BFES: Towards Optimal Bayesian Frequency Estimation Sketches in Data-Streams

Francesco Da Dalt, *ETH Zürich*, francesco.dadalt@inf.ethz.ch
Adrian Perrig, *ETH Zürich*, aperrig@inf.ethz.ch

***Abstract*—Measuring the frequency of items in data streams is a relevant and wide-spread problem in stream analysis and Internet traffic monitoring. This paper studies the problem of sketch-based frequency estimation from a Bayesian statistics point of view which captures uncertainties regarding the frequencies of items in a more flexible and quantitative way compared to the state of the art. We design and implement, based on Markov chain Monte Carlo, a Bayesian frequency estimation sketch that provides both state of the art accuracy, as well as greater functionality compared to other sketches such as confidence bounds for arbitrary levels, and error-function aware frequency estimates. In our theoretical work we derive information-theory related equations such as the expected information gain of a sketch, as well as the optimal least-squares Bayesian frequency estimator. In benchmarks comparing the state of the art, the proposed method achieves the lowest absolute error across all real world data streams, as well as outperforming all sketches on 4 out of 5 metrics on synthetic data. We also show that our method can provide, for multiple confidence levels simultaneously, good confidence levels on both synthetic as well as real data.**

***Index Terms*—Data Stream, Sketch, Bayesian Inference**

I. Introduction

Extracting the frequency of keys from data streams is useful in many domains such as Internet traffic monitoring [1]–[4], sensor networks [5]–[7], and resource provisioning [8], [9]. Measuring the frequencies of keys in a memory- and compute-efficient manner by keeping one counter per key often incurs prohibitive spatial complexity [2]. In practice however it is often sufficient to have an approximate estimate of the frequencies of items. This observation motivated lossy compression data structures known as *sketches* where frequency counters are shared amongst the keys in the data stream. A reconstruction procedure then estimates the frequency of some key based on the contents of the shared counters.

Due to the lossy compression, random error is introduced into the frequency estimates. The State Of The Art (SOTA) addresses this issue by providing confidence intervals for the estimates. The size and confidence levels of these intervals depend strictly on the amount of counters used, as well as their structure [10], [11]. However, since the pseudorandom compression of sketches induces a loss of information, there does not exist *one* best frequency estimate for all cases. Rather, the optimality of the estimate depends on the concrete error metric which one aims to minimize [12]. While SOTA methods may be aware of this fact, no existing solution so far addressed this problem constructively since the proposed algorithms are by design invariant to the error metric by which the estimates are judged [13].

Given these observations, there are two opportunities for improvement: First, a method by which the frequency estimates produced by the sketch are influenced by the error-metric used to measure performance. Secondly, a sketch that is able to provide Confidence Intervals (CIs) of arbitrary strength, irrespective of the amount of counters and their structure. This is a special case of the first point since CIs are estimates that minimize a particular error metric [14].

Main Contributions: We approach the problem of frequency estimation from a Bayesian viewpoint: By formulating prior assumptions on the frequencies of keys as probability distributions, we derive a posterior distribution for the frequencies *after* having observed the set of counters of a sketch data structure. The posterior distribution then allows the computation of arbitrary confidence intervals, as well as error-function aware frequency estimates. We construct a concrete algorithm BFES that approximately solves the problem of Bayesian frequency estimation and we demonstrate in benchmark experiments that it improves on the state of the art in the desired aspects.

Section III-D lays the theoretic foundations of Bayesian frequency estimation using counter-based sketches in a constrained setting and states the conditions required for efficient computation. Section IV then constructs and analyzes a concrete algorithm that approximately solves the Bayesian equations and we elaborate on the relation between the Bayesian frequency estimation method and other state of the art methods. Lastly, Section V evaluates our method against SOTA sketches and we demonstrate the ability to provide flexible confidence intervals, as well as robust performance across different error metrics. Given the utility of the algorithms and to enable repeatability of results, all code is open-source [15].

II. Related Work

We abstract a data stream S as a sequence of key-value pairs $S = \langle (k_i,\ v_i) | i \in \mathbb{N} \rangle$. We use the terms "key" and "item" interchangeably. The frequency a_k of key k is the sum over all values associated with k.

The most widely adopted sketching-algorithms are the Count-Sketch (CS) [10] and CountMin-Sketch (CMS) [11]. At their core, these methods use a two-dimensional array of counters c. The counter-array c has d rows and w columns. In the case of CMS, a key-value pair $(k,\ v)$ in the stream is processed by hashing k into each row of c, using row-wise independent hash functions, and increasing the indexed counter by v. When queried, the CMS estimates the frequency of k as the minimum over the d counters indexed by k. Assuming that the frequencies of all items are non-negative, CMS provides bounds that state that the estimation error is within some ε with probability at least δ. The CS is similar to the CMS and even provides ε-δ bounds on the error when frequencies are negative. For both the CS and CMS however, the bound parameters ε and δ depend on the number of rows d and columns w. Despite the CS and CMS remaining widely used, other competitive algorithms have since been developed and advancements made came generally in one of three flavors:

1) Implementation Optimization: This class of methods augments the implementation of *existing sketches* such as CMS and CS. SALSA [16] provides a statically-sized array of counters c but with dynamically sized individual counters within c, which improves memory efficiency at the cost of increased processing needs. AUGMENTED- [17], PYRAMID- [18], and STINGY-SKETCH [19] optimize the sketch throughput by improving the cache-locality and memory management of c. These methods optimize the management of the counters c, without changing the estimation algorithm at its core.

2) Sketch Specializations: Particularly in the context of computer networks, special sketches such as ELASTIC-SKETCH [20], LOFT [21], NITROSKETCH [22], and UNIVMON [23] have been developed to address network-operation specific needs such as; dissecting network-traffic into "elephant" and "mice" flows and adaptively sub-sampling the stream S based on traffic load. Recently, approaches such as META-SKETCH [24] explored how a machine can learn to estimate frequencies based on training tasks, requiring however expensive re-training to change the sketch size.

3) Estimation Advancement: The CU-SKETCH (CUS) [2] has improved upon the CMS by introducing a policy that updates c more conservatively, and other methods such as PR- [25], SEQ- [26], and CB-SKETCHES [27] (PRS, SEQS, CBS) estimate item frequencies by solving optimization problems that aim to minimize frequency estimation error as a function of c. The DIRICHLETPROCESS-SKETCH (DPS) [28] augments the CMS and derives equations for the likelihood of item frequencies given c based on a Dirichlet process prior, which improves the expected estimation error in many cases. Followup work [29] expands on the theory of DPS by using a inverse Gaussian process prior (NIGPS).

Our work is an *estimation advancement* that approaches sketch-based frequency estimation from a new point of view. It is thus orthogonal to the SOTA related to implementation optimizations and sketch specializations. Despite considerable progress, no single SOTA method has yet been able to address the following three combined desiderata:

- Detailed Uncertainty Quantification: When a sketching algorithm is queried about the frequency a_k, it will return the estimate $\hat{a}_k$ and depending on the algorithm also a confidence interval, as is the case for the CMS and CS. The confidence level is however fixed and cannot be changed without changing the dimensions of c. The method we propose returns a *distribution* of likely values for a_k and therefore enables a more detailed and flexible quantification of uncertainty. To our knowledge the only methods that allow this are the DPS and NIGPS but these methods do not meet our next desiderata; efficiency and flexibility.
- High Efficiency: Frequency estimation algorithms need to process the data stream at a high rate in realistic scenarios. All SOTA methods provide very fast stream processing but DPS and NIGPS are comparatively slow when computing the estimates $\hat{a}_k$ due to the frequent use of expensive functions such as the Log-Gamma, as well as employing iterative optimization methods for parameter estimation.
- Adaptability to Prior Knowledge and Assumptions: In practice one often knows or assumes some characteristics about the data stream. For example the CMS assumes non-negative item frequencies, while the CBS and PRS assume to a certain degree that item-frequencies follow a normal distribution, and lastly the DPS assumes item frequencies follow a Chinese restaurant process. Such prior information can be used to increase frequency estimation accuracy. All current methods lack a principled way to support a wider range of prior assumptions. The system we propose in this paper allows multiple different prior beliefs, thereby providing greater flexibility and significant performance gains when applied properly.

This paper explains how Bayesian statistics can provide an elegant approach, both in theory and practice, towards achieving these three desiderata.

III. THEORETIC MODEL

Given a stream S, we denote with $\mathcal{K}$ the set of all keys that appear in the stream. $|\mathcal{K}| = n$ therefore indicates the number of unique keys. The goal of on-line frequency estimation is to manage a small amount of memory $m \ll n$ in order to keep track of the frequencies a_k for all keys $k \in \mathcal{K}$ given one sequential pass over the stream S. We denote with $\hat{a}_k^X$ the estimate for a_k computed by some algorithm X. To simplify notation regarding the frequencies of items, we define

$$a = \begin{bmatrix} a_{k_1} \\ \vdots \\ a_{k_n} \end{bmatrix} \text{ where } a_{k_i} \text{ is the } i\text{-th element of } \mathcal{K} \qquad (1)$$

Next, we define c as a m-dimensional vector representing the flattened sketch data structure which is populated by a frequency-estimation algorithm. For many SOTA methods such as CS and CMS, the relation between c and a can be described by the following matrix-vector operation:

$$c^X = H^X \ a \qquad (2)$$

where H^X is a matrix specific to the algorithm X. For example, in the case of CMS, $H_{i,j}^{CMS}$ is 1 if key k_j contributes to the i-th counter, and 0 otherwise. We may omit the algorithm indicator X for the sake of clarity.

A. Bayesian Theory

Frequency estimation in Bayesian statistics is based on the assumption there exist more and less likely values for a. We introduce A as the n-dimensional Random Variable (RV) from which a is assumed to be a sample of. By extension, since a is a random sample, so is c. Therefore we define the RV C from which c is an instantiation of:

$$C = H\ A \iff \mathbb{P}[C = c] = \mathbb{P}[H\ A = c]$$

Where $\mathbb{P}[Y = y]$ denotes the probability density of RV Y at y. Having defined A and C, the core idea of Bayesiansim is to ask the question: Suppose we know that $H\ A = c$ where c are some counters we have measured based on 2, how does this knowledge change our belief of A? In other words, we are looking for $A \mid H\ A = c$ which denotes the distribution of A, conditioned on the knowledge $H\ A = c$. Bayes theorem tells us that this change in belief of A is described by:

$$\mathbb{P}[A = \alpha \mid H\ A = c] = \frac{\mathbb{P}[H\ A = c \mid A = \alpha]\ \mathbb{P}[A = \alpha]}{\mathbb{P}[H\ A = c]}$$

Note that $\mathbb{P}[H\ A = c]$ is constant in α so it is only a proportionality factor and that furthermore $\mathbb{P}[H\ A = c \mid A = \alpha]$ is only non-zero when $H\ \alpha = c$. Therefore we have the following proportional relation:

$$\mathbb{P}[A = \alpha \mid H\ A = c] \propto \mathbb{1}_{\{H\alpha = c\}} \cdot \mathbb{P}[A = \alpha] \qquad (3)$$

where $\mathbb{1}$ is the indicator function. The relation makes intuitive sense as after having seen c, we only consider instantiations α of A possible, if α coincides with the counters c after passing through H.

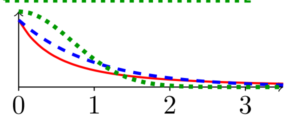
The probability measure in 3 has in general no closed form solution, however we can use computational methods to draw samples from $\mathbb{P}[A = \alpha \mid H\ A = c]$ and thereby approximate the true distribution up to arbitrary accuracy.

1) Running Example: To illustrate some core ideas presented in this paper we use a toy example with a stream S with $n = 4$ distinct keys and a sketch with $m = 2$ counters:

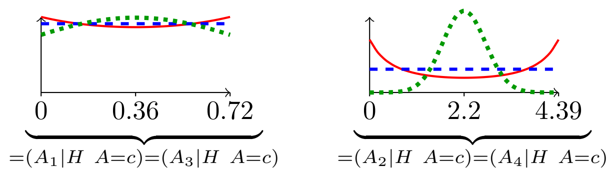
$$a = \begin{bmatrix} 0.541 \\ 0.097 \\ 0.180 \\ 4.299 \end{bmatrix} \text{ and } H = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}, \text{ thus } c = H\ a = \begin{bmatrix} 0.72 \\ 4.39 \end{bmatrix}$$

We model three different prior beliefs:

$$A = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} \text{ where } A_i \hat{=} \begin{cases} \boxed{\mathbb{P}[A_i = x] \propto (x+1)^{-2}} \\ \boxed{\mathbb{P}[A_i = x] \propto \exp(-x)} \\ \boxed{\mathbb{P}[A_i = x] \propto \exp(-x^2)} \end{cases} \qquad (4)$$



where the solid, dashed, and dotted densities illustrate three exemplary prior belief distributions – a Pareto, an exponential, and a Gaussian. Given matrix H, counters c, and prior beliefs A, the posterior beliefs of the item frequencies are given by



(5)

Sym	Definition	Sym	Definition
$\mathcal{K}$	Set of items in stream	$\mathcal{U}$	Set of possible items
n	Number of unique items	m	Number of counters
a	Item frequencies, see 1	A	The RV associated to a
c	Sketch counters, see 2	C	The RV associated to c
H	Compression matrix	N	The null-space of H
α	Samples of $A \mid HA = c$	β	Hidden variable, see 6

TABLE I: List of Symbols

where the solid, dashed, and dotted curves indicate the densities under assumption of the three different prior beliefs depicted in 4. Note that if we believe in extremely frequent items, i.e., the solid Pareto prior, then we see from the curve for $(A_4 \mid H\ A = c)$ that after having observed c we consider a_4 to be likely either very big, or very small, but unlikely to be something in between. On the other hand, conditioned on a Gaussian prior belief, our posterior belief behaves the opposite, as depicted by the dotted density function in 5.

B. Sampling

Having illustrated in the previous section what posterior beliefs convey, we now move on to the problem of how to compute these posterior distributions. In order to efficiently sample from the distribution described in 3, we must first remove $\mathbb{1}_{\{H\alpha = c\}}$. Linear algebra entails that if $n \geq m$, then $H\alpha = c$ if and only if $\alpha = H^\dagger c + N\beta$ where $H^\dagger$ is the pseudinverse of H, N is a $n \times (m - n)$ matrix representing the null-space of H, and β is a $(n - m)$ dimensional vector.

The intuition behind this decomposition of α is the following: $H^\dagger c$ captures the constraint that if α were to be the ground truth vector of frequencies a, then $H\alpha = c$. In addition, $N\beta$ provides α with $(n - m)$ degrees of freedom in which it may vary. The larger the array of counters c, the bigger m becomes, and therefore the fewer possible values α can achieve. The intuition in mathematical terms is given by:

$$\begin{aligned} \alpha = H^\dagger c + N\beta \implies H\ \alpha &= H\ (H^\dagger c + N\beta) \\ &= \underbrace{H\ H^\dagger}_{\mathbb{I}} c + \underbrace{H\ N}_{0} \beta = c \end{aligned} \qquad (6)$$

If we consider our running example from Section III-A, then

$$N = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \text{ and } H^\dagger c = \begin{bmatrix} 0 \\ 4.39 \\ 0.72 \\ 0 \end{bmatrix} \qquad (7)$$

and the intuition is that if we add some value x to a_1 and subtract x from a_3, then the array of counters c stays the same. The same happens for a_2 and a_4.

With these considerations, for any sample α' drawn from 3, there must exist a β' such that $\alpha' = H^\dagger c + N\beta'$. This correspondence is one-to-one. Hence, generating a sample α is equivalent to generating a sample β. The distribution B of the β-values is implicitly defined as:

$$\begin{aligned} \mathbb{P}[B = \beta \mid HA = c] &= \mathbb{P}[A = H^\dagger c + N\beta \mid H\ A = c] \\ &\propto \mathbb{1}_{\{H(H^\dagger c + N\beta) = c\}} \cdot \mathbb{P}[A = H^\dagger c + N\beta] \\ &= \mathbb{P}[A = H^\dagger c + N\beta] \end{aligned} \qquad (8)$$

Since the probability measure of the initial belief-distribution A is know, 8 provides us with a formula for the distribution of $(B \mid H\ A = c)$ up to an unknown scale parameter.

1) Stochastic Simulation: The primary difficulty with sampling from 8 is that the distribution is $(n-m)$-dimensional. Due to the "curse of dimensionality" [30], most sampling algorithms suffer from performance loss when applied to such high-dimensional distributions. One of the few methods known to work well even in the high-dimensional case is Markov chain Monte Carlo (MCMC). Out of the state of the art MCMC methods, we opt for Gibbs Sampling (GS) [31] over other schemes such as Hamiltonian Monte Carlo [32] and MALA [33] because GS requires no hyper-parameter tuning and puts the fewest assumptions on the distribution to be sampled.

On a high level, GS starts from an initial sample β^0 of dimension $(n-m)$. It then iteratively generates new samples β^i from β^{i-1}. This is done by first setting $\beta^i \leftarrow \beta^{i-1}$, and then changing *one* entry in β^i. Assume we change the j-th entry, then the equations describing how to sample the new j-th entry are given by

$$\mathbb{P}[\hat{B}^i = \beta^i] = \mathbb{P}[B = \beta^i \mid H\ A = c,\ \beta^i_{-j} = \beta^{i-1}_{-j}] \quad (9)$$

where $\hat{B}^i$ denotes the random distribution of β^i and the subscript $-j$ indicates all elements of a vector except for the j-th one. Note that we know the probability measure in 9 from 8. The sequence of values $\beta^0, \beta^1, \dots$ are samples drawn from the target distribution $(B \mid H\ A = c)$. These samples are however not independent from each other, but theory guarantees us that a large number of samples will follow the target distribution [34]. There are various ways of optimizing GS which we will not detail here.

2) Transforming Samples: Section III-B1 describes how to computationally evaluate the distribution of $(B \mid A\ H = c)$ using MCMC. We are however ultimately not interested in how our belief of B has changed based on $A\ H = c$, but rather what our updated belief of A is instead. In particular, we want to computationally evaluate $(A \mid A\ H = c)$. To that end, we can transform the samples $\beta^0, \beta^1, \dots$ generated in Section III-B1 into samples $\alpha^0, \alpha^1, \dots$ drawn from $(A \mid A\ H = c)$ based on the relation in 6:

$$\alpha^i = H^\dagger c + N\beta^i$$

C. Usage

Section III-B discusses how to computationally evaluate the posterior belief of A conditioned on the observed sketch counters c. We now show three concrete use cases that motivate how Bayesian posterior samples can be used in practice:

1) l_2-Optimal Frequency Estimates: Given H and some measurements c and an initial belief-distribution A, we want to find the single best estimate $\hat{a}$ that minimizes the expected loss

$$\mathbb{E}_{a\sim A}[||\hat{a} - a||_2^2 \mid H\ A = c]$$

By taking the gradient and setting it to zero we get the solution

$$\hat{a} = \mathbb{E}_{a\sim A}[a \mid H\ A = c] \approx x^{-1} \sum_{i=1}^{x} \alpha^i$$

which means that we can approximate the optimal solution by taking the average over the generated samples α^i. The more samples we use, the more accurate the prediction will be. Note that this technique of expected loss minimization can be applied to any loss function, not only the l_2 norm. For example, the single estimate that minimizes the expected l_1 loss is given by the median over the samples α^i, instead of the mean as is the case with l_2. Additionally, we can estimate the expected error of our frequency estimates in a similar fashion.

2) Confidence Intervals for Item Frequencies: Given some $p \in [0; 1]$, an initial belief A for a, and sketch counters c, we want to know for every $k \in \mathcal{K}$ a confidence interval $[l_k;\ u_k]$ such that $a_k \in [l_k;\ u_k]$ with probability p. A solution to this is given by:

$$l_k = Q_{A|A\ H=c}\Big(\frac{1-p}{2}\Big) \approx Q_{\{\alpha_k^1, \alpha_k^2, \dots\}}\Big(\frac{1-p}{2}\Big)$$
$$u_k = Q_{A|A\ H=c}\Big(\frac{1+p}{2}\Big) \approx Q_{\{\alpha_k^1, \alpha_k^2, \dots\}}\Big(\frac{1+p}{2}\Big)$$

where $Q_X(q)$ is the q-th quantile of X. We can therefore leverage the samples α^i to compute Confidence Intervals (CIs) for arbitrary confidence levels p. With respect to the running example, the centered 90% CI for the item frequencies a_3 and a_4 are given by:

$$\hat{a}_2^{90\%} = \hat{a}_4^{90\%} = \boxed{[0.1;\ 4.29]}\ ,\ \boxed{[0.22;\ 4.17]}\ ,\ \boxed{[1.38;\ 3.02]}$$

3) Frequency Estimates for Asymmetric Losses: In some settings, overestimating the frequency a_k of an item k incurs a different cost compared to underestimating it. Concretely, assume the loss between a_k and $\hat{a}_k$ is given by

$$l(a_k, \hat{a}_k) = |a_k - \hat{a}_k| \cdot c_1 \cdot \mathbb{1}_{a_k > \hat{a}_k} + |a_k - \hat{a}_k| \cdot c_2 \cdot \mathbb{1}_{a_k < \hat{a}_k}$$

Here, overestimations are weighted by c_2 while underestimations are weighted by c_1. Based on an initial belief A and the observed counters c, we can find the single best $\hat{a}_k$ that minimizes the expected asymmetric loss:

$$\begin{aligned}
&\hat{a}_k = \underset{x}{argmin}\ \mathbb{E}_{a_k\sim A_k}[l(a_k, x) \mid A\ H = c] \\
\iff &\frac{\partial}{\partial \hat{a}_k}\mathbb{E}_{a_k\sim A_k}[l(a_k, \hat{a}_k) \mid A\ H = c] = 0 \\
\iff &c_2 \cdot \mathbb{P}[A_k < \hat{a}_k] - c_1 \cdot \mathbb{P}[A_k > \hat{a}_k] = 0 \\
\iff &\mathbb{P}[A_k < \hat{a}_k] = \Big(\frac{c_2}{c_1} + 1\Big)^{-1} \\
\implies &\hat{a}_k \approx Q_{\{\alpha_k^1, \alpha_k^2, \dots\}}\Big(\Big(\frac{c_2}{c_1} + 1\Big)^{-1}\Big)
\end{aligned}$$

Note that if $c_1 = \infty$ we conceptually recover the COUNTMIN-SKETCH. Concretely, let for example $c_1 = 10$ and $c_2 = 1$, i.e. underestimations are $10\times$ worse than overestimations. Then, for a_2 and a_4 in our toy example from Section III-A we find

$$\hat{a}_2 = \hat{a}_4 = \boxed{4.17681}\ ,\ \boxed{3.99}\ ,\ \boxed{2.8659}$$

D. Analysis

Before moving on to the practical application of the ideas presented in this section, we analyse the behavior of a hypothetical optimal Bayesian Frequency Estimation (BFE) algorithm in three aspects: accuracy, optimality, and information gain. In this subsection we consider the simplifying assumption that the set of distinct keys $\mathcal{K}$, and therefore H, is known, as well as that $H^\dagger$ exist, i.e. that H has full row-rank.

Theorem 1 (l_2-optimal Prediction Performance)**.** *Assume the vector of item frequencies a is an instance of a random variable A. Furthermore, we are given a prior belief $\tilde{A}$ which may be different from A, as well as H, and an array of counters $c = Ha$. Then, the estimate $\hat{a}$ that minimizes the expected error in l_2-norm is equal to:*

$$\hat{a} = \underset{x}{argmin}\ \mathbb{E}[||x-a||_2^2] = H^\dagger c + (\mathbb{I} - H^\dagger H)\cdot \mathbb{E}[\tilde{A}]$$

and the total square error is characterized by

$$\begin{aligned}||\hat{a}-a||_2^2 &= ||(\mathbb{I}-H^\dagger H)(\mathbb{E}[\tilde{A}]-a)||_2^2\\ &\leq (n-m)||\mathbb{E}[\tilde{A}]-a||_2^2\end{aligned}$$

where $n \geq m$ are the number of columns and rows of H respectively.

Therefore, there are two ways to reduce the estimation error. The first is that we can increase the dimension m of the array of counters c up to the point where $n = m$ and we achieve zero error since we are using as many counters as there are items in the stream. Secondly, we can set $\mathbb{E}[\tilde{A}]$ such that it reduces the error.

Proof. We make use of 6 and compute

$$\begin{aligned}\hat{a} &= \mathbb{E}[\tilde{A} \mid H\tilde{A} = c]\\ &= \int x \cdot \frac{\mathbb{P}[\tilde{A}=x]\mathbb{P}[H\tilde{A}=c \mid \tilde{A}=x]}{\mathbb{P}[H\tilde{A}=c]}dx\\ &= \int (H^\dagger c + Nb)\cdot \frac{\mathbb{P}[\tilde{A}=(H^\dagger c+Nb)]}{\mathbb{P}[H\tilde{A}=c]}db\\ &= H^\dagger c + N\int b\cdot\frac{\mathbb{P}[N^\dagger(\tilde{A}-H^\dagger c)=b]}{\mathbb{P}[H\tilde{A}=c]}db\\ &= H^\dagger c + N\ \mathbb{E}[N^\dagger(\tilde{A}-H^\dagger c)]\\ &= H^\dagger c + (\mathbb{I}_n - H^\dagger H)\mathbb{E}[\tilde{A}]\end{aligned}$$

We omitted the scaling factor $\sqrt{|H^{\dagger T}H^\dagger|}$ from the change of variables formula as it cancels out. By using the Frobenius-norm matrix-vector inequality and its relation to the trace function we derive

$$\begin{aligned}||\hat{a}-a||_2^2 &= ||(\mathbb{I}_n - H^\dagger H)(\mathbb{E}[\tilde{A}]-a)||_2^2\\ &\leq ||\mathbb{I}_n - H^\dagger H||_F^2||\mathbb{E}[\tilde{A}]-a||_2^2\\ &= (trace(\mathbb{I}_n) - 2\ trace(H^\dagger H) +\\ &\qquad trace(H^\dagger H H^\dagger H))||\mathbb{E}[\tilde{A}]-a||_2^2\\ &= (n-m)||\mathbb{E}[\tilde{A}]-a||_2^2\end{aligned}$$

We made use of the fact that $(H^\dagger H)^T = H^\dagger H$. This concludes the proof. □

Theorem 2 (Optimality Gap)**.** *Assume the vector of item frequencies a is an instance of a random variable A. Furthermore, we are given a prior belief $\tilde{A}$ which may be different from A, as well as H, and an array of counters $c = Ha$. The expected loss of information due to using $\tilde{A}$ opposed to the optimal ground-truth prior A is given by*

$$\begin{aligned}&\mathbb{E}_{c\sim HA}[KL(A \mid HA = c \mid\mid \tilde{A} \mid H\tilde{A} = c)]\\ &\quad = KL(A \mid\mid \tilde{A}) - KL(HA \mid\mid H\tilde{A})\end{aligned}$$

where $KL(X \mid\mid Y)$ is the Kullback-Leibler divergence between distribution X and Y, also known as relative entropy, defined as

$$KL(X \mid\mid Y) = \int \mathbb{P}[X=x]\log\big(\frac{\mathbb{P}[X=x]}{\mathbb{P}[Y=x]}\big)dx$$

This provides two insights: If H is the identity matrix, the information loss is zero no matter the prior belief $\tilde{A}$ since we have as many counters as items in the stream and hence the prior is of no relevance. Secondly, for a fixed m and assuming the frequencies of items are independent from each other, as $n \to \infty$, HA and $H\tilde{A}$ will both converge to normal distributions by the central limit theorem. Therefore if the moments of HA and $H\tilde{A}$ match, the KL divergence between the two goes to zero, meaning that the majority of information loss is due to the divergence between A and $\tilde{A}$. This shows that the use of prior knowledge can greatly reduce the difference between the best possible estimate and the computed estimates. We now prove the theorem.

Proof.

$$\begin{aligned}&\mathbb{E}_{c\sim HA}[KL(A \mid HA = c \mid\mid \tilde{A} \mid H\tilde{A} = c)]\\ &= \int \mathbb{P}[HA=c]\int \mathbb{P}[A=x \mid HA=c]\\ &\quad \cdot\log(\frac{\mathbb{P}[A=x \mid HA=c]}{\mathbb{P}[\tilde{A}=x \mid H\tilde{A}=c]})dxdc\\ &= \int \mathbb{P}[HA=c]\int \mathbb{P}[A=x \mid HA=c]\cdot\log\left(\frac{\mathbb{P}[A=x]}{\mathbb{P}[\tilde{A}=x]}\right)dxdc\\ &+ \int \mathbb{P}[HA=c]\int \mathbb{P}[A=x \mid HA=c]\cdot\log\left(\frac{\mathbb{P}[H\tilde{A}=c]}{\mathbb{P}[HA=c]}\right)dxdc\\ &= \int\underbrace{\int \mathbb{P}[HA=c]\mathbb{P}[A=x \mid HA=c]dc}_{=\mathbb{P}[A=x]}\cdot\log\left(\frac{\mathbb{P}[A=x]}{\mathbb{P}[\tilde{A}=x]}\right)dx\\ &+ \int \mathbb{P}[HA=c]\log\left(\frac{\mathbb{P}[H\tilde{A}=c]}{\mathbb{P}[HA=c]}\right)\underbrace{\int \mathbb{P}[A=x \mid HA=c]dx}_{=1}dc\\ &= KL(A \mid\mid \tilde{A}) - KL(HA \mid\mid H\tilde{A})\end{aligned}$$

Bayes theorem applied to Line 2 yields Line 3. This concludes the proof. □

Theorem 3 (Expected Information Gain)**.** *Assume the vector of item frequencies a is an instance of a random variable A. Furthermore, we are given a prior belief $\tilde{A}$ which may be different from A, as well as H. The expected amount of information gained by observing the array of counters $c = Ha$ is equal to*

$$\begin{aligned}&\mathbb{E}_{c\sim HA}[KL(\tilde{A} \mid H\tilde{A} = c \mid\mid \tilde{A})]\\ &\quad = Entropy(|HH^T|^{\frac{-1}{2m}}HA) + KL(HA \mid\mid H\tilde{A})\end{aligned}$$

where we use the following definition of information gain for X observing Y

$$KL(X \mid Y \mid\mid X)$$

First, the scaling factor $|HH^T|^{\frac{-1}{2m}}$ arises in order for the transformation H to be volume-preserving. For example if H is a semi-orthogonal matrix, then $|HH^T|^{\frac{-1}{2m}} = 1$. We omit this scaling factor from the discussion as it is of no immediate relevance. Let us now consider a special case of the theorem: If we are using a perfect prior belief, i.e. $A = \tilde{A}$, then the expected information gain is equal to the entropy of HA. To our knowledge there is no closed form expression for the entropy of HA. However, if we study a toy example in which $A = \mathcal{N}(\mu, \sigma^2 \mathbb{I})$ is a n-dimensional normal distribution and H is semi-orthogonal, then the entropy of HA is given by

$$Entropy(HA) = \frac{m}{2}\log(2\pi e \sigma^2)$$

and therefore the information gain is linear in m, the dimension of c. If $m = n$ then the entropy of HA is equal to the entropy of A and thus we gain as much information as there is entropy, i.e., we have gained all possible information since we use as many counters as there are items in the stream.

More generally, if $A \neq \hat{A}$, then the information gain of observing c is *greater* than if $A = \hat{A}$. The reason is that observing c not only gives us some baseline information, but it additionally implicitly corrects our wrong prior belief $\tilde{A}$. Thus in a sense, if $A \neq \hat{A}$, we have *more to learn* and thus every counter c we observe contains more information.

Proof.

$$\begin{aligned}
&KL(\tilde{A} \mid H\tilde{A} = c \mid\mid \tilde{A}) \\
&= \int \mathbb{P}[\tilde{A} = x \mid H\tilde{A} = c] \cdot \log\big(\frac{\mathbb{P}[\tilde{A} = x \mid HA = c]}{\mathbb{P}[\tilde{A} = x]}\big) dx \\
&= \int \frac{\mathbb{P}[\tilde{A} = x]\mathbb{P}[H\tilde{A} = c \mid \tilde{A} = x]}{\mathbb{P}[H\tilde{A} = c]} \cdot \log\big(\frac{\mathbb{P}[H\tilde{A} = c \mid \tilde{A} = x]}{\mathbb{P}[H\tilde{A} = c]}\big) dx \\
&= \int \frac{\mathbb{P}[\tilde{A} = H^\dagger c + Nb]}{\mathbb{P}[H\tilde{A} = c]} \cdot \log\big(\frac{\sqrt{|H^{\dagger T} H^\dagger|}}{\mathbb{P}[H\tilde{A} = c]}\big) db \\
&= \log\big(\sqrt{|H^{\dagger T} H^\dagger|}\big) - \log(\mathbb{P}[H\tilde{A} = c])
\end{aligned}$$

$$\begin{aligned}
\implies &\mathbb{E}_{c \sim HA}[KL(\tilde{A} \mid H\tilde{A} = c \mid\mid \tilde{A})] \\
&= \mathbb{E}_{c \sim HA}[-\log(\mathbb{P}[H\tilde{A} = c])] + \log\big(\sqrt{|H^{\dagger T} H^\dagger|}\big) \\
&= \mathbb{E}_{c \sim HA}[-\log(\frac{\mathbb{P}[H\tilde{A} = c]}{\mathbb{P}[HA = c]}) - \log(\mathbb{P}[HA = c])] \\
&\quad - \log\big(\sqrt{|(H^{\dagger T} H^\dagger)^{-1}|}\big) \\
&= Entropy(HA) + KL(HA \mid\mid H\tilde{A}) - \log\big(\sqrt{|HH^T|}\big)
\end{aligned}$$

By picking $H' = |HH^T|^{\frac{-1}{2m}} H$ and inserting it back into the derivation, the logarithmic term vanishes. The factor $\sqrt{|H^{\dagger T} H^\dagger|}$ arises due to the metric tensor that ensures correct integration over the m-dimensional subspace in which $A \mid AH = c$ lives. We made use of the fact that $|H^{\dagger T} H^\dagger| = |(HH^T)^{-1}|$ which can be derived using the singular value decomposition of H. This concludes the proof.

□

IV. Algorithmic Implementation

This section constructs an approximate Bayesian Frequency Estimation Sketch (BFES) based on the ideas presented in the previous section. There are three primary issues with the basic idea from Section III that need to be solved in order to make the sketch generally applicable: Firstly, how to address the case when keys are not known in advance. Secondly, what should the prior A be. Lastly, how do we make Gibbs sampling efficient.

A. Unknown Keys

The model in Section III requires H to have as many columns as there are possible keys $k \in \mathcal{K}$. Assuming $\mathcal{U}$ is the set of all possible keys, such as for example the set of 64-bit integers or 128-bit IP addresses, then $\mathcal{K}$ is an a-priori unknown subset of $\mathcal{U}$ defined as

$$\mathcal{K} = \{k \in \mathcal{U} \mid a_k \neq 0\} \subset \mathcal{U}$$

Clearly, it is not computationally feasible for H to have $N = |\mathcal{U}|$ many columns. Instead, we perform an initial projection step which maps the space $\mathcal{U}$ down onto a smaller space $\{0, \dots, d-1\}$ where a key $k \in \mathcal{U}$ is projected by the function p:

$$p(k):\ \mathcal{U} \to \{0, \dots, d-1\}$$

Concretely, p may for example be any kind of reasonable pseudo-random hash function modulo d. For the sake of notation, we can write p as a projection-matrix P where

$$P_{i,j} = \mathbb{1}_{\{p(j) = i\}}$$

Therefore the augmented model looks like

$$\begin{aligned}
\tilde{a} &= P\ a & \tilde{A} &= P\ A \\
c &= H\ \tilde{a} & C &= H\ \tilde{A} = H\ P\ A
\end{aligned}$$

where H is a *known* matrix and P is an *unknown* matrix implicitly defined by the hash function p. This problem factorization allows us to apply a two-step approach towards solving Bayesian frequency estimation:

1) First, draw samples $\tilde{\alpha}^1,\ \tilde{\alpha}^2, \dots$ that approximate the distribution $\tilde{A} \mid H\tilde{A} = c$ using Gibbs sampling as described in Section III-B1.
2) Secondly, given some key k, for each $\tilde{\alpha}^j$ draw samples ${}^j\alpha_k^1, {}^j\alpha_k^2, \dots$ that approximate the distribution $A_k \mid P\ A = \tilde{\alpha}^j$ using Gibbs sampling.
3) Then, the set $\bigcup_j \{{}^j\alpha_k^1, {}^j\alpha_k^2, \dots\}$ approximates the distribution $A_k \mid H\ P\ A = c$.

Since P must in general be regarded as a random matrix, the probability $\mathbb{P}[A_k = {}^j\alpha^i \mid P\ A = \tilde{\alpha}^j]$ can in general not be directly computed efficiently as in Section III. Instead, we perform an approximation:

$$\begin{aligned}
\text{Let } k' = p(k), \text{ then } &\mathbb{P}[A_k = {}^j\alpha^i \mid P\ A = \tilde{\alpha}^j] \\
&= \mathbb{P}\Big[A_k = {}^j\alpha^i \mid \bigwedge_{l=0}^{d-1} (P\ A)_l = \tilde{\alpha}_l^j\Big]
\end{aligned}$$

Algorithm 1 Insertion subroutine

```
1: $p \leftarrow PsuedoRandomHash(\mathcal{U} \to \{0,\dots,\ d-1\})$
2: $h \leftarrow KnownHash(\{0,\dots,\ d-1\} \to \{0,\dots,\ m-1\})$
3: $c \leftarrow ZeroVec(m)$
4: $c_{total} \leftarrow 0$
5: $s \leftarrow CountUniqueItemsSketch$
6: procedure INSERT(key $k$, value $v$, hash $p$, hash $h$)
7:     $k' \leftarrow p(k)$   ▷ $p$ is the hash mapping to $\{0,\dots,d-1\}$
8:     $index\_list \leftarrow h(k')$   ▷ $h$ represents the matrix $H$
9:     $c[index\_list].add(v)$
10:    $c_{total}.add(v)$   ▷ Needed for $\tilde{\mu}_A$
11:    $s$.INSERT($k$)
```

Algorithm 2 Prior adjustment

```
1: procedure PREPAREPRIOR(distrib $A$, distrib $\tilde{A}$)
2:     $\tilde{n} \leftarrow s.count()$
3:     $\tilde{\mu}_A \leftarrow c_{total}/\tilde{n}$
4:     $A \leftarrow fitScaleToMean(\tilde{\mu}_A)$
5:     $\tilde{\mu}_{\tilde{A}} \leftarrow \tilde{\mu}_A \cdot \tilde{n}/d$
6:     $\tilde{\sigma}^2_{\tilde{A}} \leftarrow (\tilde{n}/d)^2 \cdot \mathbb{V}[A] + \tilde{n} \cdot (d-1)/d^2 \cdot \mathbb{E}[A]$
7:     $\tilde{A} \leftarrow fitToMeanAndVariance(\tilde{\mu}_{\tilde{A}}, \tilde{\sigma}^2_{\tilde{A}})$
```

$$\approx \mathbb{P}[A_k = {}^j\alpha^i \mid (P\ A)_{k'} = \tilde{\alpha}^j_{k'}]$$

$$\text{Where } (P\ A)_{k'} = A_k + \sum_{x=1}^{Q} A_{coll(k,x)}$$

In this notation, $u = coll(k, x)$ is the x-th key which collides with k, i.e., $p(u) = p(k)$. The number of collisions Q is in general unknown and therefore Q is a random variable. In theory, if p is a perfect hash, Q will be binomially distributed. To allow for efficient computation of the density of $(P\ A)_{k'}$ we approximate it by a proxy-distribution $\widetilde{(P\ A)}_{k'}$;

$$(P\ A)_{k'} \approx \widetilde{(P\ A)}_{k'} = A_k + \sum_{x=1}^{f} X_x \tag{10}$$

where we have substituted the sum over an unknown number of RVs by a fixed finite sum over f RVs. f is a user-defined parameter. Upcoming sections explain how X_x is defined but the main takeaway is that for pragmatic reasons we approximate

$$P[A_k = {}^j\alpha^i \mid P\ A = \tilde{\alpha}^j] \approx \mathbb{P}[A_k = {}^j\alpha^i \mid \widetilde{(P\ A)}_{k'} = \tilde{\alpha}^j_{k'}]$$

which allows us to apply default Gibbs sampling as described in Section III-B1 to draw samples ${}^j\alpha^1,\ {}^j\alpha^2,\dots$. We would like to point out that these simplifications imply that the frequency estimates only *approximate* the optimal method analyzed in Section III-D.

B. Prior Belief in Practice

A user providing some prior belief may know the shape of A, but not its concrete mean or variance. To fully specify A we therefore need to fix its scale, which we accomplish by making use of a secondary sketch s that estimates the

Algorithm 3 Query subroutine

```
1: $(w_1,\ w_2) \leftarrow numGibbsSamples$   ▷ $w_1 \cdot w_2$ is the total
   number of Gibbs samples.
2: procedure QUERY(key $k$, distrib $A$, distrib $\tilde{A}$)
3:     $k' \leftarrow p(k)$
4:     PREPAREPRIOR($A$, $\tilde{A}$)
5:     $fstLvlSmpls \leftarrow gibbsSample(\tilde{A}_{k'} \mid H\ \tilde{A} = c,\ w_1)$
6:             ▷ Returns the list of samples $\{\tilde{\alpha}^1_{k'},\dots,\ \tilde{\alpha}^{w_1}_{k'}\}$
7:     $totalSamples \leftarrow EmptyList(w_1 \cdot w_2)$
8:     for $\tilde{\alpha}^i_{k'}$ in $fstLvlSmpls$ do
9:         PREPAPPROXPRIOR($\widetilde{(P\ A)}_{k'},\ \tilde{\alpha}^i_{k'}$)
10:        $sndLvlSmpls \leftarrow gibbsSample(A_k \mid \widetilde{(P\ A)}_{k'} =$
           $\tilde{\alpha}^i_{k'},\ w_2)$
11:            ▷ Returns the list of samples $\{{}^i\alpha^1_k,\dots,\ {}^i\alpha^{w_2}_k\}$
12:        $totalSamples.append(sndLvlSmpls)$
13:    return $(fstLvlSmpls,\ totalSamples)$
```

Algorithm 4 Auxiliary prior adjustment

```
1: procedure PREPAPPROXPRIOR(distrib $\widetilde{(P\ A)}_{k'}$,
   scalar $\tilde{\alpha}^i_{k'}$)
2:     $X_1,\dots,\ X_f \leftarrow getFrom(\widetilde{(P\ A)}_{k'})$
3:     $p_{no\ collision} \leftarrow estimateNoCollProb(\tilde{n},\ d,\ \tilde{\alpha}^i_{k'})$
4:     $\mathbb{P}[\widetilde{(P\ A)}_{k'} = A_k] \leftarrow p_{no\ collision}$
5:     $\mu,\ \sigma^2 \leftarrow estMeanAndVariance(\widetilde{(P\ A)}_{k'} -$
   $A_k,\ \tilde{\alpha}^i_{k'})$
6:     $X_1,\dots,\ X_f \leftarrow fitToMeanAndVariance(\mu,\ \sigma^2)$
```

number of unique items in the stream n and we then estimate the average item frequency $\tilde{\mu}_A$ based on s. We then adjust the scale of A such that $\mathbb{E}[A] = \tilde{\mu}_A$. The complete insertion routine is given in Alg. 1.

When a user queries the Bayesian frequency estimator sketch and requests samples that approximate the posterior belief of a_k for some k, a number of statistics have to be adjusted before Gibbs sampling can commence. This is described in Alg. 2. The mean $\tilde{\mu}_{\tilde{A}}$ and variance $\tilde{\sigma}^2_{\tilde{A}}$ are computed based on the law to total expectation and total variance. In particular, $\tilde{\sigma}^2_{\tilde{A}}$ captures both the variance of A as well as the variance due to pseudo-random hashing.

Alg. 3 then shows the query subroutine where the user provides some key k and prior beliefs A and $\tilde{A}$, and the function returns samples that approximate the posterior belief of the frequency a_k of item k. We employ the two-stage Gibbs sampling approach described in the preceding section.

An important aspect regarding QUERY is the auxiliary function PREPAPPROXPRIOR (see Alg. 4) that prepares $\widetilde{(P\ A)}_{k'}$ for the second stage of Gibbs sampling. This function explicitly estimates the probability that key k has no collisions with other keys as this edge-case represents a discontinuity and cannot be well-approximated by the sum of continuous RVs $X_1,\dots,\ X_f$ from 10. In practice, this is only relevant when $n \approx d$.

C. Efficient Gibbs Sampling

The last aspect left to discuss are the requirements for performing fast Gibbs sampling in QUERY. As described in Section III-B1, the requirement for Gibbs sampling to generate samples from the distribution $A \mid H\ A = c$ is that we must be able to draw samples from the distribution

$$\begin{aligned}&\mathbb{P}[B = \beta^{i,j} \mid H\ A = c,\ \beta^{i,j}_{-j} = \beta^{i,j-1}_{-j}]\\ &\propto \mathbb{P}[A = H^\dagger c + N\beta^{i,j} \mid \beta^{i,j}_{-j} = \beta^{i,j-1}_{-j}]\\ &= \mathbb{P}[A = H^\dagger c + N(\beta^{i,j-1} + e_j\Delta)]\end{aligned}$$

where e_j is the j-th unit vector and $e_j\Delta$ is therefore the exact difference between the previous Gibbs sample $\hat{\beta}^{i,j-1}$ and $\hat{\beta}^{i,j}$. Therefore, we only need to sample Δ. Let D denote the RV related to Δ, then

$$\begin{aligned}\mathbb{P}[D = \Delta] &= \mathbb{P}[A = H^\dagger c + N(\hat{\beta}^{i,j-1} + e_j\Delta)]\\ &= \mathbb{P}[A - H^\dagger c - N\hat{\beta}^{i,j-1} = N_{:,j}\Delta)]\\ &\propto \prod_{l=1}^{d} \mathbb{P}[(A - H^\dagger c - N\hat{\beta}^{i,j-1})_l = N_{l,j}\Delta)] \quad (11)\end{aligned}$$

where $N_{:,j}$ is the j-th column of N. Note that if $N_{l,j} = 0$, then the l-th term in the product of 11 is constant with respect to Δ and therefore irrelevant. Hence we can reduce the computational cost of sampling D by reducing the number of nonzero elements in N. We achieve this by smartly choosing H such that both H and its nullspace N are sparse. In particular we construct H and N based on Low-Density Parity-Check (LDPC) [35] codes, inspired by the first use of LDPC codes in the domain of compressive sensing which is closely related to the field of sketching algorithms [36].

The discussion until now primarily regards the first-stage Gibbs sampling from Line 5 in QUERY. The sampling process in the second stage (Line 10) is analogous to the first stage except that the matrix H is different and dependent on f. The degrees of freedom in the second stage of GS are $f - 1$ and the density of D, from which we sample our Δ's, is proportional to the product of two densities, which ensures low computational cost.

D. Complexity Analysis

We now analyze the computational complexity of BFES. First, we assume that A and $\tilde{A}$ may either be a Pareto, exponential, or Gaussian distribution. This limits the choice of prior beliefs to a range of distributions for which we can ensure that Gibbs sampling can be done rapidly, while still leaving enough room to model different beliefs that capture distinct behaviors. In theory arbitrary prior beliefs, such as Gaussian-Mixture-Models, can be chosen at the cost of higher computational complexity of the query procedure.

Secondly, we have parametrized H and N by an integer u which specifies the length of the index list in Line 8. Informally speaking, u is analogous to the "depth" parameter from the COUNT- or COUNTMIN-SKETCH. Due to the construction of H with LDPC codes, N is guaranteed to have 2^u non-zero entries per column, and d is equal to $(m - 2\cdot(u-1))\cdot 2^{u-1}$. For details we refer to our implementation.

1) Memory Complexity: Sketch algorithms work with a limited amount of memory M that we assume to be given. Naturally, the INSERT procedure of the Bayesian frequency sketch as proposed in this section has an on-line memory complexity of

$$\mathcal{O}(M) = \mathcal{O}(m' + m)$$

where m' is the size of the sketch s used to count the number of distinct keys, and m is the size of the array of counters c. In our implementation, given M, we set $m' \in \mathcal{O}(\log(M))$ and $m = M - m'$. In practice, when employing the HYPERLOGLOG-HIP estimator [37]–[39] in our experiments, the sketch s requires a very small amount ($< 1\%$) of memory compared to m. In benchmarks, M is enforced to be identical to all other sketches.

2) Insertion Complexity: The complexity of INSERT is given by

$$\mathcal{O}(\log(m') + u)$$

if we assume constant-time hashing. The term $\log(m')$ captures the computational complexity of updating the auxiliary sketch s which has size m'. Computing the list of indices in Line 8 of INSERT can be done in $\mathcal{O}(u)$ time as well. Therefore, the insertion complexity of BFES is comparable to SOTA methods such as COUNTMIN- and COUNT-SKETCH in terms of on-line computational requirements. Empirical evaluations in Section V-E demonstrate that also in practical deployment BFES matches SOTA insertion throughput.

3) Query Complexity: The cost of computing a single Bayesian frequency query according to QUERY is in

$$\begin{aligned}&\mathcal{O}(w_1 \cdot (||N||_0 + w_2 \cdot f) + \chi_{solve\_init})\\ &= \mathcal{O}(w_1 \cdot ((d-m)2^u + w_2 \cdot f) + \chi_{solve\_init})\end{aligned}$$

The terms w_1 and w_2 come from the number of first- and second-stage Gibbs samples that have to be computed. χ_{solve_init} is the cost of finding an initial point $\tilde{\alpha}^0$ for the first-stage Gibbs sampling and varies between solvers and problem parameters. $||N||_0$ is the cost of producing one first-stage Gibbs sample and f is the cost of producing one second-stage Gibbs sample. Additionally, $||N||_0$ is the number of non-zero entries of N and as explained in Section III-B, N has $(d-m)$ columns and therefore since every column has at most 2^u nonzero entries, the total number of non-zero entries of N is bounded by $(d-m)2^u$.

Considering these asymptotic costs, u has to be chosen small in order to remain efficient and we have done benchmarks with u equal to 1, 2, and 4. If $u = 1$ then χ_{solve_init} can be omitted since the solution to the optimization problem is trivial and the computational cost of QUERY becomes

$$\mathcal{O}(w_1 \cdot w_2 \cdot f)$$

where $w_1 \cdot w_2$ is the total number of samples generated. These samples may then be used as described in Section III-C. In

Section V we show that BFES produces good results at low cost for different values of u.

E. Relation to other Sketches

Before moving on to benchmarks, we will position the proposed Bayesian Frequency Estimation (BFE) in relation to other relevant SOTA techniques.

1) CountMin-Sketch: As stated in Section III-C, the CMS [11] is from a theory point of view equivalent to the optimal Bayesian frequency estimator with asymmetric loss that penalizes underestimations by an infinite factor. The implicit prior used by the CMS is that the frequencies of items are strictly non-negative. Furthermore, the CMS provides error bounds which, for one specific instance of the sketch, state that with some fixed probability δ, the error is less than some ϵ times $||a||_1$. There are two differences to Confidence Intervals (CIs) computed based on BFE:

1) BFE-CIs can be computed for arbitrary confidence levels without alterations needed to the sketch. The CMS can only ever give performance bounds for one specific choice of δ and ϵ.
2) BFE-CIs depend on the prior belief provided, while the CMS has a fixed incorporated prior belief.

2) CountBayes-Sketch: The CBS [27] computes Maximum-A-Posteriori (MAP) estimates for frequencies of items based on the Central-Limit-Theorem (CLT). The CBS computes these estimates in closed form and prior beliefs can be provided in form of a mean μ and an uncertainty-factor χ^2. In contrast to BFE it does not allow estimating expected errors or computing estimates that minimize a specific error function.

3) PR-Sketch: At its core, the PRS [25] computes frequency estimates $\hat{a}^{PRS}$ based on

$$\hat{a}^{PRS} = \underset{a}{argmin}\ ||a||_2 \text{ such that } H^{PRS}a = c^{PRS}$$

Implicitly, $\hat{a}^{PRS}$ is also the solution to the following two optimization problems:

$$\text{Let } A \sim \mathcal{N}(\vec{0}, \sigma^2\mathbb{I})$$
$$\hat{a}^{PRS} = \underset{a}{argmax}\ \mathbb{P}[A = a \mid H^{PRS}A = c^{PRS}]$$

i.e., $\hat{a}^{PRS}$ is the MAP-estimate for a assuming a Gaussian prior. Since the mode of a Gaussian is also its mean and therefore least-squares estimator , on a high level the PRS is a special instance of BFE with Gaussian prior beliefs.

4) Seq-Sketch: Fundamentally, the frequency-estimate $\hat{a}^{Seq}$ computed by the SEQS [26] is the solution to the following problem:

$$\hat{a}^{Seq} = \underset{a}{argmin}\ ||a||_1 \text{ such that } H^{Seq}a = c^{Seq}$$

This is known as the Basis Pursuit (BP) problem [40] which has been shown [41] to be equivalent to the solution of

$$\text{Let } A \sim \mathcal{L}aplace(\vec{0}, \sigma^2\mathbb{I})$$
$$\hat{a}^{Seq} = \underset{a}{argmax}\ \mathbb{P}[A = a \mid H^{Seq}A = c^{Seq}]$$

i.e., the MAP-estimate for a assuming a Laplacian prior. While BFES, as presented in this paper, is unable to compute MAP-estimates directly, SEQS can nevertheless be conceptually understood as a special case of BFE with a Laplace prior belief.

5) DirichletProcess-Sketch: The DPS [28] assumes that the item frequencies are drawn from a Dirichlet process (DP). The DP models both the prior assumption of the frequencies of items, as well as a prior on the number of distinct items that exist. In comparison, the optimal Bayesian frequency estimator as described by Section III requires to know the set of distinct keys $\mathcal{K}$, while BFES uses an auxiliary sketch [39] to estimate the size of $\mathcal{K}$. Furthermore, DPS provides a closed-form *theoretic solution* to the frequency estimation problem, while BFES presents a *computational method*. Evaluating DPS requires many evaluations of the Gamma function as well as solving an optimization problem and is therefore in empirical experiments slower than BFES. Lastly, DPS only allows for one prior belief – the Dirichlet process – while BFE supports a wider range of prior beliefs such as Gaussians, exponentials, and Paretos.

V. BENCHMARKS AND EVALUATION

After discussing the theoretical aspects of Bayesian frequency estimation, we will now evaluate the performance of BFES in relation to other SOTA methods on both synthetic and real-world data traces, comparing both prediction accuracy as well as computational cost. In particular, we compare against all methods described as *estimation-advancements* in Section II, except for NIGPS which has shown to be computationally too expensive in our experiments.

A. Experiment Details and Notation

Regarding BFES, we measure its performance using three different priors; a Pareto (BFESP), an exponential (BFESE), and a Gaussian (BFESG), to demonstrate that the prior belief affects the frequency estimates. When we write BFES^u, u indicates the sketch parameter as described in Section IV-D. Details aside, u is the depth of the multi-level array of counters that comprise the primary sketch data structure.

Furthermore, BFES has been configured to generate 100 samples per query, i.e., $w_1 \cdot w_2 = 100$. Additionally, the algorithms BFES, CBS, and CCBS require estimating the number of distinct keys $\tilde{n}$ in the data stream. To that end, given M amount of memory, these three algorithms allocate $m' = 4 \cdot \log_2(M)$ counters to the task of estimating $\tilde{n}$ and use the remaining counters for frequency estimation. We use the HYPERLOGLOG-HIP sketch [39] to compute $\tilde{n}$.

The algorithms CCBS, SEQS, and PRS require an estimate of $\mathcal{K}$ for estimating the frequencies. For these three algorithms we allocate, as suggested by the respective publications, 12.5% of the available memory M to a Bloom filter for detecting duplicate keys and furthermore allow an additional list to keep track of $\mathcal{K}$. This causes larger effective memory footprint and an advantage

	$\overline{\mathbf{e}}^{\downarrow}$	$\overline{\mathbf{e}}^{\uparrow}$	$\overline{\mathbf{e}}_1^{rel}$	$\overline{\mathbf{e}}_1$	$\overline{\mathbf{e}}_2$
BFESP|E|G^4	40.73	9.75	0.99	7.93	17.71
BFESP|E|G^2	44.57	9.77	0.99	7.95	18.29
BFESP|E|G^1	48.73	9.78	0.99	8.04	19.14
DPS	100.00	10.00	1.00	10.00	20.13
CMS	4509.30	4.51e+4	4995.61	4509.30	4516.56
CUS	1965.31	1.97e+4	2187.31	1965.31	1965.42
CS	1001.66	1000.26	201.06	181.99	229.02
CBS	50.09	46.07	9.77	8.74	17.41
CCBS	48.93	63.64	12.11	10.23	18.18
PRS	320.87	3204.71	361.88	320.51	398.88
SEQS	413.01	347.59	70.08	69.15	492.26

(a) Pareto (n=1000000)

	$\overline{\mathbf{e}}^{\downarrow}$	$\overline{\mathbf{e}}^{\uparrow}$	$\overline{\mathbf{e}}_1^{rel}$	$\overline{\mathbf{e}}_1$	$\overline{\mathbf{e}}_2$
BFESP|E|G^4	24.78	9.63	1.00	7.02	10.08
BFESP|E|G^2	24.74	9.62	1.00	7.01	10.07
BFESP|E|G^1	24.52	9.60	1.00	6.99	10.05
DPS	100.00	10.00	1.00	10.00	14.14
CMS	4637.46	4.64e+4	4008.98	4637.46	4641.83
CUS	2007.71	2.01e+4	1742.89	2007.71	2007.76
CS	738.05	735.35	115.53	133.95	168.08
CBS	41.87	37.85	7.50	7.25	10.01
CCBS	40.05	54.78	9.31	8.62	11.28
PRS	320.49	3204.87	281.57	320.49	395.47
SEQS	348.32	264.76	43.98	55.73	443.81

(b) Exponential (n=1000000)

	$\overline{\mathbf{e}}^{\downarrow}$	$\overline{\mathbf{e}}^{\uparrow}$	$\overline{\mathbf{e}}_1^{rel}$	$\overline{\mathbf{e}}_1$	$\overline{\mathbf{e}}_2$
BFESP|E|G^4	12.70	9.03	1.00	5.05	6.22
BFESP|E|G^2	12.68	9.02	1.00	5.05	6.21
BFESP|E|G^1	12.62	8.99	0.99	5.03	6.20
DPS	100.00	10.00	1.00	10.00	11.75
CMS	4697.40	4.70e+4	1985.45	4697.40	4700.43
CUS	2038.65	2.04e+4	864.92	2038.65	2038.67
CS	613.35	611.12	46.68	111.32	139.69
CBS	29.59	25.57	3.33	5.02	6.22
CCBS	28.29	43.03	4.19	6.48	8.11
PRS	320.41	3204.14	137.94	320.41	394.04
SEQS	309.30	223.28	17.02	48.42	466.06

(c) Gaussian (n=1000000)

TABLE II: Benchmarks on synthetic data. The streams have 1 million unique items and have been normalized such that the average item frequency is 10. BFESP|E|G^u indicates that the results show BFES using a Pareto prior on the Pareto dataset, and analogously for the exponential and Gaussian benchmarks. The u indicates the parameter u as described in Section IV-D. The colors indicate the ranking of the various methods, where green means better. See Section V-B for a discussion.

Regarding CMS, CUS, DPS, CS, CBS, and CCBS, the two-dimensional counter tables were configured to have a fixed depth of 5 throughout all experiments, based on common configurations used in the literature [42]–[44].

All algorithms have been implemented in C++ with the gnu++17 language dialect and were compiled using LLVM at optimization level Ofast. The source-code is publicly available at the git repository associated with this paper [15]. We used Eigen [45] for the implementation of PRS and the commercial solver MOSEK [46] for SEQS. Benchmarking has been performed on a system with 16GB LPDDR5 memory and an Apple M1 Pro processor with 8 cores.

Frequency estimation accuracy is measured based on six different error metrics described in Tab. III. Results are always averaged over 10 runs.

Sym	Error	Sym	Error
$\overline{e}_2$	Root mean square error	$\hat{e}_2$	Root median square error
$\overline{e}_1$	Mean absolute error	$\overline{e}_1^{rel}$	Mean absolute relative error
$\overline{e}^{cal}$	Mean calibration error	$\overline{e}^{\uparrow}$, $\overline{e}^{\downarrow}$	Mean asymmetric error

TABLE III: Error metrics. Asymmetric errors are defined as in Section III-C3 where $\overline{e}^{\uparrow}$ penalizes *overestimations* 10 times more than *underestimations*, and $\overline{e}^{\downarrow}$ does the opposite. The calibration error is discussed in Section V-D.

B. Experiments on Synthetic Data

We have benchmarked BFES and SOTA methods on synthetic datasets based on three distinct generative item-frequency distributions: a Pareto, an exponential, and a Gaussian. The mean error statistics are shown in Tab. II. Each synthetic trace contains one million distinct items and each sketch was configured to use 10000 32-bit counters, i.e. , 40 KB of memory. For the BFES, CBS, and CCBS these 40 KB include also the auxiliary HYPERLOGLOG-HIP sketch. The PRS, CCBS, and SEQS require additional memory for key de-duplication which results in additional 160 KB used by these three methods.

1) Observations: Tab. II clearly shows that across the five benchmarked metrics, BFES provides the most consistent performance; it achieves the lowest error in 4 out of 5 metrics and is a close second in the fifth metric. Other SOTA methods, in particular the CBS and CCBS, achieve their best performance in terms of absolute (e_1) and root square error (e_2) where they perform similarly to BFES. This is partially to be expected since absolute and square error is the primary performance target of the benchmarked SOTA methods. The difference in performance between BFES and other methods widens when looking at $e^{\downarrow}$, $e^{\uparrow}$, and e_1^{rel} since these are the cases where BFES can make the most use out of the Bayesian item-frequency samples it has generated and employ the methods discussed in Section III-C. The only method that in theory provides similar capabilities as BFES, in the sense of providing a distribution of frequency estimates, is DPS. However, the optimization problem part of the DPS query procedure does not find a solution in a numerically stable range in these experiments on synthetic data and therefore the algorithm produces degenerate frequency estimates as is visible in Tab. II. These experiments show that BFES provides very good performance on various synthetic data streams of large size.

C. Experiments on Real Traces

To show that Bayesian frequency estimation is also competitive on real data streams, we conduct experiments on two click-stream traces Kosarak [47] and Retail [48], as well as four network-packet traces UNIV1 [49], MACCDC [50], CAIDA [51], and MAWI [52]. Throughout these experiments the sketches use $M = 10\% \cdot n$ many 32-bit counters and the remaining configuration options are the same as with the synthetic experiments in Section V-B. In particular, PRS, CCBS, and SEQS again require additional memory which ranges from an additional 91.7 KB in the case of MACCDC, to 1.45 GB in the case of the larger CAIDA trace.

1) Observations: In Tab. IV we have tabulated the performance of BFES using three different priors (Pareto, exponential, and Gaussian) against other SOTA methods. Note that $\overline{e}_2$ indicates the root mean square error, while $\widehat{e}_2$ is the root median square error.

	$\overline{e}^{\downarrow}$	$\overline{e}^{\uparrow}$	$\overline{e}_1^{rel}$	$\overline{e}_1$	$\overline{e}_2$	$\widehat{e}_2$
BFESP[1]	97.82	11.32	0.81	11.28	945.15	1.67
BFESE[1]	98.43	13.10	0.81	12.31	946.18	2.18
BFESG[1]	98.63	22.83	4.40	13.85	943.63	2.67
DPS	21.78	176.87	35.37	21.78	23.72	17.73
CMS	21.78	217.75	168.68	21.78	24.78	18.82
CUS	12.35	123.47	98.70	12.35	13.93	10.92
CS	90.50	90.06	127.25	16.41	39.67	7.85
CBS	349.42	349.73	425.66	63.56	901.24	29.77
CCBS	348.27	348.41	419.92	63.33	908.40	29.72
PRS	97.25	261.73	198.05	32.63	1013.82	1.23
SEQS	165.25	80.83	97.22	22.37	1030.42	0.24

(a) MAWI (n=339950)

	$\overline{e}^{\downarrow}$	$\overline{e}^{\uparrow}$	$\overline{e}_1^{rel}$	$\overline{e}_1$	$\overline{e}_2$	$\widehat{e}_2$
BFESP[1]	59.01	10.58	0.92	9.89	101.88	4.98
BFESE[1]	60.40	11.85	0.88	10.92	102.84	8.02
BFESG[1]	62.05	19.25	1.12	12.27	102.00	8.66
DPS	99.97	22.50	1.01	26.16	97.47	32.45
CMS	258.89	2588.88	359.19	258.89	266.87	253.78
CUS	142.83	1428.34	206.86	142.83	144.04	147.19
CS	332.21	330.51	83.84	60.25	77.97	49.67
CBS	97.96	98.58	17.17	17.87	98.49	6.71
CCBS	96.20	91.04	15.78	17.02	98.85	6.45
PRS	60.24	229.24	36.70	26.32	107.81	15.50
SEQS	965.11	140.54	59.06	100.51	3391.51	3.97

(b) Retail (n=16470)

	$\overline{e}^{\downarrow}$	$\overline{e}^{\uparrow}$	$\overline{e}_1^{rel}$	$\overline{e}_1$	$\overline{e}_2$	$\widehat{e}_2$
BFESP[1]	88.82	11.87	0.83	11.68	252.96	4.04
BFESE[1]	90.31	14.63	0.86	13.34	253.54	6.91
BFESG[1]	90.66	28.08	3.60	15.42	251.92	8.66
DPS	91.06	164.78	8.25	68.54	70.85	52.76
CMS	91.12	911.24	494.51	91.12	103.43	80.83
CUS	52.74	527.45	296.03	52.74	58.60	48.80
CS	328.44	328.54	324.11	59.73	99.10	35.22
CBS	199.72	208.23	160.36	37.09	242.62	17.43
CCBS	194.81	205.35	158.55	36.38	243.77	16.85
PRS	90.12	294.05	168.51	34.92	301.45	5.93
SEQS	495.52	179.11	276.41	61.33	1269.01	0.52

(c) UNI1 (n=63172)

	$\overline{e}^{\downarrow}$	$\overline{e}^{\uparrow}$	$\overline{e}_1^{rel}$	$\overline{e}_1$	$\overline{e}_2$	$\widehat{e}_2$
BFESP[1]	80.99	12.03	0.88	11.53	226.12	4.48
BFESE[1]	83.23	14.48	1.00	13.09	226.68	7.87
BFESG[1]	84.42	25.69	2.93	14.97	225.21	9.06
DPS	132.28	95.06	4.15	61.97	158.74	58.16
CMS	146.43	1464.32	936.81	146.43	159.84	136.64
CUS	91.17	911.65	605.13	91.17	94.56	93.51
CS	362.62	359.04	419.97	65.61	92.00	47.86
CBS	157.15	160.52	137.89	28.88	217.02	11.07
CCBS	156.34	147.30	128.04	27.60	217.70	10.74
PRS	82.28	245.97	176.69	29.84	243.06	9.55
SEQS	134.11	108.77	89.03	22.08	298.85	1.18

(d) Kosarak (n=41270)

	$\overline{e}^{\downarrow}$	$\overline{e}^{\uparrow}$	$\overline{e}_1^{rel}$	$\overline{e}_1$	$\overline{e}_2$	$\widehat{e}_2$
BFESP[1]	98.30	11.65	0.90	11.44	207.78	1.21
BFESE[1]	98.63	13.60	1.03	12.44	208.65	1.53
BFESG[1]	98.78	27.22	10.97	14.24	206.98	1.77
DPS	19.59	195.44	154.51	19.59	28.72	15.77
CMS	19.59	195.87	269.63	19.59	28.87	15.92
CUS	12.11	121.08	172.43	12.11	20.56	10.16
CS	202.97	196.74	529.77	36.34	127.79	9.86
CBS	229.63	221.64	442.74	41.02	199.48	21.83
CCBS	226.89	223.23	440.73	40.92	200.98	21.84
PRS	101.26	308.56	464.92	37.26	263.46	0.83
SEQS	122.55	123.79	132.02	22.39	302.26	0.15

(e) MACCDC (n=13500)

	$\overline{e}^{\downarrow}$	$\overline{e}^{\uparrow}$	$\overline{e}_1^{rel}$	$\overline{e}_1$	$\overline{e}_2$	$\widehat{e}_2$
BFESP[1]	85.03	10.47	0.91	10.94	179.92	3.83
BFESE[1]	88.09	12.66	0.84	12.84	180.79	6.80
BFESG[1]	88.19	26.44	1.48	15.32	179.44	8.52
DPS	103.48	137.10	2.09	67.62	75.46	57.29
CMS	105.93	1059.33	177.40	105.93	114.09	97.89
CUS	60.31	603.07	104.51	60.31	63.31	58.90
CS	297.00	296.09	90.36	53.92	102.69	29.32
CBS	180.78	178.29	42.93	32.64	172.58	16.97
CCBS	176.52	173.41	41.43	31.81	173.65	16.47
PRS	84.31	263.50	44.86	31.62	211.71	6.28
SEQS	174.51	109.24	27.36	25.79	323.41	1.23

(f) CAIDA (n=988404)

TABLE IV: Benchmarks on real data streams. The traces have been normalized such that the average item frequency is 10. The colors indicate the ranking of the various methods across the datasets. See Section V-C for a discussion.

Firstly, by comparing the three versions of BFES that have been benchmarked, we see that on these datasets, BFESP with the Pareto assumption produces on average significantly better results than BFESE or BFESG with the exponential and Gaussian assumptions. This is no surprise since all these data streams have very heavy-tailed frequency distributions: `MAWI` is the most extreme case where the kurtosis (a measure of tailed-ness of a distribution) is 105727 and where the top 1% of items make up 81% of the total stream volume. On the more moderate end we have `Retail` with a kurtosis of 5750 and where the top 1% of items make up 20% of the total volume. In comparison, the synthetic Pareto data stream has a kurtosis of 504 and the top 1% of most frequent items make up 2.7% of the total. With this in mind, amongst the available prior assumptions for BFES, Pareto is by far the most heavy-tailed distribution and therefore yields the best results. The performance of BFES is qualitatively worse in the case of real compared to empirical data because of the mismatch between the Pareto prior and the stream distribution.

This brings us to the next point where it is evident from Tab. IV that BFES does not perform best in terms of root mean square error $\overline{e}_2$. The reason is that BFESP's Pareto prior underestimates the occurrence of a few extremely frequent items. The estimation error for these few items is then squared and therefore amplifies the average error considerably. To support this explanation we have displayed the root *median* square error $\widehat{e}_2$ which shows that the median squared frequency estimation error for BFES is considerably lower than the average error. In fact, considering that as explained in Section V-C, CCBS, PRS, and SEQS make use of a considerable amount of additional memory, BFESP shows very strong performance

	Par.	Exp.	Gaus.	Kos.	Ret.	UNI1	MAWI	MAC.	CAI.
BFESP[4]	0.02	0.07	0.14	0.06	0.08	0.12	0.37	0.32	0.23
BFESE[4]	0.06	0.01	0.08	0.05	0.05	0.07	0.37	0.33	0.31
BFESG[4]	0.13	0.08	0.02	0.08	0.08	0.06	0.38	0.34	0.38
DPS	0.5	0.5	0.5	0.43	0.25	0.47	0.50	0.50	0.46

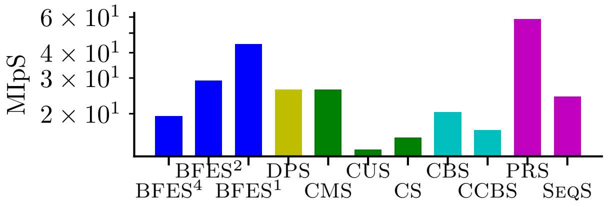
TABLE V: Average absolute calibration errors $\overline{e}_1^{cal}$ of applicable methods across synthetic and real datasets. Section V-D elaborates on the measurements.

in terms of root median square error.

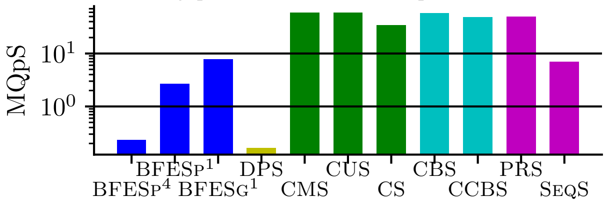
Thirdly, BFES consistently shows the strongest performance in terms of $\overline{e}^{\uparrow}$, $\overline{e}_1^{rel}$, and in particular the commonly used mean absolute error $\overline{e}_1$. CMS, CUS, and DPS display occasionally strong performance in terms of the underestimation-penalized error $\overline{e}^{\downarrow}$ which makes sense since in particular CMS and CUS always over-estimate the item frequencies, which is why these two methods show consistently very poor $\overline{e}^{\uparrow}$ performance.

D. Calibration Error and Confidence Intervals

The calibration error $\overline{e}_1^{cal}$ is the average over $p \in [0;1]$ of the absolute difference between $\mathbb{P}[a_k < Q_{\{\alpha_k^1,\alpha_k^2,\dots\}}(p)]$ and p. A calibration error of 0 therefore means that the sketch can for any arbitrary confidence level p, return confidence intervals that are on average perfect, i.e. , as tight as possible. Apart from BFES, amongst the methods we know of, only DPS and NIGPS provide functionality to give confidence intervals for any and all desired confidence levels p. We have however not been able to benchmark NIGPS due to its extreme computational cost which we found scales poorly in terms of the sketch size.



(a) Throughput in Million Insertions per Second



(b) Throughput in Million Queries per Second

Fig. 1: Computational throughput in the setting with $n = 10^6$ distinct items and $M = 10^4$ counters.

In Tab. V we show the average absolute calibration error for BFES and DPS across all datasets. Note how in the case of synthetic data, BFES achieves between 1% and 2% calibration error if the right prior is used. On the other hand the DPS displays significantly worse calibration which in the case of the synthetic data is due to the issue described in Section V-B1, while in the cases of the real data traces this is due to a systematic over-estimation of item frequencies which leads to biased and hence erroneous confidence intervals. In contrast, BFES achieves calibration errors between 5% and 6% on 3 of the 6 real streams which is still quite low. For the remaining 3 streams however BFES has a calibration error between 23% and 37% which is not very useful. This is again due to the strong tailed-ness of these three traces which is why the calibration error is the worst in the case of MAWI for reasons detailed in Section V-C1.

E. Computational Speed

To demonstrate that the BFES is not only accurate but also computationally competitive we have benchmarked the throughput of all sketches in terms of insertion and query speed. Fig. 1 shows that the BFES is competitive with the SOTA both in terms of querying and inserting items. As Section IV-D points out, the insertion speed of BFES^u changes with u. Also, in practice the query speed of the BFES varies depending on the prior belief we choose: BFESG^1 is faster than BFESP^1 since the Gibbs sampling from Section III-B1 is more difficult in the case of a Pareto prior compared to a Gaussian prior.

F. Limitations

A first point of consideration is that the implementation of BFES's QUERY is significantly more complicated compared to the other benchmarked methods discussed in this paper. Therefore the query procedure is unsuited for FPGAs, ASICS, and software-programmable network switches like Tofino. This can be worked around by implementing the insertion routine in hardware, while performing the query procedure on a general purpose CPU. So all in all while BFES can be implemented efficiently, it is in general slower than the simpler sketches such as CMS and CS, although faster than e.g. DPS.

Secondly, Bayesian frequency estimation requires an assumption (formulated as a Bayesian prior distribution) to work, unlike for example the CS. It is therefore a double-edged method which can provide great functionality, at the downside of having to provide a prior distribution. As can be seen from Tab. IV and Tab. V, if the prior belief does not match the actual distribution of item frequencies, item frequencies predicted by BFES will be biased, despite providing the lowest mean absolute error. In practice, the distribution of item frequencies in data streams often follows Zipf's law [53]–[55] which is why a power-law prior like the Pareto will in many cases provide solid baseline performance. Alternatively, statistics such as entropy [56] and variance [57] collected on representative data streams may be used to chose a prior [58].

Lastly, the analysis in Section III-D relies on the hash-matrix H to be known. This assumption creates a difference between the theoretic optimal Bayesian frequency estimator, and what can be implemented in a realistic setting. Future work can explore using additional memory such as SEQS and PRS do, in order to estimate the set of unique keys $\mathcal{K}$. In this setting one would incur a greater memory cost but come closer to the optimal Bayesian frequency estimation.

VI. Conclusion

This paper presented a novel perspective on the possibilities that Bayesian frequency estimation can provide. We have analysed the problem in a constrained setting from a theory point of view and derived sufficient requirements for efficient MCMC-based frequency estimation. The theoretic insights were translated into an algorithm that solves Bayesian frequency estimation in an approximate away. A comparison to other methods such as for example the COUNTMIN- and PR-SKETCH indicates that these methods can be seen as special instances of Bayesian frequency estimation. Benchmarks on both synthetic and real data show in particular the ability of BFES to adapt to individual error metrics, as well as provide good confidence intervals for arbitrary confidence levels. These are two qualities that existing methods lack.

Nevertheless, BFES also has its limitations, namely a more complicated frequency estimation algorithm that is difficult to implement in hardware. And although BFES requires a Bayesian prior on the item frequencies to work, empirical experiments show that BFES can keep up with SOTA methods in terms of throughput and that even when the prior assumptions are wrong, performance is still strong in multiple error metrics; in particular in terms of absolute estimation error.

We see opportunities for future research on improving the computational complexity of Bayesian frequency estimation, as well as developing efficient computational methods to use more heavy-tailed prior distributions, such that BFES can reach a better calibration and even lower error on challenging traces such as `MAWI`.

References

[1] R. Ben Basat, G. Einziger, R. Friedman, and Y. Kassner, "Randomized admission policy for efficient top-k and frequency estimation," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9.

[2] C. Estan and G. Varghese, "New Directions in Traffic Measurement and Accounting: Focusing on the Elephants, Ignoring the Mice," *ACM Trans. Comput. Syst.*, vol. 21, no. 3, p. 270–313, aug 2003. [Online]. Available: https://doi.org/10.1145/859716.859719

[3] T. Holterbach, E. C. Molero, M. Apostolaki, A. Dainotti, S. Vissicchio, and L. Vanbever, "Blink: Fast Connectivity Recovery Entirely in the Data Plane," in *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*. Boston, MA: USENIX Association, Feb. 2019, pp. 161–176. [Online]. Available: https://www.usenix.org/conference/nsdi19/presentation/holterbach

[4] H. Wu, H.-C. Hsiao, and Y.-C. Hu, "Efficient Large Flow Detection over Arbitrary Windows: An Algorithm Exact Outside an Ambiguity Region," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 209–222. [Online]. Available: https://doi.org/10.1145/2663716.2663724

[5] A. L. de Aquino, C. M. Figueiredo, E. F. Nakamura, L. S. Buriol, A. A. Loureiro, A. O. Fernandes, and C. J. J. Coelho, "Data Stream Based Algorithms For Wireless Sensor Network Applications," in *21st International Conference on Advanced Information Networking and Applications (AINA '07)*, 2007, pp. 869–876.

[6] W. Wu, J. Cao, H. Wu, and J. Li, "Robust and Dynamic Data Aggregation in Wireless Sensor Networks: A Cross-Layer Approach," in *2012 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing*, 2012, pp. 306–313.

[7] A. L. L. de Aquino, C. M. S. Figueiredo, E. F. Nakamura, L. S. Buriol, A. A. F. Loureiro, A. O. Fernandes, and C. J. N. Coelho Jr., "A Sampling Data Stream Algorithm For Wireless Sensor Networks," in *2007 IEEE International Conference on Communications*, 2007, pp. 3207–3212.

[8] C.-C. Wu, K.-T. Chen, C.-Y. Huang, and C.-L. Lei, "An empirical evaluation of VoIP playout buffer dimensioning in Skype, Google talk, and MSN Messenger," 06 2009, pp. 97–102.

[9] A. G. Kumbhare, Y. Simmhan, M. Frincu, and V. K. Prasanna, "Reactive Resource Provisioning Heuristics for Dynamic Dataflows on Cloud Infrastructure," *IEEE Transactions on Cloud Computing*, vol. 3, no. 2, pp. 105–118, 2015.

[10] M. Charikar, K. Chen, and M. Farach-Colton, "Finding frequent items in data streams," in *Automata, Languages and Programming*, P. Widmayer, S. Eidenbenz, F. Triguero, R. Morales, R. Conejo, and M. Hennessy, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 693–703.

[11] G. Cormode and S. Muthukrishnan, "An improved data stream summary: the count-min sketch and its applications," *Journal of Algorithms*, vol. 55, no. 1, pp. 58–75, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0196677403001913

[12] M. J. Kochenderfer, *Decision making under uncertainty: theory and application*. MIT press, 2015.

[13] H. Han, Z. Yan, X. Jing, and W. Pedrycz, "Applications of sketches in network traffic measurement: A survey," *Inf. Fusion*, vol. 82, no. C, p. 58–85, Jun. 2022. [Online]. Available: https://doi.org/10.1016/j.inffus.2021.12.007

[14] L. D. Brown, G. Casella, and J. T. G. Hwang, "Optimal confidence sets, bioequivalence, and the limaçon of pascal," *Journal of the American Statistical Association*, vol. 90, no. 431, pp. 880–889, 1995. [Online]. Available: http://www.jstor.org/stable/2291322

[15] F. Da Dalt, "BFES Repository," https://github.com/FrancescoDaDalt/BFES, 2024, github repository for the BFES algorithm.

[16] R. B. Basat, G. Einziger, M. Mitzenmacher, and S. Vargaftik, "Salsa: Self-adjusting lean streaming analytics," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, 2021, pp. 864–875.

[17] P. Roy, A. Khan, and G. Alonso, "Augmented sketch: Faster and more accurate stream processing," in *Proceedings of the 2016 International Conference on Management of Data*, ser. SIGMOD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1449–1463. [Online]. Available: https://doi.org/10.1145/2882903.2882948

[18] T. Yang, Y. Zhou, H. Jin, S. Chen, and X. Li, "Pyramid sketch: a sketch framework for frequency estimation of data streams," *Proc. VLDB Endow.*, vol. 10, no. 11, p. 1442–1453, aug 2017. [Online]. Available: https://doi.org/10.14778/3137628.3137652

[19] H. Li, Q. Chen, Y. Zhang, T. Yang, and B. Cui, "Stingy sketch: a sketch framework for accurate and fast frequency estimation," *Proc. VLDB Endow.*, vol. 15, no. 7, p. 1426–1438, mar 2022. [Online]. Available: https://doi.org/10.14778/3523210.3523220

[20] T. Yang, J. Jiang, P. Liu, Q. Huang, J. Gong, Y. Zhou, R. Miao, X. Li, and S. Uhlig, "Elastic sketch: adaptive and fast network-wide measurements," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 561–575. [Online]. Available: https://doi.org/10.1145/3230543.3230544

[21] S. Scherrer, C.-Y. Wu, Y.-H. Chiang, B. Rothenberger, D. E. Asoni, A. Sateesan, J. Vliegen, N. Mentens, H.-C. Hsiao, and A. Perrig, "Low-rate overuse flow tracer (loft): An efficient and scalable algorithm for detecting overuse flows," in *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*, 2021, pp. 265–276.

[22] Z. Liu, R. Ben-Basat, G. Einziger, Y. Kassner, V. Braverman, R. Friedman, and V. Sekar, "Nitrosketch: robust and general sketch-based monitoring in software switches," in *Proceedings of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 334–350. [Online]. Available: https://doi.org/10.1145/3341302.3342076

[23] Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, and V. Braverman, "One sketch to rule them all: Rethinking network flow monitoring with univmon," in *Proceedings of the 2016 ACM SIGCOMM Conference*, ser. SIGCOMM '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 101–114. [Online]. Available: https://doi.org/10.1145/2934872.2934906

[24] Y. Cao, Y. Feng, H. Wang, X. Xie, and S. K. Zhou, "Learning to sketch: A neural approach to item frequency estimation in streaming data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 11, pp. 7136–7153, 2024.

[25] S. Sheng, Q. Huang, S. Wang, and Y. Bao, "PR-Sketch: Monitoring per-Key Aggregation of Streaming Data with Nearly Full Accuracy," *Proc. VLDB Endow.*, vol. 14, no. 10, p. 1783–1796, jun 2021. [Online]. Available: https://doi.org/10.14778/3467861.3467868

[26] Q. Huang, S. Sheng, X. Chen, Y. Bao, R. Zhang, Y. Xu, and G. Zhang, "Toward Nearly-Zero-Error Sketching via Compressive Sensing," in *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, Apr. 2021, pp. 1027–1044. [Online]. Available: https://www.usenix.org/conference/nsdi21/presentation/huang

[27] F. Da Dalt, S. Scherrer, and A. Perrig, "Bayesian Sketches for Volume Estimation in Data Streams," *Proc. VLDB Endow.*, vol. 16, no. 4, p. 657–669, dec 2022. [Online]. Available: https://doi.org/10.14778/3574245.3574252

[28] D. Cai, M. Mitzenmacher, and R. P. Adams, "A bayesian nonparametric view on count-min sketch," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/0b9e57c46de934cee33b0e8d1839bfc2-Paper.pdf

[29] E. Dolera, S. Favaro, and S. Peluchetti, "A bayesian nonparametric approach to count-min sketch under power-law data streams," 2021. [Online]. Available: https://arxiv.org/abs/2102.03743

[30] R. Bellman, R. Corporation, and K. M. R. Collection, *Dynamic Programming*, ser. Rand Corporation research study. Princeton University Press, 1957. [Online]. Available: https://books.google.ch/books?id=wdtoPwAACAAJ

[31] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 721–741, 1984.

[32] S. Duane, A. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid monte carlo," *Physics Letters B*, vol. 195, no. 2, pp. 216–222, 1987. [Online]. Available: https://www.sciencedirect.com/science/article/pii/037026938791197X

[33] U. Grenander and M. I. Miller, "Representations of knowledge in complex systems," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 56, no. 4, pp. 549–603, 1994. [Online]. Available: http://www.jstor.org/stable/2346184

[34] W. Gilks, S. Richardson, and D. Spiegelhalter, Eds., *Markov Chain Monte Carlo in Practice*, 1st ed. Chapman and Hall/CRC, 1995.

[35] R. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[36] F. Zhang and H. D. Pfister, “Compressed sensing and linear codes over real numbers,” in *2008 Information Theory and Applications Workshop*, 2008, pp. 558–561.

[37] E. Cohen, “All-distances sketches, revisited: Hip estimators for massive graphs analysis,” 2015.

[38] P. Flajolet, E. Fusy, O. Gandouet, and F. Meunier, “Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm,” *Discrete Mathematics Theoretical Computer Science*, vol. DMTCS Proceedings vol. AH,..., 03 2012.

[39] T. Akiba, “hyperloglog-hip,” https://github.com/iwiwi/hyperloglog-hip, 2014, accessed: 2024-06-10.

[40] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM Review*, vol. 43, no. 1, pp. 129–159, 2001. [Online]. Available: https://doi.org/10.1137/S003614450037906X

[41] D. Wipf and B. Rao, “Sparse bayesian learning for basis selection,” *Signal Processing, IEEE Transactions on*, vol. 52, pp. 2153 – 2164, 09 2004.

[42] Y. Ben Mazziane, S. Alouf, and G. Neglia, “A Formal Analysis of the Count-Min Sketch with Conservative Updates,” in *IEEE INFOCOM WNA 2022 - The second Workshop on Networking Algorithms (WNA)*, New York, United States, May 2022. [Online]. Available: https://hal.science/hal-03613957

[43] Y. B. Mazziane, S. Alouf, and G. Neglia, “Analyzing count min sketch with conservative updates,” *Computer Networks*, vol. 217, p. 109315, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622003607

[44] D. Ting, “Count-min: Optimal estimation and tight error bounds using empirical error distributions,” 2018. [Online]. Available: https://arxiv.org/abs/1811.04150

[45] G. Guennebaud, B. Jacob *et al.*, “Eigen v3.4.0,” http://eigen.tuxfamily.org, 2010, accessed on: 23.02.2022.

[46] MOSEK ApS, “MOSEK Optimization Suite,” https://www.mosek.com/, 2023, accessed on: 30.05.2023.

[47] F. Bodon, “KOSARAK dataset,” http://fimi.uantwerpen.be/data/kosarak.dat.gz, 2003, last accessed 05.08.2022.

[48] T. Brijs, “RETAIL dataset,” http://fimi.uantwerpen.be/data/retail.dat.gz, 2003, last accessed 05.08.2022.

[49] F. Bodon, “UNIV1 dataset,” https://pages.cs.wisc.edu/~tbenson/IMC10_Data.html, 2010, last accessed 05.08.2022.

[50] Netresec, “MACCDC Dataset,” https://www.netresec.com/?page=MACCDC, 2012.

[51] CAIDA, “Caida Anonymized Internet Traces 2016.” https://data.caida.org/datasets/passive-2016/, 2016, last accessed 05.08.2022.

[52] K. Cho, K. Mitsuya, and A. Kato, “Traffic Data Repository at the WIDE Project,” in *USENIX 2000 FREENIX Track*, San Diego, CA, June 2000.

[53] P. Jurkiewicz, G. Rzym, and P. Boryło, “Flow length and size distributions in campus internet traffic,” *Computer Communications*, vol. 167, p. 15–30, Feb. 2021. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2020.12.016

[54] C. Furusawa and K. Kaneko, “Zipf’s law in gene expression,” *Physical Review Letters*, vol. 90, no. 8, p. 088102, 2003.

[55] S. T. Piantadosi, “Zipf’s word frequency law in natural language: a critical review and future directions,” *Psychonomic Bulletin & Review*, vol. 21, no. 5, pp. 1112–1130, 2014.

[56] P. Clifford and I. A. Cosma, “A simple sketching algorithm for entropy estimation,” 2013. [Online]. Available: https://arxiv.org/abs/0908.3961

[57] G. Cormode, *AMS Sketch*. New York, NY: Springer New York, 2016, pp. 76–78. [Online]. Available: https://doi.org/10.1007/978-1-4939-2864-4_578

[58] N. Ebrahimi, E. Maasoumi, and E. S. Soofi, “Ordering univariate distributions by entropy and variance,” *Journal of Econometrics*, vol. 90, no. 2, pp. 317–336, 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304407698000463