Fast, Small-Space Algorithms for Approximate Histogram Maintenance

[Extended Abstract]

Anna C. Gilbert
AT&T Labs—Research
agilbert@research.att.com

Sudipto Guha
CIS,University of Pennsylvania
sudipto@cis.upenn.edu

Piotr Indyk
Lab. Computer Science, MIT
indyk@theory.lcs.mit.edu

Yannis Kotidis
AT&T Labs—Research
kotidis@research.att.com

S. Muthukrishnan
AT&T Labs—Research
muthu@research.att.com

Martin J. Strauss
AT&T Labs—Research
mstrauss@research.att.com

ABSTRACT

A vector $\mathbf{A}$ of length N is defined implicitly, via a stream of updates of the form "add 5 to $\mathbf{A}_3$." We give a *sketching* algorithm, that constructs a small *sketch* from the stream of updates, and a *reconstruction* algorithm, that produces a B-bucket piecewise-constant representation (histogram) $\mathbf{H}$ for $\mathbf{A}$ from the sketch, such that $\|\mathbf{A}-\mathbf{H}\| \le (1+\epsilon)\|\mathbf{A}-\mathbf{H}_{\mathrm{opt}}\|$, where the error $\|\mathbf{A}-\mathbf{H}\|$ is either ℓ_1 (absolute) or ℓ_2 (root-mean-square) error. The time to process a single update, time to reconstruct the histogram, and size of the sketch are each bounded by $\mathrm{poly}(B, \log(N), \log\|\mathbf{A}\|, 1/\epsilon)$. Our result is obtained in two steps. First we obtain what we call a *robust* histogram approximation for $\mathbf{A}$, a histogram such that adding a small number of buckets does not help improve the representation quality significantly. From the robust histogram, we cull a histogram of desired accruacy and B buckets in the second step. This technique also provides similar results for Haar wavelet representations, under ℓ_2 error. Our results have applications in summarizing data distributions fast and succinctly even in distributed settings.

1. INTRODUCTION

Histograms are succinct and space-efficient approximations of distributions of numerical values. One often visualizes histograms as a sequence of vertical bars whose widths are equal but whose heights vary from bar to bar. More generally, histograms are of varying width as well; that is, they are general piecewise-constant approximations of data distributions. Formally, suppose $\mathbf{A}$ is a function (or a "distribution" or a "signal") on N points given by $\mathbf{A}[0\cdots N)$. A *B-bucket histogram* $\mathbf{H}$ of $\mathbf{A}$ is defined by a partition of the domain $[0\cdots N)$ into B intervals (buckets) B_i, as well as by B *spline parameters* b_i. For any $x \in [0\cdots N)$, the value of $\mathbf{H}(x)$ is equal to the b_i such that $x \in B_i$. Since B is typically (much) smaller than N, this is a lossy representation. The quantity $\|\mathbf{A}-\mathbf{H}\|_p$, where $\|\cdot\|_p$ is the l_p norm, is the error in approximating $\mathbf{A}$ by a B-bucket histogram $\mathbf{H}$. Typically the norms of interest are l_1 (average absolute error) or l_2 (root mean square error).

The branch of mathematics called Approximation Theory deals with finding representations for functions. Histograms are amongst the simplest class of representations, and perhaps the most fundamental. They are the easiest to visualize; statistical analyses frequently involve histograms. Histograms also find many applications in computer systems. For example, most commercial database engines keep a histogram of the various value distributions in a database for optimizing query executions and for approximately processing queries; image processing systems handle color histograms extensively, etc.

Finally, an emerging context we describe in some detail is one of distributed databases on large scale networks such as the Internet. Routers generate a *data stream* of logs of the traffic that goes over the various incident links. For real time traffic control, operators must know traffic patterns at various routers at any moment. However, it is prohibitively bandwidth-expensive to transfer data streams of traffic logs from routers to central monitoring stations on a continuous basis. Hence, histograms may be constructed at routers to summarize the traffic distribution; they will be compact, and, while not being precise, they may suffice for most trend-related analyses. Building histograms at network routers thus saves the distribution cost of transmitting raw data.

Our focus here is on maintaining a histogram representation for *dynamic* data distributions. Our study is mainly motivated by applications of histograms where data changes rapidly.

- Many commercial databases have large number of transactions, *e.g.*, stock transactions etc. during a work day. Transactions change the underlying data distribution (*e.g.*, volume of stock sold per ticker symbol, every minute). The outstanding challenge in using histograms in such transactional databases is to maintain them during these rapid changes.

- In the context of data streams comprising traffic logs, data distributions change even faster: each IP packet

STOC'02, May 19-21, 2002, Montreal, Quebec, Canada.

that passes through a router changes the data distribution (*e.g.*, the number of bytes from each IP address passing through that router) rapidly. In order to capture these traffic distributions using histograms, we inherently need mechanisms to build histograms as the data distribution evolves rapidly.

In all the above contexts, the basic problem is to find a "good" histogram for a given data distribution. For applications involving the visualization of the histogram, finding a good histogram is an art. But in a formal setting, one seeks *optimal* histograms, that is, ones that minimize $\|\mathbf{A}-\mathbf{H}\|_1$ or $\|\mathbf{A}-\mathbf{H}\|_2$. An optimal (static) histogram can be found in a straightforward way using dynamic programming, taking time $O(N^2B)$ time using $O(NB)$ space.

Formally, the problem of dynamic histogram maintenance is as follows. We are given parameter B and are to maintain a function $\mathbf{A}_0=\mathbf{A}[0\cdots N)$. (1) Suppose the j'th operation is *update*(i,x); then the operation is to add x to $\mathbf{A}_{j-1}[i]$, where x is an arbitrary integer parameter (in particular, x may be negative). In the resulting $\mathbf{A}_j$, $\mathbf{A}_j[i]=\mathbf{A}_{j-1}[i]+x$ and $\mathbf{A}_j$ is identical to $\mathbf{A}_{j-1}$ otherwise. (2) Suppose the j'th operation is *Hist*. Then the goal is to output a B-bucket histogram $\mathbf{H}$ for $\mathbf{A}_{j-1}$ such that $\|\mathbf{A}_{j-1}-\mathbf{H}\|_1$ (or $\|\mathbf{A}_{j-1}-\mathbf{H}\|_2$) is (nearly) minimized. As is standard, we must design a data structure that supports these operations with little time (here, polylogarithmic in N). Additionally, we must use working space that is *sublinear* N (again, polylogarithmic in N). This is a departure from standard dynamic data structures literature, and it is motivated by applications we cited above. For example, memory is at premium in routers and hence only a few hundred bytes may be allocated for representing data distributions even though n may be 2^{32} or higher; likewise, transactional databases allocate only a few hundred bytes for maintaining histograms on value distributions even of product codes where n may be 2^{32} or larger. Hence, B is likely to be very small; for the same reasons, the work space for maintaining histograms must also be very small with respect to N.

1.1 Our Results

Our main result is the first known algorithm that simultaneously breaks both the linear time and linear space bottlenecks for the problem of dynamic maintenance of histograms. We present

- An algorithm supporting the Update and Hist operations in space and time $\mathrm{poly}(B,1/\epsilon,\log\|\mathbf{A}\|,\log N)$. The reconstruction of the best B bucket histogram for Hist query produces one with error at most $(1+\epsilon)$ times the optimal error in ℓ_1 or ℓ_2 norm[1].
- The result is nearly best possible, since any algorithm for this problem uses $\Omega(B\log(N))$ space, and, further, any algorithm whose dependence on N is polylogarithmic uses $1/(\epsilon\log^{O(1)}(N))$ space.

The algorithm supports not only updates of individual values of $\mathbf{A}[i]$, but in fact enables *any* linear operation on one or two signals/distributions. This feature allows us to address the issues of distributed data collection, that arise in the context of large scale networks. In particular, we can combine the distribution information from many different sources (*e.g.*, routers) and compute the histogram approximation for the cummulative data. Our techniques also give similar results for Haar wavelet representations, but this holds under ℓ_2 error only.

1.2 Previous Work

The problem of histogram construction and maintenance is fundamental to databases and therefore has been studied extensively in the past decade. Different optimization versions of histograms have been formulated: see [16] for an overview. Our problem here is known as the V-optimal histogram in the database literature, and is the one sought after. Heuristic algorithms for building and maintaining histograms have been proposed using sampling [5], wavelets [13], discrete cosine transforms [12], local adjustments [1], etc. None of these approaches gives any provable bounds.

The static version of our histogram problem allows no updates. For this problem, an $O(N^2B)$ time dynamic programming algorithm appears in [11]. In addition, [11] also presented a $O((N+B\log N)\log\|\mathbf{A}\|)$-time algorithm using at most $3B$ buckets and having a guaranteed error at most 3 times the optimal histogram with B buckets. This was improved to $1+\epsilon$ approximation using B buckets in [8] in time $O(N+(B\epsilon^{-1}\log N)^3)$.

In the data stream model, let us first consider the *aggregated, sorted* model; here, (static) $\mathbf{A}[i]$'s are scanned in increasing order of i's. The result in [11] uses $O(B+\log\|\mathbf{A}\|_2)$ space and provides same $(3,3)$ guarantee as above. The algorithm of [9] provided a $(1+\epsilon)$-approximation preserving the number of buckets, using $O(B^2\log N)$ space and taking time $O(NB^2\log N)$. In [6], the authors provided a $O(B+\log N)$ space, exact algorithm for finding a B-term wavelet representation; this translates into a histogram using at most $B\log N$ buckets, preserving the error.

The most general data stream model (which we use in this paper) is known as the *cash register* model [6]. Few algorithms are known for computing on the cash register model, examples include estimating stream norms [2, 3, 10]. Computing histograms or other representations is significantly more involved than merely estimating the norm because the identification of the relevant coefficients is very crucially needed in our algorithm. Besides [6, 7, 17] that we discuss next, no other nontrivial results are known.

One of the most closely-related works is [6], which gives an algorithm for our dynamic problem (in the wavelet formulation), using $\mathrm{poly}(B,1/\epsilon,\log N)$ *space*. Our present work improves [6] in construction time, error bound, and generality of the technique. The algorithm presented here improves a construction time of $N\mathrm{poly}(B,\log(N),\log\|\mathbf{A}\|,1/\epsilon)$ to $\mathrm{poly}(B,\log(N),\log\|\mathbf{A}\|,1/\epsilon)$, improves an additive error from $\epsilon\|\mathbf{A}\|$ to $\epsilon|OPT|$ (*i.e.*, provides, for the first time, relative error with the factor $(1+\epsilon)$), and can handle the non-orthogonal norm ℓ_1 as well as ℓ_2.

Finally, papers [7, 17] present experimental evaluations and extensions of histogram construction algorithms. Paper [7] presents experimental evaluation of heuristics for constructing wavelet representations motivated by the results in this paper. Similarly, paper [17] presents experimental results for heuristics derived from the "bare bones" version of

[1]Our algorithms succeed on *each Hist* operation with user-specified probability δ over the algorithm's random choices, at a cost of the factor $\log(1/\delta)$ in the time and space. To make the algorithm succeed with probability $1-\delta'$ on *all* of t *Hist* operations, use $\delta=\delta'/t$ for cost factor $\log(t/\delta')$.

our algorithm (the resulting algorithms have running times polynomial in N). Both papers represent our ongoing work to explore the application of the ideas in this paper in a practical context.

1.3 Technical Overview

A *dyadic interval* is of the form $[i2^j, \ldots, (i+1)2^j)$, for integers i and j. A technical primitive we use in our algorithms is a *synopsis data structure* for an array that supports updates, identification of dyadic intervals with large projections, estimation of the best spline parameters, and estimation of the norm. Typically a synopsis data structure [4] is defined to be of small space, but we will additionally require them to support all necessary computations in small time as well. By "small", we mean a value at most $\mathrm{poly}(B, \log N, \frac{1}{\epsilon})$. The technical crux in building this synopsis data structure is that we will be required to sum various ranges of random variables and perform "group testing" on sets of coefficients to identify the large projections. This is described in Section 2.

Using this synopsis data structure, our algorithm proceeds by repeatedly adding to our (partial) histogram the dyadic interval which reduces the error of approximation the most. We run this process till we achieve a stable respresentation of the signal with $\mathrm{poly}(B, \log N, \frac{1}{\epsilon})$ buckets. This is what we term as a *robust* approximation of the signal which abstracts the notion that we have extracted the possible information in any B-bucket approximation of the original signal. This procedure in itself (extended to a pair of dyadic intervals) produces a B-term wavelet representation (see Section 3) which minimizes the representation error. To maintain the continuity in presentation, we first present the wavelet result, and subsequently the robust approximation in Section 4.

Finally, in Section 5, we show how to use a robust approximation $\mathbf{H}_r$ to produce a B-bucket approximation $\mathbf{H}$. At a high level, we use a dynamic programming argument introduced in [11] for construction of optimal histograms, modified for desired approximations in [9, 8]. But all of these assume knowledge of exact or approximate value of the error of a histogram when projected on a subinterval. This is not possible to have in a sketch setting since the sketch is constructed for the entire interval. The sketch may suggest the subintervals with large projection (which we use in previous section) but cannot evaluate norms projected to subintervals. For this reason we have to use a technique of creating a set of histograms, that allows us to add intervals left-to-right and circumvents the necessity of knowing projections. The argument is similar to hybridization, where we construct our final output with a series of changes, which add an interval at a time. If the error introduced by any of these intervals were *significantly* more than the error in the optimal solution, restricted to the interval in question, we would contradict the robustness we assumed.

1.4 Notation

$\mathbf{A}$ is a vector (or "signal") of length N. All uppercase bold types represent vectors. For an interval $I \subseteq [0, N)$, we write $\pi(\mathbf{A}, I)$ to denote the projection of the vector on interval I, *i.e.*, equals $\mathbf{A}$ on I and zero elsewhere. The vector χ_I equals 1 on I and 0 elsewhere. The set of vectors χ_I form a (highly redundant) basis for histograms. We use ℓ_1 norm except where noted. All results also hold under ℓ_2 norm.

2. ARRAY SKETCHES

In this section, we give a data structure for a dynamic array $\mathbf{A}$, called an *array sketch*, that supports generalized updates to $\mathbf{A}$ and several fundamental queries about $\mathbf{A}$.

The data structure is parametrized by ϵ_s, η, and N. In this section, "small" means of value at most $\mathrm{poly}(\log(N), 1/\eta, 1/\epsilon_s)$, "compact" means of small size, and "quickly" means using a small amount of time. For an understood signal $\mathbf{A}$, let c^I_{opt} denote the c that minimizes $\|\mathbf{A} - c\chi_I\|$.

Definition 1. A (ϵ_s, η, N)*-array sketch* of a signal $\mathbf{A}$ is a compact synopsis data structure that represents an array of length N and quickly supports the following operations:

- Update. Given a number c and an interval I, we can compute an array sketch for $\mathbf{A} + c\chi_I$.
- Identify. We can return a compact list that contains all dyadic intervals I such that $\left\|\mathbf{A} - c^I_{\mathrm{opt}}\chi_I\right\| \le (1-\eta)\|\mathbf{A}\|$ but contains no interval I such that $\left\|\mathbf{A} - c^I_{\mathrm{opt}}\chi_I\right\| > (1-\eta/2)\|\mathbf{A}\|$.
- Estimate norms. Return $\|\mathbf{A}\|_s$ such that
$$\|\mathbf{A}\| \le \|\mathbf{A}\|_s \le (1+\epsilon_s)\|\mathbf{A}\|.$$
- Estimate parameters. Given an interval I, return a value c such that $\|\mathbf{A} - c\chi_I\| \le (1+\epsilon_s)\left\|\mathbf{A} - c^I_{\mathrm{opt}}\chi_I\right\|$.

These operations will be used, in later sections, to build a near-best histogram representation for $\mathbf{A}$. We will proceed, roughly, as follows. Given a signal $\mathbf{A}$, find I and c such that $\|\mathbf{A} - c\chi_I\|$ is signficantly less than $\|\mathbf{A}\|$, then update the signal by $\mathbf{A} \leftarrow \mathbf{A} - c\chi_I$ and recurse. Thus we will need support for generalized interval updates (which may also be of independent interest), but support for finding a near-best histogram need not be supported directly since it can be built from the more fundamental operations.

An array sketch for signal $\mathbf{A}$ under ℓ_p norm will take the following form. As in [10], choose a random vector $\mathbf{V}$ according to a symmetric p-stable distribution. (The 1-stable distibution family is the Cauchy and the 2-stable distribution family is the Gaussian; we want distributions symmetric about zero.) For some set $S \subseteq [0, N)$ to be specified below, a sub-basic sketch of $\mathbf{A}$ is $\langle \pi(\mathbf{A}, S), \mathbf{V}\rangle$. Keeping $\mathbf{V}$ fixed, choose other sets S (specified below) and repeat the process, to get a basic sketch. Finally, generate several independent basic sketches for independent copies of $\mathbf{V}$, to drive down the distortion and probability of error. An array sketch comprises the several basic sketches.

We first give a technical lemma about p-stable distributions, then specify the sets S and show that the fully-specified data structure satisfies the above.

Definition 2. A random variable X is *Cauchy-distributed with width* w, $X \sim C(0, w)$, if the density of X is $f_w(x) = \frac{w}{\pi}\frac{1}{x^2+w^2}$. The width of a Gaussian-distributed random variable is its standard deviation.

LEMMA 1. *For $p = 1$ or 2, let X and Y be p-stably distributed random variables with widths w_X and w_Y, respectively. Let $Z = X + Y$. Then, for each number z, one can sample from a distribution indistinguishable from $X|(Z = z)$, to k digits of precision, in time (number of operations times precision) $(k \log(w_X)\log(w_Y)\log(z))^{O(1)}$.*

PROOF. We consider the Cauchy case. The conditional density of X is

$$\begin{aligned} f(x) &= \frac{f_{w_X}(x) f_{w_Y}(z-x)}{f_{w_X+w_Y}(z)} \\ &= \frac{w_X w_Y}{\pi(w_X+w_Y)} \cdot \frac{(w_X+w_Y)^2+z^2}{(w_X^2+x^2)(w_Y^2+(z-x)^2)}. \end{aligned}$$

This is a rational function of x whose coefficients are polynomially bounded in w_X, w_Y, and z. Since f is a rational function of x, the indefinite integral $F(x)$ of $f(x)$ can be computed explicitly. By [10], it thus suffices to pick $r \in [0,1]$ uniformly at random to $(k \log(w_X)\log(w_Y)\log(z))^{O(1)}$ bits of precision and solve $F(x) = r$ for x by the bisection method. It is unlikely that $F(x)$ requires more than just a small number of bits to represent. Also by [10], we can compute k bits of x in time $(k \log(w_X)\log(w_Y)\log(z))^{O(1)}$.

A similar statement holds for Gaussian random variables replacing Cauchy random variables. In this case, the conditional distribution $X|(Z=z)$ is itself a (shifted and scaled) Gaussian, and the amounts of the shift and scale are rational functions of w_X, w_Y, and z. One can use standard techniques for sampling from a Gaussian of given parameters. $\square$

We will use this lemma with w_X and w_Y at most polynomial in N, so $\log(w_X)$ and $\log(w_Y)$ are small. As in [10], with high probability, $\log(z)$ is small. Finally, a small number k of bits of precision suffices for our computation.

LEMMA 2 (NAOR AND REINGOLD [14]). *Call a set $S \subseteq [0,N)$* range-summable *if it has compact description from which, for each interval J, we can quickly compute $|S \cap J|$. For each range-summable S, we can construct a sequence $\mathbf{V}$ of pseudorandom p-stably distributed random variables r_i such that (i) the construction requires small space and (ii) given interval I, we can quickly compute the range sum of the* variables *in S,* i.e., $\sum_{i \in S\cap I} r_i$.

PROOF. The central idea of the lemma is to compute dyadic range sums in a tree and to use a pseudorandom generator for the underlying randomness. For example, suppose $S = [0,N)$, $N = 4$, and the four random variables are A, B, C, and D. First generate $A+B+C+D$, then generate $(A+B)|(A+B+C+D)$, noting that $A+B$ and $C+D$ are Cauchy or Gaussian random variables, so the sums satisfy Lemma 1. This determines $C+D$. Finally, generate $A|(A+B)$ and $C|(C+D)$, thereby determining B and D.

We first give an ideal algorithm that uses too much space; we then show how to reduce the space using Nisan's generator. Specifically, we will store a tree $\mathbf{S}$ of p-stable random variable outcomes (with widths of our choosing—not necessarily unit widths) that, ultimately, can be generated by the generator. We give the proof for Cauchy random variables.

We construct dyadic range sums of $\mathbf{A}$ in a tree, in depth-first search order. That is, we first compute $|S|$, then generate and store an outcome for $r_0^{\log(N)} = \sum_{i\in S} r_i \sim C(0,|S|)$. Next, we compute $|S \cap [0, N/2)|$ and generate and store an outcome for $r_0^{\log(N)-1} = \sum_{i\in[0,N/2)\cap S} r_i$, from the distribution conditioned on the previously-stored value for $\sum_{i\in S} r_i$, using Lemma 1. These two values determine $r_1^{\log(N)-1} = \sum_{i\in[N/2,N)\cap S} r_i$ (without further randomness). Continuining this way, we store outcomes for r_j^k, for $0 \le k \le \log(N)$ and $0 \le j < N/2^k$, with $r_j^k = r_{2j}^{k-1} + r_{2j+1}^{k-1}$.

Given dyadic interval I, we can recover $\sum_{i\in S\cap I} r_i$ from $\mathbf{S}$ by descending the tree, and using a telescopic chain of conditional probabilities. For an arbitrary I (not necessarily dyadic), partition I into dyadic intervals and proceed. We omit the simple details due to paucity of space.

Finally, instead of storing the outcomes in $\mathbf{S}$ as above, we use Nisan's generator secure against small-space tests. We then need to verify that our algorithm can be regarded as a small-space test of the randomness. That is, one needs to verify that our overall algorithm, which has oneway access to the data and twoway access to its source of randomness, could instead be implemented in small space given oneway access to the randomness and twoway access to the data, *i.e.*, the function computed by our algorithm can also be computed by an algorithm of the form fooled by Nisan's generator. Since the two implementations always compute exactly the same output, and since, with high probability, the latter implementation produces the same output on true randomness as on pseudorandomness, the pseudorandom generator suffices. We omit the details of the simulation. $\square$

The above is useful even for the degenerate case of $S = [0,N)$, as we see next. We give a data structure that supports Update, Norm Estimation, and Parameter Estimation. After that, we will use a non-trivial S to support Identification as well.

LEMMA 3. *There exists a synopsis data structure that supports the following array sketch operations: Update, Norm Estimation, and Parameter Estimation.*

PROOF. Define the synopsis data structure for $\mathbf{A}$ as multiple independent copies of $\langle \mathbf{V}, \mathbf{A}\rangle$, where $\mathbf{V}$ is a p-stable distribution. Update support follows by construction. (Observe that, by linearity of the dot product, we just need the sketch $c\langle \mathbf{V}, \chi_I\rangle$ of $c\chi_I$, a range sum, to add to an existing sketch of a signal.) Norm Estimation support is from [10]. We turn now to Parameter Estimation.

Let $h = h(\mathbf{A}) = B\mathbf{A}$, where B is a projection matrix. By Lemma 2 with $S = [0,N)$, we can quickly compute $B\chi_I$, and thus also $h'(c) = B(\mathbf{A} - c\chi_I)$, carried to any small number of bits of precision, which suffices. In the following we show how to find c minimizing $\operatorname{median}(|h'(c)_1|, \ldots, |h'(c)_d|)$, where $h'(c)_j$ is the j'th independent copy of a basic sketch. As in [10], this implies the lemma. To this end, observe that $h'(c)$ is a linear function of c. Therefore, we need to solve the following optimization program:

$$\min_{0\le c\le R} \operatorname{median}\{|a_1 c + b_1|, \ldots, |a_d c + b_d|\}$$

This problem can be trivially solved in $O(d^3)$ time as follows. Firstly, we compute all points c such that $|a_i y + b_i| = |a_j c + b_j|$, for $i \ne j$; there are at most d^2 such points. These points split the range $[0, \ldots, R]$ into at most d^2+1 intervals $i_1 \ldots i_t$. Observe that if we restrict c to any $i_j = [l_j, r_j]$, the problem amounts to finding the median of the values $\min(|a_i l_j + b_i|, |a_i r_j + b_i|)$, for $i = 1 \ldots d$, which can be done in $O(d)$ time. Therefore, the problem can be solved in $O(d^3)$ time by enumerating all intervals i_j.

Once we have found c optimizing $\|\mathbf{A} - c\chi_I\|_s$, observe that this suffices, since

$$\|\mathbf{A} - c\chi_I\|_s \le \left\|\mathbf{A} - c^I_{\mathrm{opt}}\chi_I\right\|_s \le (1+\epsilon_s)\left\|\mathbf{A} - c^I_{\mathrm{opt}}\chi_I\right\|.$$

The Gaussian/ℓ_2 case is actually easier. By [10], we need to find c minimizing $\sum_j (a_j c + b_j)^2$, a univariate quadratic in c. $\square$

We now turn to identification. First suppose there is a single overwhelmingly large interval to find. Let $H_j = \{i :$ bit j of the binary expansion of i is $1\}$ (the j'th "bit test"), to be used in Lemmas 4 and 6.

Lemma 4. *Fix j. There is a synopsis data structure for $\mathbf{A}$ that supports update and supports finding the (single) dyadic interval I of length 2^j with $\|\pi(\mathbf{A}, I)\| \ge (2/3)\|\mathbf{A}\|$, if there is such an I.*

Proof. We show the lemma for $j = 0$; other cases are similar.

For $0 \le j < \log N$, use the norm estimation data structure of Lemma 3 on the projected signals $\pi(\mathbf{A}, H_j)$. Also use the norm estimation data structure on $\mathbf{A}$ itself.

We find the position of interval I (of length 1 in this example) bit by bit. To find the most significant bit, compare $\left\|\pi(\mathbf{A}, H_{\log(N)-1})\right\|$ with $\|\mathbf{A}\|$. If $\left\|\pi(\mathbf{A}, H_{\log(N)-1})\right\|_s > (1/2)\|\mathbf{A}\|_s$ for reasonably good estimates, then

$$\left\|\pi(\mathbf{A}, H_{\log(N)-1})\right\| > (1/3)\|\mathbf{A}\|,$$

and we conclude $I \subseteq H_{\log(N)-1} = \left[0, \frac{N}{2}\right)$; otherwise, $I \subseteq \left[\frac{N}{2}, N\right)$. In general, the j'th test yields the j'th bit of the position of I. $\square$

The lemma above gives us an oracle to find a (dyadic) interval of large energy. Now we show how to reduce the general case (no overwhelmingly large interval) to the above case. This contains the elements of "group testing."

Lemma 5. *Let N be a power of 2 and let Y be a field of characteristic 2, $|Y|$ small. Fix $j \le \log(N)$. There's a function $f_s : [0, N) \to Y$, parametrized by a short random seed s, such that*

- *f_s is constant on dyadic ranges of length 2^j.*
- *f_s is a uniform pairwise independent mapping on the set of dyadic ranges of length 2^j.*
- *Given interval $I \subseteq [0, N)$, H_j, and $y \in Y$ we can quickly compute $|\{i \in I \cap H_j : f_s(i) = y\}|$.*

Proof. We illustrate the statement for $j = 0$. Other cases are similar.

Let H be the extended Hamming matrix of length N, gotten by adding a row of 1's to the collection of all columns of height $\log(N)$, in order. Let s be a vector of length $\log(N) + 1$ over Y, and let $f_s(i)$ be the i'th element of the vector-matrix product sH over Y. The claimed properties are straightforward to check. (See, *e.g.*, [3].) $\square$

Definition 3. Fix a signal, $\mathbf{A}$. *Pre-Identification* on $\mathbf{A}$ with parameter η consists of finding a compact list that contains all dyadic intervals I for which $\|\pi(\mathbf{A}, I)\| \ge \eta\|\mathbf{A}\|$.

Lemma 6. *We can construct a synopsis structure supporting updates and Pre-Identification with parameter η.*

Proof. We show the lemma for (dyadic) intervals of length 1; the other $\log(N)$ cases are similar.

Let Y be a field of characteristic 2 and size $\Theta(1/\eta)$. Pick seed $s \in Y^{\log(N)+1}$ at random and fix $y \in Y$. Let $S = \{i : f_s(i) = y\}$. We show that, with high probability, if S contains a position i with $\|\pi(\mathbf{A}, \{i\})\| \ge \eta\|\mathbf{A}\|$, then $\|\pi(\pi(\mathbf{A}, S), \{i\})\| = \|\pi(\mathbf{A}, \{i\})\| \ge (2/3)\|\pi(\mathbf{A}, S)\|$.

To see this, observe that, by pairwise independence and linearity of expectation, for each i,

$$\begin{aligned} &E\left[\|\pi(\mathbf{A}, S\setminus\{i\})\| \,\middle|\, i \in S\right] \\ &\quad = E\left[\|\pi(\mathbf{A}, S\setminus\{i\})\|\right] \le E\left[\|\pi(\mathbf{A}, S)\|\right] = (\eta/8)\|\mathbf{A}\| \end{aligned}$$

if $|Y| = \Theta(1/\eta)$ is large enough. Thus, with probability at least $3/4$,

$$E\left[\|\pi(\mathbf{A}, S\setminus\{i\})\| \,\middle|\, i \in S\right] \le (\eta/2)\|\mathbf{A}\|.$$

Since $\|\pi(\mathbf{A}, \{i\})\| \ge \eta\|\mathbf{A}\|$, it follows that $\pi(\mathbf{A}, S)$ has a single position with $2/3$ the total energy, if it contains any position of energy at least $\eta\|\mathbf{A}\|$.

For each seed s, the $y \in Y$ induce a partition of $[0, N)$; namely, $[0, N) = \bigcup_y \{i : f_s(i) = y\}$. Thus, for each position i with an η fraction of the signal energy, there is some y, namely, $y = f_s(i)$, such that $i \in \{i : f_s(i) = y\}$. The position i is identified for this choice of y. We will exhaustively consider the compact list of all y's, thereby pre-identifying *each* η-significant i with probability $3/4$. By boosting the probability of success from $3/4$ to $1 - \eta/4$ through repetition and taking the union of the lists, it follows that *all* intervals are simultaneously isolated and pre-identified this way.

Note that $S \cap H_j$ satisfies the properties of Lemma 2, where $S = \{i : f_s(i) = y\}$ and H_j is a bit test of Lemma 4. (Observe that $\{i : f_s(i) = y\} \cap H_j \subseteq [0, N)$ is equivalent to a set of the form $\{i : f_{s'}(i) = y\} \subseteq [0, N/2)$, where the new seed s' is easy to compute from s.) It follows that there's a data structure that supports Update and Norm Estimation of $\pi(\mathbf{A}, S \cap H_j)$, whence support for Pre-Identification follows. $\square$

Theorem 7. *We can construct a (ϵ_s, η, N)-array sketch that uses small space and supports its operations in small time.*

Proof. By Lemma 3, it remains only to give a structure that supports Update and Identification.

The array sketch of a signal $\mathbf{A}$ is defined as follows. Pick a random seed s. Let Y be a finite field of characteristic 2 and approximately $1/\eta^2$ elements. For each $y \in Y$ and each $j < \log(N)$, compute $\langle \pi(\mathbf{A}, \{i : f_s(i) = y\} \cap H_j), \mathbf{V}\rangle$, where $\mathbf{V}$ is a sequence of random variables, generated via Naor-Reingold's tree construction [14] of conditional probabilities, using Nisan's generator [15], for range summable $S = \{i : f_s(i) = y\} \cap H_j$. For each y, also compute $\langle \pi(\mathbf{A}, \{i : f_s(i) = y\}, \mathbf{V}\rangle$, for $\mathbf{V}$ similarly generated for $S = \{i : f_s(i) = y\}$.

Support for update follows by construction. By Lemma 6, we can find a compact list of intervals that includes all intervals I with $\|\pi(\mathbf{A}, I)\| \ge \eta\|\mathbf{A}\|$. Observe that $\|\mathbf{A}\| - \left\|\mathbf{A} - c^I_{\mathrm{opt}}\chi_I\right\| \le \|\pi(\mathbf{A}, I)\|$, so the compact list of intervals contains all I such that $\left\|\mathbf{A} - c^I_{\mathrm{opt}}\chi_I\right\| \le (1-\eta)\|\mathbf{A}\|$;

it might also contain other I's such that $\left\|\mathbf{A} - c^I_{\rm opt}\chi_I\right\| > (1-\eta/2)\left\|\mathbf{A}\right\|$. By Lemma 3, for each I on the list, we can find a parameter c such that $\left\|\mathbf{A} - c^I_{\rm opt}\chi_I\right\| \le \left\|\mathbf{A} - c\chi_I\right\|_s \le (1+\epsilon_s)\left\|\mathbf{A} - c^I_{\rm opt}\chi_I\right\|$, thereby estimating $\left\|\mathbf{A} - c^I_{\rm opt}\chi_I\right\|$ to within $\pm\epsilon_s\left\|\mathbf{A}\right\|$. This and an estimate $\left\|\mathbf{A}\right\|_s$ for $\left\|\mathbf{A}\right\|$ are sufficient to select a set of I's satisfying the desired properties, provided $\epsilon_s \le \eta/8$. □

3. EFFICIENT WAVELET APPROXIMATION

In this section, we give an algorithm that finds a $(1+\epsilon)$-approximation to the best B-term wavelet representation for the signal $\mathbf{A}$, given only a sketch of $\mathbf{A}$. We will use the array sketch synopsis data structure from the previous section. The results in this section motivate the discussion of main result which will follow, but it may be of independent interest as well. The results in this section will hold for ℓ_2 norm only. (Our main result in next section will work for both ℓ_2 and ℓ_1).

Definition 4. A (Haar) wavelet is a function ψ on $[0,N)$ of one of the following forms, for integers j and k:

- $\frac{1}{\sqrt{N}}\chi_{[0,N)}$
- $2^{-j/2}\left(-\chi_{[k2^{j-1},(k+1)2^{j-1})} + \chi_{[(k+2)2^{j-1},(k+3)2^{j-1})}\right)$.

There are N wavelets altogether, and they form an orthonormal basis, *i.e.*, $\langle\psi,\psi'\rangle$ is 1 if $\psi=\psi'$ and 0 otherwise. Let ψ_j denote the j'th wavelet basis vector.

Every signal can be reconstructed exactly from all its wavelet coefficients (its full *wavelet transform*, an orthonormal linear transformation), as $\mathbf{A} = \sum_j \langle\mathbf{A},\psi_j\rangle\psi_j$, whence a formal linear combination of distinct wavelets is its own wavelet transform.

Parseval's equality states that the L^2 norm of a signal is invariant under orthonormal change of basis: $\sum_i \mathbf{A}_i^2 = \sum_j \langle\mathbf{A},\psi_j\rangle^2$. The intuition behind our algorithm is that, by Parseval's inequality, we simply want to find the biggest B coefficients. We cannot seek them directly, however, since one of the biggest coefficients may be small, and we cannot try to identify all small coefficients since there are too many. Thus the algorithm proceeds greedily—after finding and subtracting the biggest wavelet term, the second biggest becomes bigger relative to the residual signal, and, therefore, easier to find.

In this section, we use essentially the Array Sketch data structure of Section 2. First, we require Definition 1 to hold with respect to $\|\cdot\|_2^2$, which is more convenient than $\|\cdot\|_2^1$ in this context. Note that Parameter Estimation is easily modified for wavelets—given $\mathbf{A}$ and ψ, $\left\|\mathbf{A} - d\psi\right\|_s^2$ is, by [10], a univariate quadratic in d, which is easy to optimize. Pre-Identification (of dyadic intervals I with $\|\pi(\mathbf{A},I)\|^2 \ge \eta\left\|\mathbf{A}\right\|^2$) is unchanged, so the desired Identification can be achieved by performing Pre-Identification and then sufficiently accurate Parameter Estimation. We also require parameters $\eta = \frac{\epsilon}{4B}$, and $\epsilon_s = \epsilon^2\eta/128$ to be set differently than in Section 2.

The algorithm is as follows:

Initially $\mathbf{R}=0$ and $S=\emptyset$ and sketch $=$ sketch$(\mathbf{A})$. Repeat

1. If $\mathbf{A}-\mathbf{R} = \sum_j d'_j\psi_j$ then, using the sketch, find compact Λ with $\left\{j : |d'_j|^2 > \eta\left\|\mathbf{A}-\mathbf{R}\right\|_2^2\right\} \subseteq \Lambda$ and $\Lambda \subseteq \left\{j : |d'_j|^2 > (1-\epsilon_s)\eta\left\|\mathbf{A}-\mathbf{R}\right\|_2^2\right\}$.
2. If $\Lambda=\emptyset$ or ($|S|=B$ and $S\cap\Lambda=\emptyset$) exit loop and output $\mathbf{R}$.
3. Using the sketch, for all $j\in\Lambda$ let $\tilde{d}_j \leftarrow$ estimate(j).
4. If $|S|<B$, then $S \leftarrow S\cup\{j'\}$ where j' is the index with the largest value of $|\tilde{d}_j|$.
5. For each $j\in S$ (which may have changed since start of loop)
   (a) $\mathbf{R} \leftarrow \mathbf{R} + \tilde{d}_j\psi_j$
   (b) sketch $\leftarrow$ sketch $-\ \tilde{d}_j \cdot$ sketch(ψ_j)

Define d_j and $\tilde{d}_j$ by $\mathbf{A} = \sum_j d_j\psi_j$ and $\mathbf{R} = \sum_{j\in S}\tilde{d}_j\psi_j$.

LEMMA 8. *An array sketch for wavelets can estimate any coefficient d of $\mathbf{A}$ as $\tilde{d}$ with $|d-\tilde{d}|^2 \le \epsilon_s\left\|\mathbf{A}\right\|^2$.*

PROOF. Let ψ be the basis function corresponding to coefficient d. We have

$$|d-\tilde{d}|^2 + \left\|\mathbf{A}-d\psi\right\|^2 = \left\|\mathbf{A}-\tilde{d}\psi\right\|^2 \le (1+\epsilon_s)\left\|\mathbf{A}-d\psi\right\|^2,$$

whence $|d-\tilde{d}|^2 \le \epsilon_s\left\|\mathbf{A}-d\psi\right\|^2 \le \epsilon_s\left\|\mathbf{A}\right\|^2$. □

LEMMA 9. *At termination of the above algorithm, for any $i\notin S$ and $j\in S$, we have $|d_i|^2 \le (1+\epsilon/2)|d_j|^2$.*

PROOF. Omitted. The central idea is that if otherwise, we would have chosen i instead of j to include in S. □

We will now show that the indices in S, with optimal coefficients, gives a good enough representation. Let $\mathbf{R}' = \sum_{j\in S} d_j\psi_j$, where $d_j = \langle\mathbf{A},\psi_j\rangle$ are the exact coefficients.

LEMMA 10. *If $|S|<B$ then $\left\|\mathbf{A}-\mathbf{R}'\right\|^2 \le \left\|\mathbf{A}-\mathbf{R}_{\rm opt}\right\|_2^2 + \frac{\epsilon}{4}\left\|\mathbf{A}-\mathbf{R}\right\|_2^2$.*

PROOF. Consider some $i\notin S$. Since $|S|<B$, we could have added i but did not. Thus $i\notin\Lambda$ when we decided to output. Therefore $d_i^2 \le \frac{\epsilon}{4B}\left\|\mathbf{A}-\mathbf{R}\right\|_2^2$. Let $S_{\rm opt}$ be the set of basis vectors in $\mathbf{R}_{\rm opt}$. Thus,

$$\sum_{i\in S_{\rm opt}\setminus S} d_i^2 \le |S_{\rm opt}-S|\frac{\epsilon}{4B}\left\|\mathbf{A}-\mathbf{R}\right\|_2^2 \le \frac{\epsilon}{4}\left\|\mathbf{A}-\mathbf{R}\right\|_2^2.$$

Therefore, $\left\|\mathbf{A}-\mathbf{R}'\right\|_2^2 = \sum_{j\notin S} d_j^2 \le \sum_{j\notin S_{\rm opt}} d_j^2 + \sum_{j\in S_{\rm opt}\setminus S} d_j^2 \le \left\|\mathbf{A}-\mathbf{R}_{\rm opt}\right\|_2^2 + \frac{\epsilon}{4}\left\|\mathbf{A}-\mathbf{R}\right\|_2^2$. □

LEMMA 11. $\left\|\mathbf{A}-\mathbf{R}'\right\|_2^2 \le (1+\frac{\epsilon}{2})\left\|\mathbf{A}-\mathbf{R}_{\rm opt}\right\|_2^2 + \frac{\epsilon}{2}\left\|\mathbf{A}-\mathbf{R}\right\|_2^2$.

PROOF. Once again, let $S_{\rm opt}$ be the set of basis vectors in $\mathbf{R}_{\rm opt}$. If $|S|<B$ then the previous lemma applies. In case of $|S|=B=|S_{\rm opt}|$, every $i\in S_{\rm opt}-S$ can be matched to an index $m(i)\in S-S_{\rm opt}$. By Lemma 9, $d_i^2 \le (1+\epsilon/2)d_{m(i)}^2$, and, summing over all such pairs,

$$\sum_{i\in S_{\rm opt}-S} d_i^2 \le (1+\epsilon/2)\sum_{i\in S\setminus S_{\rm opt}} d_i^2.$$

By adding terms outside $S\cup S_{\rm opt}$, $\left\|\mathbf{A}-\mathbf{R}'\right\|_2^2 = \sum_{i\notin S} d_i^2 \le (1+\epsilon/2)\sum_{i\notin S_{\rm opt}} d_i^2 = (1+\epsilon/2)\left\|\mathbf{A}-\mathbf{R}_{\rm opt}\right\|_2^2$. □

We now show that the representation $\mathbf{R}$ that we produce, whose coefficients are indexed in S but are only approximations, is accurate enough.

THEOREM 12. *The above algorithm constructs a* $(1+O(\epsilon))$*-approximation,* $\|\mathbf{A}-\mathbf{R}\|_2^2 \le (1+O(\epsilon))\,\|\mathbf{A}-\mathbf{R}_{\text{opt}}\|_2^2$.

PROOF. Let $\mathbf{A}-\mathbf{R} = \sum_j \hat{d}_j \psi_j$. If $j \notin S$ then $\hat{d}_j = d_j$. Since $\Lambda \cap S = \emptyset$ at termination, for $j \in S$, we have $|\hat{d}_j|^2 \le \frac{\epsilon}{4B}\|\mathbf{A}-\mathbf{R}\|_2^2$. Therefore,

$$\begin{aligned}
\|\mathbf{A}-\mathbf{R}\|_2^2 &= \sum_j \hat{d}_j^2 = \sum_{j\notin S} d_j^2 + \sum_{j\in S}\hat{d}_j^2\\
&\le \sum_{j\notin S} d_j^2 + B\frac{\epsilon}{4B}\|\mathbf{A}-\mathbf{R}\|_2^2\\
&\le \|\mathbf{A}-\mathbf{R}'\|_2^2 + \frac{\epsilon}{4}\|\mathbf{A}-\mathbf{R}\|_2^2\\
&\le (1+\epsilon/2)\,\|\mathbf{A}-\mathbf{R}_{\text{opt}}\|_2^2 + \frac{3\epsilon}{4}\|\mathbf{A}-\mathbf{R}\|_2^2,
\end{aligned}$$

using Lemma 11 in the last step. □

We are almost done except we need to prove that the above algorithm terminates in few steps,

LEMMA 13. *The above algorithm terminates in at most* $O(B\log(N\|\mathbf{A}\|)/\log(1/\epsilon))$ *steps.*

PROOF. Omitted. Each time we update one of only B coefficients, its square error drops by the factor $\frac{\epsilon_s}{\eta} = O(\epsilon)$, from an initial value bounded by $\|\mathbf{A}\|$. The minimum positive square error is $\Omega(1/N)$. □

Any B-bucket histogram can be regarded as a wavelet representation with $O(B\log(N))$ terms and any B-term wavelet representation can be regarded as a histogram with $O(B)$ buckets. Thus we also immediately have a result for histograms with $O(\log(N))$ blowup in the number of buckets:

THEOREM 14. *The best* $O(B\log(N))$*-term wavelet representation* $\mathbf{R}$*, found efficiently by the above algorithm, is a* $O(B\log(N))$*-bucket histogram with error at most* $(1+\epsilon)$ *times that of the best* B*-bucket histogram.*

4. FINDING A ROBUST APPROXIMATION

In this section, we define and show how to find a robust approximation $\mathbf{H}_r$ to $\mathbf{A}$ with $\text{poly}(B, \log(N), \frac{1}{\epsilon})$ buckets. We will use the synopsis data structures from Section 2. As mentioned before, intuitively, a robust approximation captures almost as much information as is contained in a B-bucket approximation of the signal. The formal definition is:

Definition 5. Given a signal $\mathbf{A}$, a histogram $\mathbf{H}_r$ is a (B_r, ϵ_r)-*robust* approximation to $\mathbf{A}$ if, given any collection X of $|X| \le B_r$ non-overlapping intervals, any histogram $\mathbf{H}$ which can be expressed as

$$\mathbf{H} = \begin{cases} H_r & \text{on } [0,N) - \bigcup_{I\in X} I \\ c_I\chi_I & \text{on } I \in X \end{cases}$$

would satisfy $(1-\epsilon_r)\,\|\mathbf{A}-\mathbf{H}_r\| \le \|\mathbf{A}-\mathbf{H}\|$.

That is, a robust histogram is not improved much if it is refined by a small number of additional buckets. Note that $|X| \le B_r$ is small, but $\bigcup_{I\in X} I$ can be large, even equal to $[0,N)$. Throughout this section we will use the term robust to denote (B_r, ϵ_r)-robust. We will eventually apply the theorems with $B_r = B$ and $\epsilon_r = O(\epsilon/B)$. Before we start, the following straightforward property of ℓ_1 norm[2]

LEMMA 15 (DECOMPOSABILITY). *If* $\mathbf{H} = \mathbf{H}'$ *everywhere except on a non-overlapping set of intervals* X*, then*

$$\begin{aligned}
&\|\mathbf{A}-\mathbf{H}\| - \|\mathbf{A}-\mathbf{H}'\|\\
&\quad = \sum_{I\in X}\left(\|\pi(\mathbf{A}-\mathbf{H}, I)\| - \|\pi(\mathbf{A}-\mathbf{H}', I)\|\right).
\end{aligned}$$

We show that if $\mathbf{H}$ is not a robust approximation to $\mathbf{A}$, it can be improved. This part is similar to the wavelet construction proof we saw in the previous section; here we improve the robust histogram by identifying and subtracting off a set of large coefficients repeatedly. First, we show that this improvement is possible; afterwards, we'll show how to perform the improvement efficiently.

LEMMA 16. *Given a histogram* $\mathbf{H}$ *which is not robust, there exists a dyadic interval* I *and a parameter* c *such that a histogram* $\mathbf{H}'$ *which agrees with* $\mathbf{H}$ *everywhere except* I*, and takes the value* c *on* I*, approximates* $\mathbf{A}$ *better than* $\mathbf{H}$ *by a factor* $1-\epsilon_r/(4B_r\log N)$.

PROOF. (sketch) Since $\mathbf{H}$ is not robust, Definition 5 guarantees a set X of intervals that improves $\mathbf{H}$. Let X^D be a similar set of at most $B_r\log(N)$ *dyadic* intervals. Some interval in X^D gives at least the average improvement. □

We can construct a robust approximation greedily. Using the Identification operation of an array sketch, we can find a dyadic interval I such that refining $\mathbf{H}$ by I and changing the value only on I to c_{opt}, getting $\mathbf{H}_I^{c_{\text{opt}}}$, gives $\left\|\mathbf{A}-\mathbf{H}_I^{c_{\text{opt}}}\right\| \le (1-\epsilon_r/(4B_r\log N))\,\|\mathbf{A}-\mathbf{H}\|$. Furthermore, using Parameter Estimation, we can find a parameter c with $\|\mathbf{A}-\mathbf{H}_I^c\| \le (1-\epsilon_r/(8B_r\log N))\,\|\mathbf{A}-\mathbf{H}\|$. Call this process one *round of improvement*; each round of improvement is quick. For an integer-valued signal, one easily sees that if a histogram $\mathbf{H}$ has error less than $\Theta(1)$, then it can trivially and efficiently be improved to have error zero by changing spline parameters only. Thus, starting from $\mathbf{H} = 0$ (*i.e.*, $\|\mathbf{A}-\mathbf{H}\| = \|\mathbf{A}\|$), by attempting to perform $R \le O(B_r\log(N)\log\|\mathbf{A}\|/\epsilon_r)$ rounds of improvement on $\mathbf{H}$, either no improvement is possible beyond some round $R' < R$ (in which case $\mathbf{H}$ at that point is robust by Lemma 16), or, after round R, we have $\|\mathbf{A}-\mathbf{H}\| \le \Theta(1)$, so, by changing spline parameters in $\mathbf{H}$, we get $\mathbf{H}' = \mathbf{A}$, which is robust.

LEMMA 17. *Given* ϵ_r *and* B_r*, in time* t *we can find a* (B_r, ϵ_r)*-robust histogram* $\mathbf{H}_r$ *of* $O((B_r/\epsilon_r)\log(N)\log\|\mathbf{A}\|)$ *buckets from an* (ϵ_s, η)*-array sketch for* $\mathbf{A}$*, where* t, ϵ_s, η *are in* $\text{poly}(B_r, \frac{1}{\epsilon_r}, \log N, \log\|\mathbf{A}\|)$.

In particular, consider refining $\mathbf{H}_r$ by the optimal histogram $\mathbf{H}_{\text{opt}}$ with $B \le B_r$ buckets (so that none of $\mathbf{H}_r$ remains). It follows that

LEMMA 18. *Let* $\mathbf{H}_r$ *be a* (B, ϵ_r) *robust approximation to* $\mathbf{A}$ *and let* $\mathbf{H}_{\text{opt}}$ *be an optimal* B*-bucket representation. Then* $\|\mathbf{A}-\mathbf{H}_r\| \le \|\mathbf{A}-\mathbf{H}_{\text{opt}}\|/(1-\epsilon_r)$.

[2]Similar intermediate statements hold for ℓ_2 norm or its square, as appropriate, so our main result will hold for ℓ_2.

5. A $(1+\epsilon)$-APPROXIMATION HISTOGRAM CONSTRUCTION

Before we present the approximation algorithm, we recall an optimal dynamic program for histogram construction, which uses $O(BN)$ space and $O(N^2B)$ time. Inductively, for each $k < B$, the algorithm stores, for each $x \in [0, N)$, the best histogram with k buckets which approximates the signal on the interval $[0, x)$. Using this, it constructs the best $(k+1)$-bucket approximation for each $x \in [0, N)$. The best $(k+1)$-bucket histogram extends some k-bucket histogram on $[0, x')$. By optimality, this k-bucket histogram has to be also the best possible k-bucket histogram for $[0, x')$. Since the algorithm is allowed $O(BN)$ space it can store the signal and appropriate prefix sums to compute the best value to attribute to a bucket.

Intuitively, the dynamic program constructs best approximation of subintervals and tries to extend them. In that sense, for a fixed k, we call

$$\{\text{best } k\text{-bucket histogram on } [0, x) | 0 \leq x < N\}$$

a *k-extension-basis* (briefly, a k-basis) since it consists of k-bucket histograms and supports constructing the best $(k+1)$-bucket histogram on $[0, N)$ for any x by extending one of the stored histograms. Although not explicitly stated, this idea of constructing an extension basis was considered in [9], but for a weaker non-dynamic model.

We will proceed along similar lines. But this similarity will be restricted to the high level of constructing an *Extension Basis* and performing extensions to any $[0, x)$. Since we are allowed $\text{poly}(\log N, \frac{1}{\epsilon}, B)$ space our extension basis size has to be small—we'll construct

$$\{\text{best } k\text{-bucket histogram on } [0, x_i) | i = 0, 1, \ldots, \ell\}$$

for $\ell \leq \text{poly}(\log N, \frac{1}{\epsilon}, B)$, and show that we can still (approximately) construct a $(k+1)$-basis from a k-basis. Furthermore, assigning the best spline parameter to an interval is approximate (through sketches). Finally, the dynamic programming algorithm recalled above, if implemented naturally, needs to evaluate $\|\pi(\mathbf{A} - \mathbf{H}, I)\|$. As we discuss below, we cannot even estimate this quantity directly from sketches. We can only estimate the norm of $\mathbf{A}$ or $\mathbf{A} - \mathbf{H}$ on the entire interval $[0, N)$; there are too many intervals to store sketches on each in order to estimate norms on each. Instead of approximating $\pi(\mathbf{A}, [0, x))$ by a histogram $\mathbf{H}$, we approximate $\mathbf{A}$ by a histogram $\mathbf{H}'$ that equals $\mathbf{H}$ on $[0, x)$ and equals $\mathbf{H}_r$ on $[x, N)$—this is where we use $\mathbf{H}_r$. Thus we estimate $\|\mathbf{A} - \mathbf{H}'\|$ instead of $\|\pi(\mathbf{A} - \mathbf{H}, [0, x)\|$, and use only sketches on all $[0, N)$.

In Subsection 5.1, we define and show how to use (recursively) an extension basis for signal $\mathbf{A}$, given just an array sketch representation for $\mathbf{A}$ (from which we can also build a robust approximation to $\mathbf{A}$). In Subsection 5.2, we instantiate parameters of the extension basis and show that the histogram ultimately produced has small enough error. Finally, in Subsection 5.3, we show how to build a suitable extension basis quickly (in particular, the extension basis we build is small).

5.1 The Extension Basis and Recursive Extension

Definition 6. A collection of histograms-interval pairs $\{(\mathbf{H}_i^k, I_i)\}$ is a *Δ-separated k-basis of size ℓ, with error parameters ϵ_k and $E(k)$*, if the following holds.

1. $\emptyset = I_0 \subseteq [0, 1) = I_1 \subseteq \cdots \subseteq I_\ell = [0, N)$ and $0 \in I_i$ for all $i \geq 1$. (That is, the I_i's are a sequence of nested prefixes increasing from $\emptyset$ to $[0, N)$.)
2. Each $\mathbf{H}_i^k$ is a k-bucket histogram on I_i and is equal to $\mathbf{H}_r$ on $I_\ell - I_i$.
3. $\|\mathbf{A} - \mathbf{H}_i^k\| \leq \|\mathbf{A} - \mathbf{H}_{i-1}^k\| + \Delta$ if I_i extends I_{i-1} by more than one element, $|I_i| > |I_{i-1}| + 1$.
4. Error Property: Each $\mathbf{H}_i^k$ is a good k-bucket approximation for $\mathbf{A}$ amongst all histograms $\mathbf{H}$ that have k buckets in I_i, and agree with $\mathbf{H}_r$ on $I_\ell - I_i$. Formally, $\|\mathbf{A} - \mathbf{H}_i^k\| \leq (1 + \epsilon_k) \|\mathbf{A} - \mathbf{H}\| + E(k)$.

The reason we call the above a k-extension basis is immediate from the following construction procedure for a nearly optimal histogram $\mathbf{H}_{\text{sol}}$ that approximates $\mathbf{A}$ on the interval $[0, x)$ while using at most $k+1$ buckets in $[0, x)$. In fact it is immediate that if we could generate such an approximation, we could generate a Δ-separated $(k+1)$-basis by iterating through the values of x and retaining the necessary histograms. Of course, iterating through $[0, N)$ is prohibitive; we will show how to construct an extension basis efficiently, but first we need to give a construction process that preserves the Error Property for $\mathbf{H}_{\text{sol}}$ with a small error $E(k+1)$. But recall that we can only compare histograms that are defined on the entire domain $[0, N)$. Consider the following construction procedure.

Definition 7. Given a k-extension basis $\{(\mathbf{H}_i^k, I_i)\}$ and an interval $I = [0, x)$, the following *extension procedure* finds a $(k+1)$-bucket histogram on I.

1. For each $i \leq \ell$
   (a) Consider extending $\mathbf{H}_i^k$ to $\mathbf{H}_i$ by adding a bucket $I - I_i$.
   (b) $\mathbf{H}_i$ agrees with $\mathbf{H}_i^k$ on I_i, has the value c_i on $I - I_i$ and agrees with $\mathbf{H}_r$ on $[0, N) - I$.
   (c) c_i is chosen such that $\|\mathbf{A} - \mathbf{H}_i\|$ is minimized over all choices of c_i.
2. Pick j minimizing $\|\mathbf{A} - \mathbf{H}_j\|$ over all choices $j \leq \ell$ ($\|\mathbf{A} - \mathbf{H}_j\| \leq \|\mathbf{A} - \mathbf{H}_i\|$ for any i).
3. Set $\mathbf{H}_{\text{sol}} = \mathbf{H}_j$.

Lemma 19. *If $\mathbf{H}^*$ is a histogram with at most $k+1$ buckets intersecting the interval I and $\mathbf{H}^*$ agrees with robust histogram $\mathbf{H}_r$ on $I_\ell - I$, and if $\mathbf{H}_{\text{sol}}$ is produced by Definition 7 from a Δ-separated k basis with error parameters ϵ_k and $E(k)$, then*

$$\begin{aligned} &\|\mathbf{A} - \mathbf{H}_{\text{sol}}\| \\ &\leq \quad (1 + \epsilon_k) \|\mathbf{A} - \mathbf{H}^*\| + E(k) + \Delta + (1 + \epsilon_k)\epsilon_r \|\mathbf{A} - \mathbf{H}_r\| . \end{aligned}$$

Proof. Omitted. ☐

In Definition 7, we used $\|\mathbf{A} - \mathbf{H}_i\|$. Using a sketch of $\mathbf{A}$, however, we only have access to $\|\mathbf{A} - \mathbf{H}_i\|_s$. We now modify Lemma 19 to use the norm approximation.

Definition 8. Given a k-extension basis $\{(\mathbf{H}_i^k, I_i)\}$ and an interval $I = [0, x)$, the *sketched extension procedure* finds a $(k+1)$-bucket histogram on I by using Definition 7 except by evaluating $\|\cdot\|_s$ from a sketch instead of $\|\cdot\|$.

LEMMA 20. *If* $\mathbf{H}^*$ *is a histogram with at most* $k+1$ *buckets intersecting the interval* I *and* $\mathbf{H}^*$ *agrees with robust histogram* $\mathbf{H}_r$ *on* $I_\ell - I$*, and if* $\mathbf{H}_{\rm sol}$ *is produced by Definition 8 from a* Δ*-separated* k *basis with error parameters* ϵ_k *and* $E(k)$*, then*

$$\begin{aligned}\|\mathbf{A}-\mathbf{H}_{\rm sol}\| \le\ & (1+\epsilon_s)\Big[(1+\epsilon_k)\|\mathbf{A}-\mathbf{H}^*\| + E(k) \\ & +\Delta + (1+\epsilon_k)\epsilon_r\|\mathbf{A}-\mathbf{H}_r\|\Big].\end{aligned}$$

PROOF. Define $\mathbf{H}_i^s$ to agree with $\mathbf{H}_i^k$ on I_i, to have the value c_i^s on $I \setminus I_i$, and to agree with $\mathbf{H}_r$ on $[0,N)\setminus I$, where c_i^s chosen to minimize $\|\mathbf{A}-\mathbf{H}_i^s\|_s$. Define $\mathbf{H}_j$ to agree with $\mathbf{H}_j^k$ on I_j, to have value c_j on $I\setminus I_j$, and to agree with $\mathbf{H}_r$ on $[0,N)\setminus I$, where c_j chosen to minimize $\|\mathbf{A}-\mathbf{H}_j\|$. Finally, suppose i is chosen to mimimize $\|\mathbf{A}-\mathbf{H}_i^s\|_s$, and put $\mathbf{H}_{\rm sol} = \mathbf{H}_i^s$. Then

$$\begin{aligned}\|\mathbf{A}-\mathbf{H}_{\rm sol}\| &= \|\mathbf{A}-\mathbf{H}_i^s\| \le \|\mathbf{A}-\mathbf{H}_i^s\|_s \\ &\le \left\|\mathbf{A}-\mathbf{H}_j^s\right\|_s \le \|\mathbf{A}-\mathbf{H}_j\|_s \\ &\le (1+\epsilon_s)\|\mathbf{A}-\mathbf{H}_j\|.\end{aligned}$$

The lemma follows. □

5.2 The Final Approximation Guarantee

In this subsection, we instantiate Δ and the error parameters ϵ_k and $E(k)$.

LEMMA 21. *Given a signal* $\mathbf{A}$*, distortion* ϵ*, and desired number* B *of buckets, let* $|OPT| = \|\mathbf{A}-\mathbf{H}_{\rm opt}\|$*, where* $\mathbf{H}_{\rm opt}$ *is the best* B*-bucket histogram. Put* $\epsilon_s = \epsilon/(4B)$, $\epsilon_r = \epsilon/B$*, and* $B_r = B$*. Put* $\Delta = \epsilon|OPT|/B$*, put* $E(1) = 0$ *and* $E(k+1) = (1+\epsilon_s)\left[E(k)+\Delta+\frac{2\epsilon_r|OPT|}{1-\epsilon_r}\right]$*, and put* $(1+\epsilon_k) = (1+\epsilon_s)^k$*. Then*

1. *If we use Definition 8 on a* Δ*-separated* k*-basis with error parameters* ϵ_k *and* $E(k)$*, we get a* Δ*-separated* $(k+1)$*-basis with error parameters* ϵ_{k+1} *and* $E(k+1)$*.*
2. *If we use Definition 8 on a* $(B-1)$*-basis to produce a* B*-bucket histogram* $\mathbf{H}_{\rm sol}$ *on* $[0,N)$*, we get* $\|\mathbf{A}-\mathbf{H}_{\rm sol}\| \le (1+O(\epsilon))|OPT|$*.*

PROOF. From Lemma 20 and the definition of $(k+1)$-basis, for the first statement, we need to show that

$$\begin{aligned}&(1+\epsilon_{k+1})\|\mathbf{A}-\mathbf{H}^*\| + E(k+1) \\ &\quad\ge (1+\epsilon_s)\Big[(1+\epsilon_k)\|\mathbf{A}-\mathbf{H}^*\| \\ &\qquad +E(k)+\Delta+(1+\epsilon_k)\epsilon_r\|\mathbf{A}-\mathbf{H}_r\|\Big].\end{aligned}$$

By defintion of $E(k+1)$,

$$\begin{aligned}E(k+1) &= (1+\epsilon_s)\left[E(k)+\Delta+\frac{2\epsilon_r|OPT|}{1-\epsilon_r}\right] \\ &\ge (1+\epsilon_s)\left[E(k)+\Delta+\frac{(1+\epsilon_k)\epsilon_r|OPT|}{1-\epsilon_r}\right].\end{aligned}$$

By Lemma 18, $\frac{|OPT|}{1-\epsilon_r} \ge \|\mathbf{A}-\mathbf{H}_r\|$, so

$$E(k+1) \ge (1+\epsilon_s)\left[E(k)+\Delta+(1+\epsilon_k)\epsilon_r\|\mathbf{A}-\mathbf{H}_r\|\right].$$

Since $(1+\epsilon_{k+1})\|\mathbf{A}-\mathbf{H}^*\| \ge (1+\epsilon_s)(1+\epsilon_k)\|\mathbf{A}-\mathbf{H}^*\|$, the first statement follows.

As for the second statement, observe that

$$\begin{aligned}E(B) &= \sum_{i=1}^{B}(1+\epsilon_s)^i\left[\Delta+\frac{2\epsilon_r|OPT|}{1-\epsilon_r}\right] \\ &\le O(\epsilon)|OPT|\end{aligned}$$

and $(1+\epsilon_B) = (1+O(\epsilon))$. It follows that

$$\begin{aligned}\|\mathbf{A}-\mathbf{H}_{\rm sol}\| &\le (1+\epsilon_B)|OPT| + E(B) \\ &\le (1+O(\epsilon))|OPT|.\end{aligned}$$

□

5.3 The Construction of a Small Extension Basis

We will now show how to construct a *small* extension basis with parameters Δ, ϵ_k, and $E(k)$ as above; that is, a basis of size $\ell \le \text{poly}(B, \log N, \frac{1}{\epsilon})$. First let us assume that we know $|OPT|$ up to a factor between 1 and $1+\epsilon$, by exhaustively trying powers of $(1+\epsilon)$ from $\Theta(1)$ to (an upper bound for) $\|\mathbf{A}\|$ as candidates for $|OPT|$. Recall that a $\Delta = 0$-separated extension basis can have N elements, which is prohibitively large, so we non-trivially use the fact that $\Delta = \epsilon|OPT|/B$.

Definition 9. A *small basis construction procedure* for fixed $k < B$ is the following.

1. $I_0 = \emptyset$ and $I_1 = [0,1)$.
2. If $\left\|\mathbf{A}-\mathbf{H}_i^k\right\|_s \ge 1.5(1+\epsilon_s)|OPT|$, then put $[0,x)$ into the basis for all $x \in [x_i, N)$, and halt.
3. Let $\widehat{\mathbf{H}}(x)$ be the extension basis element we would generate for interval $I = [0,x)$. After constructing $\mathbf{H}_i^k$, we consider the possibility of $I_{i+1} = [0, x_i+1)$. Three cases can happen.
   (a) If $\left\|\mathbf{A}-\widehat{\mathbf{H}}(x_i+1)\right\|_s \ge \left\|\mathbf{A}-\mathbf{H}_i^k\right\|_s + \frac{\Delta}{2}$, set $x_{i+1} = x_i+1$.
   (b) If $\left\|\mathbf{A}-\widehat{\mathbf{H}}(N)\right\|_s \le \left\|\mathbf{A}-\mathbf{H}_i^k\right\|_s + \frac{\Delta}{2}$, then set $x_{i+1} = N$.
   (c) Otherwise, perform a bisection search on $[x_i+1, N)$, and set $x_{i+1} = x$ and $x_{i+2} = x+1$, and set $\mathbf{H}_{i+1}^k = \widehat{\mathbf{H}}(x)$ and $\mathbf{H}_{i+2}^k = \widehat{\mathbf{H}}(x+1)$, where x satisfies
   $$\left\|\mathbf{A}-\widehat{\mathbf{H}}(x)\right\|_s \le \left\|\mathbf{A}-\mathbf{H}_i^k\right\|_s + \frac{\Delta}{2}$$
   and
   $$\left\|\mathbf{A}-\widehat{\mathbf{H}}(x+1)\right\|_s \ge \left\|\mathbf{A}-\mathbf{H}_i^k\right\|_s + \frac{\Delta}{2}.$$
4. Repeat for i up to ℓ.

Let $\mathcal{B}$ denote the resulting basis. Let $\mathcal{B}'$ denote

$$\left\{(I_i, \mathbf{H}_i^k) \in \mathcal{B} \;\middle|\; \left\|\mathbf{A}-\mathbf{H}_i^k\right\|_s \le 1.5(1+\epsilon_s)|OPT|\right\}.$$

LEMMA 22. *We have:*

1. *Definition 9 produces a* Δ*-separated* k*-basis* $\mathcal{B}$*.*

2. *The best B-bucket histogram on $[0, N)$ produced from a recursively-built $\mathcal{B}'$ is the same as the best produced from a recursively-built $\mathcal{B}$.*

3. *$\mathcal{B}'$ can be produced quickly (by a natural modification of the procedure in Definition 9).*

PROOF. First consider the Δ-separtion property.

Observe that since $[x_{i+1}, x_{i+2})$ is an interval of length 1, we have nothing to prove about x_{i+2}. Similarly, we may assume that $\left\|\mathbf{A} - \mathbf{H}_i^k\right\|_s < 1.5|OPT|$, since otherwise $[x_i, x_{i+1})$ is an interval of length 1. Now,

$$\begin{aligned}\left\|\mathbf{A} - \mathbf{H}_{i+1}^k\right\| &\leq \left\|\mathbf{A} - \mathbf{H}_{i+1}^k\right\|_s \\ &\leq \left\|\mathbf{A} - \mathbf{H}_i^k\right\|_s + \frac{\Delta}{2} \\ &\leq (1+\epsilon_s)\left\|\mathbf{A} - \mathbf{H}_i^k\right\| + \frac{\Delta}{2}.\end{aligned}$$

Since $\left\|\mathbf{A} - \mathbf{H}_i^k\right\|_s < 1.5(1+\epsilon_s)|OPT|$, it follows that $\left\|\mathbf{A} - \mathbf{H}_i^k\right\| < 2|OPT|$, and

$$\epsilon_s\left\|\mathbf{A} - \mathbf{H}_i^k\right\| \leq 2\epsilon_s|OPT| \leq 2\frac{\epsilon}{4B}\frac{B\Delta}{\epsilon} \leq \frac{\Delta}{2},$$

whence

$$\left\|\mathbf{A} - \mathbf{H}_{i+1}^k\right\| \leq \left\|\mathbf{A} - \mathbf{H}_i^k\right\| + \Delta.$$

This proves the first statement.

Now consider second statement. We need to show that discarding some $\mathbf{H}_i^k$'s do not change the B-bucket histograms produced. Suppose $\mathbf{H}_i^k \in \mathcal{B}\backslash\mathcal{B}'$ for some k. Then $\left\|\mathbf{A} - \mathbf{H}_i^k\right\| \geq 1.5|OPT|$. Such a $\mathbf{H}_i^k$ and its extensions will never be useful for a $1+\epsilon$ approximation for $\epsilon < 0.5$; that is, i will never be the minimum j at Step 2 in Definition 8. All other $\mathbf{H}_i^k$'s are retained by Definition 9.

Finally, consider the size of $\mathcal{B}'$. For every two elements we add to $\mathcal{B}$, the error goes up by $\frac{\Delta}{2}$ at least. Thus we cannot have $\ell > 2 \cdot (2|OPT|)/\frac{\Delta}{2}$ elements in $\mathcal{B}'$, since, otherwise, $\left\|\mathbf{A} - \mathbf{H}_\ell^k\right\|$ would be more than $2|OPT|$. Thus the size of $\mathcal{B}'$ will be at most $O(B/\epsilon)$. One can construct $\mathcal{B}'$ directly by using Definition 9 but halting at Step 2 *without* including any additional intervals into the basis. □

5.4 Summary

THEOREM 23. *Fix ϵ, B, and N. Consider a signal $\mathbf{A}$ of length N defined implicitly by updates of the form "add a to $\mathbf{A}_i$," where a can be a positive or negative integer. There is a data structure that supports updates and production, on demand, with high probability over the algorithms random choices, of a B-bucket histogram $\mathbf{H}$ such that $\|\mathbf{A} - \mathbf{H}\| \leq (1+\epsilon)\|\mathbf{A} - \mathbf{H}_{\mathrm{opt}}\|$, where $\mathbf{H}_{\mathrm{opt}}$ is the best possible B-bucket histogram under ℓ_1 or ℓ_2 error $\|\cdot\|$. The data structure requires space $\mathrm{poly}(B\log(N)\log(\|\mathbf{A}\|)/\epsilon)$ and time $\mathrm{poly}(B\log(N)\log(\|\mathbf{A}\|)/\epsilon)$ for each of its operations.*

PROOF. Process updates into an array sketch. Use the array sketch to produce a robust approximation $\mathbf{H}_r$ to $\mathbf{A}$. Use $\mathbf{H}_r$ and the array sketch to produce a k-extension basis, for $k = 0, \ldots, B-1$. Use the $(B-1)$-extension basis to produce a near-optimal histogram representation for $\mathbf{A}$ on $[0, N)$. □

Acknowledgment

We are very grateful to Moni Naor and Omer Reingold for providing a general construction for Lemma 2 and for generously allowing us to include our variant of the construction here. Although published results [6] give a construction for Bernoulli random variables (that can substitute for Gaussian random variables, leading to our result under ℓ_2 error), Naor-Reingold is much more elegent than other constructions. Furthermore, theirs is the *only* known construction for Cauchy random variables, which is needed for our result under ℓ_1 error.

6. REFERENCES

[1] A. Aboulnaga, S. Chaudhuri. Self-tuning Histograms: Building Histograms Without Looking at Data. SIGMOD 1999, 181–192.

[2] N. Alon, Y. Matias, M. Szegedy. The Space Complexity of Approximating the Frequency Moments. JCSS 58(1): 137–147 (1999).

[3] J. Feigenbaum, S. Kannan, M. Strauss, M. Viswanathan. An Approximate L1-Difference Algorithm for Massive Data Streams. FOCS 1999, 501–511.

[4] P. B. Gibbons, Y. Matias. Synopsis Data Structures for Massive Data Sets SODA 1999, 909–910.

[5] P. B. Gibbons, Y. Matias, V. Poosala. Fast Incremental Maintenance of Approximate Histograms. VLDB 1997, 466–475.

[6] A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, M. Strauss. Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries. VLDB 2001, 79–88.

[7] A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, M. Strauss. QuickSAND: Quick Summary and Analysis of Network Data DIMACS Technical Report 2001-43.

[8] S. Guha, N. Koudas. Approximating a Data Stream for Querying and Estimation: Algorithms and Performance Evaluation. ICDE 2002.

[9] S. Guha, N. Koudas, K. Shim. Data-streams and histograms. STOC 2001, 471–475.

[10] P. Indyk. Stable Distributions, Pseudorandom Generators, Embeddings and Data Stream Computation. FOCS 2000, 189–197.

[11] H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. C. Sevcik, T. Suel. Optimal Histograms with Quality Guarantees. VLDB 1998, 275–286.

[12] J.-H. Lee, D.-H. Kim, C.-W. Chung. Multi-dimensional selectivity estimation using compressed histogram information. SIGMOD 1999, 205–214.

[13] Y. Matias, J. S. Vitter, M. Wang. Dynamic Maintenance of Wavelet-Based Histograms. VLDB 2000, 101–110.

[14] M. Naor, O. Reingold. Private communication, March, 1999.

[15] N. Nisan Pseudorandom Generators for Space-Bounded Computation. STOC 1990, 204–212.

[16] V. Poosala. Histograms for selecitivty estimation. PhD Thesis, U. Wisconsin, Madison. 1997.

[17] N. Thaper, S. Guha, P. Indyk, N. Koudas. Dynamic Multidimensional Histograms. SIGMOD 2002.