An Empirical Study of Moment Estimators for Quantile Approximation

RORY MITCHELL, Nvidia and University of Waikato, New Zealand
EIBE FRANK, University of Waikato, New Zealand
GEOFFREY HOLMES, University of Waikato, New Zealand

We empirically evaluate lightweight moment estimators for the single-pass quantile approximation problem, including maximum entropy methods [13] and orthogonal series with Fourier, Cosine, Legendre, Chebyshev and Hermite basis functions. We show how to apply stable summation formulas to offset numerical precision issues for higher-order moments, leading to reliable single-pass moment estimators up to order 15. Additionally we provide an algorithm for GPU-accelerated quantile approximation based on parallel tree reduction. Experiments evaluate the accuracy and runtime of moment estimators against the state-of-the-art KLL [17] quantile estimator on 14,072 real-world datasets drawn from the OpenML [2] database. Our analysis highlights the effectiveness of variants of moment-based quantile approximation for highly space efficient summaries: their average performance using as few as five sample moments can approach the performance of a KLL sketch containing 500 elements. Experiments also illustrate the difficulty of applying the method reliably and showcases which moment-based approximations can be expected to fail or perform poorly.

CCS Concepts: • **Mathematics of computing** → *Distribution functions*; *Statistical software*; • **Information systems** → *Data mining*; • **Computing methodologies** → *Machine learning algorithms.*

Additional Key Words and Phrases: density estimation, quantiles, data streams

ACM Reference Format:
Rory Mitchell, Eibe Frank, and Geoffrey Holmes. 2020. An Empirical Study of Moment Estimators for Quantile Approximation. *ACM Trans. Datab. Syst.* 1, 1, Article 1 (January 2020), 21 pages. https://doi.org/10.1145/3442337

1 INTRODUCTION

Quantile estimators are fundamental for characterising and manipulating data in a wide range of applications, either as building blocks for other algorithms or as data inspection and visualization tools in their own right. The moments sketch algorithm [13] is a method for extracting approximate quantile information from a data stream using a data structure based on the sample moments of the input stream. This approach is unique and appealing for a number of reasons. The sketch uses a fixed size data structure with a constant memory requirement, typically less than 200 bytes, making it suitable for tracking many data streams concurrently using cache memory only, and rendering it a good candidate for implementation on devices like graphics processing units (GPUs). Incremental updates and merge operations to the moments sketch are trivial and require only basic arithmetic. Across a distributed system, merge operations can be implemented using a single call

Authors' addresses: Rory Mitchell, ramitchellnz@gmail.com, Nvidia, University of Waikato, Department of Computer Science, New Zealand; Eibe Frank, eibe.frank@waikato.ac.nz, University of Waikato, Department of Computer Science, New Zealand; Geoffrey Holmes, geoffrey.holmes@waikato.ac.nz, University of Waikato, Department of Computer Science, New Zealand.

0362-5915/2020/1-ART1 $15.00
https://doi.org/10.1145/3442337

to AllReduce [12], a widely available primitive for efficient communication. Moreover, quantile estimates derived from the sketch typically produce estimates of less than 1% error (as we show in this paper), making it a useful tool in applications where this level of error is acceptable.

However, the basic algorithm presented in [13] suffers from drawbacks that make it difficult to use in practice. The primary challenge is numerical imprecision caused by maintaining higher-order moments using finite-precision arithmetic across a large range of real number inputs. In exact math, the algorithm works in a single pass, but in practice this is only true for well conditioned data. Our empirical evaluation shows that estimates for sample moments on many real-world inputs have unacceptable loss of precision.

In this paper, we investigate the behaviour of moments sketching on 14,072 datasets extracted from the OpenML database [2], evaluating accuracy, speed, and reliability. These datasets vary greatly in size, number of duplicate values, and proportion and scale of floating point values, providing a challenging testbed for an algorithm relying on numerical optimisation and numerical integration for accurate approximations. Considering the presence of duplicate values in real-world data, it is important to note that the theory of some of the quantile approximation methods we consider is based on the assumption of a density function. Our experiments with real-world data containing duplicates give an indication of how well these methods perform when this assumption is violated.

We consider several modifications of the basic moments sketch method: using stable higher order moment summation to improve accuracy on poorly conditioned inputs, using massively parallel GPUs to accumulate sample moments, and solving for the output distribution using Legendre polynomials instead of the more complicated maximum entropy method. Additionally, we compare the accuracy of the moments sketch to a related family of two-pass orthogonal series estimators and the state-of-the-art KLL [17] quantile sketch under a set of space constraints.

2 QUANTILE APPROXIMATION BACKGROUND

The ϕ-quantile q_ϕ is defined as $q_\phi = F^{-1}(\phi)$, where $F(x) = P(X \leq x)$ is the cumulative distribution function of random variable X, and F^{-1} is the generalised inverse, where $F^{-1}(u) = \min\{x : F(x) \geq u\}$. Given a finite sample dataset S of size n that is sorted in ascending order, an estimate of q_ϕ is the element in S whose rank is $\lfloor n\phi \rfloor$, where $\text{rank}(x)$ is defined as the number of elements in S smaller than x. The median is equivalent to the $\phi = 0.5$ quantile.

The naïve algorithm for computing quantiles is to sort the input data and extract the element at index $\lfloor n\phi \rfloor$. One of the earliest efficient approaches is the selection algorithm [3], which can find any given quantile in linear time. The problem of approximating quantiles using sublinear memory and a single pass is well studied. In [23], it was shown that computing the median using p passes over the data requires $\Omega(n^{1/p})$ space. As a result, any algorithm that computes quantiles in sublinear space, and in a single pass, must be an approximation. Approximate algorithms become necessary in the streaming setting, where only a partial view of the dataset can be observed at any given point, or in a distributed setting, where it becomes computationally infeasible to provide access to the entire dataset at each node.

The performance of quantile approximation algorithms delivering a quantile approximation $\hat{q}$ is typically described in terms of space required to achieve ϵ accurate quantile queries, where the true rank is bounded by $(\phi - \epsilon)n$ and $(\phi + \epsilon)n$. The algorithm of Greenwald and Khanna (GK sketch) [14] accumulates samples from an input stream into a buffer and applies a pruning scheme to the active set that maintains error bounds. The algorithm uses $O(\frac{1}{\epsilon}\log(n))$ space and is widely considered to be state-of-the-art. The Q-Digest algorithm of Shrivastava et. al. [29] operates on a fixed size universe $[u]$ (for example, the representable range of a floating-point variable). The

algorithm constructs a sparse tree representation of incoming data over the input universe using dyadic ranges, achieving space complexity of $O(\frac{1}{\epsilon}\log(u))$.

The count-sketch [6] and related count-min-sketch [8] algorithm were designed for tracking frequent elements in data streams. Input elements are accumulated into a set of constant-size hash tables, allowing well-defined probabilistic queries for summary statistics. Both algorithms may be adapted to return approximate quantile queries by partitioning the input universe into dyadic ranges and maintaining hash tables for the corresponding ranges (see [20]). The approximation guarantee of these algorithms relative to size is inferior to the GK sketch or Q-Digest, but they have the advantage of allowing deletions as well as insertions.

We recommend [20] for an in-depth and accessible summary of the above methods. More recent methods such as T-Digest [10] and DDSketch [22] provide improved performance for quantile queries on heavily skewed data and around the tails of the distribution. Of particular relevance to this paper is the KLL sketch [17], which is a randomized sketch providing ϵ accurate quantiles with probability $1-\delta$ using space $O(\frac{1}{\epsilon}\log\log(1/\delta))$.

The above quantile approximation algorithms are sample based — they store a sub-linear number of input elements as a sketch, approximating the empirical quantile function $F^{-1}(\phi)$ as a discontinuous step function via queries to these stored elements. Another approach is to define some parameterised uniformly continuous function closely approximating $F^{-1}(\phi)$. The moments sketch [13], which we discuss in detail in the next section, uses this approach. Another example is [30], in which an orthogonal Hermite series is used to generate quantile estimates. We discuss approaches based on orthogonal series in Section 4 and consider estimation using Legenrde series in particular in Section 5.

3 MOMENTS SKETCH

The strength of the moments sketch [13] lies in providing a low-memory, constant-size data structure that can be trivially updated and merged. It maintains a working set of power sums from orders 0 to l, where the sum at order k is

$$S_k = \sum_{i=0}^{n} x_i^k. \tag{1}$$

The sample moments $\mu_k = E[x^k]$ are obtained from the sketch as

$$\mu_k = \frac{S_k}{S_0}. \tag{2}$$

The moments sketch also maintains the values *min* and *max* bounding the range of the input data so that moments can be scaled and centered into the range $-1 \le x \le 1$. This rescaling and centering is necessary so that operations to obtain a density estimate from sample moments will be sufficiently well conditioned.

We rescale the moments using $s = \frac{2}{max-min}$ to obtain

$$\hat{\mu}_k = s^k \mu_k \tag{3}$$

and then shift by $-c = -s \cdot \frac{min+max}{2}$ using the binomial formula:

$$\tilde{\mu}_k = \sum_{i=0}^{k} \binom{k}{i} \hat{\mu}_i (-c)^{k-i}. \tag{4}$$

Given sample moments in the range $[-1, 1]$, a matching distribution based on a density function $f(x)$ is constructed. Many distributions may exist that match a finite set of sample moments [1].

One method of obtaining a unique distribution makes use of the principle of maximum entropy [19]. The entropy of a distribution's density function $f(x)$ is

$$H = -\int_{-\infty}^{\infty} f(x) \log f(x) dx.$$

Based on the maximum entropy principle, we search for the density function $f(x)$ that maximises H subject to the moment matching constraint

$$\mu_k = \int_{-\infty}^{\infty} x^k f(x) dx.$$

This maximum entropy density represents the least informative (i.e., "simplest") distribution that matches the moments. It is unique and has the form

$$f(x) = \exp(\sum_{k=0}^{l} \theta_k x^k). \tag{5}$$

The parameters θ_k are determined by minimising the loss function

$$L(\theta) = \int_{x_{min}}^{x_{max}} \exp(\sum_{k=0}^{l} \theta_k x^k) dx - \sum_{k=0}^{l} \theta_k \mu_k. \tag{6}$$

Minimising this convex loss function yields the distribution of maximum entropy among those that match the sample moments [16]. This minimisation can be performed by applying Newton's method with gradient

$$\frac{\partial L(\theta)}{\partial \theta_a} = \int_{x_{min}}^{x_{max}} x^a \exp(\sum_{k=0}^{l} \theta_k x^k) dx - \mu_a \tag{7}$$

and Hessian

$$\frac{\partial^2 L(\theta)}{\partial \theta_a \partial \theta_b} = \int_{x_{min}}^{x_{max}} x^a x^b \exp(\sum_{k=0}^{l} \theta_k x^k) dx. \tag{8}$$

Note that $x^a x^b = x^{(a+b)}$, leading to computational efficiencies because integrals from Equation 7 can be reused.

Figure 1 depicts the order $l = 10$ maximum entropy distribution fit to a range of simple datasets, comparing it to the fit obtained using other moment-based methods discussed in Section 4. In particular, we can see the maximum entropy distribution avoids the oscillations of approximations based on Legendre polynomials or the Fourier series.

It was noted in [13] that Chebyshev polynomials of the first kind improve the conditioning of the optimisation problem. This can be achieved by converting the moments $E[x^k]$ to 'Chebyshev moments' $E[T_k(x)]$, using expressions derived from the recurrence relations of Chebyshev polynomials [21] and replacing occurrences of x^k with $T_k(x)$. Our implementation uses this variant. See [9] for a more in-depth discussion of solving maximum entropy problems using Chebyshev polynomials instead of monomials. Additionally, in [13], the authors define a variant of the moments sketch utilising log moments $T_k(\log(x))$, although this requires strictly positive inputs. In Section 8, we evaluate versions of the moments sketch using log Chebyshev moments.

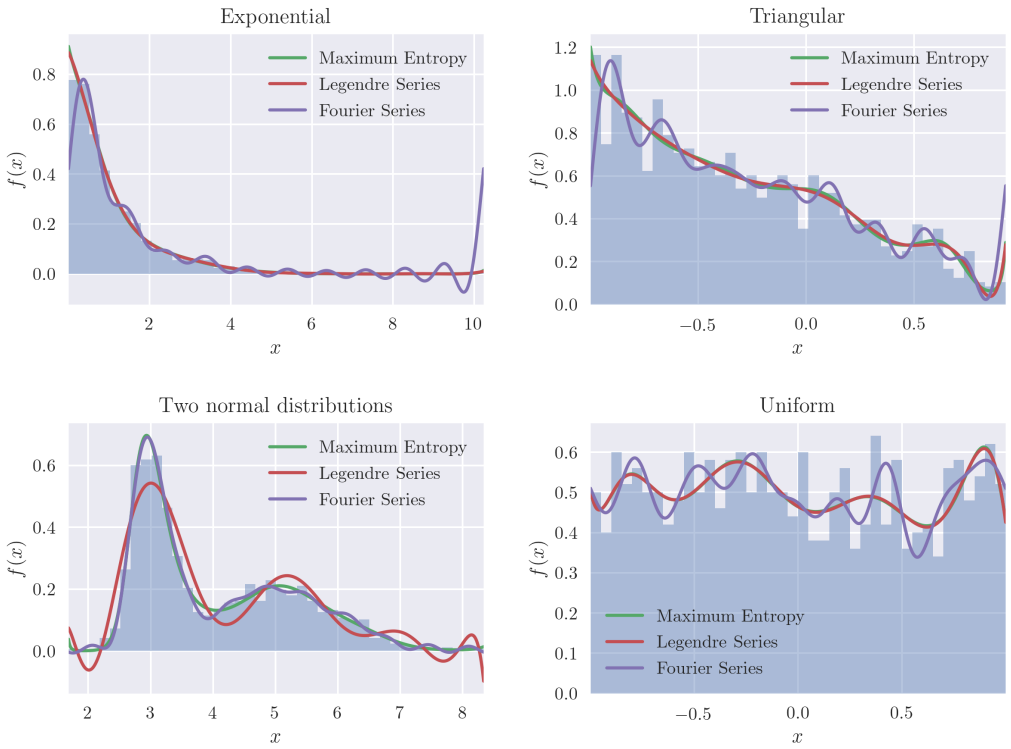


Fig. 1. Various density estimates of order $l = 10$

3.1 Solving for q_ϕ from a density function

Given the moment-matching distribution of maximum entropy, numerical integration techniques can be used to find q_ϕ satisfying $\phi = F(q_\phi) = \int_{-\infty}^{q_\phi} f(x)dx$, for example, integrating $f(x)$ with the trapezoid rule to approximate $F(x)$ and applying binary search to find q_ϕ. Alternatively, the problem can be posed as a system of ordinary differential equations (ODEs) so that widely available software packages will output q_ϕ for given ϕ. Consider the first-order differential equation

$$y'(t) = g(t, y)$$

$$y(t_0) = y_0$$

where we seek the value of function $y(t)$ at time points $t_1, t_2, \cdots$ and have access to the function g and y_0. We substitute t for ϕ, $y(t)$ for the inverse $F^{-1}(\phi)$, and y_0 for $F^{-1}(0) = q_0 = x_{min}$. To apply an ODE solver, we need $g(t) = (F^{-1})'(\phi)$. We know that

$$F(F^{-1}(\phi)) = \phi$$

and taking derivatives with respect to ϕ using the chain rule yields

$$f(F^{-1}(\phi))(F^{-1})'(\phi) = 1$$

which means

$$(F^{-1})'(\phi) = \frac{1}{f(F^{-1}(\phi))}$$

Algorithm 1 Moments sketch algorithm

Input: Datastream $(x_0, x_1, \cdots, x_n)$, maximum order l, $(\phi_0, \phi_1, \cdots, \phi_m)$
Output: Quantiles $(q_{\phi_0}, q_{\phi_1}, \cdots, q_{\phi_m})$

1: Compute power sums $(S_0, S_1, \cdots, S_l)$ from the input stream (Eq. 1)
2: Obtain sample moments $(\mu_0, \mu_1, \cdots, \mu_l)$ (Eq. 2)
3: Rescale sample moments to get $(\hat{\mu}_0, \hat{\mu}_1, \cdots, \hat{\mu}_l)$ (Eq. 3)
4: Shift sample moments to get $(\tilde{\mu}_0, \tilde{\mu}_1, \cdots, \tilde{\mu}_l)$ (Eq. 4)
5: Solve $\theta^* = \arg\min_\theta \int_{x_{min}}^{x_{max}} \exp(\sum_{k=0}^{l} \theta_k x^k) dx - \sum_{k=0}^{l} \theta_k \tilde{\mu}_k$ (Eq. 6)
6: Solve ODE with $y'(t, y) = \frac{1}{\exp(\sum_{k=0}^{l} \theta_k^* y^k)}$, $y_0 = x_{min}$ and $(t_0, t_1, \cdots) = (\phi_0, \phi_1, \cdots)$, yielding $(\tilde{q}_{\phi_0}, \tilde{q}_{\phi_1}, \cdots, \tilde{q}_{\phi_m})$
7: Shift and scale $(\tilde{q}_{\phi_0}, \tilde{q}_{\phi_1}, \cdots, \tilde{q}_{\phi_m})$ back to original domain, output $(q_{\phi_0}, q_{\phi_1}, \cdots, q_{\phi_m})$

where f is the density function from Equation 5. Now, when applying the ODE solver, $F^{-1}(\phi)$ corresponds to the state variable y provided by the solver on each iteration.

The end-to-end moments sketch algorithm for quantile approximation based on maximum entropy density estimation is summarised in Algorithm 1.

4 ORTHOGONAL FUNCTION DENSITY ESTIMATION

An alternative method of approximating a density function for the purposes of quantile approximation is possible via orthogonal functions. A set of functions $g_0(x)$, $g_1(x) \ldots g_k(x)$ is defined to be orthogonal over the interval $a < x < b$ when

$$\int_a^b g_i(x) g_j(x) w(x) dx = \delta_{ij} c_i$$

where $w(x)$ is a weight function, c_i is some scaling constant, and δ_{ij} is the Kronecker delta yielding 0 when $i \neq j$ and 1 when $i = j$.

A function $f(x)$ may be represented by an infinite series in a basis of orthogonal functions

$$f(x) = \sum_{i=0}^{\infty} a_i g_i(x) \tag{9}$$

with coefficients

$$a_i = \frac{1}{c_i} \int_a^b f(x) g_i(x) w(x) dx.$$

This is of practical relevance because the truncation of this series gives a useful approximation to $f(x)$. Some examples of orthogonal basis functions are listed in Table 1. As the table shows, the first three expansions are only applicable to restricted domains, but it is possible to map arbitrary data to the corresponding ranges. Note that the Fourier series is also an example of an orthogonal expansion, but it generates two sets of coefficients instead of one.

The series expansion based on orthogonal functions can naturally be used to estimate a probability density $f(x)$ from samples in the domain $[a, b]$ by realising that

$$\int_a^b f(x) g_i(x) w(x) dx = E[g_i(x) w(x)] \approx \frac{1}{n} \sum_{j=0}^{n} g_i(x_j) w(x_j) \tag{10}$$

In Equation 10, the series coefficients a_i from Equation 9 are estimated by a weighted average of the orthogonal basis functions applied to sample data, yielding a simple and computationally efficient density estimate.

Table 1. Orthogonal expansions

Expansion	Weight	Constant	Domain
Cosine	1	$c_0 = \pi, c_i = \frac{\pi}{2}$	$[0, \pi]$
Chebyshev	$\frac{1}{\sqrt{1-x^2}}$	$c_0 = \pi, c_i = \frac{\pi}{2}$	$[-1, 1]$
Legendre	1	$c_i = \frac{2}{2i+1}$	$[-1, 1]$
Hermite	$e^{-\frac{x^2}{2}}$	$c_i = \sqrt{2\pi} \cdot i!$	$[-\infty, \infty]$

This method of density estimation goes back to Cencov [4]. For a more recent summary, see [11]. Note that most work on orthogonal density estimation is concerned with the estimation of the unobserved density from sample data and truncates the series to obtain a smoother estimate. In contrast, for the quantile approximation problem, we aim to approximate the sample quantiles as closely as possible, and will not truncate the series except out of computational necessity.

A drawback of density estimation based on orthogonal functions is that the estimate may be negative in places. Correspondingly, the cumulative distribution function may not be monotonically increasing, as it would for a true probability density function. Simple corrections can be made by shifting the density function upwards and rescaling such that $\int f(x)dx = 1$. However, these corrections are not required for obtaining valid quantile estimates (see Section 5) and consistently have a negative effect on accuracy, so we do not consider them in this paper.

In Section 8, we provide an evaluation for the methods in Table 1 implemented as two-pass estimators, using a first pass to compute the range of the data, and a second pass to compute the orthogonal series coefficients (Equation 10) on the data mapped into the relevant domain. The Legendre series is an exception, and can be implemented in one pass directly from the moments sketch as explained in the next section. Note that, while the domain of the Hermite series is technically unbounded, its weight function $e^{-\frac{x^2}{2}}$ is poorly conditioned away from 0 and requires scaling to use in practice, so we implement it as a two-pass estimator.

5 LEGENDRE POLYNOMIAL SERIES

The Legendre series can be implemented as a one-pass estimator directly from the moments sketch as an alternative to the method of maximum entropy, providing the second moment-based quantile approximation method we consider in this paper. Given the Legendre polynomials defined by the recurrence relation

$$L_0(x) = 1$$
$$L_1(x) = x$$
$$(n+1)L_{n+1}(x) = (2n+1)xL_n(x) - nL_{n-1}(x),$$

the truncated Legendre series of order k is

$$f(x) \approx \sum_{n=0}^{k} a_n L_n(x)$$

where the coefficients a_n are

$$a_n = \frac{2n+1}{2} \int_{-1}^{1} f(x)L_n(x)w(x)dx$$

and the weight function is $w(x) = 1$. We can form the coefficients a_n directly from the scaled and shifted sample moments of the moments sketch, using the fact that these sample moments represent

data in the domain $[-1, 1]$, the weight function $w(x) = 1$ cancels out, and there are expressions directly converting monomials to Legendre polynomials.

To obtain the coefficients from sample moments, we use the following formula relating monomials to the Legendre polynomials:

$$L_0(x) = 1$$
$$L_1(x) = x$$
$$L_n(x) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-1)^k}{2^n} \binom{n}{k} \binom{2n-2k}{n} x^{n-2k}. \tag{11}$$

Applying the above formula with x^{n-2k} replaced by the sample moments μ_{n-2k} gives us the Legendre moments $\mu_n^{Leg} = \int_{-1}^{1} f(x) L_n(x) dx$, and we have the series coefficients by

$$a_n = \frac{2n+1}{2} \mu_n^{Leg}.$$

To get the cumulative distribution function $F(x) \approx \sum_{n=0}^{k} a_n \int_{-1}^{x} L_n(y) dy$, we use the integral of the Legendre polynomials

$$\int L_n(x) dx = \frac{L_{n+1}(x) - L_{n-1}(x)}{2n+1}.$$

As discussed in the context of the other orthogonal series estimators considered in the previous section, the density estimate can be negative in places, the cumulative distribution function $F(x)$ may not be monotonic and so its direct inverse—the quantile function $F^{-1}(y)$—may not exist. We can still find solutions to the inverse, with the caveat that they may not be unique. A standard root finding algorithm such as the bracketed Newton-Raphson approach of [27] is effective. In our implementation, we take the first root found by the algorithm. In practice, if multiple roots occur, they occur close together and all represent valid solutions, so any deterministic method for selecting roots could be used.

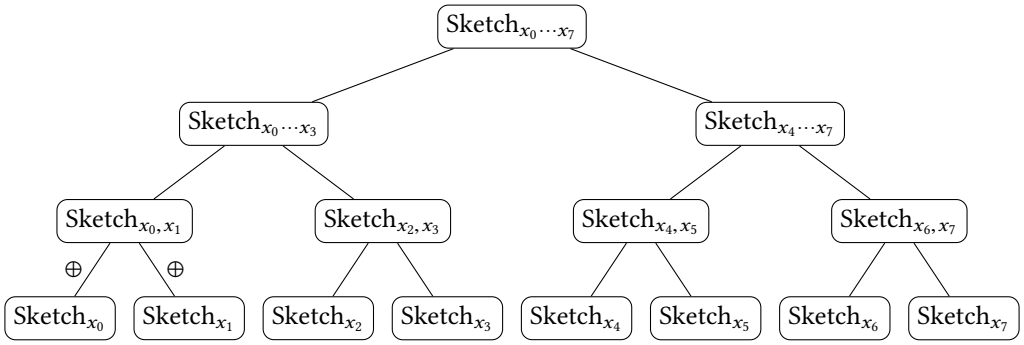
Considering the two moment-based density estimation approaches for quantile approximation we have now discussed, the Legendre series has some advantages over the maximum entropy distribution. It is comparatively simple to compute, avoiding the need to solve an optimisation problem with numerical integrals. The series is a polynomial and therefore has an analytic cumulative distribution function, useful for computing quantiles. Its disadvantage is also that it is a polynomial. The series will be effective for functions that are sufficiently smooth to be well approximated by a polynomial, otherwise it is common to encounter ringing artefacts as in Runge's phenomenon [28]. We include experiments comparing it to the maximum entropy approach in Section 8, concluding that it is a less accurate but considerably simpler one-pass method than the moments sketch with maximum entropy.

6 NUMERICALLY STABLE POWER SUMS

Methods based on the collection of sample moments suffer from limited available numerical precision when implemented on floating-point hardware. Early works such as [31] and [5] discuss the accumulation of rounding errors in the sample variance calculation and propose single-pass update formulas with significantly improved conditioning over naïve methods. More recently, Pébay *et al.* [26] generalise these formulas to higher-order central moments, providing both an incremental update formula for processing elements one at a time and a parallel update formula for merging partial sample moment estimates. The latter enables us to compute the power sum

$$M_k = \sum_{i=0}^{N} (x_i - \mu)^k,$$

Fig. 2. Parallel reduction



where μ is the sample mean, by splitting the full dataset into two multisets A and B of sizes N^A and N^B and combining the central moment estimates for those two multisets M_k^A and M_k^B in a numerically stable manner. Define $\delta_{B,A}$ as

$$\delta_{B,A} = \mu^B - \mu^A.$$

where μ^A, μ^B are the means of the respective multisets, then, for any integer $k \geq 2$,

$$M_k = N^A\left(\frac{-N^B}{N}\delta_{B,A}\right)^k + N^B\left(\frac{N^A}{N}\delta_{B,A}\right)^k + \sum_{i=0}^{k-2}\binom{k}{i}\left[M_{k-i}^A\left(\frac{-N^B}{N}\delta_{B,A}\right)^i + M_{k-i}^B\left(\frac{N^A}{N}\delta_{B,A}\right)^i\right] \quad (12)$$

Equation 12 facilitates numerically stable merging of power sums in parallel or distributed environments. In the case of incrementally summing elements, where $N^B = 1$ and $M_k^B = 0$ for $k > 0$, the formula simplifies to

$$M_k = \left[\frac{N-1}{(-N)^k} + \left(\frac{N-1}{N}\right)^k\right]\delta_{B,A}^k + \sum_{i=0}^{k-2}\binom{k}{i}M_{k-i}^A\left(\frac{-\delta_{B,A}}{N}\right)^i. \quad (13)$$

Given centred power sums from the above formulas, the moment estimates of Section 3 are trivially obtained, replacing S_k with M_k and shifting by $(\mu - c)$ instead of $(-c)$ in Equation 4.

In Section 8, we perform experiments comparing the naïve power sums of Equation 1 to the above formulas, showing that they significantly extend the usable order of sample moments at the cost of some speed due to extra computation.

7 GPU IMPLEMENTATION

Moment-based sketching is an attractive option for GPU accelerated quantile approximation. Computation of power sums by naïve summation or by the pairwise method of Section 6 can be performed in a data-parallel manner, taking advantage of the small fixed-size data structure and simple merge operations of the sketch. Other quantile sketch algorithms such as GK [14] or KLL [17] enable merging of sketches, but effective implementation of these algorithms on GPUs is nontrivial, due to dynamic resizing of the sketch and limited register and shared memory resources available to each GPU thread. In contrast, moment-based sketches can be stored entirely in registers on a per thread basis.

Implementing moment-based sketching in parallel using tree reduction on a GPU provides fast sketching at large input sizes as well as some numerical advantages over incremental sketching. The standard GPU tree reduction algorithm (see [7, 24] for reference) illustrated in Figure 2 takes an array $[x_0, x_1, x_2, \cdots, x_{n-1}]$ and a binary associative operator $\oplus$, returning as a single output value the result of $(x_0 \oplus x_1 \oplus x_2 \oplus \cdots \oplus x_{n-1})$. In the context of moment-based sketching, we can perform a massively parallel reduction by initialising n sketches in n threads, each with a single data point, implementing an operator $\oplus$ for merging two sketches, and applying reduction from the pycuda library [18]. When performing moment-based sketching by maintaining power sums (Equation 1) as in the original moments sketch implementation of [13], two sketches are merged trivially by adding together the pair of power sums for each moment and establishing the minimum and maximum data point over both sketches. Using the numerically stable method from Section 6 instead, the two centered power sums for each moment are merged via Equation 12, and the minimum and maximum element are updated as before.

As well as providing the opportunity for faster sketching, parallel merging of sketches has the advantage of reducing round-off error in floating-point summation. [15] shows that the round-off error from tree-based summation grows proportionally to $O(\epsilon \log_2(n))$, whereas the error from naïve summation grows proportionally to $O(\epsilon n)$. Reduced round-off error is a useful property that arises naturally from parallel reduction on GPUs. As we will see from the experiments in Section 8, the number of usable sample moments is heavily constrained by available numerical precision.

8 EVALUATION

We perform numerical experiments evaluating the accuracy, speed, and numerical stability of moment-based sketching methods. We also include the KLL algorithm (specifically the first of two algorithms described in [17]) as a baseline state-of-the-art one-pass quantile estimator. Additionally, we compare against two-pass algorithms based on orthogonal density estimation. Algorithms are evaluated against the OpenML-CC18 benchmark suite [2], containing 72 machine learning datasets. Each dataset consists of tabular data with varying numbers of rows and columns. Sketches are evaluated for all columns of each dataset, for a combined total of 14,072 unique sketch inputs. We differentiate between an OpenML 'dataset' and 'sketch input' as each dataset contains multiple columns used independently in our experiments. Figure 3 shows a histogram of the sizes of the OpenML sketch inputs, and Figure 4 shows a histogram of the proportion of unique values contained in each sketch input. It is noteworthy that many sketch inputs contains duplicate values, showing the importance of considering each method's ability to deal with such inputs even if their theoretical derivation requires the existence of a probability density.

8.1 Measuring the accuracy of the quantile approximations

To evaluate the accuracy of the quantile approximations provided by the methods considered in this paper, we use the normalised integrated absolute error (NIAE):

$$NIAE = \frac{1}{b-a} \int_{0.0}^{1.0} |\hat{F}^{-1}(\phi) - F^{-1}(\phi)| d\phi \tag{14}$$

where $\hat{F}^{-1}(\phi)$ is the approximation of the quantile function, $F^{-1}(\phi)$ is the empirical quantile function, and (a, b) is the range of the data.

Note that the NIAE error metric has an equivalent formulation in terms of the cumulative distribution function $F(x)$, i.e.,

$$NIAE = \frac{1}{b-a} \int_{a}^{b} |\hat{F}(x) - F(x)| dx.$$

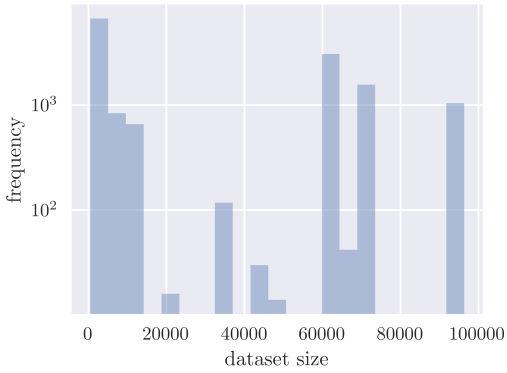


Fig. 3. Histogram of OpenML sketch input size

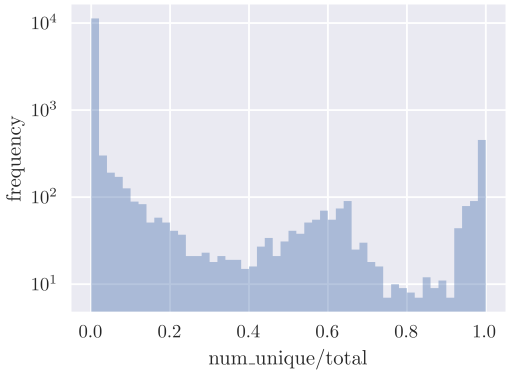
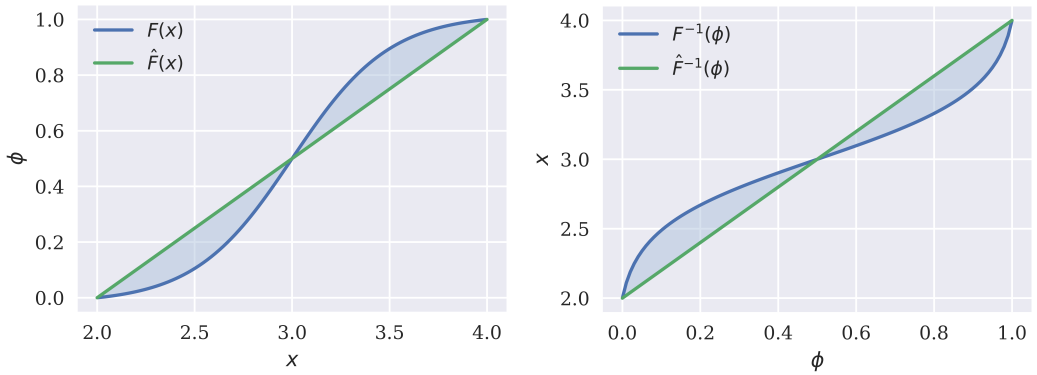


Fig. 4. Histogram of OpenML unique value ratio



(a) $\int_a^b |\hat{F}(x) - F(x)|dx$ (b) $\int_{0.0}^{1.0} |\hat{F}^{-1}(\phi) - F^{-1}(\phi)|d\phi$

Fig. 5. Integrated absolute error visualised

It can be seen from Figure 5 that the two formulations are equivalent. This gives another interpretation of NIAE as the expected value of $|\hat{F}(x) - F(X)|$ assuming x is drawn uniformly from the range (a, b):

$$X \sim U(a, b)$$

$$NIAE = E[|\hat{F}(X) - F(X)|].$$

In our experiments, we evaluate the integral from Equation 14 using the trapezoid rule with 1000 function evaluations. For each of the 14,072 sketch inputs obtained from OpenML, we evaluate the NIAE for our variants of the two moment-based sketching methods based on maximum entropy and Legendre polynomials respectively, and the KLL sketch as one-pass estimators, and various orthogonal series as two-pass estimators. We report overall performance of each estimator using violin plots of the distribution of the NIAE across the 14,072 sketch inputs and also provide the mean NIAE, its standard deviation, the median NIAE, and the minimum and maximum NIAE. Additionally, we report the number of failures due to numerical error, and exclude failures from final statistics (mean, median, std, min, max).

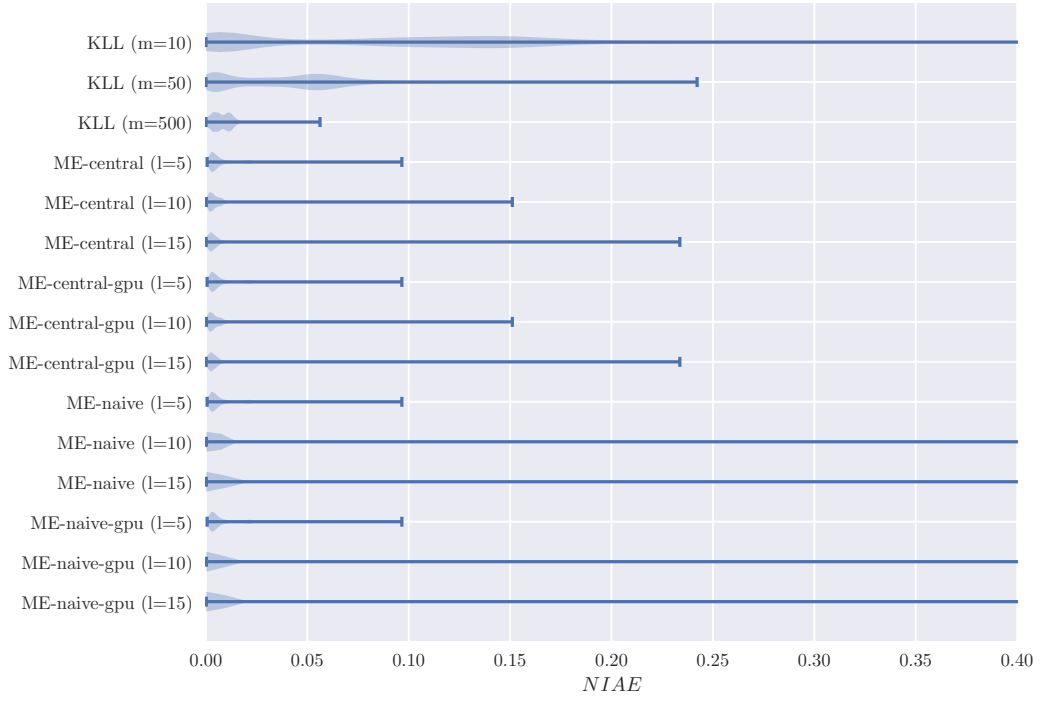


Fig. 6. Violin plot of NIAE measured on OpenML datasets - KLL and ME

8.1.1 One-pass Estimators. Figure 6, Figure 7, and Table 2 summarise the accuracy of the one-pass estimators. We evaluate several variants of the moment-based quantile approximation methods, differentiating on the number of moments l, the method of fitting the distribution (maximum entropy or Legendre polynomials), the algorithm used to accumulate moments (central moment formulas (Eq. 12) or naïve summation (Eq. 1)), and whether the moments were computed using tree reduction on the GPU, or in the standard manner on the CPU. The KLL estimator provides a state-of-the-art baseline. The m in KLL(m=10), etc. indicates the maximum size of the quantile summary.

Results verify the effectiveness of moment-based quantile estimators for answering queries with low memory requirements. Maximum entropy moment-based estimators (ME) with $l = 5$ store only 8 elements in memory ($l + 1$ moments, min and max) while providing a mean error of approximately 0.0084—very close to the mean error of 0.0081 for the KLL (m=500) estimator, which requires storing 500 elements in memory. The Legendre polynomial moment-based estimators (see Figure 7) are all outperformed by the equivalent maximum entropy estimators in terms of mean, median, and maximum error, but they compare favourably to KLL estimators as a low memory summary. Assuming some degree of error can be tolerated, they provide a simpler alternative to the method of maximum entropy, not requiring numerical integrals or the solution of a nonlinear optimisation problem.

Of the two strategies for sample moment summation, the naïve method quickly becomes unstable at $l > 5$, resulting in a number of failures and considerably reduced accuracy when solving for the output distribution using either the method of maximum entropy or the Legendre series. Using the central moment formulas of Pébay *et al.* [26] improves numerical stability dramatically, allowing the

Table 2. Error statistics on OpenML datasets

Estimator	mean	std	median	min	max	failures
KLL (m=10)	0.107872	0.096799	0.103475	0.000014	0.673525	0
KLL (m=50)	0.046058	0.037219	0.045693	0.000000	0.242182	0
KLL (m=500)	0.008104	0.004924	0.007611	0.000000	0.056235	0
Legendre-central (l=5)	0.081809	0.067712	0.083347	0.000062	0.163962	0
Legendre-central (l=10)	0.057979	0.050471	0.055742	0.000063	0.120809	0
Legendre-central (l=15)	0.048106	0.042234	0.046309	0.000052	0.100216	0
Legendre-central-gpu (l=5)	0.081809	0.067712	0.083347	0.000062	0.163962	0
Legendre-central-gpu (l=10)	0.057979	0.050471	0.055742	0.000063	0.120809	0
Legendre-central-gpu (l=15)	0.048106	0.042234	0.046309	0.000052	0.100216	0
Legendre-naive (l=5)	0.081811	0.067714	0.083355	0.000062	0.163962	1
Legendre-naive (l=10)	0.059386	0.054452	0.056886	0.000063	0.938904	131
Legendre-naive (l=15)	0.051749	0.051907	0.049966	0.000052	0.992505	467
Legendre-naive-gpu (l=5)	0.081811	0.067712	0.083347	0.000062	0.163962	0
Legendre-naive-gpu (l=10)	0.058923	0.052333	0.056635	0.000063	0.918821	87
Legendre-naive-gpu (l=15)	0.052065	0.054244	0.049645	0.000052	0.964786	355
ME-central (l=5)	0.008444	0.008461	0.005030	0.000570	0.096600	0
ME-central (l=10)	0.006171	0.008792	0.003628	0.000232	0.151073	0
ME-central (l=15)	0.005955	0.010254	0.003081	0.000119	0.233678	0
ME-central-gpu (l=5)	0.008444	0.008461	0.005030	0.000570	0.096600	0
ME-central-gpu (l=10)	0.006178	0.008791	0.003632	0.000232	0.151073	0
ME-central-gpu (l=15)	0.005962	0.010467	0.003048	0.000120	0.233682	0
ME-naive (l=5)	0.008447	0.008479	0.005030	0.000570	0.096600	0
ME-naive (l=10)	0.007116	0.015522	0.003754	0.000232	0.707496	20
ME-naive (l=15)	0.009518	0.032835	0.003232	0.000170	0.982960	155
ME-naive-gpu (l=5)	0.008447	0.008479	0.005030	0.000570	0.096600	0
ME-naive-gpu (l=10)	0.007112	0.019299	0.003701	0.000232	0.891338	15
ME-naive-gpu (l=15)	0.009329	0.033306	0.003212	0.000170	0.955920	112

Table 3. Two Pass Orthogonal Estimators: Error statistics on OpenML datasets

Estimator	mean	std	median	min	max	failures
Chebyshev (l=5)	0.063520	0.052235	0.067856	0.002066	0.975388	30
Chebyshev (l=10)	0.073640	0.067511	0.062730	0.001530	0.970769	30
Chebyshev (l=15)	0.032477	0.036878	0.037495	0.001335	0.969974	30
Cosine (l=5)	0.054256	0.043132	0.062123	0.001364	0.103496	0
Cosine (l=10)	0.031929	0.025494	0.033809	0.001363	0.062232	0
Cosine (l=15)	0.023803	0.018731	0.025451	0.001363	0.045502	0
Fourier (l=5)	0.220277	0.207780	0.166646	0.000053	0.499986	0
Fourier (l=10)	0.216825	0.208501	0.156493	0.000052	0.499986	0
Fourier (l=15)	0.215401	0.208858	0.151379	0.000051	0.499986	0
Hermite (l=5)	0.227591	0.195979	0.253604	0.001332	0.518415	0
Hermite (l=10)	0.255721	0.237913	0.229341	0.001083	0.593443	0
Hermite (l=15)	0.241525	0.223023	0.227826	0.001115	0.568650	0

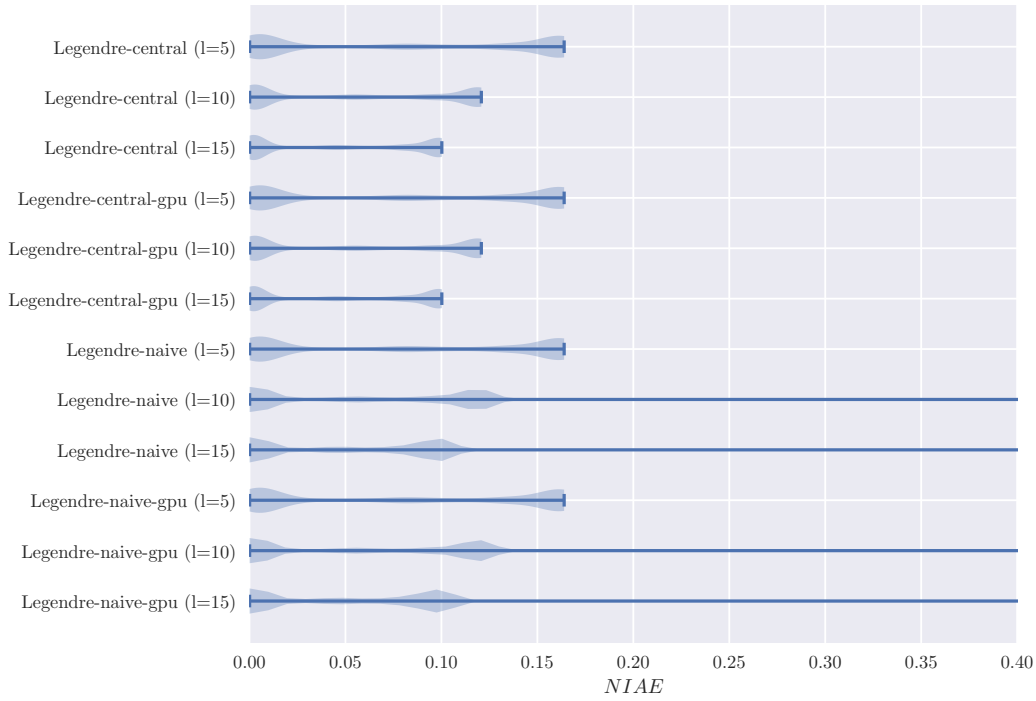


Fig. 7. Violin plot of NIAE measured on OpenML datasets - Legendre series

use of up to $l = 15$ moments without failures. Use of higher-order moments with stable summation allows a moderate reduction in mean error for both the maximum entropy and Legendre series estimators.

Parallel summation of moments on the GPU has a moderate effect on accuracy when naïve summation is used: for example, the mean error of ME-naïve (l=15) reduces from 0.009518 to 0.009329 with ME-naïve-gpu (l=15), and the number of failures reduces from 155 to 112. Overall, it is less effective than the use of stable central moment formulas; when parallel summation is applied in addition to stable moment summation, its effect is not significant.

We also evaluate a variant of the ME estimator using 'log Chebyshev moments' $E[T_k(\log(x))]$ instead of Chebyshev moments $E[T_k(x)]$, with the goal of improving stability for large-valued positive inputs. We test three estimators on the subset of 1,421 OpenML datasets where $x_{min} > 0$. ME-central acts as a baseline (although differing from the above charts as only positive datasets are considered), ME-log-naive computes moments using standard summation on log-transformed inputs, and ME-log-central computes moments using stable summation formulas on log-transformed inputs. The results are summarised by Figure 9 and Table 4.

In general, the use of log moments instead of conventional moments results in significantly reduced accuracy, and requires a priori knowledge that data is positive, limiting its usefulness for general database queries. Stable summation formulas provide considerable benefits even on log transformed inputs, showing that log transformation is not a solution to precision loss during summation.

Additionally, to test the robustness of one pass methods on large inputs, we draw 10^9 samples from the normal distribution $X \sim \mathcal{N}(1000, 1)$, numerically challenging due to its offset from zero.

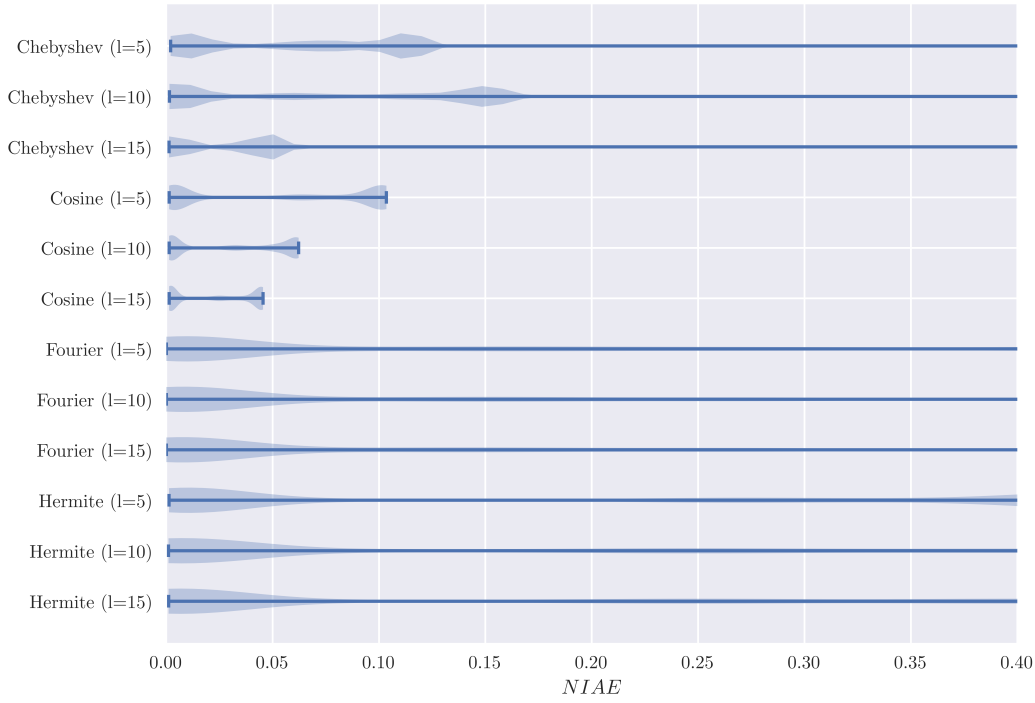


Fig. 8. Two Pass Orthogonal Estimators: Violin plot of NIAE measured on OpenML datasets

Table 4. ME Estimators: Error statistics on positive OpenML datasets

Estimator	mean	std	median	min	max	failures
ME-central (l=5)	0.009323	0.009896	0.004281	0.000913	0.063738	0
ME-central (l=10)	0.005444	0.005474	0.002916	0.000381	0.077327	0
ME-central (l=15)	0.005803	0.006204	0.002591	0.000330	0.080504	0
ME-log-central (l=5)	0.087116	0.107243	0.035912	0.002847	0.714728	1
ME-log-central (l=10)	0.085741	0.104820	0.033300	0.002477	0.779287	1
ME-log-central (l=15)	0.088461	0.109431	0.032717	0.002098	0.788754	1
ME-log-naive (l=5)	0.089040	0.106900	0.042254	0.002849	0.714728	1
ME-log-naive (l=10)	0.149801	0.127953	0.094661	0.003077	0.780095	2
ME-log-naive (l=15)	0.253283	0.161318	0.252858	0.002282	0.756054	35

Table 5 shows the resulting NIAE for one-pass moment estimators. Central moment summation formulas provide robust results, even on poorly conditioned inputs and at large sizes.

8.1.2 Two-pass Estimators. Figure 8 and Table 3 summarise the accuracy of the two-pass estimators based on orthogonal series. The estimators perform a first pass to establish the range of the data and then compute an orthogonal series per Equation 10 after the data has been scaled into the applicable domain. The Chebyshev estimator encounters some numerical failures, attributable to the weight function $w(x) = \frac{1}{\sqrt{1-x^2}}$, which approaches infinity at the endpoints of the domain

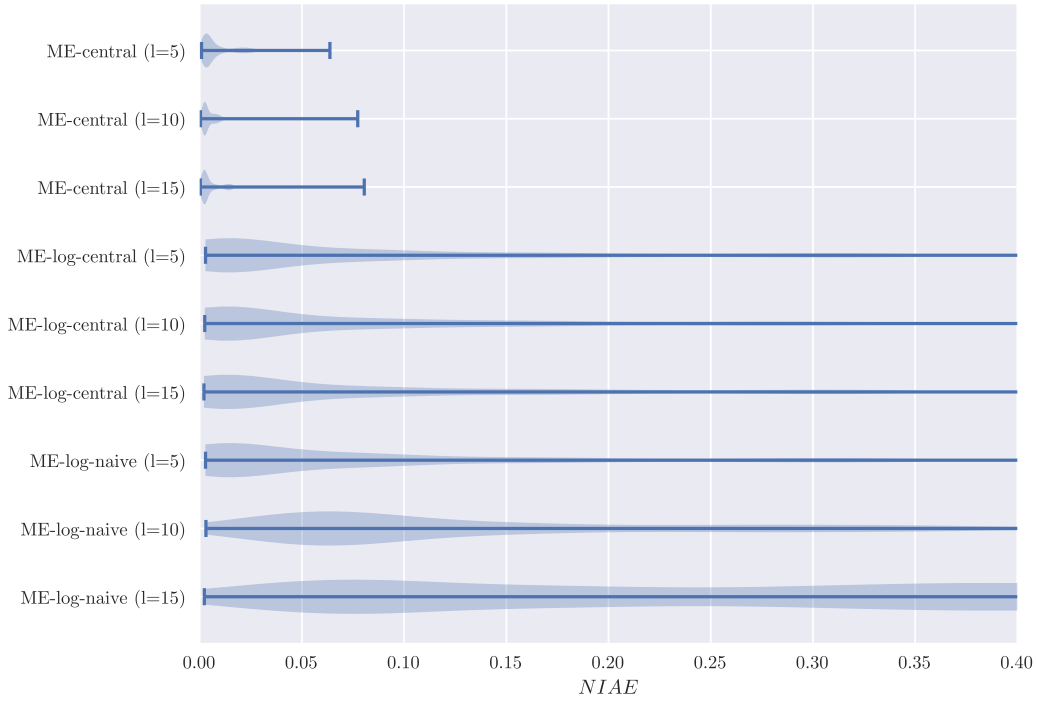


Fig. 9. Violin plot of NIAE measured on positive OpenML datasets

Table 5. NIAE for 10^9 samples from $X \sim \mathcal{N}(1000, 1)$

Estimator	NIAE
Legendre-central (l=5)	0.042877
Legendre-central (l=10)	0.005369
Legendre-central (l=15)	0.000997
Legendre-central-gpu (l=5)	0.042877
Legendre-central-gpu (l=10)	0.005369
Legendre-central-gpu (l=15)	0.000997
Legendre-naive (l=5)	1.000000
Legendre-naive (l=10)	1.000000
Legendre-naive (l=15)	1.000000
ME-central (l=5)	0.001777
ME-central (l=10)	0.001777
ME-central (l=15)	0.001808
ME-central-gpu (l=5)	0.001777
ME-central-gpu (l=10)	0.001777
ME-central-gpu (l=15)	0.001924
ME-naive (l=5)	0.225655
ME-naive (l=10)	1.000000
ME-naive (l=15)	1.000000

Table 6. Summary: Error statistics on OpenML datasets for the top performers

Estimator	mean	std	median	min	max	failures
Cosine (l=15)	0.023803	0.018731	0.025451	0.001363	0.045502	0
KLL (m=500)	0.008104	0.004924	0.007611	0.000000	0.056235	0
Legendre-central-gpu (l=15)	0.048106	0.042234	0.046309	0.000052	0.100216	0
ME-central-gpu (l=5)	0.008444	0.008461	0.005030	0.000570	0.096600	0
ME-central-gpu (l=15)	0.005962	0.010467	0.003048	0.000120	0.233682	0

$(-1.0, 1.0)$. The standout performer of the two-pass orthogonal estimators is Cosine ($l = 15$), with a mean error of 0.023803, being outperformed only by the maximum entropy estimators and KLL at $m = 500$ when comparing to the results in Table 2. It also has the lowest maximum error of any estimator tested, at 0.045502. This, as well as the simpler implementation of the cosine series, make it a compelling quantile estimator in settings where the range of the data is known or two passes are acceptable.

8.1.3 Comparing All Estimators. Table 6 summarises estimators with the lowest mean error using either one or two passes. For the moment-based estimators we show the most accurate summation method, using central moment update formulas and GPU tree reduction. ME-central-gpu (l=15) is the most accurate of all estimators in terms of mean and median NIAE, but with a significantly higher maximum error. KLL (m=500) follows closely behind in mean and median NIAE, but uses more than an order of magnitude more space. ME-central-gpu (l=5) is slightly less accurate on average than its (l=15) version, but benefits from a much lower maximum error and smaller space footprint, while being almost as accurate as KLL (m=500). Legendre-central-gpu (l=15) has more than 5 times the mean or median error of the KLL or maximum entropy methods, but its maximum error is significantly smaller than the maximum entropy estimator and its implementation is simpler. The two-pass Cosine (l=15) estimator sits between KLL and Legendre polynomials in terms of mean and median error, but notably, it has the lowest maximum error of any estimator and a simple implementation.

8.2 Sketch Time

We measure the runtime of sketches with respect to data size, evaluated on the standard normal distribution at varying sizes. Runtime is measured as time taken to accumulate the data stream into the sketch (excluding time to return quantile queries). All algorithms are implemented in native code and run on an AMD Ryzen 7 2700 @3.2GHz CPU and Nvidia 1080Ti GPU.

8.2.1 One-pass. Figure 10 shows the runtime of moment-based sketches and the KLL sketch implemented on the CPU. There is a considerable gap in performance between the KLL sketch and moment based sketches. This is in part due to the simplicity of accumulating power sums with basic arithmetic operations on a static data structure compared to maintaining a more complicated dynamic data structure in memory. Moment-based sketching using the stable central moment formulas is noticeably slower than naïve summation as the algorithmic complexity of Equations 12 and 13 is quadratic in the number of moments. GPU versions of the moments sketch are shown in Figure 11. The GPU versions computing the central moment formulas are considerably faster than CPU versions at sizes $> 10^5$ and GPU versions using naïve summation are moderately faster than their CPU alternatives at sizes $> 10^7$. This is an expected result as GPU architecture is comparatively optimised for throughput over latency whereas CPU architecture is optimised for latency over

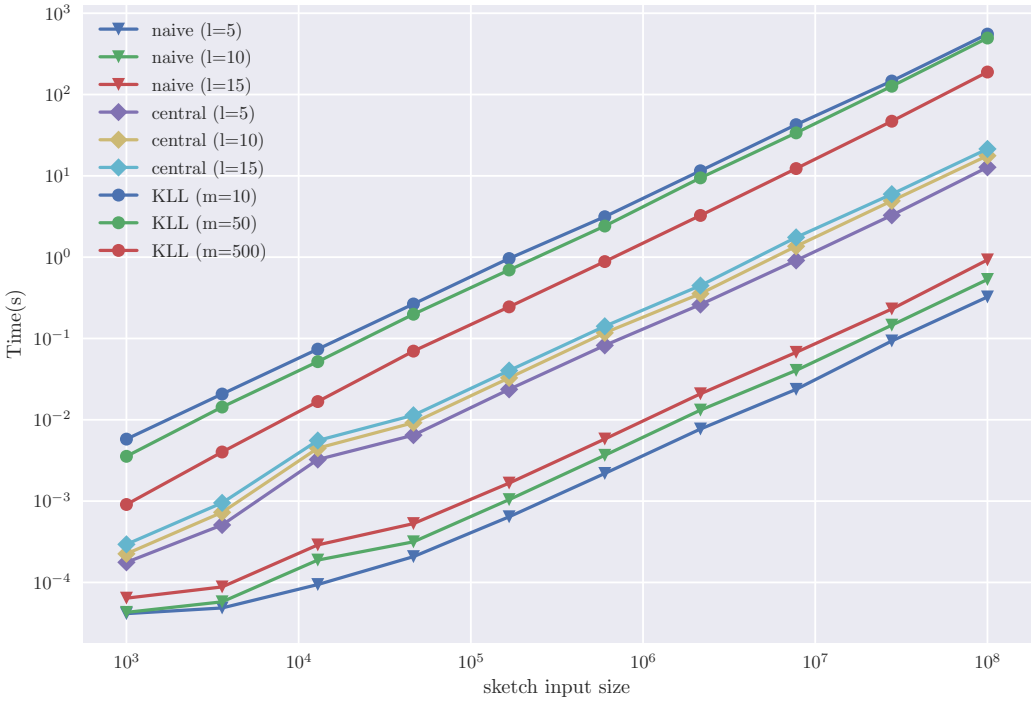


Fig. 10. Runtime of one-pass sketches on CPU

throughput [25]. Naïve summation implemented on the CPU remains the fastest method at smaller input sizes, but the GPU central moments sketch provides a compelling way to compute stable sample moments at higher orders while still retaining good performance.

8.2.2 Two-pass. Figure 12 shows the runtime of orthogonal estimators implemented on the CPU, including a first pass to establish the bounds of the data. This represents the cost of evaluating Equation 10 with different basis functions and weight functions. According to Figure 12, the effective speed of the orthogonal series estimators varies by a constant factor depending on the selection of basis and number of coefficients. Comparing to Figure 10, performance is similar to the moments sketch (in scenarios where the domain of the data is known).

9 CONCLUSION

We perform a study of moment-based sketching methods for the quantile estimation problem on 14,072 real-world datasets, comparing the state-of-the-art KLL estimator, a moment-based maximum entropy method, and orthogonal series estimation based on Legendre polynomials. We verify the result of [13] that moment-based sketching is effective at approximating quantiles with low memory requirements, but we show that its numerical stability deteriorates rapidly at order $l > 5$. We propose the use of stable higher-order moment summation formulas to address issues of numerical stability, and show that a reliable sketching based on higher-order moments is possible at the cost of some extra computation. We show that GPUs can be used to efficiently aggregate stable moment-based sketches at higher orders, allowing moment-based sketches that are both accurate and fast.

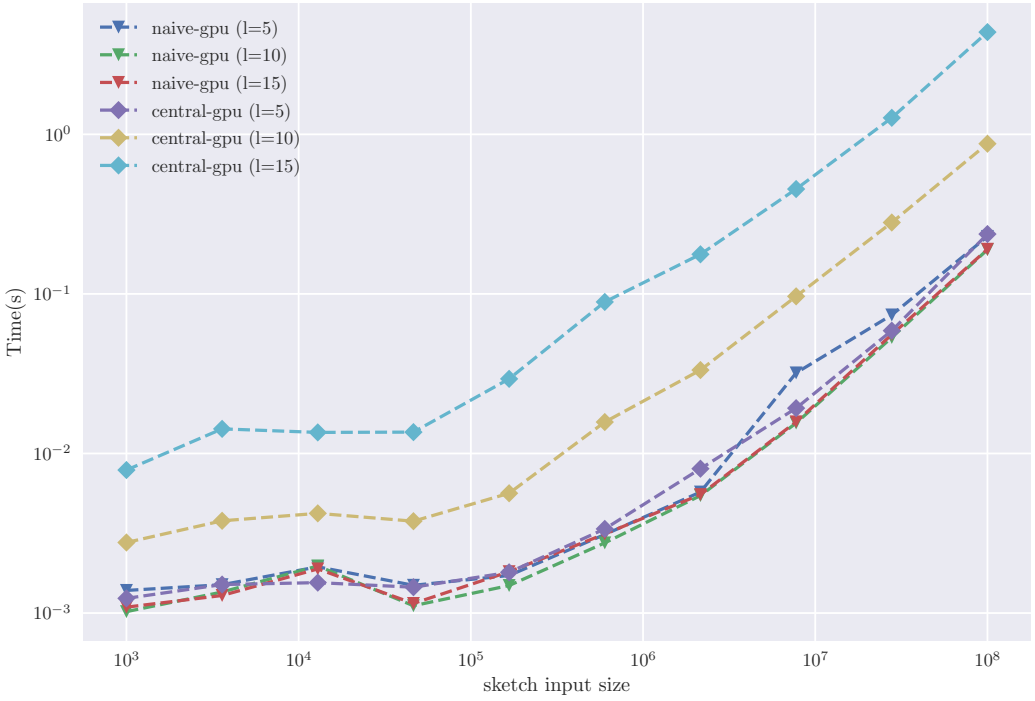


Fig. 11. Runtime of one-pass sketches on GPU

Additionally, we compare single-pass estimators against a related set of two-pass orthogonal series estimators, concluding that the cosine series is also a competitive, space-efficient estimator where two passes over the dataset are acceptable or the domain of the data is known.

Our primary conclusions for practitioners are the following:

- Moments sketch is accurate and fast in space-constrained settings compared to the state-of-the-art sample-based estimator, KLL.
- Moment-based sketching with naïve summation of sample moments is unstable at order $l > 5$.
- More sophisticated moment summation formulas can be used to improve reliability at some computational cost.
- Implementation of moment-based sketching for GPUs is practical and has benefits for speed and accuracy.
- The method of maximum entropy can be substituted with Legendre polynomials for a simpler implementation with reduced accuracy.
- If the domain of the input data is known or two passes are acceptable, we recommend the cosine orthogonal series as a simple, accurate estimator.

REFERENCES

[1] Naum IlÊźich Akhiezer. 1965. *The classical moment problem: and some related questions in analysis.* Vol. 5. Oliver & Boyd, Edinburgh.

[2] Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Frank Hutter, Michel Lang, Rafael G. Mantovani, Jan N. van Rijn, and Joaquin Vanschoren. 2017. OpenML Benchmarking Suites. arXiv:stat.ML/1708.03731

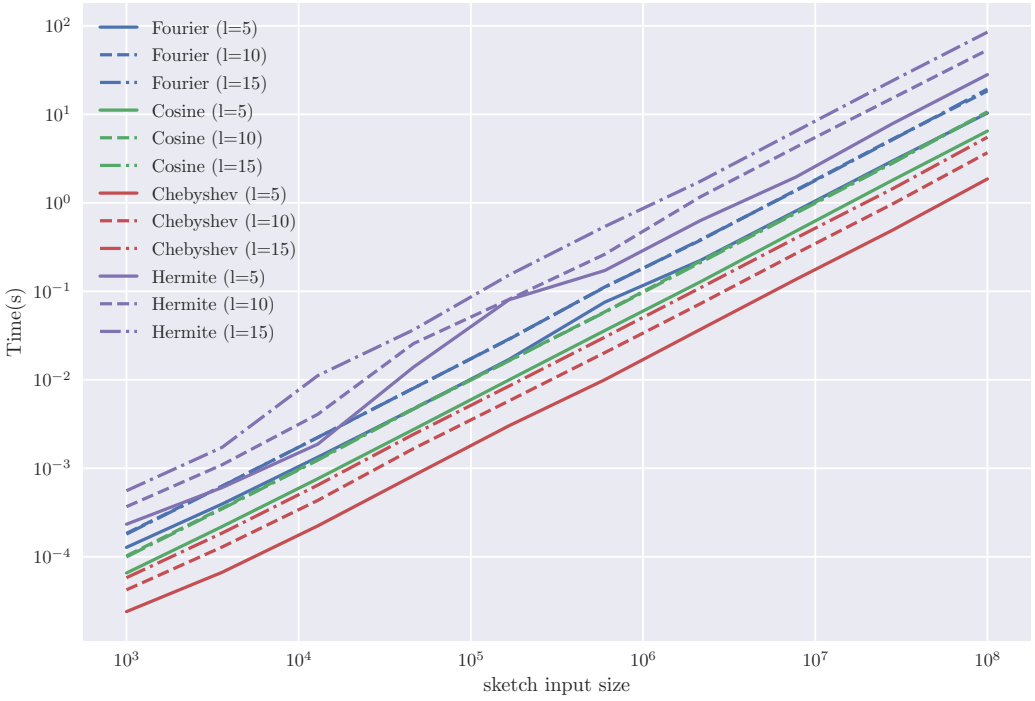


Fig. 12. Runtime of two-pass sketches

[3] Manuel Blum, Robert W. Floyd, Vaughan Pratt, Ronald L. Rivest, and Robert E. Tarjan. 1973. Time Bounds for Selection. *J. Comput. Syst. Sci.* 7, 4 (Aug. 1973), 448–461. https://doi.org/10.1016/S0022-0000(73)80033-9

[4] Nikolai N Cencov. 1962. Estimation of an unknown distribution density from observations. *Soviet Math.* 3 (1962), 1559–1566.

[5] Tony F. Chan, Gene H. Golub, and Randall J. LeVeque. 1983. Algorithms for Computing the Sample Variance: Analysis and Recommendations. *The American Statistician* 37, 3 (1983), 242–247. http://www.jstor.org/stable/2683386

[6] Moses Charikar, Kevin Chen, and Martin Farach-Colton. 2002. Finding Frequent Items in Data Streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP '02)*. Springer-Verlag, Berlin, Heidelberg, 693–703. http://dl.acm.org/citation.cfm?id=646255.684566

[7] John Cheng, Max Grossman, and Ty McKercher. 2014. *Professional CUDA C Programming* (1st ed.). Wrox Press Ltd., GBR.

[8] Graham Cormode and Shan Muthukrishnan. 2005. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms* 55, 1 (2005), 58–75.

[9] Jiu Ding, Noah H. Rhee, and Chenhua Zhang. 2016. On Polynomial Maximum Entropy Method for Classical Moment Problem. *Advances in Applied Mathematics and Mechanics* 8, 1 (2016), 117–127. https://doi.org/10.4208/aamm.2014.m504

[10] Ted Dunning and Otmar Ertl. 2019. Computing Extremely Accurate Quantiles Using t-Digests. arXiv:stat.CO/1902.04023

[11] Sam Efromovich. 2010. Orthogonal series density estimation. *Wiley Interdisciplinary Reviews: Computational Statistics* 2, 4 (2010), 467–476.

[12] Message P Forum. 1994. *MPI: A Message-Passing Interface Standard.* Technical Report. MPI Forum, Knoxville, TN, USA.

[13] Edward Gan, Jialin Ding, Kai Sheng Tai, Vatsal Sharan, and Peter Bailis. 2018. Moment-based quantile sketches for efficient high cardinality aggregation queries. *Proceedings of the VLDB Endowment* 11, 11 (2018), 1647–1660.

[14] Michael Greenwald and Sanjeev Khanna. 2001. Space-efficient Online Computation of Quantile Summaries. *SIGMOD Rec.* 30, 2 (May 2001), 58–66. https://doi.org/10.1145/376284.375670

[15] Nicholas J. Higham. 1993. The Accuracy Of Floating Point Summation. *SIAM J. Sci. Comput* 14 (1993), 783–799.

[16] Edwin T Jaynes. 1957. Information theory and statistical mechanics. *Physical review* 106, 4 (1957), 620.

[17] Z. Karnin, K. Lang, and E. Liberty. 2016. Optimal Quantile Approximation in Streams. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, New Brunswick, NJ, USA, 71–78.
[18] Andreas Klöckner, Nicolas Pinto, Yunsup Lee, Bryan Catanzaro, Paul Ivanov, and Ahmed Fasih. 2012. PyCUDA and PyOpenCL: A scripting-based approach to GPU run-time code generation. *Parallel Comput.* 38, 3 (2012), 157–174.
[19] Solomon Kullback. 1997. *Information theory and statistics*. Dover Publications, Massachusetts.
[20] Ge Luo, Lu Wang, Ke Yi, and Graham Cormode. 2016. Quantiles over Data Streams: Experimental Comparisons, New Analyses, and Further Improvements. *The VLDB Journal* 25, 4 (Aug. 2016), 449–472. https://doi.org/10.1007/s00778-016-0424-7
[21] John C Mason and David C Handscomb. 2002. *Chebyshev polynomials*. Chapman and Hall/CRC.
[22] Charles Masson, Jee E. Rim, and Homin K. Lee. 2019. DDSketch. *Proceedings of the VLDB Endowment* 12, 12 (Aug 2019), 2195–2205. https://doi.org/10.14778/3352063.3352135
[23] J Ian Munro and Mike S Paterson. 1980. Selection and sorting with limited storage. *Theoretical computer science* 12, 3 (1980), 315–323.
[24] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. 2008. Scalable Parallel Programming with CUDA. *Queue* 6, 2 (March 2008), 40–53. https://doi.org/10.1145/1365490.1365500
[25] CUDA Nvidia. 2011. Nvidia cuda c programming guide. *Nvidia Corporation* 120, 18 (2011), 8.
[26] Philippe Pébay, Timothy B Terriberry, Hemanth Kolla, and Janine Bennett. 2016. Numerically stable, scalable formulas for parallel and online computation of higher-order multivariate central moments with arbitrary weights. *Computational Statistics* 31, 4 (2016), 1305–1325.
[27] William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. 2007. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, Cambridge.
[28] Carl Runge. 1901. Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten. *Zeitschrift für Mathematik und Physik* 46, 224-243 (1901), 20.
[29] Nisheeth Shrivastava, Chiranjeeb Buragohain, Divyakant Agrawal, and Subhash Suri. 2004. Medians and Beyond: New Aggregation Techniques for Sensor Networks. In *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*. ACM, New York, NY, USA, 239–249. https://doi.org/10.1145/1031495.1031524
[30] Michael Stephanou, Melvin Varughese, Iain Macdonald, et al. 2017. Sequential quantiles via Hermite series density estimation. *Electronic Journal of Statistics* 11, 1 (2017), 570–607.
[31] Edward A Youngs and Elliot M Cramer. 1971. Some results relevant to choice of sum and sum-of-product algorithms. *Technometrics* 13, 3 (1971), 657–665.