA and LogLog are both $\Theta(m)$. Scheuermann and Mauve [34] experimented with entropy compressed versions of PCSA and HyperLogLog and found PCSA to be slightly superior. Rather than use the given estimators of [20, 19, 16], Lang [28] used Maximum Likelihood-type Estimators and found entropy-compressed PCSA to be significantly better than entropy-compressed LogLog (with MLE estimators). Pettie and Wang [31] defined the Fisher-Shannon (Fish)[5] number of a sketch as the ratio of its Shannon entropy (controlling its entropy-compressed size) to its Fisher information (controlling the variance of a statistically efficient estimator), and proved that the Fish-number of any base-q PCSA is H_0/I_0, and that the Fish-number of base-q LogLog is worse, but tends to H_0/I_0 in the limit as $q \to \infty$. (The constants H_0, I_0 were defined earlier.)

Table 1 A selection of results on composable sketches (top) and non-composable Martingale sketches (bottom) in terms of their limiting memory-variance product (MVP). Logarithms are base 2.

Mergeable Sketch		Limiting MVP	Notes
PCSA	[20]	$.6 \log U \approx 38.9$	For $U = 2^{64}$
LogLog	[16]	$1.69 \log\log U \approx 10.11$	For $U = 2^{64}$
MinCount	[23, 9, 29]	$\log U = 64$	For $U = 2^{64}$
HyperLogLog	[19]	$1.08 \log\log U \approx 6.48$	For $U = 2^{64}$
Fishmonger	[31]	$H_0/I_0 \approx 1.98$	
Non-Mergeable Sketch			
S-Bitmap	[10]	$O(\log^2(U/m))$	
Recordinality	[24]	$O(\log(\lambda/m)\log U)$	
Martingale PCSA	**new**	$0.35 \log U \approx 22.4$	For $U = 2^{64}$
Martingale LogLog	[14, 35]	$0.69 \log\log U \approx 4.16$	For $U = 2^{64}$
Martingale MinCount	[14, 35]	$0.5 \log U = 32$	For $U = 2^{64}$
Martingale Fishmonger	**new**	$H_0/2 \approx 1.63$	$H_0 = (\ln 2)^{-1} + \sum_{k>1} \frac{\log_2(1+1/k)}{k}$
Martingale Curtain	**new**	≈ 2.31	Theorem 4 with $(q, a, h) = (2.91, 2, 1)$
Non-Mergeable Lower Bound			
Martingale X	**new**	$\geq H_0/2$	X is a *linearizable* sketch

1.2 Prior Work: Non-Mergeable Sketches

Chen, Cao, Shepp, and Nguyen's S-Bitmap [10] consists of a bit string $S \in \{0,1\}^m$ and m known constants $0 \leq \tau_0 < \tau_1 < \cdots < \tau_{m-1} < 1$. It interprets $h(a) = (j, \rho) \in [m] \times [0,1]$ as an index j and real ρ and when processing a, sets $S(j) \leftarrow 1$ iff $\rho > \tau_{\mathrm{HammingWeight}(S)}$. One may confirm that S is insensitive to duplicates in the stream $\mathcal{A}$, but its state depends on the *order* in which $\mathcal{A}$ is scanned. By setting the τ-thresholds and estimator properly, the standard error is $\approx \ln(eU/m)/(2\sqrt{m})$ and MVP $= O(\log^2(U/m))$.

Recordinality [24] is based on MinCount; it stores (S, cnt), where S is the m smallest hash values encountered and cnt is the *number of times* that S has changed. The estimator looks only at cnt, not S, and has standard error $\approx \sqrt{\ln(\lambda/em)/m}$ and MVP $= O(\log(\lambda/m)\log U)$.

[5] Fish is essentially the same as MVP, under the assumption that the sketch state is compressed to its entropy.

Cohen [14] and Ting [35] independently described how to turn any sketch into a non-mergeable sketch using what we call the Martingale transform. Let S_i be the state of the original sketch after seeing $(a_1, \ldots, a_i)$ and $P_{i+1} = \Pr(S_{i+1} \neq S_i \mid S_i, a_{i+1} \notin \{a_1, \ldots, a_i\})$ be the probability that it changes state upon seeing a *new* element a_{i+1}.[6] The state of the Martingale sketch is $(S_i, \hat{\lambda}_i)$. Upon processing a_{i+1} it becomes $(S_{i+1}, \hat{\lambda}_{i+1})$, where

$$\hat{\lambda}_{i+1} = \hat{\lambda}_i + P_{i+1}^{-1} \cdot \left[\!\left[S_{i+1} \neq S_i \right]\!\right].$$

Here $\left[\!\left[\mathcal{E} \right]\!\right]$ is the indicator variable for the event $\mathcal{E}$. We assume the original sketch is insensitive to duplicates, so

$$\mathbb{E}(\hat{\lambda}_{i+1}) = \begin{cases} \hat{\lambda}_i & \text{when } a_{i+1} \in \{a_1, \ldots, a_i\} \text{ (and hence } S_{i+1} = S_i) \\ \hat{\lambda}_i + 1 & \text{when } a_{i+1} \notin \{a_1, \ldots, a_i\}. \end{cases}$$

Thus, with $\hat{\lambda}_0 = \lambda_0 = 0$, $\hat{\lambda}_i$ is an unbiased estimator of the true cardinality $\lambda_i = |\{a_1, \ldots, a_i\}|$ and $(\hat{\lambda}_i - \lambda_i)_i$ is a *martingale*. The Martingale-transformed sketch requires the same space, plus just $\log U$ bits to store the estimate $\hat{\lambda}$.

Cohen and Ting [14, 35] both proved that Martingale MinCount has standard error $1/(2\sqrt{m})$ and MVP $= (\log U)/2$. They gave different estimates for the standard error of Martingale LogLog. Ting's estimate is quite accurate, and tends to $\sqrt{\ln 2/m}$ as $m \to \infty$, giving it an MVP $= \ln 2 \log\log U \approx 0.69 \log\log U$.

▶ Remark 3. We call Martingale sketches *non-mergeable* because, in a distributed environment, there is no obvious way to merge the cardinality estimates $(\hat{\lambda})$. On the other hand, Ting [36] has shown that if $(S^A, \hat{\lambda}^A)$ and $(S^B, \hat{\lambda}^B)$ are Martingale MinCount sketches obtained by sequentially processing A and B, that $\hat{\lambda}^A, \hat{\lambda}^B$ carry useful information for estimating $|A \cup B|$ and $|A \cap B|$ beyond that contained in S^A, S^B.

1.3 The Dartboard Model

The ***dartboard model*** [31] is useful for describing cardinality sketches with a single, uniform language. The dartboard model is essentially the same as Ting's [35] *area cutting* process, but with a specific, discrete cell partition and state space fixed in advance.

The *dartboard* is the unit square $[0, 1]^2$, partitioned into a set $\mathcal{C} = \{c_0, \ldots, c_{|\mathcal{C}|-1}\}$ of *cells* of various sizes. Every cell may be either *occupied* or *unoccupied*; the *state* is the set of occupied cells and the state space some $\mathcal{S} \subseteq 2^{\mathcal{C}}$.

We process a stream of elements one by one; when a *new* element is encountered we throw a *dart* uniformly at random at the dartboard and update the state in response. The relationship between the state and the dart distribution satisfies two rules:

(R1) Every cell with at least one dart is occupied; occupied cells may contain no darts.

(R2) If a dart lands in an occupied cell, the state does not change.

As a consequence of (R1) and (R2), if a dart lands in an empty cell the state *must* change, and occupied cells may never become unoccupied. Dart throwing is merely an intuitive way of visualizing the hash function. Base-q PCSA and LogLog use the same cell partition but with different state spaces; see Figure 1.

[6] These probabilities are over the choice of $h(a_{i+1})$, which, in the RANDOM ORACLE MODEL, is independent of all other hash values.

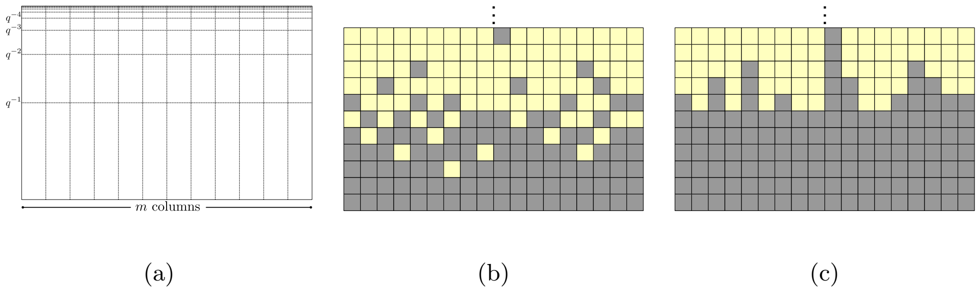

Figure 1 The unit square is partitioned into m columns. Each column is partitioned into cells. Cell j covers the vertical interval $[q^{-(j+1)}, q^{-j})$. (b) The state of a PCSA sketch records precisely which cells contain a dart (gray); all others are empty (yellow). (c) The state of the corresponding LogLog sketch.

It was observed [31] that the dartboard model includes all mergeable sketches, and some non-mergeable ones like S-Bitmap. Recordinality and the Martingale sketches obey rules (R1),(R2) but are not strictly dartboard sketches as they maintain some small state information (cnt or $\hat{\lambda}$) outside of the set of occupied cells. Nonetheless, it is useful to speak of the *dartboard part* of their state information.

1.4 Linearizable Sketches

The lower bound of [31] applies to *linearizable* sketches, a subset of mergeable sketches. A sketch is called linearizable if it is possible to encode the occupied/unoccupied status of its cells in some fixed linear order $(c_0, \ldots, c_{\mathcal{C}-1})$, so whether c_i is occupied only depends on the status of $c_0, \ldots, c_{i-1}$ and whether c_i has been hit by a dart. (Thus, it is independent of $c_{i+1}, \ldots, c_{\mathcal{C}-1}$.) Specifically, let Y_i, Z_i be the indicators for whether c_i is occupied, and has been hit by a dart, respectively, and $\mathbf{Y}_i = (Y_0, \ldots, Y_i)$. The state of the sketch is $\mathbf{Y}_{\mathcal{C}-1}$; it is called linearizable if there is some monotone function $\phi : \{0,1\}^* \to \{0,1\}$ such that

$$Y_i = Z_i \vee \phi(\mathbf{Y}_{i-1}).$$

I.e., if $\phi(\mathbf{Y}_{i-1}) = 1$, c_i is forced to be occupied and the state is forever independent of Z_i.

PCSA-type sketches [20, 18] are linearizable, as are (Hyper)LogLog [19, 16], and all MinCount, MinHash, and Bottom-m type sketches [13, 7, 23, 9, 29]. It is very easy to engineer non-linearizable sketches; see [31]. The open problem is whether this is ever a *good idea* in terms of memory-variance performance.

1.5 Organization

In Section 2 we introduce the Curtain sketch, which is a linearizable (hence mergeable) sketch in the dartboard model. In Section 3 we prove some general theorems on the bias and asymptotic relative variance of Martingale-type sketches, and in Section 4 we apply this framework to bound the limiting MVP of Martingale PCSA, Martingale Fishmonger, and Martingale Curtain.

In Section 5 we prove some results on the optimality of the Martingale transform itself, and that Martingale Fishmonger has the lowest variance among those based on linearizable sketches.

Section 6 presents some experimental findings that demonstrate that the conclusions drawn from the asymptotic analysis of Martingale sketches are extremely accurate in the pre-asymptotic regime as well, and that Martingale Curtain has lower variance than Martingale LogLog.

All the missing proofs can be found in the full version [32].

2 The Curtain Sketch

Design Philosophy

Our goal is to strike a nice balance between the simplicity and time-efficiency of (Hyper)LogLog, and the superior information-theoretic efficiency of PCSA, which can only be fully realized under extreme (and time-inefficient) compression to its entropy bound [31, 28]. Informally, if we are dedicating at least 1 bit to encode the status of a cell, the *best* cells to encode have mass $\Theta(\lambda^{-1})$ and we should design a sketch that maximizes the number of such cells encoded.

We assume the dartboard is partitioned into m columns; define $\mathrm{Cell}(j, i)$ to be the cell in column i covering the vertical interval $[q^{-(j+1)}, q^{-j})$. In a PCSA sketch, the occupied cells are precisely those with at least one dart. In LogLog, the occupied cells in each column are contiguous, extending to the highest cell containing a dart. In Figure 1, cells are drawn with uniform sizes for clarity.

Consider the vector $v = (g_0, g_1, \ldots, g_{m-1})$ where $\mathrm{Cell}(g_i, i)$ is the highest occupied cell in LogLog/PCSA. The *curtain* of v w.r.t. allowable offsets $\mathcal{O}$ is a vector $v_{\text{curt}} = (\hat{g}_0, \hat{g}_1, \ldots, \hat{g}_{m-1})$ such that (i) $\forall i \in [1, m-1].\ \hat{g}_i - \hat{g}_{i-1} \in \mathcal{O}$, and (ii) v_{curt} is the minimal such vector dominating v, i.e., $\forall i.\ \hat{g}_i \geq g_i$. Although we have described v_{curt} as a function of v, it is clearly possible to maintain v_{curt} as darts are thrown, without knowing v.

We have an interest in $|\mathcal{O}|$ being a power of 2 so that curtain vectors may be encoded efficiently, as a series of offsets. On the other hand, it is most efficient if $\mathcal{O}$ is symmetric around zero. For these reasons, we use a base-q "sawtooth" cell partition of the dartboard; see Figure 2. Henceforth $\mathrm{Cell}(j, i)$ is defined as usual, except j is an integer when i is even and a half-integer when i is odd. Then the allowable offsets are $\mathcal{O}_a = \{-(a - 1/2), -(a - 3/2), \ldots, -1/2, 1/2, \ldots, a - 3/2, a - 1/2\}$, for some a that is a power of 2.

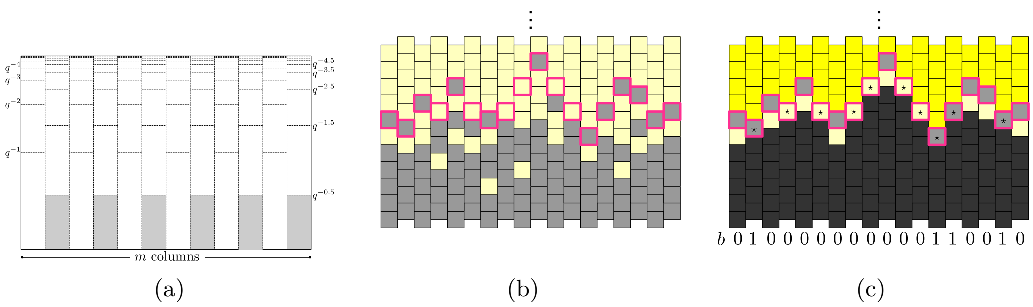

Figure 2 (a) The base-q "sawtooth" cell partition. (b) and (c) depict a Curtain sketch w.r.t. $\mathcal{O} = \{-3/2, -1/2, 1/2, 3/2\}$ and $h = 1$. (b) Gray cells contain at least one dart; light yellow cells contain none. The curtain $v_{\text{curt}} = (\hat{g}_i)$ is highlighted with a pink boundary. (c) Columns that are in *tension* have a $\star$ in their curtain cell. All *dark* gray cells are occupied and all *dark* yellow cells are free according to Rule 3. All other cells are occupied/free (light gray, light yellow) according to Rules 1 and 2.

Let $\mathrm{Cell}(g_i, i)$ be the highest cell containing a dart in column i in the *sawtooth* cell partition and $v_{\mathrm{curt}} = (\hat{g}_i)$ be the curtain vector of $v = (g_i)$ w.r.t. offsets $\mathcal{O} = \mathcal{O}_a$. We say column i is *in tension* if $(\cdots, \hat{g}_{i-1}, \hat{g}_i - 1, \hat{g}_{i+1}, \cdots)$ is not a valid curtain, i.e., if $\hat{g}_i - \hat{g}_{i-1} = \min(\mathcal{O})$ or $\hat{g}_{i+1} - \hat{g}_i = \max(\mathcal{O})$. In particular, if column i is *not* in tension, then $\mathrm{Cell}(\hat{g}_i, i)$ must contain at least one dart, for if it contained no darts the curtain would be dropped to $\hat{g}_i - 1$ at column i. However, if column i *is* in tension, then $\mathrm{Cell}(\hat{g}_i, i)$ might not contain a dart.

The Curtain sketch encodes $v_{\mathrm{curt}} = (\hat{g}_i)$ w.r.t. the base-q sawtooth cell partition and offsets $\mathcal{O}_a$, and a bit-array $b = \{0,1\}^{h\times m}$. This sketch designates each cell *occupied* or *free* as follows.

Rule 1. If column i is not in tension then $\mathrm{Cell}(\hat{g}_i, i)$ is occupied, and $b(\cdot, i)$ encodes the status of the h cells below the curtain, i.e., $\mathrm{Cell}(\hat{g}_i - (j+1), i)$ is occupied iff $b(j,i) = 1$, $j \in \{0, \ldots, h-1\}$.

Rule 2. If column i is in tension, then $\mathrm{Cell}(\hat{g}_i - j, i)$ is occupied iff $b(j,i) = 1$, $j \in \{0, \ldots, h-1\}$.

Rule 3. Every cell above the curtain is free ($\mathrm{Cell}(\hat{g}_i + j, i)$, when $j \ge 1$) and all remaining cells are occupied.

Figure 2 gives an example of a Curtain sketch, with $\mathcal{O} = \{-3/2, -1/2, 1/2, 3/2\}$ and $h = 1$. (The base q of the cell partition is unspecified in this example.)

▶ **Theorem 4.** *Consider the Martingale Curtain sketch with parameters q, a, h (base q, $\mathcal{O}_a = \{-(a-1/2), \ldots, a-1/2\}$, and $b \in \{0,1\}^{h\times m}$), and let $\hat{\lambda}$ be its estimate of the true cardinality λ.*

1. *$\hat{\lambda}$ is an unbiased estimate of λ.*
2. *The relative variance of $\hat{\lambda}$ is:*

$$\frac{1}{\lambda^2}\operatorname{Var}(\hat{\lambda} \mid \lambda) = \frac{(1 + o_{\lambda/m}(1) + o_m(1))q\ln q}{2m(q-1)}\left(\frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2}-1)} + \frac{1}{q^{h+1}}\right),$$

As a result, the limiting MVP of Martingale Curtain is

$$\mathit{MVP} = (\log_2(2a) + h) \times \frac{q\ln q}{2(q-1)}\left(\frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2}-1)} + \frac{1}{q^{h+1}}\right).$$

Proof. Follows from Theorems 11 and 17. ◀

Here $o_{\lambda/m}(1)$ and $o_m(1)$ are terms that go to zero as m and λ/m get large. Recall that for practical reasons we want to parameterize Theorem 4 with a a power of 2 and h an integer, but it is realistic to set $q > 1$ to be any real. Given these constraints, the optimal setting is $q = 2.91$, $a = 2$, and $h = 1$, exactly as in the example in Figure 2. This uses $\log\log U + 3(m-1)$ bits to store the sketch proper, $\log U$ bits[7] to store $\hat{\lambda}$, and achieves a limiting MVP ≈ 2.31. In other words, to achieve a standard error $1/\sqrt{b}$, we need about $2.31b$ bits.

Implementation Considerations

We encode a curtain $(\hat{g}_0, \hat{g}_1, \ldots, \hat{g}_{m-1})$ as $\hat{g}_0$ and an offset vector $(o_1, o_2, \ldots, o_{m-1})$, $o_i = \hat{g}_i - \hat{g}_{i-1}$, where $\hat{g}_0$ takes $\log_2\log_q U \le 6$ bits and o_i takes $\log_2|\mathcal{O}| = \log_2(2a)$ bits. Clearly, to evaluate $\hat{g}_i$ we need to compute the prefix sum $\hat{g}_0 + \sum_{i' \le i} o_{i'}$.

[7] It is fine to store an approximation $\tilde{\lambda}$ of $\hat{\lambda}$ with $O(\log m)$ bits of precision.

▶ **Lemma 5.** *Let $(x_0, \ldots, x_{\ell-1})$ be a vector of t-bit unsigned integers packed into $\lceil t\ell/w \rceil$ words, where each word has $w = \Omega(\log(t\ell))$ bits. The prefix sum $\sum_{j\in[0,i]} x_j$ can be evaluated in $O(t\ell/w + \log w)$ time.*

Proof. W.l.o.g. we can assume $i = \ell - 1$, so the task is to sum the entire list. In $O(\lceil (t\ell)/w \rceil)$ time we can halve the number of summands, by masking out the odd and even summands and adding these vectors together. After halving twice in this way, we have a vector of $\ell/4$ $(t+2)$-bit integers, each allocated $4t$ bits. At this point we can halve the number of words by adding the $(2i+1)$th word to the $2i$th word. Thus, if $T_w(\ell, t)$ is the time needed to solve this problem, $T_w(\ell, t) = T_w(\ell/8, 4t) + O(\lceil (t\ell)/w \rceil)$, which is $O((t\ell)/w + \log w)$. ◀

In our context $t = \log_2(2a) = 2$, so even if m is a medium-size constant, say at most 256 or 512, we only have to do prefix sums over 8 or 16 consecutive 64-bit words. If m is much larger then it would be prudent to partition the dartboard into m/c independent curtains, each with $c = 256$ or 512 columns. This keeps the update time independent of m and increases the space overhead negligibly.

We began this section by highlighting the design philosophy, which emphasizes conceptual simplicity and efficiency. Our encoding uses fixed-length codes for the offsets, and can be decoded very efficiently by exploiting bit-wise operations and word-level parallelism. That said, we are mainly interested in analyzing the *theoretical* performance of sketches, and will not attempt an exhaustive experimental evaluation in this work.

3 Foundations of the Martingale Transform

In this section we present a simple framework for analyzing the limiting variance of Martingale sketches, which is strongly influenced by Ting's [35] work. Theorem 7 gives simple unbiased estimators for the cardinality and the variance of the the cardinality estimator. The upshot of Theorem 7 is that to analyze the variance of the estimator, we only need to bound $\mathbb{E}(P_k^{-1})$, where P_k is the probability the kth distinct element changes the sketch. Theorem 11 further shows that for sketches composed of m subsketches (like Curtain, HyperLogLog, and PCSA), the limiting variance tends to $\frac{1}{2\kappa m}$, where κ is a constant that depends on the sketch scheme. Section 4 analyzes the constant κ for each of PCSA, LogLog, and Curtain. Using results of [31] on the entropy of PCSA we can calculate the limiting MVP of PCSA, LogLog, Curtain, and Fishmonger.

3.1 Martingale Estimators and Retrospective Variance

Consider an arbitrary sketch with state space $\mathcal{S}$. We assume the sketch state does not change upon seeing duplicated elements, hence it suffices to consider streams of *distinct* elements. We model the evolution of the sketch as a Markov chain $(S_k)_{k\geq 0} \in \mathcal{S}^*$, where S_k is the state after seeing k *distinct* elements. Define $P_k = \Pr(S_k \neq S_{k-1} \mid S_{k-1})$ to be the *state changing probability*, which depends only on S_{k-1}. In the dartboard terminology P_k is the total size of all unoccupied cells in S_{k-1}.

▶ **Definition 6.** *Let* $\llbracket \mathcal{E} \rrbracket$ *be the indicator variable for event* $\mathcal{E}$*. For any* $\lambda \geq 0$*, define:*

$$E_\lambda = \sum_{k=1}^{\lambda} \llbracket S_k \neq S_{k-1} \rrbracket \cdot \frac{1}{P_k}, \qquad \text{the martingale estimator,}$$

$$\text{and } V_\lambda = \sum_{k=1}^{\lambda} \llbracket S_k \neq S_{k-1} \rrbracket \cdot \frac{1-P_k}{P_k^2}, \qquad \text{the "retrospective" variance.}$$

Note that $E_0 = V_0 = 0$*.*

The Martingale transform of this sketch stores $\hat{\lambda} = E_\lambda$ in one machine word and returns it as a cardinality estimate. It can also store V_λ in one machine word as well. Theorem 7 shows[8] that the retrospective variance V_λ is a good running estimate of the empirical squared error $(E_\lambda - \lambda)^2$.

▶ **Theorem 7.** *The martingale estimator* E_λ *is an unbiased estimator of* λ *and the retrospective variance* V_λ *is an unbiased estimator of* $\mathrm{Var}(E_\lambda)$*. Specifically, we have,*

$$\mathbb{E}(E_\lambda) = \lambda, \text{ and } \mathrm{Var}(E_\lambda) = \mathbb{E}(V_\lambda) = \sum_{k=1}^{\lambda} \mathbb{E}\left(\frac{1}{P_k}\right) - \lambda.$$

▶ Remark 8. Theorem 7 contradicts Ting's claim [35], that V_λ is unbiased *only at "jump" times*, i.e., those λ for which $S_\lambda \neq S_{\lambda-1}$, and therefore inadequate to estimate the variance. In order to correct for this, Ting introduced a Bayesian method for estimating the time that has passed since the last jump time. The reason for thinking that jump times are different is actually quite natural. Suppose we record the list of *distinct* states $s_0, \ldots, s_k$ encountered while inserting λ elements, λ being unknown, and let p_i be the probability of changing from s_i to some other state. The amount of time spent in state s_i is a geometric random variable with mean p_i^{-1} and variance $(1-p_i)/p_i^2$. Furthermore, these waiting times are independent. Thus, $\sum_{i\in[0,k)} p_i^{-1}$ and $\sum_{i\in[0,k)}(1-p_i^{-1})/p_i^2$ are unbiased estimates of the cardinality λ' and squared error *upon entering state* s_k. These exactly correspond to E_λ and V_λ, but they *should* be biased since they do not take into account the $\lambda - \lambda'$ elements that had no effect on s_k. As Theorem 7 shows, this is a mathematical optical illusion. The history is a random variable, and although the last $\lambda - \lambda'$ elements did not change the state, *they could have*, which would have altered the observed history $s_0, \ldots, s_k$ and hence the estimates E_λ and V_λ.

3.2 Asymptotic Relative Variance

3.2.1 The ARV Factor

We consider classes of sketches composed of m *subsketches*, which controls the size and variance. In LogLog, PCSA, and Curtain these subsketches are the m columns. When considering a sketch with m subsketches, instead of using λ as the total number of insertions, we always use λ to denote the number of insertions *per subsketch* and therefore the total number of insertions is λm. We care about the *asymptotic relative variance* (ARV) as m and λ both go to infinity (defined below). A reasonable sketch should have relative variance $O(1/m)$. Informally, the ARV factor is just the leading constant of this expression.

---

[8] The proof can be found in the full version [32].

▶ **Definition 9** (ARV factor). *Consider a class of sketches whose size is parameterized by m. For any $k \geq 0$, define $P_{m,k}$ to be the probability the sketch changes state upon the kth insertion and $E_{m,k}$ the martingale estimator. The* ARV factor *of this class of sketches is defined as*

$$\lim_{\lambda\to\infty} \lim_{m\to\infty} m \cdot \frac{\mathrm{Var}(E_{m,\lambda m})}{(\lambda m)^2}. \tag{1}$$

3.2.2 Scale-Invariance and the Constant κ

Few sketches have *strictly* well-defined ARV factors. In Martingale LogLog, for example, the quantity $\left(\lim_{m\to\infty} m\frac{\mathrm{Var}(E_{m,\lambda m})}{(\lambda m)^2}\right)$ is not constant, but periodic in $\log_2 \lambda$; it does not converge as $\lambda \to \infty$. We explain how to fix this issue using *smoothing* in Section 3.2.3. *Scale-invariant* sketches must have well-defined ARV factors.

▶ **Definition 10** (scale-invariance and constant κ). *A combined sketch is* scale-invariant *if*
1. *For any λ, there exists a constant κ_λ such that $\lambda \cdot P_{m,\lambda m}$ converges to κ_λ almost surely as $m \to \infty$.*
2. *The limit of κ_λ as $\lambda \to \infty$ exists, and $\kappa \stackrel{\text{def}}{=} \lim_{\lambda\to\infty} \kappa_\lambda$.*

The constant of a sketch A is denoted as κ_A, where the subscript A is often dropped when the context is clear.

The next theorem proves that under mild regularity conditions, all scale-invariant sketches have well defined ARV factors and there is a direct relation between the ARV factor and the constant κ.

▶ **Theorem 11** (ARV factor of a scale-invariant sketch). *Consider a sketching scheme satisfying the following properties.*
1. *It is scale-invariant with constant κ.*
2. *For any $\lambda > 0$, the limit operator and the expectation operator of $\{\frac{1}{P_{m,\lambda m}}\}_m$ can be interchanged.*

Then the ARV factor of the sketch exists and equals $\frac{1}{2\kappa}$.

The constant κ together with Theorem 11 is useful in that it gives a simple and systematic way to evaluate the asymptotic performance of a well behaved (scale-invariant) sketch scheme.

MinCount [23, 9, 29] is an example of a scale-invariant sketch. The function $h(a) = (i, v) \in [m] \times [0, 1]$ is interpreted as a pair containing a bucket index and a real hash value. A (k, m)-MinCount sketch stores the smallest k hash values in each bucket.

▶ **Theorem 12.** *(k, m)-MinCount is scale-invariant and $\kappa_{(k,m)\text{-MinCount}} = k$.*

Proof. When a total of λm elements are inserted to the combined sketch, each subsketch receives $(1 + o(1))\lambda$ elements as $\lambda \to \infty$. Since we only care the asymptotic behavior, we assume for simplicity that each subsketch receives exactly λ elements.

Let $P^{(i)}_\lambda$ be the probability that the sketch of the ith bucket changes after the λth element is thrown into the ith bucket. Then by definition, we have

$$P_{m,\lambda m} = \frac{\sum_{i=1}^m P^{(i)}_\lambda}{m}.$$

Since all the subsketches are i.i.d., by the law of large numbers, $\lambda \cdot P_{m,\lambda} \to \lambda \cdot \mathbb{E}\left(P^{(1)}_\lambda\right)$ almost surely as $m \to \infty$.

Let X be the kth smallest hash value among λ uniformly random numbers in $[0, 1]$, which distributes identically with $P_\lambda^{(1)}$. By standard order statistics, X is a Beta random variable $\mathrm{Beta}(k, \lambda - 1 + k)$ which has mean $\frac{k}{\lambda+1}$. Thus $\kappa_\lambda = \lambda \cdot \mathbb{E}(X) = \frac{k\lambda}{\lambda+1}$. We conclude that

$$\kappa = \lim_{\lambda\to\infty} \kappa_\lambda = \lim_{\lambda\to\infty} \frac{k\lambda}{\lambda+1} = k. \qquad ◀$$

Applying Theorem 11 to (k, m)-MinCount, we see its ARV is $\frac{1}{2km}$,[9] matching Cohen [14] and Ting [35]. Technically its MVP is unbounded since hash values were real numbers, but any realistic implementation would store them to $\log U$ bits of precision, for a total of $km \log U$ bits. Hence we regard its MVP to be $\frac{1}{2} \cdot \log_2 U$.

3.2.3 Smoothing Discrete Sketches

Sketches that partition the dartboard in some exponential fashion with base q (like LogLog, PCSA, and Curtain) have the property that their estimates and variance are periodic in $\log_q \lambda$. Pettie and Wang [31] proposed a simple method to *smooth* these sketches and make them truly scale-invariant as $m \to \infty$.

We assume that the dartboard is partitioned into m columns. The base-q *smoothing* operation uses an *offset vector* $\vec{r} = (r_0, \ldots, r_{m-1})$. We scale down all the cells in column i by the factor q^{-r_i}, then add a dummy cell spanning $[q^{-r_i}, 1)$ which is always occupied. (Phrased algorithmically, if a dart is destined for column i, we filter it out with probability $1 - q^{-r_i}$ and insert it into the sketch with probability q^{-r_i}.) When analyzing variants of (Hyper)LogLog and PCSA, we use the uniform offset vector $(0, 1/m, 2/m, \ldots, (m-1)/m)$. The Curtain sketch can be viewed as having a built-in offset vector of $(0, 1/2, 0, 1/2, 0, 1/2, \ldots)$ which effects the “sawtooth” cell partition. To smooth it, we use the offset vector[10]

$$(0,\ 1/2,\ 1/m,\ 1/2 + 1/m,\ 2/m,\ 1/2 + 2/m,\ \ldots,\ 1/2 - 1/m,\ 1 - 1/m).$$

As $m \to \infty$, $\vec{r}$ becomes uniformly dense in $[0, 1]$.

The smoothing technique makes the empirical estimation more scale-invariant (see [31, Figs. 1& 2]) but also makes the sketch theoretically scale-invariant according to Definition 10. Thus, in the analysis, we will always assume the sketches are smoothed. However, in practice it is probably not necessary to do smoothing if $q < 3$.

In the next section, we will prove that *smoothed* q-LL, q-PCSA, and Curtain are all scale-invariant.

4 Analysis of Dartboard Based Sketches

Consider a dartboard cell that covers the vertical interval $[q^{-(t+1)}, q^{-t})$. We define the *height* of the cell to be t. In a smoothed cell partition, no two cells have the same height and all heights are of the form $t = j/m$, for some integer j. Thus, we may refer to it unambiguously as *cell* t. Note that cell t is an $m^{-1} \times \frac{1}{q^t}\frac{q-1}{q}$ rectangle.

---

[9] For simplicity, we assume the second condition of Theorem 4 holds for all the sketches analyzed in this paper.

[10] In [31], the smoothing was implemented via *random* offsetting, instead of the *uniform* offsetting. In Curtain we need to use uniform offsetting so that the offset values of columns are similar to their neighbors.

4.1 Poissonized Dartboard

Since we care about the asymptotic case where $\lambda \to \infty$, we model the process of "throwing darts" by a Poisson point process on the dart board (similar to the "poissonization" in the analysis of HyperLogLog [19]). Specifically, after throwing λm darts (events) to the dartboard, we assume the number of darts in cell t is a Poisson random variable with mean $\lambda \frac{1}{q^t}\frac{q-1}{q}$ and the number of darts in different cells are independent. For the poissonized dartboard, the range of height of cells naturally extend to the whole set of real numbers, instead of just having cells with positive height.

For any $t \in \mathbb{R}$, let $Y_{t,\lambda}$ be the indicator whether cell t contains at least one dart. Note that the probability that a Poisson random variable with mean λ' is zero is $e^{-\lambda'}$. Thus we have,

$$\Pr(Y_{t,\lambda} = 0) = e^{-\frac{\lambda}{q^t}\frac{q-1}{q}}.$$

Here, we note some simple identities for integrals that we will use frequently in the analysis.

▶ **Lemma 13.** *For any $q > 1$, we have*

$$\int \frac{1}{q^t} e^{-\frac{1}{q^t}} dt = \frac{1}{\ln q} e^{-\frac{1}{q^t}} + C.$$

Furthermore, let c_0, c_1 be any positive numbers, we have

$$\int_{-\infty}^{\infty} \frac{c_0}{q^t} e^{-\frac{c_1}{q^t}} dt = \frac{c_0}{c_1}\frac{1}{\ln q}.$$

4.2 The Constant κ

Let $Z_{t,\lambda}$ be the indicator of whether the cell t is *free*. Unlike $Y_{t,\lambda}$, $Z_{t,\lambda}$ depends on which sketching algorithm we are analyzing. Since the state changing probability is equal to the sum of the area of free cells, we have

$$P_{m,\lambda m} = \sum_{j=0}^{\infty} \frac{1}{m}\left(\frac{1}{q^{j/m}} - \frac{1}{q^{j/m+1}}\right) Z_{j/m,\lambda}. \tag{2}$$

If $P_{m,\lambda m}$ converges to κ_λ/λ almost surely as $m \to \infty$, then $\mathbb{E}(P_{m,\lambda m})$ also converges to κ_λ/λ as $m \to \infty$. Thus we have, from (2),

$$\begin{aligned}\kappa_\lambda/\lambda &= \lim_{m\to\infty} \mathbb{E}(P_{m,\lambda m}) = \lim_{m\to\infty} \sum_{j=0}^{\infty} \frac{1}{m}\left(\frac{1}{q^{j/m}} - \frac{1}{q^{j/m+1}}\right) \mathbb{E}(Z_{j/m,\lambda}) \\ &= \int_0^{\infty} \left(\frac{1}{q^t} - \frac{1}{q^{t+1}}\right) \mathbb{E}(Z_{t,\lambda}) dt \approx \int_{-\infty}^{\infty} \left(\frac{1}{q^t} - \frac{1}{q^{t+1}}\right) \mathbb{E}(Z_{t,\lambda}) dt, \end{aligned} \tag{3}$$

where we can extend the integration range to negative infinity without affecting the limit of κ_λ as $\lambda \to \infty$.[11] We conclude that

$$\kappa = \lim_{\lambda\to\infty} \kappa_\lambda = \lim_{\lambda\to\infty} \lambda \int_{-\infty}^{\infty} \left(\frac{1}{q^t} - \frac{1}{q^{t+1}}\right) \mathbb{E}(Z_{t,\lambda}) dt. \tag{4}$$

The formula (4) is novel in the sense that, in order to evaluate κ, we now only need to understand the probability that $Z_{t,\lambda}$ is 1 for fixed t and λ.[12]

[11] See [32].

[12] Technically, to apply formula (4) one needs to first prove that the state changing probability $P_{m,\lambda m}$ converges almost surely to some constant κ_λ/λ for any λ, which is a mild regularity condition for any reasonable sketch. Thus in this paper we will assume the sketches in the analysis all satisfy this regularity condition and claim that a sketch is scale-invariant if formula (4) converges.

4.3 Analysis of Smoothed q-PCSA and q-LL

The sketches q-PCSA and q-LL are the natural smoothed base-q generalizations of PCSA [20] and LogLog [16].

▶ **Theorem 14.** *q-PCSA and q-LL are scale-invariant. In particular, we have,*

$$\kappa_{q\text{-PCSA}} = \frac{1}{\ln q}, \text{ and } \kappa_{q\text{-LL}} = \frac{1}{\ln q}\frac{q-1}{q}.$$

Proof. For q-LL, cell t is free iff both itself and all the cells above it in its column contain no darts. Thus we have

$$\mathbb{E}(Z_{t,\lambda}) = \prod_{i=0}^{\infty} \Pr(Y_{t+i,\lambda} = 0) = \prod_{i=0}^{\infty} e^{-\frac{\lambda}{q^{t+i}}\frac{q-1}{q}} = e^{-\frac{\lambda}{q^t}}.$$

Insert it to formula (4) and we get

$$\kappa_{q\text{-LL}} = \lim_{\lambda\to\infty} \lambda \int_{-\infty}^{\infty} \left(\frac{1}{q^t} - \frac{1}{q^{t+1}}\right) e^{-\frac{\lambda}{q^t}}\, dt = \frac{1}{\ln q}\frac{q-1}{q}.$$

For q-PCSA, cell t is free iff it has no dart. Thus $Z_{t,\lambda} = 1 - Y_{t,\lambda}$ and by formula (4) we have

$$\kappa_{q\text{-PCSA}} = \lim_{\lambda\to\infty} \lambda \int_{-\infty}^{\infty} \left(\frac{1}{q^t} - \frac{1}{q^{t+1}}\right) e^{-\frac{\lambda}{q^t}\frac{q-1}{q}}\, dt = \frac{1}{\ln q}.$$ ◀

The Fishmonger [31] sketch is based on a smoothed, entropy compressed version of base-e PCSA. The memory footprint of Fishmonger approaches its entropy as $m \to \infty$, which was calculated to be mH_0 [31, Lemma 4]. From Theorem 14, we know $\kappa_{e\text{-PCSA}} = 1$.

▶ **Corollary 15.** *Fishmonger has limiting MVP $H_0/2 \approx 1.63$.*

Proof. By Theorem 11, limiting MVP equals $mH_0 \cdot \frac{1}{2m} = \frac{H_0}{2}$. ◀

4.4 Asymptotic Local View

For any t and λ, since we want to evaluate $Z_{t,\lambda}$, whose value may depend on its "neighbors" on the dartboard, we need to understand the configurations of the cells near cell t. Since we consider the case where m goes to infinity, we may ignore the effect of smoothing to the cells in the immediate vicinity of cell t.

After taking these asymptotic approximations, we can index the cells near cell t as follows.

▶ **Definition 16** (neighbors of cell t). *Fix a cell t. Let $i \in \mathbb{Z}$ and $c \in \mathbb{R}$. The (i, c)-neighbor of cell t is a cell whose column index differs by i (negative i means to the left, positive to the right) and has height $t + c$, it covers the vertical interval $[q^{-(t+c+1)}, q^{-(t+c)})$. In the sawtooth partition, c is an integer when i is even and a half-integer when i is odd. (Note that we are locally ignoring the effect of smoothing.)*

Once cell t is fixed, define $W(i, c)$ to be the indicator for whether the (i, c)-neighbor of cell t has at least one dart in it. Thus, for fixed t, λ, we have

$$\Pr(W(i, c) = 0) = \Pr(Y_{t+c,\lambda} = 0) = e^{-\frac{\lambda}{q^{t+c}}\frac{q-1}{q}}.$$

In the asymptotic local view, we lose the property that a cell can be uniquely identified by its height, hence the need to refer to nearby cells by their position *relative* to cell t.

4.5 Analysis of Curtain

We first briefly state some properties of curtain. For any $a \ge 1$, recall that $\mathcal{O}_a = \{-(a - 1/2), -(a-3/2), \ldots, -1/2, 1/2, \ldots, a-3/2, a-1/2\}$. It is easy to see that for any vector $v = (g_0, g_1, \ldots, g_{m-1})$, $v_{\text{curt}} = (\hat{g}_i)$ can be expressed as

$$\hat{g}_i = \max_{j \in [0,m-1]} \{g_j - |i-j|(a-1/2)\}.$$

For each i, we define the *tension point* τ_i to be the lowest allowable value of $\hat{g}_i$, given the context of its neighboring columns.

$$\tau_i = \max_{j \in [0,m-1] \setminus \{i\}} \{g_j - |i-j|(a-1/2)\},$$

and thus we have $\hat{g}_i = \max(g_i, \tau_i)$. We see that the column i is *in tension* iff $g_i \le \tau_i$, that is, $\hat{g}_i = \tau_i$.

▶ **Theorem 17.** *Curtain is scale-invariant with*

$$\kappa_{\mathsf{Curtain}} = \frac{1}{\ln q} \frac{q-1}{q} \frac{1}{\frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2}-1)} + \frac{1}{q^{h+1}}}.$$

Proof. Fix cell t and λ. Define $W_1(k)$ to be the height of the highest cell containing darts in the column k away from t's column. I.e., define $\iota = [\![k \text{ is odd}]\!]/2$ to be $1/2$ if k is odd and zero if k is even, and $W_1(k) \stackrel{\text{def}}{=} \max\{t + i + \iota \mid i \in \mathbb{Z} \text{ and } W(k, i+\iota) = 1\}$.

We have for any $i \in \mathbb{Z}$,

$$\Pr(W_1(k) \le t + i + \iota) = \prod_{j=1}^{\infty} \Pr(W(k, i+j+\iota) = 0) = e^{-\frac{\lambda}{q^{t+1+i+\iota}}}.$$

Let T_1 be the tension point of the column of cell t, which equals $\max_{j \in \mathbb{Z}\setminus\{0\}} \{W_1(j) - |j|(a-1/2)\}$. We have for any $i \in \mathbb{Z}$,

$$\begin{aligned}
\Pr(T_1 \le i + t) &= \Pr\left(\max_{j \in \mathbb{Z}\setminus\{0\}} \{W_1(j) - |j|(a-1/2)\} \le i + t\right) \\
&= \prod_{j \in \mathbb{Z}\setminus\{0\}} \Pr(W_1(j) - |j|(a-1/2) \le i+t) \\
&= \left(\prod_{j=1}^{\infty} e^{-\lambda \frac{1}{q^{t+i+1+j(a-1/2)}}}\right)^2 = e^{-\lambda \frac{2}{q^{t+i+1}} \frac{1}{q^{a-1/2}-1}}.
\end{aligned}$$

From the rules of $\mathsf{Curtain}$, we know that a cell is free iff it contains no dart, it is at most $h-1$ below its column's tension point, and at most h below the highest cell in its column containing darts. Thus,

$$Z_{t,\lambda} = [\![Y_{t,\lambda} = 0]\!] \cdot [\![t \ge T_1 - (h-1)]\!] \cdot [\![t \ge W_1(0) - h]\!],$$

Note that T_1 is independent from $Y_{t,\lambda}$ and $W_1(0)$. In addition, $Y_{t,\lambda}$ is also independent from $[\![t \ge W_1(0) - h]\!]$, since the latter only depends on $Y_{t',\lambda}$ with $t' \ge h + t + 1$. Thus, we have

$$\begin{aligned}
\mathbb{E}(Z_{t,\lambda}) &= \Pr(Y_{t,\lambda} = 0) \cdot \Pr(T_1 \le t + h - 1) \cdot \Pr(W_1(0) \le t + h) \\
&= e^{-\frac{\lambda}{q^t}\frac{q-1}{q}} e^{-\lambda \frac{2}{q^{t+h}} \frac{1}{q^{a-1/2}-1}} e^{-\frac{\lambda}{q^{t+h+1}}} \\
&= \exp\left(-\frac{\lambda}{q^t}\left(\frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2}-1)} + \frac{1}{q^{h+1}}\right)\right).
\end{aligned}$$

Thus by formula (4), we have

$$\begin{aligned}\kappa_{\mathsf{Curtain}} &= \lim_{\lambda\to\infty} \lambda \int_{-\infty}^{\infty} \left(\frac{1}{q^t} - \frac{1}{q^{t+1}}\right) \exp\left(-\frac{\lambda}{q^t}\left(\frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2}-1)} + \frac{1}{q^{h+1}}\right)\right) dt \\ &= \frac{1}{\ln q}\frac{q-1}{q}\frac{1}{\frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2}-1)} + \frac{1}{q^{h+1}}}.\end{aligned}$$ ◀

5 Optimality of Martingale Fishmonger

Martingale sketches have several attractive properties, e.g., being strictly unbiased and insensitive to duplicate elements in the data stream. In Section 5.1 we argue that any sketch that satisfies these natural assumptions can be systematically transform into a Martingale X sketch with equal or lesser variance, where X is a dartboard sketch. In other words, the Martingale *transform* is optimal.

In Section 5.2 we prove that within the class of *linearizable* dartboard sketches, Martingale Fishmonger is optimal. The class of linearizable sketches is broad and includes state-of-the-art sketches, which lends strong *circumstantial* evidence that the memory-variance product of Martingale Fishmonger cannot be improved.

5.1 Optimality of the Martingale Transform

Consider a non-mergeable sketch processing a stream $\mathcal{A} = (a_1, a_2, \ldots)$. Let S_i be its state after seeing $(a_1, \ldots, a_i)$, $\lambda_i = |\{a_1, \ldots, a_i\}|$, and $\hat{\lambda}(S_i)$ be the estimate of cardinality λ_i when in state S_i. We make the following natural assumptions.

Randomness. The random oracle h is the only source of randomness. In particular, S_i is a function of $(h(a_1), h(a_2), \ldots, h(a_i))$.

Duplicates. If $a_i \in \{a_1, \ldots, a_{i-1}\}$, $S_i = S_{i-1}$, i.e., duplicates do not trigger state transitions.

Unbiasedness. Suppose one examines the data structure at time i and sees $S_i = \mathbf{s}_i$ and then examines it at time j. Then $\hat{\lambda}(S_j) - \hat{\lambda}(\mathbf{s}_i)$ is an unbiased estimate of $\lambda_j - \lambda_i$.

In the full version [32], we show that as a consequence of the *Randomness*, *Duplicates*, and *Unbiased* assumptions, the Martingale estimator has minimum variance.

▶ Remark 18. We should note that under some circumstances it is possible to achieve smaller variance by violating the *duplicates* and *unbiasedness* assumptions. For example, suppose the sketch state after seeing i elements were $(\hat{\lambda}_i, S_i, i)$. If the stream is duplicate-heavy, "i" carries no useful information, but if nearly all elements are distinct, i is also a good cardinality estimate. Since $\lambda_i \le i$, the cardinality estimate $\min\{\hat{\lambda}_i, i\}$ is never worse than $\hat{\lambda}_i$ alone, but when $\lambda_i \approx i$, it is biased and has a constant factor lower variance.

5.2 Optimality of Martingale Fishmonger

Given an abstract linearizable sketching scheme X, its space is minimized by compressing it to its entropy. On the other hand, by Theorem 11 the variance of Martingale X is controlled by the normalized expected probability of changing state: $2\lambda \cdot \mathbb{E}(P_\lambda)$. Theorem 19 lower bounds the ratio of these two quantities for any sketch that behaves well over a sufficiently large interval of cardinalities $\lambda \in [e^a, e^b]$. The proof technique is very similar to [31], as is the take-away message (that X=Fishmonger is optimal up to some assumptions). However, the two proofs are mathematically distinct as [31] focuses on Fisher information while Theorem 19 focuses on the *probability of state change*.

▶ **Theorem 19.** *Fix reals $a < b$ with $d = b - a > 1$. Let $\bar{H}, \bar{R} > 0$. For any linearizable sketch, let $H(\lambda)$ be the entropy of its state and P_λ be the probability of state change*[13] *at cardinality λ satisfies that*

1. *for all $\lambda > 0$, $H(\lambda) \le \bar{H}$, and*
2. *for all $\lambda \in [e^a, e^b]$, $2\lambda\mathbb{E}(P_\lambda) \ge \bar{R}$, then*

$$\frac{\bar{H}}{\bar{R}} \ge \frac{H_0}{2} \frac{1 - \max(8d^{-1/4}, 5e^{-d/2})}{1 + \frac{(344+4\sqrt{d})}{d}\frac{H_0}{I_0}\left(1 - \max(8d^{-1/4}, 5e^{-d/2})\right)} = \frac{H_0}{2}(1 - o_d(1)).$$

▶ **Corollary 20.** *The MVP of any linearizable and scale-invariant sketch is at least $\frac{H_0}{2}$.*

6 Experimental Validation

Throughout the paper we have maintained a possibly unhealthy devotion to asymptotic analysis, taking $m \to \infty$ whenever it was convenient. In practice m will be a constant, and possibly a smallish constant. How do the sketches perform in the pre-asymptotic region?

In turns out that the theoretical analysis predicts the performance of Martingale sketches pretty well, even whem m is small. In the experiment of Figure 3, we fixed the sketch size at a tiny 128 bits. Therefore HyperLogLog uses $m_1 = \lfloor 128/6 \rfloor = 21$ counters. The Martingale LogLog and Martingale Curtain sketches encode the martingale estimator with a floating point *approximation* of $\hat{\lambda}$ in 14 bits, with a 6-bit exponent and 8-bit mantissa. Thus, Martingale LogLog uses $m_2 = (128 - 14)/6 = 19$ counters, and Martingale Curtain uses $m_3 = 37$.[14]

For larger sketch sizes, the distribution of $\hat{\lambda}/\lambda$ is more symmetric, and closer to the predicted performance. Figure 4 gives the empirical distribution of $\hat{\lambda}/\lambda$ over 100,000 runs when $\lambda = 10^6$ and the sketch size is fixed at 1,200 bits. Here MartingaleCurtain uses $m = 400$, and both Martingale LogLog and HyperLogLog use $m = 200$. The experimental and predicted relative variances and standard errors are given in Table 2.

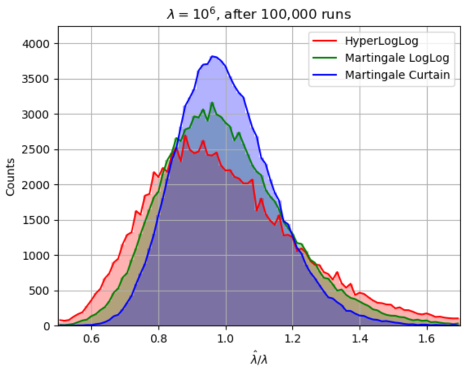

Figure 3 The sketch size is fixed at 128 bits.

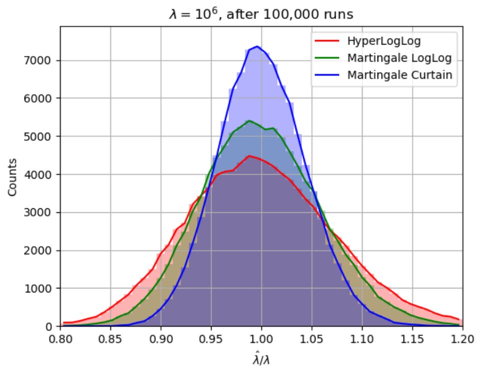

Figure 4 The sketch size is fixed at 1200 bits.

[13] The probability of state change P_λ is itself a random variable.
[14] It uses the optimal parameterization $(q, a, h) = (2.91, 2, 1)$ of Theorem 4.

Table 2 The relative variance is $\frac{1}{\lambda^2}\operatorname{Var}(\hat{\lambda} \mid \lambda)$ and standard error is $\frac{1}{\lambda}\sqrt{\operatorname{Var}(\hat{\lambda} \mid \lambda)}$. The predictions for Martingale LogLog and Martingale Curtain use Theorems 11, 14, and 17. The predictions for HyperLogLog are from Flajolet et al. [19, p. 139].

Sketch	Using 128 bits				Using 1200 bits			
	Experiment		Prediction		Experiment		Prediction	
	Var	StdErr	Var	StdErr	Var	StdErr	Var	StdErr
HyperLogLog	0.0573	23.94%	0.0549	23.44%	0.00541	7.36%	0.00539	7.35%
Martingale LogLog	0.0348	18.65%	0.0365	19.10%	0.00350	5.91%	0.00347	5.89%
Martingale Curtain	0.0211	14.54%	0.0208	14.43%	0.00189	4.35%	0.00193	4.39%

7 Conclusion

The Martingale transform is attractive due to its simplicity and low variance, but it results in *non*-mergeable sketches. We proved that under natural assumptions,[15] it generates optimal estimators automatically, allowing one to design structurally more complicated sketches, without having to worry about designing or analyzing *ad hoc* estimators. We proposed the Curtain sketch, in which each subsketch only needs a constant number of bits of memory, for *arbitrarily large* cardinality U.[16]

The analytic framework of Theorems 7 and 11 simplifies Cohen [14] and Ting [35], and gives a user-friendly formula for the asymptotic relative variance (ARV) of the Martingale estimator, as a function of the sketch's constant κ. We applied this framework to Martingale Curtain as well as the Martingale version of the classic sketches (MinCount, HLL and PCSA).

Assuming perfect compression, one gets the *memory-variance product* (MVP) of an sketch by multiplying its entropy and ARV. It is proved that for linearizable sketches, Fishmonger is optimal for mergeable sketches [31] (limiting MVP $= H_0/I_0 \approx 1.98$). In this paper we proved that in the sequential (non-mergeable) setting, if we restrict our attention to linearizable sketches, that Martingale Fishmonger is optimal, with limiting MVP $= H_0/2 \approx 1.63$ (Section 5.2). We conjecture that these two lower bounds hold for general, possibly *non*-linearizable sketches.

References

1 Noga Alon, Phillip B. Gibbons, Yossi Matias, and Mario Szegedy. Tracking join and self-join sizes in limited storage. In *Proceedings 18th ACM Symposium on Principles of Database Systems (PODS)*, pages 10–20, 1999. doi:10.1145/303976.303978.

2 Daniel N Baker and Ben Langmead. Dashing: Fast and accurate genomic distances with hyperloglog. *bioRxiv*, 2019. doi:10.1101/501726.

3 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, D. Sivakumar, and Luca Trevisan. Counting distinct elements in a data stream. In *Proceedings 6th International Workshop on Randomization and Approximation Techniques (RANDOM)*, volume 2483 of *Lecture Notes in Computer Science*, pages 1–10, 2002. doi:10.1007/3-540-45726-7_1.

4 Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. Reductions in streaming algorithms, with an application to counting triangles in graphs. In *Proceedings 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 623–632, 2002.

[15] insensitivity to duplicates, and unbiasedness
[16] Note that an $O(\log\log U)$-bit offset register is needed for the whole sketch.

5 Ran Ben-Basat, Gil Einziger, Shir Landau Feibish, Jalil Moraney, and Danny Raz. Network-wide routing-oblivious heavy hitters. In *Proceedings of the 2018 Symposium on Architectures for Networking and Communications Systems (ANCS)*, pages 66–73, 2018. doi:10.1145/3230718.3230729.

6 Jarosław Błasiok. Optimal streaming and tracking distinct elements with high probability. *ACM Trans. Algorithms*, 16(1):3:1–3:28, 2020. doi:10.1145/3309193.

7 Andrei Z. Broder. On the resemblance and containment of documents. In *Proceedings of Compression and Complexity of SEQUENCES*, pages 21–29, 1997. doi:10.1109/SEQUEN.1997.666900.

8 Thilina Buddhika, Matthew Malensek, Sangmi Lee Pallickara, and Shrideep Pallickara. Synopsis: A distributed sketch over voluminous spatiotemporal observational streams. *IEEE Trans. Knowl. Data Eng.*, 29(11):2552–2566, 2017. doi:10.1109/TKDE.2017.2734661.

9 Philippe Chassaing and Lucas Gerin. Efficient estimation of the cardinality of large data sets. In *Proceedings of the 4th Colloquium on Mathematics and Computer Science Algorithms, Trees, Combinatorics and Probabilities*, 2006.

10 Aiyou Chen, Jin Cao, Larry Shepp, and Tuan Nguyen. Distinct counting with a self-learning bitmap. *Journal of the American Statistical Association*, 106(495):879–890, 2011. doi:10.1198/jasa.2011.ap10217.

11 Min Chen, Shigang Chen, and Zhiping Cai. Counter tree: A scalable counter architecture for per-flow traffic measurement. *IEEE/ACM Trans. Netw.*, 25(2):1249–1262, 2017. doi:10.1109/TNET.2016.2621159.

12 Pern Hui Chia, Damien Desfontaines, Irippuge Milinda Perera, Daniel Simmons-Marengo, Chao Li, Wei-Yen Day, Qiushi Wang, and Miguel Guevara. KHyperLogLog: Estimating reidentifiability and joinability of large data at scale. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy*, pages 350–364, 2019. doi:10.1109/SP.2019.00046.

13 Edith Cohen. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. Syst. Sci.*, 55(3):441–453, 1997. doi:10.1006/jcss.1997.1534.

14 Edith Cohen. All-distances sketches, revisited: HIP estimators for massive graphs analysis. *IEEE Trans. Knowl. Data Eng.*, 27(9):2320–2334, 2015. doi:10.1109/TKDE.2015.2411606.

15 Edith Cohen and Haim Kaplan. Tighter estimation using bottom k sketches. *Proc. VLDB Endow.*, 1(1):213–224, 2008. doi:10.14778/1453856.1453884.

16 Marianne Durand and Philippe Flajolet. Loglog counting of large cardinalities. In *Proceedings 11th Annual European Symposium on Algorithms (ESA)*, volume 2832 of *Lecture Notes in Computer Science*, pages 605–617. Springer, 2003. doi:10.1007/978-3-540-39658-1_55.

17 R. A. Leo Elworth, Qi Wang, Pavan K. Kota, C. J. Barberan, Benjamin Coleman, Advait Balaji, Gaurav Gupta, Richard G. Baraniuk, Anshumali Shrivastava, and Todd J. Treangen. To petabytes and beyond: recent advances in probabilistic and signal processing algorithms and their application to metagenomics. *Nucleic Acids Research*, 48(10):5217–5234, 2020. doi:10.1093/nar/gkaa265.

18 Cristian Estan, George Varghese, and Michael E. Fisk. Bitmap algorithms for counting active flows on high-speed links. *IEEE/ACM Trans. Netw.*, 14(5):925–937, 2006. doi:10.1145/1217709.

19 Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. HyperLogLog: the analysis of a near-optimal cardinality estimation algorithm. In *Proceedings of the 18th International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods for the Analysis of Algorithms (AofA)*, 2007.

20 Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, 1985. doi:10.1016/0022-0000(85)90041-8.

21 Michael J. Freitag and Thomas Neumann. Every row counts: Combining sketches and sampling for accurate group-by result estimates. In *Proceedings of the 9th Biennial Conference on Innovative Data Systems Research (CIDR)*, 2019.

22 Phillip B. Gibbons and Srikanta Tirthapura. Estimating simple functions on the union of data streams. In *Proceedings 13th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 281–291, 2001. doi:10.1145/378580.378687.

23 Frédéric Giroire. Order statistics and estimating cardinalities of massive data sets. *Discret. Appl. Math.*, 157(2):406–427, 2009. doi:10.1016/j.dam.2008.06.020.

24 Ahmed Helmi, Jérémie Lumbroso, Conrado Martínez, and Alfredo Viola. Data Streams as Random Permutations: the Distinct Element Problem. In *Proceedings of the 23rd International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods for the Analysis of Algorithms (AofA)*, pages 323–338, 2012.

25 Piotr Indyk and David P. Woodruff. Tight lower bounds for the distinct elements problem. In *Proceedings 44th IEEE Symposium on Foundations of Computer Science (FOCS), October 2003, Cambridge, MA, USA, Proceedings*, pages 283–288, 2003. doi:10.1109/SFCS.2003.1238202.

26 T. S. Jayram and David P. Woodruff. Optimal bounds for Johnson-Lindenstrauss transforms and streaming problems with subconstant error. *ACM Trans. Algorithms*, 9(3):26:1–26:17, 2013. doi:10.1145/2483699.2483706.

27 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings 29th ACM Symposium on Principles of Database Systems (PODS)*, pages 41–52, 2010. doi:10.1145/1807085.1807094.

28 Kevin J. Lang. Back to the future: an even more nearly optimal cardinality estimation algorithm. *CoRR*, abs/1708.06839, 2017. arXiv:1708.06839.

29 Jérémie Lumbroso. An optimal cardinality estimation algorithm based on order statistics and its full analysis. In *Proceedings of the 21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA)*, pages 489–504, 2010.

30 Guillaume Marçais, Brad Solomon, Rob Patro, and Carl Kingsford. Sketching and sublinear data structures in genomics. *Annual Review of Biomedical Data Science*, 2(1):93–118, 2019. doi:10.1146/annurev-biodatasci-072018-021156.

31 Seth Pettie and Dingyu Wang. Information theoretic limits of cardinality estimation: Fisher meets Shannon. In *Proceedings 53rd ACM Symposium on Theory of Computing (STOC)*, 2021.

32 Seth Pettie, Dingyu Wang, and Longhui Yin. Non-mergeable sketching for cardinality estimation, 2021. arXiv:2008.08739.

33 Ninh Pham. Hybrid LSH: faster near neighbors reporting in high-dimensional space. In *Proceedings of the 20th International Conference on Extending Database Technology (EDBT)*, pages 454–457, 2017. doi:10.5441/002/edbt.2017.43.

34 Björn Scheuermann and Martin Mauve. Near-optimal compression of probabilistic counting sketches for networking applications. In *Proceedings of the 4th International Workshop on Foundations of Mobile Computing (DIALM-POMC)*, 2007.

35 Daniel Ting. Streamed approximate counting of distinct elements: beating optimal batch methods. In *Proceedings 20th ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 442–451, 2014. doi:10.1145/2623330.2623669.

36 Daniel Ting. Towards optimal cardinality estimation of unions and intersections with sketches. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1195–1204. ACM, 2016. doi:10.1145/2939672.2939772.

37 Jake Wires, Stephen Ingram, Zachary Drudi, Nicholas J. A. Harvey, and Andrew Warfield. Characterizing storage workloads with counter stacks. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 335–349, 2014.

38 Derrick E. Wood, Jennifer Lu, and Ben Langmead. Improved metagenomic analysis with Kraken 2. *bioRxiv*, 2019. doi:10.1101/762302.

39 Qingjun Xiao, Shigang Chen, You Zhou, Min Chen, Junzhou Luo, Tengli Li, and Yibei Ling. Cardinality estimation for elephant flows: A compact solution based on virtual register sharing. *IEEE/ACM Trans. Netw.*, 25(6):3738–3752, 2017. doi:10.1109/TNET.2017.2753842.