Discrete Mathematics and Theoretical Computer Science **4**, 2001, 301–322

An Approximate L^p-Difference Algorithm for Massive Data Streams

Jessica H. Fong[1 †] and Martin Strauss[2]

[1]*Department of Computer Science, Princeton University, 35 Olden Street, Princeton, NJ, 08544*
jfong@cs.Princeton.EDU
[2]*AT&T Labs—Research, 180 Park Avenue, Florham Park, NJ 07932 USA*
mstrauss@research.att.com

received May 15, 2000, *revised Feb 13, 2001*, *accepted Mar 2001*.

Several recent papers have shown how to approximate the difference $\sum_i |a_i - b_i|$ or $\sum |a_i - b_i|^2$ between two functions, when the function values a_i and b_i are given in a data stream, and their order is chosen by an adversary. These algorithms use little space (much less than would be needed to store the entire stream) and little time to process each item in the stream. They approximate with small relative error. Using different techniques, we show how to approximate the L^p-difference $\sum_i |a_i - b_i|^p$ for any rational-valued $p \in (0, 2]$, with comparable efficiency and error. We also show how to approximate $\sum_i |a_i - b_i|^p$ for larger values of p but with a worse error guarantee. Our results fill in gaps left by recent work, by providing an algorithm that is precisely tunable for the application at hand.

These results can be used to assess the difference between two chronologically or physically separated massive data sets, making one quick pass over each data set, without buffering the data or requiring the data source to pause. For example, one can use our techniques to judge whether the traffic on two remote network routers are similar without requiring either router to transmit a copy of its traffic. A web search engine could use such algorithms to construct a library of small "sketches," one for each distinct page on the web; one can approximate the extent to which new web pages duplicate old ones by comparing the sketches of the web pages. Such techniques will become increasingly important as the enormous scale, distributional nature, and one-pass processing requirements of data sets become more commonplace.

1 Introduction

[Some of the following material is excerpted from [FKSV99], with the authors' permission. Readers familiar with [FKSV99] may skip to Section 1.1.]

Massive data sets are becoming more and more important in a wide range of applications, including observational sciences, product marketing, and monitoring and operations of large systems. In network operations, raw data typically arrive in *streams*, and decisions must be made by algorithms that make one pass over each stream, throw much of the raw data away, and produce "synopses" or "sketches" for further processing. Moreover, network-generated massive data sets are often *distributed*. Several different, physically separated network elements may receive or generate data streams that, together, comprise one

[†]Part of this work was done while the first author was visiting AT&T Labs.

logical data set. To be of use in operations, the streams must be analyzed locally and their synopses sent to a central operations facility. The enormous scale, distributed nature, and one-pass processing requirement on the data sets of interest must be addressed with new algorithmic techniques.

In [AMS96, KOR98, AGMS99, FKSV99], the authors presented a new technique: a space-efficient, one-pass algorithm for approximating the L^1 difference $\sum_i |a_i - b_i|$ or L^2 difference[‡] $\left(\sum_i |a_i - b_i|^2\right)^{1/2}$ between two functions, when the function values a_i and b_i are given as data streams, and their order is chosen by an adversary. Here we continue that work by showing how to compute $\sum_i |a_i - b_i|^p$ for any rational-valued $p \in (0,2]$. These algorithms fit naturally into a toolkit for Internet-traffic monitoring. For example, Cisco routers can now be instrumented with the NetFlow feature [CN98]. As packets travel through the router, the NetFlow software produces summary statistics on each *flow*.[§] Three of the fields in the flow records are source IP-address, destination IP-address, and total number of bytes of data in the flow. At the end of a day (or a week, or an hour, depending on what the appropriate monitoring interval is and how much local storage is available), the router (or, more accurately, a computer that has been "hooked up" to the router for monitoring purposes) can assemble a set of values $(x, f_t(x))$, where x is a source-destination pair, and $f_t(x)$ is the total number of bytes sent from the source to the destination during a time interval t. The L^p difference between two such functions assembled during different intervals or at different routers is a good indication of the extent to which traffic patterns differ.

Our algorithm allows the routers and a central control and storage facility to compute L^p differences efficiently under a variety of constraints. First, a router may want the L^p difference between f_t and f_{t+1}. The router can store a small "sketch" of f_t, throw out all other information about f_t, and still be able to approximate $\|f_t - f_{t+1}\|_p$ from the sketch of f_t and (a sketch of) f_{t+1}.

The functions $f_t^{(i)}$ assembled at each of several remote routers R_i at time t may be sent to a central tape-storage facility C. As the data are written to tape, C may want to compute the L^p difference between $f_t^{(1)}$ and $f_t^{(2)}$, but this computation presents several challenges. First, each router R_i should transmit its statistical data when R_i's load is low and the R_i-C paths have extra capacity; therefore, the data may arrive at C from the R_i's in an arbitrarily interleaved manner. Also, typically, the x's for which $f(x) \neq 0$ constitute a small fraction of all x's; thus, R_i should only transmit $(x, f_t^{(i)}(x))$ when $f_t^{(i)}(x) \neq 0$. The set of transmitted x's is not predictable by C. Finally, because of the huge size of these streams,[¶] the central facility will not want to buffer them in the course of writing them to tape (and cannot read from one part of the tape while writing to another), and telling R_i to pause is not always possible. Nevertheless, our algorithm supports approximating the L^p difference between $f_t^{(1)}$ and $f_t^{(2)}$ at C, because it requires little workspace, requires little time to process each incoming item, and can process in one pass all the values of both functions $\{(x, f_t^{(1)}(x))\} \cup \{(x, f_t^{(2)}(x))\}$ in any permutation.

Our L^p-difference algorithm achieves the following performance for rational $p \in (0,2]$:

---

‡ Approximating the L^p difference, $\|\langle a_i\rangle - \langle b_i\rangle\|_p = \left(\sum |a_i - b_i|^p\right)^{1/p}$, is computationally equivalent to approximating the easier-to-read expression $\sum |a_i - b_i|^p$. We will use these interchangeably when discussing computational issues.

§ Roughly speaking, a "flow" is a semantically coherent sequence of packets sent by the source and reassembled and interpreted at the destination. Any precise definition of "flow" would have to depend on the application(s) that the source and destination processes were using to produce and interpret the packets. From the router's point of view, a flow is just a set of packets with the same source and destination IP-addresses whose arrival times at the routers are close enough, for a tunable definition of "close."

¶ In 1999, a WorldNet gateway router generated more that 10Gb of NetFlow summary data each day.

Consider two data streams of length at most n, each representing the non-zero points on the graph of an integer-valued function on a domain of size n. Assume that the maximum value of either function on this domain is M. Then a one-pass streaming algorithm can compute with probability $1-\delta$ an approximation A to the L^p-difference B of the two functions, such that $|A-B| \leq \varepsilon B$, using total space and per-item processing time $(\log(M)\log(n)\log(1/\delta)/\varepsilon)^{O(1)}$. The input streams may be interleaved in an arbitrary (adversarial) order.

1.1 L^p-Differences for p other than 1 or 2

While the L^1- and L^2- differences are most commonly used, the L^p-differences for other p, say the $L^{1.5}$-difference, provide additional information. In particular, there are $\langle a_i\rangle$, $\langle b_i\rangle$, $\langle a'_i\rangle$, and $\langle b'_i\rangle$ such that $\sum|a_i-b_i| = \sum|a'_i-b'_i|$ and $\sum|a_i-b_i|^2 = \sum|a'_i-b'_i|^2$ but $\sum|a_i-b_i|^{1.5}$ and $\sum|a'_i-b'_i|^{1.5}$ are different. As p increases, the measure $\sum|a_i-b_i|^p$ attributes more significance to a large individual difference $|a_{i_0}-b_{i_0}|$, while reducing the significance of a large number of differences, $|\{i : |a_i-b_i| > 0\}|$. By showing how to compute the L^p difference for varing p, we provide an approximate difference algorithm that is precisely tunable for the application at hand.

We also give an algorithm for $p > 2$, though with an error guarantee somewhat worse than the guarantee available for the $p \leq 2$ cases. Still, that result is a randomized algorithm with the correct mean, which is an advantage in some situtations.

1.2 Organization

The rest of this paper is organized as follows. In Section 2, we describe precisely our model of computation and its complexity measure. We present our main technical results in Section 3. We discuss the relationship of our algorithm to other recent work and present some open problems, in Section 4.

Proofs of lemmas end with a $\Box$ and other proofs and definitions terminate with a $\blacksquare$

2 Background

We describe the details of our algorithm in terms of the streaming model used in [FKSV99]. This model is closely related to that of [HRR98].

2.1 Model of Computation

A *data stream* is a sequence of data items $\sigma_1, \sigma_2, \ldots, \sigma_n$ such that, on each *pass* through the stream, the items are read once in increasing order of their indices. We assume the items σ_i come from a set of size M, so that each σ_i has size $\log M$. In the computational model, we assume that the input is one or more data streams. We focus on two resources—the *workspace* required in words and the *time to process* an item in the stream, but disregard pre- and post-processing time.

It is immediate to adapt our algorithm to the sketch model of [FKSV99, BCFM98]. The latter used sketches to check whether two documents are nearly duplicates. A sketch can also be regarded as a *synopsis data structure* [GM98].

2.2 Medians and Means of Unbiased Estimators

We now recall a general technique of randomized approximation schemes.

Lemma 1 *Let X be a real-valued random variable such that, for some c, $E[X^2] \leq c \cdot \mathrm{var}[X]$. Then, for any $\varepsilon, \delta > 0$, there exists a random variable Z such that $\Pr(|Z - E[X]| \geq \varepsilon E[X]) \leq \delta$. Furthermore, Z is a function of $O(\log(1/\delta)/\varepsilon^2)$ independent samples of X.*

Proof. Let Y be the average of $8c/\varepsilon^2$ independent copies of X. Then $E[Y] = E[X]$ and $\mathrm{var}[Y] \leq \varepsilon^2 E^2[X]/8$. By the Chebychev inequality, $\Pr(|Y - E[X]| > \varepsilon E[X]) \leq \frac{\mathrm{var}(Y)}{\varepsilon^2 E^2[X]} \leq \frac{1}{8}$. Let Z be the median of $4\log(1/\delta)$ independent copies of Y. Then $|Z - E[X]| \geq \varepsilon E[X]$ iff for at least half the Y_i's, $|Y_i - E[X]| \geq \varepsilon E[X]$. Since, for each i, this happens only with probability $1/8$, the Chernoff inequality implies that $\Pr(|Z - E[X]| \geq \varepsilon E[X]) \leq \delta$. □

3 The Algorithm

In this section we prove our main theorem:

Theorem 2 *For rational $p \in (0,2]$, the L^p-difference of two functions $\langle a_i \rangle$ and $\langle b_i \rangle$ can be computed in time and space $(\log(n)\log(M)\log(1/\delta)/\varepsilon)^{O(1)}$, where (1) the input is a data stream of values a_i or b_i, $0 \leq i < n$, from a set of size M, (2) one can output a random variable X such that $|X - f| < \varepsilon f$ with probability at least $1 - \delta$, and (3) computation of X can be done by making a single pass over the data.*

3.1 Intuition

We first give an intuitive overview of the algorithm. Our goal is to approximate $L_p = \sum |a_i - b_i|^p$, where the values a_i, $b_i \in [0, M)$ are presented in a stream in any order, and the index i runs up to n. We are given tolerance ε and maximum error probability δ.

The input is a stream consisting of tuples of the form (i, c, θ), where $0 \leq i < n$, $0 \leq c < M$, and $\theta \in \{\pm 1\}$. The tuple (i, c, θ) denotes the data item a_i with value c if $\theta = +1$ and indicates that $b_i = c$ if $\theta = -1$. We wish to output a random variable Z such that $\Pr(|Z - L_p| > \varepsilon L_p) < \delta$, using total space and per-item processing time polynomial in $(\log(n)\log(M)\log(1/\delta)/\varepsilon)$.

In the next few sections, we will construct a randomized function $f(r,x)$ such that

$$E\left[(f(r,b) - f(r,a))^2\right] \approx |b - a|^p. \tag{1}$$

In a first reading of the algorithm below, the reader may assume f to be a deterministic function with $(f(b) - f(a))^2 = |b - a|^p$. The algorithm proceeds as in Figure 1.

To see how the algorithm works, first focus on single values for k and ℓ. Let Z be an abbreviation for $Z_{k,\ell} = \sum_i \sigma_i (f(a_i) - f(b_i))$. We separate the diagonal and off-diagonal terms of Z^2 to simplify,

$$\begin{aligned} E\left[Z^2\right] &= E\left[\sum_i \sigma_i^2 (f(a_i) - f(b_i))^2 + \sum_{i \neq i'} \pm \sigma_i \sigma_{i'} (f(a_i) - f(b_i))(f(a_{i'}) - f(b_{i'}))\right] \\ &\approx E\left[\sum_i \sigma_i^2 |a_i - b_i|^p + \sum_{i \neq i'} \pm \sigma_i \sigma_{i'} (f(a_i) - f(b_i))(f(a_{i'}) - f(b_{i'}))\right] \\ &= E\left[\sum_i |a_i - b_i|^p\right]. \end{aligned} \tag{2}$$

Fig. 1: Main algorithm, intuition

Algorithm $L^p(\langle(i,c,\theta)\rangle)$

Initialize:

  For $k = 1$ to $O(\log(1/\delta))$ do
    For $\ell = 1$ to $O(1/\varepsilon^2)$ do
      $Z_{k,\ell} = 0$
      pick sample points for
      a family $\{\sigma_i\}$ of n 4-wise independent ± 1-valued random variables and
      a family $\{\vec{r}_i\}$ of n 4-wise independent random variables (described further below)

Stream processing:

  For each tuple (i,c,θ) in the input stream do
    For $k = 1$ to $O(\log(1/\delta))$ do
      For $\ell = 1$ to $O(1/\varepsilon^2)$ do
        $Z_{k,\ell}$ += $\sigma_i \theta f_{\vec{r}_i}(c)$

Report:

  Output $\text{median}_k\ \text{avg}_\ell\ Z_{k,\ell}^2$.

The result follows since $\sigma_i^2 \equiv 1$, $E[\sigma_i] = 0$, and σ_i and $\sigma_{i'}$ are independent for $i \neq i'$. Similarly, $\text{var}(Z^2) \leq O(E^2[Z^2])$. We can apply Lemma 1 and take a median of means of independent copies of Z^2 to get the desired result $\sum |a_i - b_i|^p$.

3.2 Construction of f

3.2.1 Overview

Construction of f is the main technical content of this paper. We construct a function $f : \mathbb{Z} \to \mathbb{Z}$ such that

$$E\left[(f(b) - f(a))^2\right] = (1 \pm \varepsilon)|b - a|^p.$$

We put $f_{\vec{r}}(x) = c\phi(\vec{r})d(T_{\vec{r}}(0), T_{\vec{r}}(x))$ (rounding appropriately from reals to integers), defining the component functions as follows. When the choice of $\vec{r}$ is clear, we drop the subscript. We will come back to f in the overview in the next subsection. The function $d(a,b)$ satisfies

- $|d(a,b)| \in O(|b-a|^{p/2})$ for all a and b,
- $|d(a,b)| \in \Omega(|b-a|^{p/2})$ for a significant fraction of a and b, and
- $d(c,b) - d(c,a) = d(a,b)$ for all c.

The family $\{T_{\vec{r}}\}$ of *transformations* on the reals, with corresponding *inverse scale factors* $\phi(\vec{r})$ are such that:

- the transformation is an approximate isometry, *i.e.*, $|a-b|^p \approx c^2\phi^2(\vec{r})|T_{\vec{r}}(b)-T_{\vec{r}}(a)|^p$, and,
- the distribution on $\phi(\vec{r})d(T_{\vec{r}}(a),T_{\vec{r}}(b))/|b-a|^{p/2}$ is approximately constant, independent of a and b.

Due to the above properties, our function, acting on parameters a and b, is tightly bounded near $|b-a|^p$. We compute an upper bound of

$$\begin{aligned}
E_{\vec{r}}\left[(f(b)-f(a))^2\right] &= E_{\vec{r}}\left[c^2\phi^2(\vec{r})(d(T_{\vec{r}}(0),T_{\vec{r}}(a))-d(T_{\vec{r}}(0),T_{\vec{r}}(b)))^2\right] \\
&= E_{\vec{r}}\left[c^2\phi^2(\vec{r})(d(T_{\vec{r}}(a),T_{\vec{r}}(b)))^2\right] \\
&\in O\left(E_{\vec{r}}\left[c^2\phi^2(\vec{r})|T_{\vec{r}}(a)-T_{\vec{r}}(b)|^p\right]\right) \\
&\approx O(|a-b|^p).
\end{aligned} \tag{3}$$

Because $|d(\alpha,\beta)| \in \Omega\left(|\beta-\alpha|^{p/2}\right)$ for a significant fraction of α,β (according to the distribution $(\alpha,\beta) = (T_{\vec{r}}(a),T_{\vec{r}}(b))$), the Markov inequality gives

$$E_{\vec{r}}\left[d(T_{\vec{r}}(a),T_{\vec{r}}(b))^2\right] \geq \Omega\left(|T_{\vec{r}}(a)-T_{\vec{r}}(b)|^p\right).$$

We get the lower bound similarly as above. It follows that $E_{\vec{r}}\left[(f(b)-f(a))^2\right] \in \Theta\left(|b-a|^p\right)$.

We then show that the distribution on $\phi(\vec{r})d(T_{\vec{r}}(a),T_{\vec{r}}(b))/|b-a|^{p/2}$ is approximately independent of a and b, thus $E_{\vec{r}}\left[(f(b)-f(a))^2\right] \approx c'(1\pm\varepsilon)|b-a|^p$, for c' independent of a and b. By choosing c appropriately, we can arrange that $c'=1$. Finally, we address the issue of precision.

We now proceed with a detailed construction of f.

3.2.2 Construction of d

The function $d(a,b)$ takes the form $d(a,b) = \sum_{a\leq j<b}\pi_j$, where π_j is a ±1-valued function of j, related to a function described in [FKSV99]. This function is selected to fulfil the properties listed in the overview.

First, find integers u,v such that $\frac{\log(v-u)}{\log(v+u)} = p/2$, and, for technical reasons, $v-u \geq 17$ and $u \geq 2$. To do this, find integers $\alpha > 1$ and $\beta > 1$ with $p/2 = \alpha/\beta$ (by hypothesis, a rational number). Put $v = 2^{\beta-1}+2^{\alpha-1}$ and $u = 2^{\beta-1}-2^{\alpha-1}$; thus $\frac{\log(v-u)}{\log(v+u)} = p/2$. If $v-u < 17$ or $u = 1$, then (repeatedly, if necessary) replace v by v^2+u^2 and replace u by $2uv$. Observe that $\frac{\log(v^2+u^2-2uv)}{\log(v^2+u^2+2uv)} = \frac{\log(v-u)}{\log(v+u)}$ and the new value $v^2+u^2-2uv = (v-u)^2$ is greater than the old value $v-u$. Also note that $(v+u)^{p/2} = v-u$.

Now, we define a sequence π as a string of $+1$'s and -1's, as follows. Let $\pi = \lim_{i\to\infty}\pi_{(i)}$ where $\pi_{(i)}$ is defined recursively for $i \geq 1$ as

$$\pi_{(1)} = (+1)^u(-1)^v \tag{4}$$

$$\pi_{(i+1)} = \pi_{(i)}^u\overline{\pi_{(i)}^v}, \tag{5}$$

and $\overline{\pi_{(i)}}$ denotes $\pi_{(i)}$ with all $+1$'s replaced by -1's and all -1's replaced by $+1$'s. Note that $\pi_{(i)}$ is a prefix of $\pi_{(i+1)}$. For example, a graph of π with $u=1$ and $v=3$ is given in Figure 2 (Figure 2 also describes sets $S_{s,t}$, to be defined later).

Let π_j (differentiate this from $\pi_{(j)}$) denote the j'th symbol of π. Now $d(a,b) = \sum_{j=a}^{b-1} \pi_j$ is the discrepancy between $+1$'s and -1's in the interval $[a,b)$. The self-similarity of π allows the value of d, applied on random transformations of the parameters a and b to have the desired expected value. Note that d and π depend on u and v, which, in turn, depend on p. We only consider one set of values for u,v at a time and drop them from the notation.

3.2.3 Adding randomness

We now define the transformation $T_{\vec{r}}()$ and the reconstruction $\phi(\vec{r})$.

Definition 3 Set

$$\begin{aligned}
u,v: & \quad \text{integers such that } \frac{\log(v-u)}{\log(v+u)} = p/2 \\
c_1 &= \min\left(\frac{1}{2}, \frac{3v}{4(v-u)}\right) \\
c_2 &= 3u+3v \\
\eta &= \frac{\log(v-6/8)-\log(v-10/8)}{\log(v+u)} \cdot \frac{5/8-3/8}{v+u} \\
N_1 &= \log(8)/\log(u+v) \\
N_2 &= N_1 + 8c_2\log(M)/c_1\eta\varepsilon \\
N_3 &= \text{ least power of } (u+v) \text{ such that } N_3 \geq (u+v)MN_2.
\end{aligned}$$

These values are motivated by several subsequent proofs, particularly those for Lemma 12 and (*average*); we state them now for clarity. Let r be $(u+v)^s$, where s is chosen uniformly at random from the real interval $[N_1,N_2]$. Let r' be an integer chosen uniformly at random from $[0,N_3)$. Put $T_{\vec{r}}(a) = ra + r'$ and put $\phi(\vec{r}) = r^{-p/2}$. ∎

Let $\hat{d}(a,b)$ denote $\phi(\vec{r})d(T_{\vec{r}}(a),T_{\vec{r}}(b))$, *i.e.*, d acting in the transformed domain, rounding the arguments of d appropriately from reals to integers (we will specify the rounding precisely below).

3.2.4 Expectation of $d(a,b)$

We now prove that $|d(a,b)|$ has tight upper and lower bounds. (More precisely, we show the lower bound for $\hat{d}$ as defined above.) This utilizes a succession of properties about π, d, $T_{\vec{r}}$, and $\phi(\vec{r})$. Some of the following assume that $v-u \geq 17$. The constants $\gamma_0,\gamma_1,\gamma_2$ below may depend on p,M, and ε, but are bounded uniformly in M and ε. The dependence of $\gamma_0,\gamma_1,\gamma_2$ on p,M, and ε is hard to find analytically; in Section 4.3 we discuss in more detail a method for determining the gammas.

$$d((u+v)a,(u+v)b) \;=\; -(v-u)d(a,b). \qquad (\textit{homogeneity})$$

$$\text{For all } r, \text{ all } a \le b < (u+v)^r \text{ and all } x, |d(a,b)| = |d(a+x(u+v)^r, b+x(u+v)^r)|. \qquad (\textit{periodicity})$$

$$\text{For some } c_2,\ |d(a,b)| \;\le\; c_2(b-a)^{p/2}. \qquad (\textit{upper bound})$$

$$\text{For some } c_1 > 0 \text{ and some } \eta > 0, \Pr_{\vec{r}}\left(\left|\hat{d}(a,b)\right| \;\ge\; c_1|b-a|^{p/2}\right) > \eta. \qquad (\textit{averaged lower bound})$$

$$\left.\begin{array}{llcl} \text{For some } \gamma_0 > 0, & E_{\vec{r}}\left[\left|\hat{d}(a,b)\right|\right] & = & \gamma_0|(b-a)|^{p/2}(1\pm\varepsilon). \\ \text{For some } \gamma_1 > 0, & E_{\vec{r}}\left[\hat{d}^2(a,b)\right] & = & \gamma_1|(b-a)|^{p}(1\pm\varepsilon). \\ \text{For some } \gamma_2 > 0, & E_{\vec{r}}\left[\hat{d}^4(a,b)\right] & = & \gamma_2|(b-a)|^{2p}(1\pm\varepsilon). \end{array}\right\} \qquad (\textit{average})$$

Claim 4 *The sequence $\pi_{(i+1)}$ can be obtained by starting with $\pi_{(i)}$ and replacing each $+1$ with $\pi_{(1)}$ and each -1 with $\overline{\pi_{(1)}}$.*

Proof. Consider a top-down rather than a bottom-up recursive definition of π. ∎

Proof. [(*homogeneity*) and (*periodicity*)] The homogeneity and periodicity properties are immediate from the definition of π and from Claim 4. ∎

Proof. [(*upper bound*)] Next, consider the upper bound property. Note that (*homogeneity*) implies (*upper bound*) infinitely often, *i.e.*, for bounded a and b, we have

$$\begin{aligned} |d(a(u+v)^s, b(u+v)^s)| &= (v-u)^s d(a,b) \\ &= (v+u)^{sp/2} d(a,b) \\ &= \left(\frac{d(a,b)}{|b-a|^{p/2}}\right)(b(u+v)^s - a(u+v)^s)^{p/2} \end{aligned} \tag{6}$$

Intuitively, since $d(a,b) \approx d(a',b')$ for $a \approx a'$ and $b \approx b'$, the result follows.

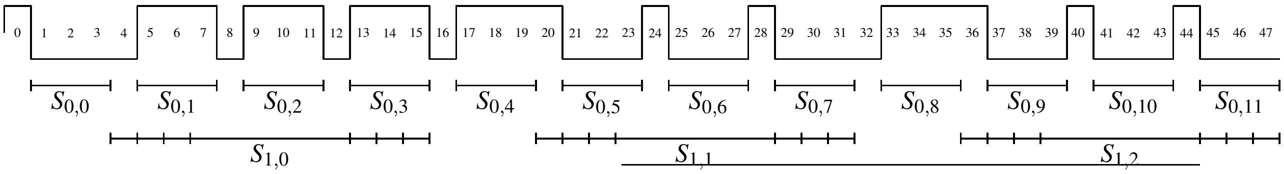
Formally, assume by induction that, for all a and b with $0 \le b-a \le (u+v)^r$, we have $|d(a,b)| \le q_r|b-a|^{p/2}$. Now assume that $(u+v)^r < b-a \le (u+v)^{r+1}$. Let a' be the smallest multiple of $(u+v)$ that is at least a and let b' be the largest multiple of $(u+v)$ that is at most b. Then $|d(a',b')| \le q_r(b'-a')^{p/2}$ by homogeneity and by induction. Also, $|d(a,b)-d(a',b')| \le 2(u+v)$. Thus $|d(a,b)| \le q_r(b'-a')^{p/2} + 2(u+v)$. Let q_{r+1} be the maximum over $(u+v)^r < b-a \le (u+v)^{r+1}$ of $|d(a,b)|/|b-a|^{p/2}$; then

$$\begin{aligned} q_{r+1} &\le q_r + 2(u+v)|b-a|^{-p/2} \\ &\le q_r + 2(u+v)(u+v)^{-rp/2} \\ &= q_r + 2(u+v)(v-u)^{-r} \end{aligned} \tag{7}$$

Similarly,

$$q_r \le q_{r-1} + 2(u+v)(v-u)^{-(r-1)}.$$

Fig. 2: Geometric view of π (continuous polygonal curve) for $u = 1$ and $v = 3$. The sets $S_{s,t}$ are indicated by segments with vertical ticks. Each element of $S_{s,t}$ is a pair (α, β), indicated in the diagram by two of the vertical ticks near opposite ends of the interval labeled $S_{s,t}$. The discrepancy of π is relatively high over intervals with endpoints in $S_{s,t}$.



Unwinding the recursion, we have

$$\begin{aligned} q_{r+1} &= q_1 + 2(u+v)\left[(v-u)^{-r} + (v-u)^{-(r-1)} + \cdots + (v-u)^{-1}\right] \\ &\le q_1 + 2(u+v), \end{aligned} \tag{8}$$

where q_1 is such that $d(a,b) \le q_1|b-a|^{p/2}$ for all $0 < b-a \le v+u$. Since, for these a and b, we have $d(a,b) \le v+u$ and $|b-a|^{p/2} \ge 1$, we can take $q_1 = v+u$, and so we can take $c_2 = 3u+3v$. ∎

Proof. [(*averaged lower bound*)]
The proof consists of two lemmas. We identify a set S of (a,b) values. We then show in Lemma 6 that $|d(a,b)|$ is large on S and we show in Lemma 7 that the set S itself is big. The result follows.

Definition 5 Fix u and v. We define a set S of (a,b) values as follows. For each integer s and t, let $S_{s,t}$ consist of the pairs (a,b) such that

$$\begin{cases} t(u+v)^{s+1} + u(u+v)^s & \le \quad a \quad < \quad t(u+v)^{s+1} + (u+1)(u+v)^s \\ t(u+v)^{s+1} + (u+v-1)(u+v)^s & < \quad b \quad \le \quad (t+1)(u+v)^{s+1} \end{cases}$$

Let S be the (disjoint) union of all $S_{s,t}$. ∎

Geometrically speaking, s adjusts the size of the set and t linearly translates the range of the set. Note that $S_{0,0}$ is the singleton $(u, u+v)$, *i.e.*, the endpoints of the interval of v -1's (the interval of maximum discrepancy in $\pi_{(1)}$). Elements of $S_{s,0}$ are close to analogs of $S_{0,0} = \{(u, u+v)\}$, scaled-up (by $(u+v)^s$). Elements of $S_{s,t}$ are analogs of elements of $S_{s,0}$, translated (by $t(u+v)^{s+1}$).

Figure 2 shows the case of $u = 1$ and $v = 3$. Here,

$$\pi_{(2)} = +1-1-1-1 \quad -1+1+1+1 \quad -1+1+1+1 \quad -1+1+1+1.$$

The component $S_{0,2}$ is the singleton $\{(9,12)\}$. The component $S_{1,0}$ is $\{4,5,6,7\} \times \{13,14,15,16\}$. The element $(a,b) = (5,14) \in S_{1,0}$ leads to $d(5,14)$ which is the sum of the 9 symbols

$$+1+1+1 \quad -1+1+1+1 \quad -1+1.$$

It has a relatively high discrepancy.

Lemma 6 *For some constant c_1, for each $(a,b) \in S$, we have*

$$|d(a,b)| \geq c_1|b-a|^{p/2}.$$

Proof. We show the lemma for $(a,b) \in S_{s,0}$. The general result follows immediately from (*periodicity*).

One can easily show that the lower bound property is satisfied if $u = 0$. We now assume $u > 0$.

For each s, we define close to maximal q_s such that for all a,b as defined in Definition 5 (i.e. with $u(v+u)^s \leq a < (u+1)(v+u)^s$ and $(v+u-1)(v+u)^s < b \leq (v+u)^{s+1}$), we have $|d(a,b)| \geq q_s(b-a)$. (Recall $t = 0$.) We will consider q_s for $s = 0$, show that $q_{s-1} - q_s$ drops exponentially in s, then bound the sum of $|q_{s-1} - q_s|$. We consider two cases, depending on whether or not $u \leq v/2$.

First, suppose $u \leq v/2$. Note that $\pi_{(1)} = \overbrace{+1+1\cdots+1}^{u}\overbrace{-1-1\cdots-1}^{v}$. If $s = 0$, we have $a = u$ and $b = v+u$. The discrepancy $d(a,b)$ consists of the sum of v copies of -1, so $|d(a,b)| = v = b-a$, and we want $v \geq q_0 v^{p/2}$. Since $v \geq v^{p/2}$, we can put $q_0 = 1$.

Consider $s \geq 1$. Suppose a,b are such that $u(v+u)^s \leq a < (u+1)(v+u)^s$ and $(v+u-1)(v+u)^s < b \leq (v+u)^{s+1}$. Define a' to be the greatest multiple of $(v+u)$ that is at most a and define b' to be the smallest multiple of $(v+u)$ that is at least b.

Using the homogeneity property and induction,

$$\begin{aligned}
|d(a',b')| &= (v-u)\left|d\left(\frac{a'}{v+u},\frac{b'}{v+u}\right)\right| \\
&\geq q_{s-1}(v-u)\left(\frac{b'-a'}{v+u}\right)^{p/2} \\
&\geq q_{s-1}(b'-a')^{p/2} \\
&\geq q_{s-1}(b-a)^{p/2}.
\end{aligned} \tag{11}$$

It follows that

$$|d(a,b)| \geq q_{s-1}(b-a)^{p/2} - 2v.$$

Therefore we want to define q_s such that $|d(a,b)| \geq q_{s-1}(b-a)^{p/2} - 2v \geq q_s(b-a)^{p/2}$, *i.e.*, such that

$$q_s \leq q_{s-1} - 2v(b-a)^{-p/2}. \tag{12}$$

Note that $v - u \geq 17$ implies $(v-2) \geq \frac{v+u}{2}$ and $u \leq v/2$ implies $2(v-u) \geq v$. Also, note that $b - a \geq (v-2)(u+v)^s$. We have

$$\begin{aligned}
q_{s-1} - 2v(b-a)^{-p/2} &\geq q_{s-1} - 2v\left((v-2)(v+u)^s\right)^{-p/2} \\
&\geq q_{s-1} - 2v\left((v+u)^{s+1}/2\right)^{-p/2} \\
&\geq q_{s-1} - 4v(v-u)^{-(s+1)} \\
&\geq q_{s-1} - 8(v-u)(v-u)^{-(s+1)} \\
&= q_{s-1} - 8(v-u)^{-s}.
\end{aligned} \tag{13}$$

Thus it suffices to make $q_s \leq q_{s-1} - 8(v-u)^{-s}$, for $s \geq 1$. Since

$$8\sum_{s\geq 1}(v-u)^{-s} \leq 8\sum_{s\geq 1} 17^{-s} = 1/2,$$

we can make all $q_s = q_0 - 1/2 = 1/2$.

Now, assume $u \geq v/2$. When $s = 0$, so that $a = u$ and $b = v + u$, then $d(a,b) = -v$ and we need $|d(a,b)| \geq q_0 v^{p/2}$. Since

$$\begin{aligned} \frac{v}{v^{p/2}} &\geq \frac{v}{(v+u)^{p/2}} \\ &= \frac{v}{v-u}, \end{aligned} \tag{15}$$

We can put $q_0 = \frac{v}{v-u}$.

Now consider $s \geq 1$ when $u \geq v/2$. As in (12), we want to define q_s such that

$$q_s \leq q_{s-1} - 2v(b-a)^{-p/2}.$$

Since $b - a \geq (v-2)(v+u)^s \geq 1/2(v+u)^{s+1}$, we have

$$(b-a)^{-p/2} \leq 2^{p/2}(v+u)^{-(p/2)(s+1)} \leq 2(v-u)^{-(s+1)}.$$

Thus

$$q_{s-1} - 2v(b-a)^{-p/2} \geq q_{s-1} - 4v(v-u)^{-(s+1)},$$

and it suffices to make

$$q_s \leq q_{s-1} - 4v(v-u)^{-(s+1)}.$$

Unwinding the recursion, this becomes

$$q_s \leq q_0 - 4v \sum_{s=2}^{\infty} (v-u)^{-s}.$$

We get

$$\begin{aligned} q_0 - 4v \sum_{s=2}^{\infty} (v-u)^{-s} &= \frac{v}{v-u} - 4v \sum_{s=2}^{\infty} (v-u)^{-s} \\ &= \frac{v}{v-u} - \frac{4v}{(v-u)^2} \frac{1}{1 - 1/(v-u)} \\ &= \frac{v}{v-u} \left(1 - \frac{4}{(v-u)} \frac{1}{1 - 1/(v-u)}\right) \\ &= \frac{v}{v-u} \left(1 - \frac{4}{v-u-1}\right) \\ &\geq \frac{3v}{4(v-u)}, \end{aligned} \tag{16}$$

using the fact that $v - u \geq 17$ in the last line. It suffices to put $q_s = \frac{3v}{4(v-u)}$.

We can make c_1 in (*averaged lower bound*) be the minimum of $1/2$ and $\frac{3v}{4(v-u)}$. □

We now return to the proof of (*averaged lower bound*) by showing that S has positive probability.

Lemma 7 *Let $a, b < M$ be arbitrary. Then for any u, v there exists $\eta > 0$ such that* $\Pr\left((T_{\vec{r}}(a), T_{\vec{r}}(b)) \in S\right) \geq \eta$.

Proof.
With probability $\frac{\log(v-6/8)-\log(v-10/8)}{\log(v+u)}$, we have

$$\log(v-10/8) \leq \log(r(b-a)) \bmod \log(v+u) < \log(v-6/8),$$

i.e., for some integer s,

$$(v-10/8)(u+v)^s \leq r(b-a) < (v-6/8)(u+v)^s. \tag{17}$$

Find the integer t with $t(u+v)^{s+1} \leq ra+r' < (t+1)(u+v)^{s+1}$. With probability $\frac{5/8-3/8}{v+u}$,

$$t(u+v)^{s+1} + (u+3/8)(u+v)^s \leq ra+r' < t(u+v)^{s+1} + (u+5/8)(u+v)^s. \tag{18}$$

If both (17) and (18) hold, then the following also holds:

$$t(u+v)^{s+1} + (v+u-7/8)(u+v)^s \leq rb+r' < t(u+v)^{s+1} + (v+u-1/8)(u+v)^s.$$

It follows that

$$\begin{cases} t(u+v)^{s+1} + (u+3/8)(u+v)^s & \leq & ra+r' & \leq & t(u+v)^{s+1} + (u+5/8)(u+v)^s \\ t(u+v)^{s+1} + (v+u-7/8)(u+v)^s & \leq & rb+r' & \leq & t(u+v)^{s+1} + (v+u-1/8)(u+v)^s, \end{cases} \tag{19}$$

whence $(T_{\vec{r}}(a), T_{\vec{r}}(b)) \in S_{s,t} \subseteq S$, with positive probability. If $N_1 \geq \log(8)/\log(u+v)$, then $(u+v)^s/8 \geq 1$. It follows that $(T_{\vec{r}}(a) \pm 1, T_{\vec{r}}(b) \pm 1) \in S_{s,t} \subseteq S$ with positive probability. (The ± 1's allow us to round $T_{\vec{r}}()$ to a nearby integer in an arbitrary way.) □

Let $\eta > 0$ denote the probability that $ra+r'$ and $rb+r'$ are as above. (This concludes the proof of (*averaged lower bound*).) ∎

Proof. [(*average*)]
We show $E\left[\hat{d}(a_1, b_1)\right] \approx \gamma_0(b_1-a_1)^{p/2}$. The other conditions are similar.
From property (*upper bound*), we conclude that

$$E\left[\hat{d}(a_1, b_1)\right] \leq c_2(b_1-a_1)^{p/2}. \tag{20}$$

Similarly, from property (*averaged lower bound*) and the Markov inequality, we conclude

$$E\left[\hat{d}(a_1, b_1)\right] \geq \eta c_1(b_1-a_1)^{p/2}. \tag{21}$$

Equations 20 and 21 indicate that some multiple of $E[\hat{d}(a,b)]$ gives a $c_2/(\eta c_1)$-factor approximation to $|b-a|^{p/2}$. The equations leave open the possibility, however, that

$$E\left[\left|\hat{d}(a_1, b_1)\right|\right] = \eta c_1(b_1-a_1)^{p/2}$$

while

$$E\left[\left|\hat{d}(a_2, b_2)\right|\right] = c_2(b_2-a_2)^{p/2},$$

where $c_2/(\eta c_1) > (1+\varepsilon)$. If that were the case, no multiple of $E\left[\left|\hat{d}(a_2,b_2)\right|\right]$ would be a $(1\pm\varepsilon)$-factor approximation. Fortunately, as we show, a $(1\pm\varepsilon)$-approximation *does* result from our algorithm. Our proof proceeds by showing that

$$\frac{\left|\hat{d}(a_1,b_1)\right|}{|b_1-a_1|^{p/2}} \quad \text{and} \quad \frac{\left|\hat{d}(a_2,b_2)\right|}{|b_2-a_2|^{p/2}}$$

have approximately the same distribution. It follows that these distributions have approximately the same expectancy, γ_0, whence it follows that, for this universal γ_0,

$$E\left[\left|\hat{d}(a,b)\right|\right] \approx \gamma_0(b-a)^{p/2}$$

for all a and b.

Note that, if $a=b$, then $d(a,b)=\hat{d}(a,b)=|b-a|^{p/2}=0$, identically. Thus, in approximating $\sum_i|b_i-a_i|^p$ by $\sum_i\hat{d}(a_i,b_i)$, we may restrict attention only to i such that $a_i\neq b_i$.

Definition 8 The **randomized rounding** $[x]_\rho$ of a real number x by a (random) real $\rho\in[0,1]$ is defined by

$$[x]_\rho = \begin{cases} \lceil x\rceil, & x\geq \rho+\lfloor x\rfloor \\ \lfloor x\rfloor, & \text{otherwise} \end{cases}$$

∎

Note that $E([x]_\rho)=x$.

Lemma 9 *For any real numbers $a\leq b$, the distribution on $[b-a]_\rho$ is the same as the distribution on $[b]_{\rho'}-[a]_{\rho'}$.*

Note that it is *not* the case that, for all a,b,ρ, $[b-a]_\rho=[b]_\rho-[a]_\rho$.

Proof. (Tedious, straightforward, omitted.) Note that the expected values of $[b-a]_\rho$ and $[b]_{\rho'}-[a]_{\rho'}$ are the same. One can show that these random variables take on the same two values, $\lfloor b-a\rfloor$ and $\lceil b-a\rceil$. The result follows. □

We now clarify the definition of $\hat{d}$ and indicate how to do rounding.

Definition 10 Let $\vec{r}=(r,r',\rho)$. Define $\hat{d}(\vec{r},a,b)$ by $r^{-p/2}d([ra]_\rho+r',[rb]_\rho+r')$. ∎

Lemma 11 *For all $a<b$ and for all $i\in[M(u+v)^{N_1},(u+v)^{N_2}]$, the probability $\Pr_{r,\rho}([rb]_\rho-[ra]_\rho=i)$ does not depend on a or b.*

Proof. By Lemma 9, for each fixed r, $\Pr_\rho([rb]_\rho-[ra]_\rho=i)=\Pr_\rho([r(b-a)]_\rho=i)$. It follows that

$$\Pr_{r,\rho}([rb]_\rho-[ra]_\rho=i)=\Pr_{r,\rho}([r(b-a)]_\rho=i).$$

Next, for each fixed ρ,

$$\begin{aligned}\Pr_r([r(b-a)]_\rho = i) &= \Pr_r(i-1+\rho \le r(b-a) < i+\rho) \\ &= \Pr_r\left(\log\left(\frac{i-1+\rho}{b-a}\right) \le \log(r) < \log\left(\frac{i+\rho}{b-a}\right)\right) \\ &= \log\left(\frac{i+\rho}{i-1+\rho}\right).\end{aligned} \tag{22}$$

Thus

$$\Pr_{r,\rho}([rb]_\rho - [ra]_\rho = i) = \Pr_{r,\rho}([r(b-a)]_\rho = i) = \int_0^1 \log\left(\frac{i+\rho}{i-1+\rho}\right) d\rho,$$

and, in any case, this probability does not depend on a or b. □

Note that if $i \in [M(u+v)^{N_1}, (u+v)^{N_2}]$, then $\Pr([rb]_\rho - [ra]_\rho = i) > 0$.

Lemma 12 *For all distinct a and b, the probability* $\Pr\left([rb]_\rho - [ra]_\rho \notin [M(u+v)^{N_1}, (u+v)^{N_2}]\right)$ *is at most*

$$2\log(M)/(N_2 - N_1).$$

Proof. Immediate from the definition of r and the fact that $0 \le b-a \le M$. □

Put $N_2 - N_1 = 8c_2 \log(M)/(c_1\eta\varepsilon)$. Then $\Pr([rb]_\rho - [ra]_\rho \notin [M(u+v)^{N_1}, (u+v)^{N_2}]) \le c_1\eta\varepsilon/(4c_2)$.

Lemma 13 *For $i \in [M(u+v)^{N_1}, (u+v)^{N_2}]$, given that $[rb]_\rho - [ra]_\rho = i$, the conditional expectation of*

$$\frac{|\hat{d}(\vec{r},a,b)|}{|b-a|^{p/2}} = \frac{\left|\phi(\vec{r})d([ra]_\rho + r', [rb]_\rho + r')\right|}{|b-a|^{p/2}}$$

is independent of a and b.

Proof. Follows from (*periodicity*), using the fact that $u \ge 2$. □

We return to the proof of (*average*). Fix a_2, b_2, a_1, b_1, such that $E\left[\hat{d}(\vec{r},a,b)/|b-a|^{p/2}\right]$ is minimized at $(a,b) = (a_1,b_1)$ and maximized at (a_2,b_2). Write

$$E\left[\hat{d}(\vec{r},a,b)/|b-a|^{p/2}\right] = \sum_i E\left[\hat{d}(\vec{r},a,b)/|b-a|^{p/2} \middle| [rb]_\rho - [ra]_\rho = i\right] \cdot \Pr([rb]_\rho - [ra]_\rho = i).$$

Using Lemmas 11, 12, and 13, we have

$$
\begin{aligned}
&E\left[\hat{d}(\vec{r},a_2,b_2)/|b_2-a_2|^{p/2}\right]-E\left[\hat{d}(\vec{r},a_1,b_1)/|b_1-a_1|^{p/2}\right]\\
&=\sum_i \Pr\left([rb_2]_\rho-[ra_2]_\rho=i\right)E\left[\hat{d}(\vec{r},a_2,b_2)\big|[rb_2]_\rho-[ra_2]_\rho=i\right]/|a_2-b_2|^{p/2}\\
&\quad-\sum_i \Pr\left([rb_1]_\rho-[ra_1]_\rho=i\right)E\left[\hat{d}(\vec{r},a_1,b_1)\big|[rb_1]_\rho-[ra_1]_\rho=i\right]/|a_1-b_1|^{p/2}\\
&\le \sum_{i\in[M(u+v)^{N_1},(u+v)^{N_2}]}\Pr\left([rb_2]_\rho-[ra_2]_\rho=i\right)\left(\frac{E\left[\hat{d}(\vec{r},a_2,b_2)\big|[rb_2]_\rho-[ra_2]_\rho=i\right]}{|a_2-b_2|^{p/2}}\right.\\
&\qquad\qquad\left.-\frac{E\left[\hat{d}(\vec{r},a_1,b_1)\big|[rb_1]_\rho-[ra_1]_\rho=i\right]}{|a_1-b_1|^{p/2}}\right) \qquad (23)\\
&\quad+\sum_{i\notin[M(u+v)^{N_1},(u+v)^{N_2}]}\Big(\Pr\left([rb_2]_\rho-[ra_2]_\rho=i\right)+\Pr\left([rb_1]_\rho-[ra_1]_\rho=i\right)\Big)\\
&\qquad\cdot\left(\frac{E\left[\hat{d}(\vec{r},a_2,b_2)\big|[rb_2]_\rho-[ra_2]_\rho=i\right]}{|a_2-b_2|^{p/2}}+\frac{E\left[\hat{d}(\vec{r},a_1,b_1)\big|[rb_1]_\rho-[ra_1]_\rho=i\right]}{|a_1-b_1|^{p/2}}\right)\\
&\le 2\max_{\vec{r}}\hat{d}(\vec{r},a_2,b_2)/|a_2-b_2|^{p/2}\cdot\sum_{i\notin[M(u+v)^{N_1},(u+v)^{N_2}]}\left(\Pr\left([rb_2]_\rho-[ra_2]_\rho=i\right)+\Pr\left([rb_1]_\rho-[ra_1]_\rho=i\right)\right)\\
&\le 2c_2\cdot(c_1\eta\varepsilon/(4c_2))\\
&\le \varepsilon E\left[\hat{d}(\vec{r},a_1,b_1)/|b_1-a_1|^{p/2}\right].
\end{aligned}
$$

It follows that

$$E\left[\hat{d}(\vec{r},a_2,b_2)/|b_2-a_2|^{p/2}\right]=(1\pm\varepsilon)E\left[\hat{d}(\vec{r},a_1,b_1)/|b_1-a_1|^{p/2}\right], \qquad (24)$$

so that for γ_0 equal to the (approximate) common value (24), for all a,b,

$$E\left[\hat{d}(\vec{r},a,b)\right]=\gamma_0|b-a|^{p/2}(1\pm\varepsilon). \qquad (25)$$

∎

3.2.5 Precision

Last, we look at the precision needed to implement our solution. Above, r and ρ were specified as real numbers, but, in practice, they are limited-precision floating point numbers. Let $\hat{r}$ and $\hat{\rho}$ denote the floating point equivalents of r and ρ.

Note that the proofs of (*upper bound*) and Lemma 6 hold for any r and any ρ. The proof of Lemma 7 works provided $[\hat{r}a]_{\hat{\rho}} = ra \pm 1$ and $[\hat{r}b]_{\hat{\rho}} = rb \pm 1$; this is the case if $\hat{r} - r < O(1/M)$, *i.e.*, if $\hat{r}$ has $O(\log(M))$ bits of precision to the right of the radix point.

The proof of (*average*) requires that $\Pr\left([rb_1]_\rho - [ra_1]_\rho = i\right)$ be close to $\Pr\left([\hat{r}b_1]_{\hat{\rho}} - [\hat{r}a_1]_{\hat{\rho}} = i\right)$, in several places—in Lemma 11, in (23) directly, and in (23) buried in the conditional expectation (in the numerator and denominator). It is tedious but straightforward to check that the required number of bits of precision to the right of the radix point is within the overall space bound claimed. To get an approximation of $\Pr\left([\hat{r}b_1]_{\hat{\rho}} - [\hat{r}a_1]_{\hat{\rho}} = i\right)$ to within some error ζ, one needs to maintain ρ and $r(b-a)$ to within ζ, *i.e.*, maintain r to within ζ/M. This requires $\log(M) + \log(1/\zeta)$ bits of precision to the right of the radix point. The total error in the sum of n terms may be n times as much, so we need to maintain ρ to within $\Omega(\delta/n)$ and r to within $\Omega(\delta/(Mn))$. This requires $O(\log(M) + \log(n) + \log(1/\delta))$ bits of precision to the right of the radix point. Analysis of the expectancy in (23) is slightly more involved, but similar.

A completely analogous argument shows that, for $k = 2,4$, $E[\hat{d}^k(a,b)]/|b-a|^{kp/2}$ is the same, up to the factor $(1 \pm \varepsilon)$, for all values of a and b.

3.3 Top Level Algorithm, Correctness

We now revisit the overall algorithm in detail, thereby proving Theorem 2.

Proof. (of Theorem 2)

Put
$$f(x) = \gamma_1^{-1/2}\hat{d}(0,x). \tag{26}$$

Then
$$E\left[(f(b) - f(a))^2\right] = (b-a)^p(1 \pm \varepsilon), \tag{27}$$

and, similarly,
$$E\left[(f(b) - f(a))^4\right] = \frac{\gamma_2}{\gamma_1^2}(b-a)^{2p}(1 \pm \varepsilon). \tag{28}$$

Thus
$$\mathrm{var}\left((f(b) - f(a))^2\right) \le O(E^2\left[(f(a) - f(b))^2\right]) \tag{29}$$

and, similarly,
$$\mathrm{var}\left((f(a) - f(b))^4\right) \le O(E^2\left[(f(a) - f(b))^4\right]). \tag{30}$$

Choose families of 4-wise independent random variables $\{r_i\}, \{r'_i\}$, and $\{\rho_i\}$ such that each r, r' is distributed as in Definition 3 and ρ_i is uniformly distributed in $[0,1]$. Also choose a 4-wise independent family $\{\sigma_i\}$ of ± 1-valued random variables. The 4-wise independence is needed below to bound Z^4, *i.e.*, to bound $\mathrm{var}(Z^2)$.

Each parallel repetition of the algorithm computes $Z = \sum_i \sigma_i(f(a_i) - f(b_i))$. Then

$$\begin{aligned} E[Z^2] &= \sum_i \sigma_i^2(f(a_i) - f(b_i))^2 + \sum_{i \ne i'} \sigma_i\sigma_{i'}(f(a_i) - f(b_i))(f(a_{i'}) - f(b_{i'})) \\ &= \sum_i |b_i - a_i|^p(1 \pm \varepsilon). \end{aligned} \tag{31}$$

Here we used the fact that $\sigma_i^2 \equiv 1$ and $E[\sigma_i\sigma_{i'}] = 0$ for $i \ne i'$.

Similarly,

$$\begin{aligned} E[Z^4] &= \sum_i \sigma_i^4 (f(a_i) - f(b_i))^4 + 6 \sum_{i \neq i'} \sigma_i^2 \sigma_{i'}^2 (f(a_i) - f(b_i))^2 (f(a_{i'}) - f(b_{i'}))^2 \\ &= \left(A \sum_i |b_i - a_i|^{2p} + B \Big(\sum_i |b_i - a_i|^p \Big)^2 \right) (1 \pm \varepsilon), \end{aligned} \tag{32}$$

where A and B are nonzero constants. Thus $\mathrm{var}(Z^2) \leq O(E^2[Z^2])$, and we can apply Lemma 1. We conclude that

$$\Pr\left(\left|X - \sum |a_i - b_i|^p\right| > \varepsilon \sum |a_i - b_i|^p\right) \leq \delta,$$

where X is a median of means of independent copies of Z^2.

3.4 Algorithm in the Sketch Model

We can apply this algorithm to the sketching model, also. Perform $O(\log(1/d)/\varepsilon^2)$ parallel repetitions of the following. Given one function $\langle a_i \rangle$, construct the small sketch $\sum_i \sigma_i f(a_i)$. Later, given two sketches $A = \sum_i \sigma_i f(a_i)$ and $B = \sum_i \sigma_i f(b_i)$, one can reconstruct $\sum |a_i - b_i|^p$ by outputting a median of means of independent copies of $|A - B|^2$. ∎

3.5 Cost

We first consider the space. The algorithm needs to store seeds for families of random variables and values of the Z's. Each family is of size n, so requires space $O(\log n)$ times the number of bits in a single random variable [AS92]. The variables r and r' each require $O(\log(M)/\varepsilon)$ bits to the left of the radix point. We need $O(\log(1/d)/\varepsilon^2)$ parallel repetitions of the random variables, so the total space for random variables is $(\log(M)\log(n)\log(1/d)/\varepsilon)^{O(1)}$. The counters Z are bounded by $M^{O(1/\varepsilon)}$, so they also require comparable space.

In [FKSV99], the authors show that, for $M = 2$, any randomized algorithm that approximates $\sum |a_i - b_i|$ to within the factor $(1 \pm \varepsilon)$ uses more than $\omega(\log^{\alpha}(n)/\varepsilon^{1-\beta})$ space, for any (large) α and any (small) $\beta > 0$. A similar argument applies here, too. Thus the space used by our algorithm is within a constant power of optimal.

We now consider the time to process an item in the stream. To process (i, c, θ), the algorithm first produces values for the random variables, which is quick. The algorithm then needs to compute $\sum_{j=0}^{C} \pi_j$, where $C \leq cM^{O(1/\varepsilon)}$. From the recursive structure of π, it is clear that this computation requires time at most $\log^{O(1)}(cM)^{1/\varepsilon}$, *i.e.*, polynomial in the size of (i, c, θ).

3.6 Top Level Algorithm, $2 \leq p \leq 4$

We can also approximate $\sum_i |a_i - b_i|^p$ for $p \in [2,4]$, though with worse error. The algorithm should use 8-wise independent random variables instead of 4-wise independent random variables. Given $p \in [2,4]$, use (32) to approximate $A \sum_i |b_i - a_i|^p + B\left(\sum_i |b_i - a_i|^{p/2}\right)^2$. This approximation will have small relative error, *i.e.*, error small compared with the larger term, $\left(\sum_i |b_i - a_i|^{p/2}\right)^2$. Next, approximate $\sum_i |b_i - a_i|^{p/2}$ and solve for $\sum_i |a_i - b_i|^p$, getting error that is small compared with $\left(\sum_i |b_i - a_i|^{p/2}\right)^2$. Thus we cannot, by this method, approximate $\sum_i |a_i - b_i|^p$ with small relative error, for $p > 2$.

We now analyze this error guarantee compared with error guarantees of related work. In previous work, there are three types of error guarantees given. In typical sampling arguments, using resources $(\log(n)\log(M)/\varepsilon)^{O(1)}$, an error is produced which is bounded by εn or even εMn. The techniques of [BCFM98] can be used to approximate $\sum|a_i - b_i|$ when $a_i, b_i \in \{0,1\}$. In this case, the error is bounded by $\varepsilon\sum|a_i + b_i|$—already a substantial improvement over error εn and the best possible in the original context of [BCFM98]. Relative error, *i.e.*, error smaller than ε times the returned value, is better still, and is achievable for $p \in (0,2]$. Our error guarantee for $p > 2$ falls between relative error and $O(\sum|a_i + b_i|)$. In particular, our error in approximating $\sum|a_i - b_i|^p$ is small compared with $(\sum|a_i - b_i|^p)^2$, so our error gets small as the returned value $\sum|a_i - b_i|^p$ gets small—this is not true for an error bound of, say, $\varepsilon\sum|a_i + b_i|^p$ or $\varepsilon\sum|a_i + b_i|$.

Since

$$\sum|a_i - b_i|^{p/2} \le \sum|a_i - b_i|^p \le \left(\sum|a_i - b_i|^{p/2}\right)^2, \tag{33}$$

one could also approximate $\sum|a_i - b_i|^p$ by $\left(\sum|a_i - b_i|^{p/2}\right)^{3/2}$. This will be correct to within the factor $\left(\sum|a_i - b_i|^{p/2}\right)^{1/2}$. Note that our algorithm is an unbiased estimator, *i.e.*, it has the correct mean—an advantage in some contexts; this is not true of $\left(\sum|a_i - b_i|^{p/2}\right)^{3/2}$. Furthermore, our algorithm provides a smooth trade-off between guaranteed error and cost, which is not directly possible with the trivial solution. We hope that, in some applications, our approximation to $\sum|a_i - b_i|^p$ provides information not contained in $\sum|a_i - b_i|^{p/2}$.

We conjecture that one can approximate $\sum|a_i - b_i|^p$ for any p this way. To do this, one needs to show that $E[Z^{2\lfloor p/2\rfloor}]$ is a polynomial in $\sum|a_i - b_i|^p, \sum|a_i - b_i|^{p/2}, \ldots$, in which $\sum|a_i - b_i|^p$ appears with nonzero coefficient. The algorithm may be quite expensive in p.

4 Discussion

4.1 Relationship with Previous Work

We give an approximation algorithm for, among other cases, $p = 1$ and $p = 2$. The $p = 1$ case was first solved in [FKSV99], using different techniques. Our algorithm is less efficient in time and space, though by no more than a power. The case $p = 2$ was first solved in [AMS96, AGMS99], and it is easily seen that our algorithm for the case $p = 2$ coincides with the algorithm of [AMS96, AGMS99].

Our algorithm is similar to [AMS96, FKSV99] at the top level, using the strategy proposed by [AMS96].

4.2 Random-Self-Reducibility

Our proof technique can be regarded as exploitation of a random-self-reduction [F93] of the L^p difference. Roughly, a function $f(x)$ is random-self-reducible via (σ, ϕ) if, for random r, $f(x) = \phi(r, f(\sigma(r,x)))$, where the distribution $\sigma(\cdot, x)$ does not depend on x. That is, to compute $f(x)$, one can transform x to a random $y = \sigma(r,x)$, compute $f(y)$, then transform $f(y)$ back to $f(x) = \phi(r, f(y))$. If one can show that a function f is random-self-reducible then one can show

- The function f is hard on average (with respect to σ) if it is hard in worse case. Average-case hard functions are useful in cryptography.

- A program P that is guaranteed to compute f correctly only on $3/4$ of the inputs to x can be made to compute f correctly, with high probability, on all inputs. That is, P has a self-corrector. (On input x, repeatedly run P on $\sigma(r,x)$ (using independent r's), then output the majority of $(\phi(r,P(\sigma(r,x))))$.)

- If Alice has x and Bob can compute $f()$, then Alice can get Bob to compute $f(x)$ without Bob learning x. (Alice gives Bob $\sigma(r,x)$, which is uncorrelated with x. She receives $f(\sigma(r,x))$ and then computes $\phi(r,f(\sigma(r,x)))$.)

In [FS97], the authors generalized existing notions of random-self-reducibility to include transformations σ that are slightly correlated with x, but the correlation is tightly bounded. They called such random-self-reductions "leaky."

Our construction involves producing a form of self-corrector for programs for a function, L^p, that have a sort of leaky random-self-reduction. The function L^p of two streams is random-self-reducible in the sense that

$$|a_i - b_i|^p = \frac{1}{r^p}\left|(ra_i + r') - (rb_i + r')\right|^p,$$

where the distribution $(ra_i + r', rb_i + r')$ is only weakly dependent on (a_i, b_i). We present a deterministic function, $d(a,b)$, that produces a result guaranteed to be correct only up to a large constant factor and only on a set S that is small but has non-negligible probability η; from that, we produce the function, $r^{-p/2}d(T_r(a),T_r(b))$, that produces a correct result with high probability, on all inputs.

It is easiest to present our current work without invoking random-self-reducibility machinery. We hope to investigate further the random-self-reducibility issues for massive data streams, for sketches, and for real-valued functions. A theory of random-self-reducibility for streams may make it easier to produce streaming algorithms, to give performance guarantees for heuristics thought to work in many cases, and to characterize functions that have or do not have efficient streaming algorithms.

4.3 Determination of Constants

Our function f involves some constant c, such that $E[(f(a)-f(b))^2] \approx c|b-a|^p$, but we do not explicitly provide the constant c. This needs to be investigated further. We give a few comments here.

One can approximate c using a randomized experiment. Due to our fairly tight upper and lower bounds for c, we can, using Lemma 1, estimate c reliably as $\left|\hat{d}(a,b)\right| \cdot |b-a|^{-p/2}$. At worst, this occurs once for each p, M, n, and ε; it is not necessary to do this once for each item or even once for each stream. Furthermore, one can fix generously large M and n and generously small ε to avoid repeating the estimation of c for changes in these values.

In some practical cases, not knowing c may not be a drawback. In practice, as in [BCFM98], one may use the measure $\sum|a_i - b_i|^p$ to quantify the difference between two web pages, where a_i is the number of occurrences of feature i in page A and b_i is the number of occurrences of feature i in page B. For example, one may want to keep a list of non-duplicate web pages, where two web pages that are close enough may be deemed to be duplicates. According to this model, there are sociological empirical constants $\hat{c}$ and $\hat{p}$ such that web pages with $\hat{c}\sum|a_i - b_i|^{\hat{p}} < 1$ are considered to be duplicates. To apply this model, one must estimate the parameters $\hat{c}$ and $\hat{p}$ by doing sociological experiments, *e.g.*, by asking human subjects whether they think pairs of webpages, with varying measures of $\hat{c}\sum|a_i - b_i|^{\hat{p}}$ for various values of $\hat{c}$ and $\hat{p}$, are or are not duplicates. If one does not know c, one can simply estimate $\hat{c}/c$ at once by a single sociological experiment.

4.4 Non-grouped Input Representation

Often, in practice, one wants to compare $\langle a_i \rangle$ and $\langle b_i \rangle$ when the values a_i and b_i are represented differently. For example, suppose there are two grocery stores, A and B, that sell the same type of items. Each time either store sells an item it sends a record of this to headquarters in an ongoing stream. Suppose item i sells a_i times in store A and b_i times in store B. Then headquarters is presented with two streams, A and B, such that i appears a_i times in A and b_i times in B; $\sum |a_i - b_i|^p$ measures the extent to which sales differ in the two stores. Unfortunately, we don't see how to apply our algorithm in this situation. Apparently, in order to use our algorithm, each store would have to aggregate sales data and present a_i or b_i, rather than present a_i or b_i non-grouped occurrences of i. The algorithm of [AMS96, AGMS99] solves the $p = 2$ case in the non-grouped case, but the problem for other p is important and remains open.

We have recently learned of a solution the non-grouped problem [I00]. Note that, in general, a solution A in the non-grouped representation yields a solution in the function-value representation, since, on input a_i, an algorithm can simulate A on a_i occurrences of i; this simulation takes time exponential in the size of a_i to process a_i. The proposed solution, however, appears to be of efficiency comparable to ours in the function-value representation, at least in theory, but there may be implementation-related reasons to prefer our algorithm in the grouped case.

References

[AGMS99] N. Alon, P. Gibbons, Y. Matias, and M. Szegedy. Tracking Join and Self-Join Sizes in Limited Storage. In *Proc. of the 18'th Symp. on Principles of Database Systems*, ACM Press, New York, pages 10–20, 1999.

[AMS96] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proc. of 28'th STOC*, pages 20–29, 1996. To appear in Journal of Computing and System Sciences.

[AS92] N. Alon and J. Spencer. *The Probabilistic Method*. Wiley, 1992.

[BCFM98] A. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *Proc. of the 30'th STOC*, pages 327–336, 1998.

[CN98] Cisco NetFlow, 1998. `http://www.cisco.com/warp/public/732/netflow/`.

[F93] J. Feigenbaum. Locally random reductions in interactive complexity theory. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 13, pages 73–98. American Mathematical Society, Providence, 1993.

[FKSV99] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. An Approximate L^1-Difference Algorithm for Massive Data Streams. In *Proc. of the 40'th IEEE Symposium on Foundataions of Computer Science*, IEEE Computer Society, Los Alamitos, CA, pages 501-511, 1999.

[FS97] J. Feigenbaum and M. Strauss. An Information-Theoretic Treatment of Random-Self-Reducibility. *Proc. of the 14'th Symposium on Theoretical Aspects of Computer Science*, pages 523–534. Lecture Notes in Computer Science, vol. 1200, Springer-Verlag, New York, 1997.

[GM98] P. Gibbons and Y. Matias. Synopsis Data Structures for Massive Data Sets. To appear in *Proc. 1998 DIMACS Workshop on External Memory Algorithms.* DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence. Abstract in *Proc. Tenth Symposium on Discrete Algorithms*, ACM Press, New York and Society for Industrial and Applied Mathematics, Philadelphia, pages S909–910, 1999.

[HRR98] M. Rauch Henzinger, P. Raghavan, and S. Rajagopalan. Computing on data streams. Technical Report 1998-011, Digital Equipment Corporation Systems Research Center, May 1998.

[I00] P. Indyk. Stable Distributions, Pseudorandom Generators, Embeddings and Data Stream Computation. *Proc. of the 41'st IEEE Symposium on Foundations of Computer Science*, IEEE Computer Society, Los Alamitos, CA, to appear.

[KOR98] E. Kushilevitz, R. Ostrovsky, Y. Rabani. Efficient Search for Approximate Nearest Neighbor in High Dimensional Spaces. *Proc. of The 30'th ACM Symposium on Theory of Computing*, ACM Press, New York, pages 514-523.