Wavelet Synopses with Error Guarantees

Minos Garofalakis   Phillip B. Gibbons[*]

Information Sciences Research Center
Bell Labs, Lucent Technologies
Murray Hill, NJ 07974
minos@research.bell-labs.com

ABSTRACT

Recent work has demonstrated the effectiveness of the wavelet decomposition in reducing large amounts of data to compact sets of wavelet coefficients (termed "wavelet synopses") that can be used to provide fast and reasonably accurate approximate answers to queries. A major criticism of such techniques is that unlike, for example, random sampling, conventional wavelet synopses do not provide informative error guarantees on the accuracy of individual approximate answers. In fact, as this paper demonstrates, errors can vary widely (without bound) and unpredictably, even for identical queries on nearly-identical values in distinct parts of the data. This lack of error guarantees severely limits the practicality of traditional wavelets as an approximate query-processing tool, because users have no idea of the quality of any particular approximate answer. In this paper, we introduce Probabilistic Wavelet Synopses, the first wavelet-based data reduction technique with guarantees on the accuracy of individual approximate answers. Whereas earlier approaches rely on deterministic thresholding for selecting a set of "good" wavelet coefficients, our technique is based on a novel, probabilistic thresholding scheme that assigns each coefficient a probability of being retained based on its importance to the reconstruction of individual data values, and then flips coins to select the synopsis. We show how our scheme avoids the above pitfalls of deterministic thresholding, providing highly-accurate answers for individual data values in a data vector. We propose several novel optimization algorithms for tuning our probabilistic thresholding scheme to minimize desired error metrics. Experimental results on real-world and synthetic data sets evaluate these algorithms, and demonstrate the effectiveness of our probabilistic wavelet synopses in providing fast, highly-accurate answers with error guarantees.

1. INTRODUCTION

Approximate query processing has recently emerged as a viable solution for dealing with the huge amounts of data, the high query complexities, and the increasingly stringent response-time requirements that characterize today's Decision Support Systems (DSS) applications. Typically, DSS users pose very complex queries to the underlying Database Management System (DBMS) that require complex operations over gigabytes or terabytes of disk-resident data and, thus, take a very long time to execute to completion and produce exact answers. Due to the *exploratory nature* of many DSS applications, there are a number of scenarios in which an exact answer may not be required, and a user may in fact prefer a fast, approximate answer. For example, during a drill-down query sequence in ad-hoc data mining, initial queries in the sequence frequently have the sole purpose of determining the truly interesting queries and regions of the database. Providing (reasonably accurate) approximate answers to these initial queries gives users the ability to focus their explorations quickly and effectively, without consuming inordinate amounts of valuable system resources. An approximate answer can also provide useful feedback on how well-posed a query is, allowing DSS users to make an informed decision on whether they would like to invest more time and resources to execute their query to completion. Finally, for DSS queries requesting a numerical answer, it is often the case that the full precision of the exact answer is not needed and the first few digits of precision will suffice (e.g., the leading few digits of a total in the millions or the nearest percentile of a percentage).

Wavelets provide a mathematical tool for the hierarchical decomposition of functions, with a long history of successful applications in signal and image processing [12, 18]. Recent studies have also demonstrated the applicability of wavelets to selectivity estimation [14] and to approximate query processing over massive relational tables [4, 19] and data streams [9, 15]. Briefly, the idea is to apply wavelet decomposition to the input relation (attribute column(s) or OLAP cube) to obtain a compact data synopsis that comprises a select small collection of *wavelet coefficients*. The results of Chakrabarti et al. [4] and Vitter and Wang [19] have shown that fast and accurate approximate query processing engines can be designed to operate solely over such compact *wavelet synopses*.

The Problem with Wavelets. A major criticism of wavelet-based techniques for approximate query processing is the fact that, unlike, e.g., random samples, conventional wavelet-coefficient synopses cannot provide guarantees on the error of individual approximate query answers. (Coefficients in the synopsis are typically chosen to optimize an overall error metric, e.g., the L^2 error in the approximation [18].) In fact, even for the simplest case of approximating a value in the original data set, the absolute errors can vary widely (same for the relative errors). Consider the following example:

Original data values	127	71	87	31	59	3	43	99
	100	42	0	58	30	88	72	130
Wavelet answers	65	65	65	65	65	65	65	65
	100	42	0	58	30	88	72	130

Table 1: Errors with Conventional Wavelet Synopses.

The first pair of lines shows the 16 original data values (the exact answer), whereas the second pair shows the 16 approximate answers returned when using conventional wavelet synopses and storing 8 coefficients (details are given in Section 3.1). Although

[*] Current affiliation: Intel Research Pittsburgh, 417 South Craig St., Pittsburgh, PA 15213 *phillip.b.gibbons@intel.com*

ACM SIGMOD '2002 June 4-6, Madison, Wisconsin, USA
Copyright 2002 ACM 1-58113-497-5/02/06 ...$5.00.

the first half of the values is basically a mirror image of the second half, all the approximate answers for the first half are 65, whereas all the approximate answers for the second half are exact! Similar data values have widely different approximations, e.g., 30 and 31 have approximations 30 and 65, respectively. The approximate answers make the first half appear as a uniform distribution, with widely different values, e.g., 3 and 127, having the same approximate answer 65. Moreover, the results do not improve when one considers the presumably easier problem of approximating the sum over a range of values: for *all possible* ranges within the first half involving $x = 2$ to 7 of the values, the approximate answer will be $65 \cdot x$, while the actual answers vary widely. For example, for both the range d0 to d2 and the range d3 to d5, the approximate answer is 195, while the actual answer is 285 and 93, respectively. On the other hand, *exact* answers are provided for all possible ranges within the second half (although the user will not know this).

This illustrates the following serious problems for approximate query processing with wavelet synopses, due to their deterministic approach to selecting coefficients and their lack of error guarantees:

1. The user is provided *no guarantees on the quality of a given wavelet synopsis* for individual answers.
2. The quality of the answers can vary widely, even
   - within the same data set,
   - when the range is the same width, and is large (e.g., almost half the range),
   - when the synopsis is large (e.g., half the data vector size),
   - when the data values in the range are nearly the same, *and*
   - when the actual answer is the same or nearly the same.

   These are circumstances where one might expect approximate answers of similar quality.
3. The user has *no way of knowing whether or not a particular answer is any good.*
4. Moreover, the approximate answers are *biased*, and there is no bound on this bias[1] (in contrast, sampling typically yields unbiased answers: the expected value of the approximate answer is the exact answer).

Our Solution: Probabilistic Wavelet Synopses. In this paper, we propose a novel approach to building wavelet synopses that enables *unbiased* approximate query answers with *error guarantees on the accuracy of individual answers*, thereby mitigating the four serious problems outlined above. Whereas conventional wavelet synopses rely on deterministic thresholding for selecting a set of "good" wavelet coefficients, our technique is based on a novel, probabilistic thresholding scheme that assigns each coefficient a probability of being retained based on its importance to the reconstruction of individual data values, and then flips coins to select the synopsis. Our basic scheme deterministically retains the most important coefficients while randomly rounding the other coefficients either up to a larger value or down to zero. This *randomized rounding* enables unbiased, guaranteed-error reconstruction of individual data values as well as unbiased, guaranteed-error answers for any range aggregate query. The basic scheme is contrasted with an alternative scheme we propose in which coefficients are either selected or not (but never rounded up) according to the assigned probabilities, resulting in low-bias approximate answers but often with tighter error guarantees.

The contributions of this paper are as follows.

1. We provide a quantitative and qualitative demonstration of the unpredictable, widely varying errors arising using conventional wavelet synopses for approximate query answers.
2. We provide the *first* wavelet-based compression technique that provably enables unbiased data reconstruction of a data vector, with error guarantees on individual answers.
3. We provide novel optimization algorithms for tuning our probabilistic thresholding scheme to minimize (a) the expected mean-squared error, and (b) an upper bound on the maximum error in the reconstruction of the data. For reconstruction error, we focus on relative error with a sanity bound (details in Section 3), as this is arguably the most important for approximate query answers. (We can also handle absolute error.)
4. We present a variation on our scheme that allows for reconstruction bias, but selects coefficient probabilities to minimize this bias, often resulting in tighter error guarantees.
5. We describe how our approach for data vectors can be extended for use with multi-dimensional data sets.
6. We demonstrate the effectiveness of our probabilistic wavelet synopses in providing fast, highly-accurate answers with error guarantees, using real-world and synthetic data sets.

There is an extensive literature on wavelets and yet, to the best of our knowledge, this is the first paper to present a wavelet-based compression technique that enables unbiased data reconstruction with error guarantees.

2. WAVELET BASICS

2.1 One-Dimensional Haar Wavelets

Suppose we are given the one-dimensional data vector A containing the $N = 16$ data values depicted in Table 1. The Haar wavelet transform of A can be computed as follows. We first average the values together pairwise to get a new "lower-resolution" representation of the data with the following average values [99, 59, 31, 71, 71, 29, 59, 101]. In other words, the average of the first two values (that is, 127 and 71) is 99, that of the next two values (that is, 87 and 31) is 59, and so on. Obviously, some information has been lost in this averaging process. To be able to restore the original values of the data array, we need to store some *detail coefficients*, that capture the missing information. In Haar wavelets, these detail coefficients are simply the differences of the (second of the) averaged values from the computed pairwise average. Thus, in our simple example, for the first pair of averaged values, the detail coefficient is 28 since $99 - 71 = 28$, for the second we again need to store 28 since $59 - 31 = 28$. Note that no information has been lost in this process – it is fairly simple to reconstruct the sixteen values of the original data array from the lower-resolution array containing the eight averages and the eight detail coefficients. Recursively applying the above pairwise averaging and differencing process on the lower-resolution array containing the averages, we get the following full decomposition:

Resolution	Averages	Detail Coefficients
4	[127, 71, 87, 31, 59, 3, 43, 99, 100, 42, 0, 58, 30, 88, 72, 130]	—
3	[99, 59, 31, 71, 71, 29, 59, 101]	[28, 28, 28, -28, 29, -29, -29, -29]
2	[79, 51, 50, 80]	[20, -20, 21, -21]
1	[65, 65]	[14, -15]
0	[65]	[0]

The *wavelet transform* (also known as the *wavelet decomposition*) of A is the single coefficient representing the overall average of the data values followed by the detail coefficients in the order of increasing resolution. Thus, the one-dimensional Haar wavelet

[1] The *bias* of an estimator $\hat{\Theta}$ for a quantity Θ is $|E[\hat{\Theta}] - \Theta|$. If $E[\hat{\Theta}] = \Theta$, the estimator is *unbiased*. For a deterministic estimator, we will equate the bias with the absolute error $|\hat{\Theta} - \Theta|$.

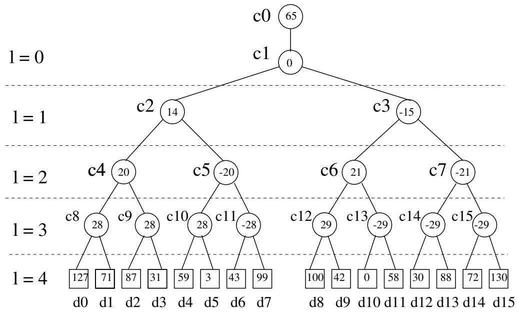

Figure 1: Error tree for our example data array A ($N = 16$).

transform of A is given by W_A = [65, 0, 14, -15, 20, -20, 21, -21, 28, 28, 28, -28, 29, -29, -29, -29]. Each entry in W_A is called a *wavelet coefficient*. The main advantage of using W_A instead of the original data vector A is that for vectors containing similar values most of the detail coefficients tend to have very small values. (However, this is obviously *not* the case with our "bad" example data array.) Thus, eliminating such small coefficients from the wavelet transform (i.e., treating them as zeros) introduces only small errors when reconstructing the original data, giving a very effective form of lossy data compression [4, 18, 19].

Note that, intuitively, wavelet coefficients carry different weights with respect to their importance in rebuilding the original data values. For example, the overall average is obviously more important than any detail coefficient since it affects the reconstruction of all entries in the data array. In order to equalize the importance of all wavelet coefficients, we need to *normalize* the final entries of W_A appropriately. A common normalization scheme (also discussed in Section 2.3) is to divide each wavelet coefficient by $\sqrt{2^l}$, where l denotes the *level of resolution* at which the coefficient appears (with $l = 0$ corresponding to the "coarsest" resolution level, as depicted in the above table). Thus the normalized coefficient, c_i^*, is $c_i/\sqrt{2^{\texttt{level}(c_i)}}$.

Basic Haar Wavelet Properties and Notational Conventions. A helpful tool for exploring and understanding the key properties of the Haar wavelet decomposition is the *error tree* structure [14]. The error tree is a hierarchical structure built based on the wavelet transform process. Figure 1 depicts the error tree for our simple example data vector A. Each internal node c_i ($i = 0, \ldots, 15$) is associated with a wavelet coefficient value, and each leaf d_i ($i = 0, \ldots, 15$) is associated with a value in the original data array; in both cases, the index i denotes the positions in the (data or wavelet transform) array. For example, c_0 corresponds to the overall average of A. Note that the values associated with the error tree nodes c_j are the *unnormalized* coefficient values; the resolution levels l for the coefficients (corresponding to levels in the tree) are also depicted. We use the terms "node" and "node value" interchangeably in what follows. For ease of reference, Table 2 summarizes some of the notation used in this paper with a brief description of its semantics. Detailed definitions of all these parameters are provided at the appropriate locations in the text.

Given an error tree T and an internal node t of T, $t \neq c_0$, we let $\texttt{leftleaves}(t)$ ($\texttt{rightleaves}(t)$) denote the set of leaf (i.e., data) nodes in the subtree rooted at t's left child. Also, given any (internal or leaf) node u, we let $\texttt{path}(u)$ be the set of all (internal) nodes in T that are proper ancestors of u (i.e., the nodes on the path from u to the root of T, including the root but not u) with non-zero coefficients. Finally, for any two leaf nodes d_l and d_h, we let $d(l : h)$ denote the range sum $\sum_{i=l}^{h} d_i$. Using the error tree representation T, we can outline the following important reconstruction properties of the one-dimensional Haar wavelet decomposition [14, 19].

Symbol ($i \in 0..N$-1)	**Semantics**
A, W_A	Input data array and wavelet-transform array
N, N_z	Number of cells (non-zero cells, resp.) in A
B	Target number of coefficients retained
d_i	Data value at cell i of the data array
$\hat{d}_i$	Reconstructed (approximate) data value at cell i
$d(l\!:\!h) = \sum_{i=l}^{h} d_i$	Range sum of data values betw. cells l and h
c_i, c_i^*	Unnormalized / normalized Haar coeff. at cell i
$\texttt{level}(c_i)$	Level of resolution of Haar coefficient c_i
$\texttt{leftleaves}(t)$	Leaves in error subtree rooted at t's left child
$\texttt{rightleaves}(t)$	Leaves in error subtree rooted at t's right child
$\texttt{path}(t)$	All non-zero proper ancestors of t in error tree
T_i	Error subtree rooted at node corr. to c_i
$\texttt{PATHS}_i$	Set of all root-to-leaf paths in T_i
$E[X]$, Var(X)	Expectation and variance of random variable X
$\texttt{WS}_A$, $\lvert\texttt{WS}_A\rvert$	Probabilistic wavelet synopsis for array A, and number of retained coefficients
λ_i	Probabilistic rounding value used for coef. c_i
C_i	Weighted Bernoulli random variable for coeff. c_i
$y_i = c_i/\lambda_i$	"Success" probability for random variable C_i
$\texttt{NSE}(\hat{d}_i)$	Normalized standard error for reconstructed $\hat{d}_i$
$\texttt{VAR}(i, y_i)$	Variance of C_i for a given y_i

Table 2: Notation.

(P1). The reconstruction of any data value d_i depends only on the values of the nodes in path(d_i). More specifically, we have $d_i = \sum_{c_j \in \texttt{path}(d_i)} \delta_{ij} \cdot c_j$, where $\delta_{ij} = +1$ if $d_i \in \texttt{leftleaves}(c_j)$ or $j = 0$, and $\delta_{ij} = -1$ otherwise. For example, in Figure 1, $d_5 = c_0 - c_2 + c_5 - c_{10} = 65 - 14 + (-20) - 28 = 3$.

(P2). An internal node c_j contributes to the range sum $d(l : h)$ only if $c_j \in \texttt{path}(d_l) \cup \texttt{path}(d_h)$. More specifically, $d(l : h) = \sum_{c_j \in \texttt{path}(d_l) \cup \texttt{path}(d_h)} x_j$, where

$$x_j = \begin{cases} (h - l + 1) \cdot c_j, & \text{if } j = 0 \\ (|\texttt{leftleaves}(c_j, l : h)| - \\ \quad |\texttt{rightleaves}(c_j, l : h)|) \cdot c_j, & \text{otherwise.} \end{cases} \qquad (1)$$

where $\texttt{leftleaves}(c_j, l : h) = \texttt{leftleaves}(c_j) \cap \{d_l, d_{l+1}, \ldots, d_h\}$ (i.e., the intersection of $\texttt{leftleaves}(c_j)$ with the summation range) and $\texttt{rightleaves}(c_j, l : h)$ is defined similarly. For example, in Figure 1, $d(3 : 5) = 3c_0 + (1 - 2)c_2 - c_4 + 2c_5 - c_9 + (1 - 1)c_{10} = 93$.

Thus, reconstructing a single data value involves summing at most $\log N{+}1$ coefficients and reconstructing a range sum involves summing at most $2 \log N + 1$ coefficients, regardless of the width of the range.

2.2 Multi-Dimensional Haar Wavelets

The Haar wavelet decomposition can be extended to *multi-dimensional* data arrays using two distinct methods, namely the *standard* and *nonstandard* Haar decomposition [18]. Each of these transforms results from a natural generalization of the one-dimensional decomposition process described above, and both have been used in a wide variety of applications, including approximate query answering over high-dimensional DSS data sets [4, 19].

As in the one-dimensional case, the Haar decomposition of a d-dimensional data array A results in a d-dimensional wavelet-coefficient array W_A with the same dimension ranges and number of entries. Error tree structures for multi-dimensional Haar wavelets can be constructed in a manner similar to those for the one-dimensional case, and properties (P1)-(P2) can be extended to multi-dimensional Haar wavelets. (The full details as well as efficient decomposition algorithms can be found in [4, 19].) Our novel technical results and algorithms rely solely on these key properties of Haar wavelets and, therefore, are applicable for general, multi-dimensional wavelet synopses. However, to simplify the exposition and remain within the specified page constraints, the development in this paper is based primarily on the one-dimensional case, and the extensions to the multi-dimensional case are only sketched briefly in Section 3.6.

2.3 Wavelet-Coefficient Thresholding

Given a limited amount of storage for the *wavelet synopsis* of a data array A, we can only retain a certain number B of the coefficients stored in W_A. (The remaining coefficients are implicitly set to 0.) Letting N_z denote the number of non-zero entries in the data array A, we typically have $B \ll N_z$; that is, the chosen B wavelet coefficients form a highly compressed approximate representation of the original data. The goal of coefficient thresholding is to determine the "best" subset of B coefficients to retain, so that some overall error measure in the approximation is minimized. Conventional coefficient thresholding is a completely deterministic process that typically retains the B largest wavelet coefficients in *absolute normalized value* (an example is given in the next section). It is a well-known fact that, for Haar wavelets, this thresholding method is in fact *provably optimal* with respect to minimizing the overall root-mean-squared error (i.e., *L^2-norm average error*) in the data compression [18]. More formally, letting $\hat{d}_i$ denote the (approximate) reconstructed data value for cell i, retaining the B largest normalized coefficients implies that the resulting synopsis minimizes the quantity $\sqrt{\frac{1}{N}\sum_i (d_i - \hat{d}_i)^2}$ (for the given amount of space B).

3. PROBABILISTIC WAVELET SYNOPSES

In this section, we first detail the problems with conventional wavelet synopses, and then present our probabilistic approach based on randomized rounding. We present three schemes for selecting rounding values, then outline extensions for multidimensional data, and finally, summarize our approach, with an example.

3.1 The Problem with Conventional Wavelets

As discussed above, conventional wavelet synopses retain the B wavelet coefficients with the largest absolute value after normalization (according to level); this deterministic process minimizes the overall L^2 error in reconstructing all the data values. Unfortunately, these guarantees on overall error cannot provide any interesting/non-trivial error guarantees for the approximation of the individual data values or the results of individual range-sum queries.

The example in Figure 1 discussed previously illustrates this failing. Table 3 depicts the wavelet coefficients in the wavelet transform (W_A) for the data array in Figure 1, followed by the level of each coefficient ($\texttt{level}(c_i)$) and the normalized coefficients (c_i^*). With conventional wavelet synopses, we retain the B coefficients c_i with largest $|c_i^*|$. In this example, $|c_0^*|$, $|c_3^*|$, $|c_6^*|$, $|c_7^*|$, $|c_{12}^*|$, $|c_{13}^*|$, $|c_{14}^*|$, and $|c_{15}^*|$ are all greater than 10, while the rest are at most 10. Thus for $B = 8$, the conventional wavelet synopsis is $\{c_0, c_3, c_6, c_7, c_{12}, c_{13}, c_{14}, c_{15}\}$, as shown in the table.

index i	0	1	2	3	4	5	6	7
	8	9	10	11	12	13	14	15
wav. coeff. c_i	65	0	14	-15	20	-20	21	-21
	28	28	28	-28	29	-29	-29	-29
$\texttt{level}(c_i)$	0	0	1	1	2	2	2	2
	3	3	3	3	3	3	3	3
norm. coeff. c_i^*	65	0	$\frac{14}{\sqrt{2}}$	$\frac{-15}{\sqrt{2}}$	10	-10	$\frac{21}{2}$	$\frac{-21}{2}$
	$\frac{14}{\sqrt{2}}$	$\frac{14}{\sqrt{2}}$	$\frac{14}{\sqrt{2}}$	$\frac{-14}{\sqrt{2}}$	$\frac{29}{2\sqrt{2}}$	$\frac{-29}{2\sqrt{2}}$	$\frac{-29}{2\sqrt{2}}$	$\frac{-29}{2\sqrt{2}}$
retained coeff.	65	0	0	-15	0	0	21	-21
	0	0	0	0	29	-29	-29	-29

Table 3: Wavelet Synopsis Using Deterministic Thresholding.

For a given wavelet synopsis, approximate answers are obtained by assuming all non-retained coefficients are zero, and either applying property (P1) from Section 2.1 to estimate individual values or applying property (P2) from Section 2.1 to estimate range sums. For the wavelet synopsis above, Table 1 depicts the (error-prone) individual-value estimates obtained. For example, $d_5 = c_0 - c_2 + c_5 - c_{10}$, and so $\hat{d}_5 = 65 - 0 + 0 - 0 = 65$; since $d_5 = 3$, the estimate has over 2000% relative error! Likewise, $d(3 : 5) = 3c_0 - c_2 - c_4 + 2c_5 - c_9$, and so the estimate for this range sum is $195 - 0 - 0 + 0 - 0 = 195$; since $d(3 : 5) = 93$, the estimate has over a 100% relative error!

The reader may verify that each of the four problems outlined in Section 1 occur in this example, when using a conventional wavelet synopsis. For example, even when the synopsis happens to produce an exact answer to a range sum query that involves the right half of the data values, there is no way of knowing this from the synopsis, because c_1 is not retained.

Moreover, it is not difficult to construct examples with arbitrarily large relative and absolute error. For example, a simple linear transformation of the data array in Figure 1 yields this example:

Original data values	124M	68M	84M	28M	56M	0	40M	96M
	101M	43M	1M	59M	31M	89M	73M	131M
Wavelet transform	64M	-2M	14M	-15M	20M	-20M	21M	-21M
	28M	28M	28M	-28M	29M	-29M	-29M	-29M
Wavelet synopsis	64M	0	0	-15M	0	0	21M	-21M
	0	0	0	0	29M	-29M	-29M	-29M
Wavelet answers	64M	64M	64M	64M	64M	64M	64M	64M
	99M	41M	-1M	57M	29M	87M	71M	129M

Note that the data value d_5 is 0, but its estimate using a conventional wavelet synopsis with $B = 8$ is 64 million, the same as the estimate for $d_0 = 124$ million.

Root Causes. As can be seen from these examples, conventional wavelet synopses suffer from (1) strict deterministic thresholding (i.e., 100% above the threshold are retained, and 0% below the threshold are retained), (2) independent thresholding (i.e., there is no attempt to adapt the thresholding based on what is happening to neighboring coefficients, in order to avoid large regions with no retained coefficients), and (3) the bias resulting from dropping coefficients without compensating for their loss. For example, the retained coefficients (from the right half of the tree in Figure 1) are only slightly larger than the non-retained coefficients (from the left half), and yet *all* the right half coefficients are retained while *none* of the left half coefficients are retained, and there is no attempt to compensate for the resulting bias.

Our approach, outlined next, is to address these three root causes of wavelet synopsis errors, by devising a scheme based on *randomized rounding*[2], using carefully chosen rounding values.

3.2 Our General Approach

We seek to overcome the problems with conventional wavelet synopses outlined thus far by introducing a new approach for building wavelet synopses from wavelet-transform arrays. In a nutshell, our scheme deterministically retains the most important coefficients while randomly rounding the other coefficients either up to a larger value (called a *rounding value*) or down to zero. The

[2] Randomized rounding [16] has been used previously as a technique for obtaining approximate solutions to integer programs [17] and approximate sums of integers in parallel [13]. Our application of randomized rounding is in a completely different domain, requiring novel techniques.

probability of rounding up vs. down is selected so that the expected value of the rounded coefficient equals the original coefficient. By carefully selecting the rounding values, we ensure that (1) we expect a total of B coefficients to be retained, and (2) we minimize a desired error metric in the reconstruction of the data, e.g., the maximum relative error.

The key idea in our thresholding scheme is to associate, with each non-zero coefficient c_i in the wavelet-transform of A, a random variable C_i such that (1) C_i takes the value zero (i.e., c_i is discarded from the synopsis) with some (possibly zero) probability, and (2) $E[C_i] = c_i$. Then $\mathtt{WS}_A$, the probabilistic wavelet synopsis for A, is comprised of the values for those random variables C_i with non-zero values. We determine the general form of these random variables using a randomized rounding scheme, where we select a *rounding value*, λ_i, for each non-zero c_i such that $C_i \in \{0, \lambda_i\}$, $0 < \frac{c_i}{\lambda_i} \leq 1$, and

$$C_i = \begin{cases} \lambda_i & \text{with probability } \frac{c_i}{\lambda_i} \\ 0 & \text{with probability } 1 - \frac{c_i}{\lambda_i} \end{cases}$$

Thus, our proposed thresholding scheme essentially "rounds" each non-zero wavelet coefficient c_i *independently* to either λ_i or zero by flipping a biased coin with success probability $\frac{c_i}{\lambda_i}$. It is easy to see that for this rounding process the expected value of each rounded coefficient is $E[C_i] = \lambda_i \cdot \frac{c_i}{\lambda_i} + 0 \cdot (1 - \frac{c_i}{\lambda_i}) = c_i$ (i.e., the actual coefficient value), and its variance is simply

$$\text{Var}(C_i) = E[C_i^2] - (E[C_i])^2 = \lambda_i^2 \cdot \frac{c_i}{\lambda_i} - c_i^2 = (\lambda_i - c_i) \cdot c_i \quad (2)$$

For the special case where we deterministically retain the coefficient, we set $\lambda_i = c_i$, and indeed $\text{Var}(C_i) = 0$.

Unbiased Estimation. Let $\hat{d}_i$ denote the estimate for the data value d_i, as calculated based on the coefficient values retained in our probabilistic wavelet synopsis $\mathtt{WS}_A$, using property (P1) above. Moreover, let $\hat{d}(l : h)$ ($\hat{d}_{avg}(l : h)$) denote the estimate for the range sum $d(l : h)$ (the range average $d(l : h)/(h - l + 1)$, resp.), as calculated based on the coefficient values retained in $\mathtt{WS}_A$, using property (P2) above. Clearly, $\hat{d}_i$, $\hat{d}(l : h)$, and $\hat{d}_{avg}(l : h)$ are random variables. In the full paper [7] we show (using the two properties and the linearity of expectation):

THEOREM 3.1. *Each of $\hat{d}_i$, $\hat{d}(l : h)$, and $\hat{d}_{avg}(l : h)$ are unbiased estimators for the data value d_i, the range sum $d(l : h)$, and the range average $d(l : h)/(h - l + 1)$, respectively.*

For example, suppose we select $\lambda_0 = c_0$, $\lambda_{10} = 2 \cdot c_{10}$, and $\lambda_i = \frac{3c_i}{2}$ for all other non-zero coefficients c_i in Figure 1. Let $y_i = \frac{c_i}{\lambda_i}$ be the probability of rounding up. Then $E[\hat{d}_5] = E[C_0] - E[C_2] + E[C_5] - E[C_{10}] = y_0\lambda_0 - y_2\lambda_2 + y_5\lambda_5 - y_{10}\lambda_{10} = 65 - \frac{2}{3} \cdot 21 + \frac{2}{3} \cdot (-30) - \frac{1}{2} \cdot 56 = 65 - 14 - 20 - 28 = 3$, which is exactly d_3. Likewise, $E[\hat{d}(3 : 5)] = 3 \cdot E[C_0] - E[C_2] - E[C_4] + 2 \cdot E[C_5] - E[C_9] = 3 \cdot 65 - \frac{2}{3} \cdot 21 - \frac{2}{3} \cdot 30 + 2 \cdot \frac{2}{3} \cdot (-30) - \frac{2}{3} \cdot 42 = 195 - 14 - 20 - 40 - 28 = 93$, which is exactly $d(3 : 5)$.

The Impact of the λ_i's. Everything thus far holds for *any* choice of λ_i's, as long as $0 < \frac{c_i}{\lambda_i} \leq 1$. The choice of the λ_i's is crucial, however, because it determines the variances of our estimators as well as the expected number of coefficients retained. Indeed, the key to providing "good" error guarantees for individual data values (for range sums) lies in selecting the λ_i's to ensure small variances $\text{Var}(\hat{d}_j)$ of the reconstructed data values (data paths, resp.) while not exceeding the prescribed space limit for the synopsis. Because each coefficient is rounded independently, we have by Equation 2:

$$\begin{aligned} \text{Var}(\hat{d}_j) &= \text{Var}(\sum_{c_i \in \mathtt{path}(d_j)} \delta_{ji} \cdot C_i) = \sum_{c_i \in \mathtt{path}(d_j)} (\delta_{ji})^2 \cdot \text{Var}(C_i) \\ &= \sum_{c_i \in \mathtt{path}(d_j)} (\lambda_i - c_i) \cdot c_i \quad (3) \end{aligned}$$

Thus, having a λ_i closer to c_i reduces the variance. On the other hand, we retain all non-zero coefficients after the rounding step, and $|\mathtt{WS}_A|$, the number of non-zero coefficients after rounding, is a random variable such that:

$$E[|\mathtt{WS}_A|] = \sum_{i | c_i \neq 0} \frac{c_i}{\lambda_i} \quad (4)$$

Thus, having λ_i's further from their respective c_i's reduces the expected number of retained coefficients. For a given target B on the number of retained coefficients, our choice of λ_i's needs to ensure that $E[|\mathtt{WS}_A|] \leq B$.

3.3 Rounding to Minimize the Expected Mean-Squared Error

A reasonable approach is to select the λ_i values in a way that minimizes some overall error metric (e.g., L^2) in the approximation. Because such error metrics are obviously random variables under our probabilistic methodology, we seek to minimize their *expectation*. The theorem below follows from the orthogonality properties of the Haar basis [7].

THEOREM 3.2. *For any choice of λ_i's for the non-zero c_i's such that $0 < \frac{c_i}{\lambda_i} \leq 1$, the expected value of the overall L^2 error in reconstructing the data values from a probabilistic wavelet synopsis, $E[L^2] = E[\sum_j (\hat{d}_j - d_j)^2]$, is $\sum_{i|c_i \neq 0} \frac{Var(C_i)}{2^{\mathtt{level}(c_i)}} = \sum_{i|c_i \neq 0} \frac{(\lambda_i - c_i) \cdot c_i}{2^{\mathtt{level}(c_i)}}$.*

Theorem 3.2 shows that the variance of the non-zero coefficients at lower levels of resolution (that is, closer to the root of the error tree) has a higher impact on the overall L^2 error. This is a very intuitive result since, by virtue of the Haar decomposition, such coefficients contribute to the reconstruction of a larger number of data values. Based on Theorem 3.2, selecting the rounding values λ_i to minimize the expected L^2 error subject to a given expected space constraint[3] B can be formally stated as the following optimization problem:

> **[Expected L^2 Error Minimization]** Find the rounding values λ_i that *minimize* the expected L^2 error $\sum_{i|c_i \neq 0} \frac{(\lambda_i - c_i) \cdot c_i}{2^{\mathtt{level}(c_i)}}$, subject to the constraints $0 < \frac{c_i}{\lambda_i} \leq 1$ for all non-zero c_i and $E[|\mathtt{WS}_A|] = \sum_{i|c_i \neq 0} \frac{c_i}{\lambda_i} \leq B$. ▮

An Optimal Algorithm for Computing the λ_i Values. The above-stated problem is a continuous, non-linear optimization problem with an objective function that is *convex* in the problem variables. In general, such *convex programming* problems are solved using computationally-intensive numerical methods (e.g., Sequential Quadratic Programming (SQP) or interior-point methods), that are typically not intended to scale beyond a few thousand variables [8].

Fortunately, the specific form of our L^2 error minimization problem allows us to derive an efficient optimal algorithm for computing the rounding values λ_i. More specifically, letting $y_i = \frac{c_i}{\lambda_i}$

[3] For simplicity, we discuss *expected* space constraints in this paper, although one can ensure we are within an absolute space bound B^*, by targeting an expectation slightly less than B^* and possibly repeating the coin tossing a few times until $|\mathtt{WS}_A| \leq B^*$. In our experiments comparing wavelet approaches, we ensure that all synopses have the same number of coefficients.

(i.e., the "success" probability for random variable C_i) and $k_i = \frac{c_i^2}{2^{\mathtt{level}(c_i)}}$, it is easy to see that our expected L^2 error minimization problem is equivalent to:

$$\text{Minimize} \sum_{i|c_i \neq 0} \frac{k_i}{y_i} \text{ subject to } \sum_{i|c_i \neq 0} y_i \leq B \text{ and } y_i \in (0,1].$$

(Note that terms involving only c_i's are constant in our minimization problem, and hence safely ignored.) Based on the Cauchy-Schwarz inequality, the minimum value of the objective is reached when $\frac{\sqrt{k_i}}{y_i}$ is the same for all i with $c_i \neq 0$. Let $w = \frac{\sqrt{k_i}}{y_i}$, so that we require $\sum_{i|c_i \neq 0} y_i = \sum_{i|c_i \neq 0} \frac{\sqrt{k_i}}{w} \leq B$. It is not hard to see that using exactly B is better than using less than B, so we get $w = \frac{1}{B} \sum_i \sqrt{k_i}$ and hence $y_i = \frac{B \cdot \sqrt{k_i}}{\sum_i \sqrt{k_i}}$, for all i with non-zero c_i. Thus, setting

$$\lambda_i = \frac{c_i}{y_i} = \frac{c_i \cdot \sum_i \sqrt{k_i}}{B \cdot \sqrt{k_i}}, \tag{5}$$

for all i with non-zero c_i, is the optimal solution, ignoring the second constraint that $y_i \in (0,1]$.

This leads to our MinL2 algorithm, which we sketch here. We first compute the $\sqrt{k_i}$'s and their sum. We would like to apply Equation 5 to set the rounding values, but this may result in one or more $y_i > 1$. Thus we consider the indices i in non-increasing order of $\sqrt{k_i}$. While $y_i \geq 1$, we set λ_i to c_i, so that $y_i = 1$, and then loop to recurse on the subproblem without i. Once we encounter a $y_i < 1$, we are guaranteed that all remaining y_i are less than 1 as well, and we can safely apply Equation 5 to set the rounding values. Using convexity arguments, it can be shown that MinL2 produces the optimal solution to our optimization problem. Our algorithm runs in linear time, plus the time for sorting (a total of $O(N \log N)$ time). It uses $O(N)$ total storage space. Using the rounding values produced by the algorithm results in *unbiased* approximate answers for individual values and for ranges, with the minimum expected mean-squared error over all individual values.

3.4 Rounding to Minimize the Maximum Relative Error

In the previous section, we described how to obtain unbiased approximate answers while minimizing an *overall* error metric. However, there were no error guarantees for *individual* answers, and indeed there can be wide discrepancies in the accuracy of reconstructed values. (Recall from Section 3.1 that conventional wavelet synopses suffer from this same problem, as well as from bias.)

In this section, we present techniques for minimizing the maximum reconstruction error for individual data values and ranges. Thus, we will achieve our goal of providing *guarantees* on the accuracy of each reconstructed value. We focus on minimizing the *relative error*, and only briefly describe our approach for *absolute error* at the end of the section. Although minimizing absolute error is somewhat easier to achieve, we believe that minimizing relative error is more important to approximate query answering.[4] On the other hand, relative error is unduly dominated by small data values (e.g., for a data value = 3, returning 2 as the reconstructed value is a 50% relative error!), so various techniques have been studied for combining absolute and relative error (see, e.g., [10, 19]). In this paper, we study an error metric that combines relative error with a *sanity bound* S. Our goal is to produce estimates $\hat{d}_i$ for each data value d_i such that

$$|\hat{d}_i - d_i| \leq \epsilon \cdot \max\{|d_i|, \mathrm{S}\} \tag{6}$$

for a given sanity bound $\mathrm{S} > 0$, where the relative error bound $\epsilon > 0$ is minimized, subject to the prescribed space limit for the synopsis.

In the context of our probabilistic wavelet synopses, we know that the expected value of $\hat{d}_i$ is d_i, and we would like to minimize the variance, in order to make it highly likely that Equation 6 is satisfied. More precisely, we seek to minimize the *normalized standard error* for a reconstructed data value:

$$\mathrm{NSE}(\hat{d}_i) = \frac{\sqrt{\mathrm{Var}(\hat{d}_i)}}{\max\{|d_i|, \mathrm{S}\}} \tag{7}$$

Note that by applying Chebyshev's Inequality, we obtain (for all $\alpha > 1$)

$$\Pr\left(|\hat{d}_i - d_i| > \alpha \cdot \mathrm{NSE}(\hat{d}_i) \cdot \max\{|d_i|, \mathrm{S}\}\right) < \frac{1}{\alpha^2}, \tag{8}$$

so that minimizing NSE will indeed minimize the probabilistic bounds on our relative error metric.

Based on the earlier development, letting $\mathrm{PATHS} = \{\mathtt{path}(d_i) : i = 1, \ldots, N\}$ (i.e., the set of all root-to-leaf paths in the error tree, where paths again ignore both data value nodes and nodes whose coefficient is zero), and applying Equation 3, we can define our maximum NSE minimization problem as follows.

[Maximum Normalized Standard Error Minimization] Find the rounding values λ_i that *minimize* the maximum NSE for each reconstructed data value; that is,

$$\text{Minimize} \max_{\mathtt{path}(d_k) \,\in\, \mathrm{PATHS}} \frac{\sqrt{\sum_{i \,\in\, \mathtt{path}(d_k)} (\lambda_i - c_i) \cdot c_i}}{\max\{|d_k|, \mathrm{S}\}} \tag{9}$$

subject to the constraints $0 < \frac{c_i}{\lambda_i} \leq 1$ for all non-zero c_i and $E[|\mathrm{WS}_A|] = \sum_{i|c_i \neq 0} \frac{c_i}{\lambda_i} \leq B$. ▮

Our solution to the maximum NSE minimization problem relies on applying four key technical ideas, which we will describe throughout this section. The first is *Exploiting the Error-Tree Structure for Coefficients:* As explained above, the variance for the reconstruction of individual data values is computed by summing the contributions of independent random variables C_i along all root-to-leaf paths in the error-tree structure for Haar wavelet coefficients; our algorithm takes advantage of this *hierarchical* problem structure to efficiently explore the solution space using *dynamic programming*, as discussed next.

Formulating a Dynamic Programming Recurrence. We would like to formulate a dynamic programming recurrence for this problem. Accordingly, we first simplify Equation 9 by squaring the main ($\mathrm{NSE}(\hat{d}_k)$) term; this does not alter the optimality of any solution. As before, we let $y_i = \frac{c_i}{\lambda_i}$, where $y_i \in (0,1]$ for a non-zero coefficient c_i. Then, let $\mathrm{VAR}(i, y_i) = (\lambda_i - c_i)c_i = \frac{1-y_i}{y_i} \cdot c_i^2$ denote the variance of C_i (the random variable associated with c_i) for the given y_i. Let T_j be the subtree of the error-tree rooted at the node corresponding to coefficient c_j. Let PATHS_j denote the set of all root-to-leaf paths in T_j. Finally, let $M[j, B]$ denote the optimal (i.e., minimum) value of the maximum $\mathrm{NSE}(\hat{d}_k)^2$ among all data values d_k in T_j assuming a space budget of B; that is,

$$M[j, B] = \min_{\substack{y_i \in (0,1], i \in T_j | c_i \neq 0: \\ \sum_{i \in T_j | c_i \neq 0} y_i \leq B}} \left\{ \max_{\substack{\mathtt{path}(d_k) \\ \in \mathrm{PATHS}_j}} \left\{ \sum_{i \in \mathtt{path}(d_k)} \frac{\mathrm{VAR}(i, y_i)}{\max\{d_k^2, \mathrm{S}^2\}} \right\} \right\} \tag{10}$$

[4] For example, when (exact) answers can vary by orders of magnitude, it is often preferable to have each answer be within say 1% relative error than have each answer be within the same absolute error, because the same absolute error may correspond to orders of magnitude differences in relative error (say, .1% to 100%).

$$M[j,B] = \begin{cases} \min\limits_{\substack{y_j \in (0,\min\{1,B\}];\\ b_L \in [0, B-y_j]}} \left\{ \max\left\{ \begin{array}{l} \frac{\mathrm{VAR}(j,y_j)}{\mathrm{NORM}(2j)} + M[2j, b_L]\,, \\ \frac{\mathrm{VAR}(j,y_j)}{\mathrm{NORM}(2j+1)} + M[2j+1, B-y_j-b_L] \end{array} \right\} \right\} & \text{if } j < N, c_j \neq 0, \\ & \quad \text{and } B > 0 \\ \min_{b_L \in [0,B]} \; \{ \max\{ \, M[2j, b_L]\,, \; M[2j+1, B-b_L] \, \} \, \} & \text{if } j < N \text{ and } c_j = 0 \\ 0 & \text{if } j \geq N \\ \infty & \text{otherwise} \end{cases} \tag{11}$$

$$M[j,B] = \min_{\substack{y_j \in (0,\min\{1,B\}];\\ b_L \in [0,B-y_j]}} \left\{ \max \left\{ \begin{array}{l} \max_{\mathtt{path}(d_L) \in \mathtt{PATHS}_{2j}} \left\{ \frac{\mathrm{VAR}(j,y_j)}{\max\{d_L^2,\mathrm{S}^2\}} + \sum_{i \in \mathtt{path}(d_L)} \frac{\mathrm{VAR}(i,y_i(b_L))}{\max\{d_L^2,\mathrm{S}^2\}} \right\}, \\ \max_{\mathtt{path}(d_R) \in \mathtt{PATHS}_{2j+1}} \left\{ \frac{\mathrm{VAR}(j,y_j)}{\max\{d_R^2,\mathrm{S}^2\}} + \sum_{i \in \mathtt{path}(d_R)} \frac{\mathrm{VAR}(i,y_i(B-y_j-b_L))}{\max\{d_R^2,\mathrm{S}^2\}} \right\} \end{array} \right\} \right\} \tag{12}$$

Consider the recurrence for $M[j,B]$ depicted in Equation 11 (omitting the $j=0$ case), where $\mathrm{NORM}(i) = \max\{\min_{k \in T_i}\{d_k^2\}, \mathrm{S}^2\}$ is the normalization term for subtree T_i. Note that the indices $2j$ and $2j+1$ in this recurrence correspond to the left and right child (respectively) of node j in the error-tree structure (Figure 1). Intuitively, Equation 11 states that, given a space budget of B for the subtree rooted at node j, the optimal solution for the y_i's and the corresponding cost (maximum NSE^2) needs to minimize the larger of the cost for paths via j's two child subtrees (including the root in all paths), where the cost for a path via a subtree is the sum of: (1) the variance penalty incurred at node j itself, assuming a setting of y_j, divided by the normalization term for that subtree, and (2) the optimal cost for the subtree, assuming the given space budget. This minimization, of course, is over all possible values of y_j and, given a setting of y_j, over all possible allotments of the remaining $B - y_j$ budget "units" amongst the two child subtrees of node j. Of course, if c_j is zero then none of our space budget needs to be allocated to node j, which results in the simpler recurrence in the second clause of Equation 11. Finally, data nodes ($j \geq N$) cost no space and incur no cost, and the "otherwise" clause handles the case where we have a non-zero coefficient but zero budget ($c_j \neq 0$ and $B = 0$).

Equation 11 is a significant simplification of Equation 10, so why might it be correct? Consider a node with coefficient $c_j \neq 0$. Starting with Equation 10 and separating the root from the rest of the path, we can rewrite the expression for $M[j,B]$ as shown in Equation 12, where the $y_i(b)$'s are the optimal choice for y_i's given space b. To proceed from here, we need to apply a second key technical idea: *Exploiting Properties of Optimal Solutions*. Consider the following claim:

CLAIM 3.1. *Consider a subtree T_i and a space budget $B > 0$. Then in an optimal setting of the y_i's, a min data value d in T_i has the maximum cost, i.e.,* $\mathrm{NSE}(\hat{d})^2 = M[i,B]$.

Assuming for now that Claim 3.1 is correct, we have, for all $d_L \in T_{2j}$: $\frac{\mathrm{VAR}(j,y_j)}{\max\{d^2,\mathrm{S}^2\}} = \frac{\mathrm{VAR}(j,y_j)}{\mathrm{NORM}(2j)} \geq \frac{\mathrm{VAR}(j,y_j)}{\max\{d_L^2,\mathrm{S}^2\}}$, and $\sum_{i \in \mathtt{path}(d)} \frac{\mathrm{VAR}(i,y_i(b_L))}{\max\{d^2,\mathrm{S}^2\}} = M[2j,b_L] \geq \sum_{i \in \mathtt{path}(d_L)} \frac{\mathrm{VAR}(i,y_i(b_L))}{\max\{d_L^2,\mathrm{S}^2\}}$. Thus, the maximum cost path via the left subtree is $\mathtt{path}(d)$, and its cost is $\frac{\mathrm{VAR}(j,y_j)}{\mathrm{NORM}(2j)} + M[2j,b_L]$. By a similar argument for the right subtree, it follows that Equation 11 is correct, assuming Claim 3.1 is correct.

Unfortunately, there are corner cases which make Claim 3.1 false![5] Thus we need to apply a third key technical idea: *Perturbing Coefficients to Avoid Harmful Corner Cases.* We can ensure these corner case do not occur by "perturbing" certain zero coefficients by a very small perturbation amount Δ, making them either $+\Delta$ or $-\Delta$ with equal probability (to ensure no reconstruction bias is introduced):

> **[Perturbation Rule]** For each subtree T_i such that (1) one of its child subtrees, say T_{2i}, has all zero coefficients, (2) its other child subtree, T_{2i+1}, has at least one non-zero coefficient, and (3) the min data value in T_{2i} is less than the min data value in T_{2i+1}, then perturb c_{2i}.

Note that each data value has at most one perturbed coefficient on its path, so the effect of this perturbation is minimal. We determined experimentally that a good choice for Δ is $\min\{0.01, \frac{\mathrm{S}}{100}\}$. In the full paper, we prove:

THEOREM 3.1. *Claim 3.1 holds for any error-tree after applying the perturbation rule. Thus, for such trees, Equation 11 correctly computes the optimal $M[j,B]$, as defined in Equation 10. Hence, computing $M[0,B]$ using Equation 11, saving the optimal choices for y_j and b_L at each step, yields the optimal y_i's (and hence λ_i's) that solve the Maximum Normalized Standard Error Minimization problem.*

The problem with the recurrence in Equation 11 is that the y_j and b_L each range over a continuous interval, making it infeasible to use. Thus, rather than insisting on an exact solution, we instead propose an efficient approximation algorithm that produces near-optimal solutions to our maximum NSE minimization problem.

An Efficient Approximation Algorithm for Minimizing the Relative Error. The fourth and final key technical idea applied in our dynamic programming algorithm is to *Quantize the Solution Space:* Instead of allowing the y_i variables to vary continuously over $(0,1]$, we assume that these variables take values from a discrete set of q choices corresponding to integer multiples of $1/\mathrm{q}$, where q is an input integer parameter to our algorithm; that is, we modify the constraint $y_i \in (0,1]$ to $y_i \in \{\frac{1}{\mathrm{q}}, \frac{2}{\mathrm{q}}, \ldots, 1\}$, and the constraint $b_L \in [0,B]$ to $b_L \in \{0, \frac{1}{\mathrm{q}}, \ldots, B\}$. Obviously, our approximate solution converges to the optimal solution for the maximum NSE minimization problem as q becomes larger. On the other hand, as we show below, larger values for q also imply higher running-time and memory requirements for our algorithm. Thus, the input "quantizing" parameter q provides a convenient "knob" for tuning the tradeoff between resource requirements and solution quality for our approximate dynamic-programming algorithm.

The pseudo-code for our algorithm (termed MinRelVar) is based on the above quantization of the recurrence in Equation 11, and is depicted in Figure 2. The initial invocation of this recursive algorithm is done with $\mathtt{root} = 0$ and B equal to the total (expected) number of coefficients to be retained in the wavelet synopsis.[6]

[5] Details are in the full paper. In a nutshell, the problem arises in subtrees T_i described below in the "perturbation rule". In such corner cases, it is in general not possible to make the min data value in T_i have the maximum cost while minimizing the maximum cost. In the worst case, this problem compounds itself up the tree, destroying any hope of developing a fast dynamic-programming algorithm.

[6] Note that node 0, corresponding to the overall average, should be handled as a special case, since it only has one child in the error tree (Figure 1); the modifications required are straightforward and omitted from the description for clarity.

```
procedure MinRelVar( $W_A$ , $B$ , q, root )
Input: Array $W_A = [c_0, \ldots, c_{N-1}]$ of $N$ Haar wavelet coefficients,
    space budget $B$ (expected number of retained coefficients),
    quantizing parameter $\mathtt{q} \geq 1$, error-subtree root-node index root.
Output: Value of $M[\mathtt{root}, B]$ according to the quantized version of
    Eqn. 11 ($M[\mathtt{root}, B]$.value), with the optimal space allotments for the
    root ($M[\mathtt{root}, B]$.yValue) & left child subtree ($M[\mathtt{root}, B]$.leftAllot).
Note: We assume that prior to the initial invocation, the perturbation rule
    has been applied, using a given perturbation value parameter $\Delta$. Also,
    both $\textsc{norm}(i)$, using a given sanity bound parameter S, and $NZ[i]$, the
    number of non-zero coefficients in the subtree rooted at $i$, have been
    computed for all $i$. Finally, $M[i, B]$.computed is initialized to false, for
    all $i$ and all $B = 0, \frac{1}{\mathtt{q}}, \ldots, B^*$, where $B^*$ is the root's space budget.
begin
1.  if ($\mathtt{root} > N - 1$ or $NZ[\mathtt{root}] \leq B$) then
2.      return 0            // no nodes left or all coeffs can be retained
3.  if ($NZ[\mathtt{root}] > B * \mathtt{q}$) then
4.      return $\infty$            // not enough space even for minimal allocations
5.  if ($M[\mathtt{root}, B]$.computed = true) then
6.      return $M[\mathtt{root}, B]$.value     // optimal value already in $M[]$
7.  $M[\mathtt{root}, B]$.value := $\infty$
8.  for $l$ := 1 to q do
9.      if ($c_{\mathtt{root}}$ = 0) then
10.         rootLeft := rootRight := rootSpace := 0
11.     else
12.         rootLeft := $(\mathtt{q} - l) * c^2_{\mathtt{root}} / (l * \textsc{norm}(2 * j))$
13.         rootRight := $(\mathtt{q} - l) * c^2_{\mathtt{root}} / (l * \textsc{norm}(2 * j + 1))$
14.         rootSpace := $l/\mathtt{q}$
15.     endif
16.     for $b$ := 0 to $B-$rootSpace step $1/\mathtt{q}$ do
17.         L := MinRelVar( $W_A$, $b$, q, $2 * \mathtt{root}$ )
18.         R := MinRelVar( $W_A$, $B-$rootSpace$-b$, q, $2 * \mathtt{root} + 1$ )
19.         if ( max{rootLeft+L , rootRight+R} $< M[\mathtt{root}, B]$.value ) then
20.             $M[\mathtt{root}, B]$.value := max{ rootLeft+L , rootRight+R }
21.             $M[\mathtt{root}, B]$.yValue := rootSpace
22.             $M[\mathtt{root}, B]$.leftAllot := b
23.         endif
24.     endfor
25.     if ($c_{\mathtt{root}}$ = 0) then break        // no need to iterate over multiple $l$
26. endfor
27. $M[\mathtt{root}, B]$.computed := true
28. return $M[\mathtt{root}, B]$.value
end
```

Figure 2: The MinRelVar Algorithm: Rounding to Minimize Maximum Normalized Standard Error.

Time/Space Complexity and the Quantizing Parameter q**.** Given a node/coefficient j in the error tree and a budget B, MinRelVar computes the optimal value by examining q possible space allotments for j (Step 8) and, for each of these, a maximum of $O(\mathtt{q} \cdot B)$ ways of distributing the remaining budget among the two children of node j (Step 16). (Once an optimal value is computed it is recorded in the dynamic-programming array for future reference.) Thus, the overall running-time complexity of algorithm MinRelVar is $O(N\mathtt{q}^2 B)$. Moreover, typically the running time will be faster due to the considerable pruning during the search. With respect to the space requirements of our algorithm, note that, even though the size of the full dynamic-programming array M is $O(N\mathtt{q}B)$, MinRelVar does not require the entire array to be memory resident. In fact, it is easy to verify that, at any given point during the execution of MinRelVar, there will be at most one active "line" (of size $O(\mathtt{q}B)$) of array M per level of the error tree. This is because the results for all descendants of a node j can be swapped out once the results for j have been computed. Thus, the memory resident working set size is only $O(\mathtt{q}B \log N)$.

An Optimization. Recall that for nonzero coefficients, we select a $y_i \in \{\frac{1}{\mathtt{q}}, \frac{2}{\mathtt{q}}, \ldots, 1\}$. Because it is often useful to permit very small y_i's for unimportant coefficients, this forces a larger choice for q, in order to make the smallest value, $\frac{1}{\mathtt{q}}$, be sufficiently small. Instead, we propose the following simple optimization: allow $y_i = 0$ even for nonzero coefficients. The problem, of course, is that then the variance, $\textsc{var}(i, y_i) = \frac{1-y_i}{y_i} \cdot c_i^2$, for this coefficient is infinite, and hence this choice for y_i will never be selected as the optimal choice by the MinRelVar algorithm. To get around this problem, we observe that because the coefficient is always rounded down when $y_i = 0$, its contribution to the squared error of any data value in its subtree is c_i^2, not infinite. Thus, we can set $\textsc{var}(i, 0) = c_i^2$ in our dynamic programming algorithm. The downside of this optimization is that it introduces a small bias in the reconstruction. On balance, however, it leads to a faster algorithm (because a larger q can be used) and highly-accurate answers (see Section 4).

Minimizing the Maximum Absolute Error. Minimizing the maximum absolute error is essentially equivalent to minimizing the *maximum total variance across all root-to-leaf paths* in the error tree of the wavelet decomposition. In the full paper, we formulate the optimization problem of rounding values in a probabilistic wavelet synopsis to minimize the maximum reconstruction variance. Our formulation gives rise to a non-linear, convex programming problem that is significantly more complex than that of minimizing the expected L^2 error (Section 3.3); thus, it is unlikely that efficient, specialized solution procedures exist [8]. Given the shortcomings of generic convex-program solvers, we present an efficient approximate algorithm for the problem based on dynamic programming that runs in $O(N\mathtt{q}^2B)$ time and $O(N + \mathtt{q}B \log^2 N)$ space.

3.5 Low-Bias Probabilistic Wavelet Synopses

A key feature of our probabilistic wavelet synopses is their use of randomized rounding, in order to achieve unbiased, guaranteed-error reconstruction of data vectors as well as unbiased, guaranteed-error answers for range queries. In this section, we propose an alternative scheme that does not perform randomized rounding. Instead, each coefficient is either retained or discarded, according to the probabilities y_i, where as before the y_i's are selected to minimize a desired error metric.

The high-level approach is the same as before, except that now there is a random variable, C_i, associated with each coefficient that is c_i with probability y_i, and 0 with probability $1 - y_i$. Clearly, C_i is no longer an unbiased estimator for c_i, so this scheme introduces bias into the reconstruction of data values. To combat this, we select y_i's to minimize the maximum normalized bias for a reconstructed data value, where the *normalized bias* for a data value d_k is $\frac{\sum_{i \,\in\, \mathtt{path}(d_k)} |c_i| \cdot (1-y_i)}{\max\{|d_k|, \mathtt{S}\}}$. To see why the $|c_i| \cdot (1 - y_i)$ term above makes sense, observe that $E[C_i] = c_i \cdot y_i$. It follows that each C_i contributes either plus or minus $c_i(1 - y_i)$ to the expected reconstruction bias of all data values in its subtree. Thus, $|c_i| \cdot (1 - y_i)$ upper bounds the expected contribution of this coefficient to the reconstruction bias.

We define our maximum normalized bias minimization problem as follows.

[Maximum Normalized Bias Minimization] Find the y_i's that *minimize* the maximum normalized bias for each reconstructed data value; that is,

$$\text{Minimize} \quad \max_{\mathtt{path}(d_k) \,\in\, \mathtt{PATHS}} \frac{\sum_{i \,\in\, \mathtt{path}(d_k)} |c_i| \cdot (1 - y_i)}{\max\{|d_k|, \mathtt{S}\}}$$

subject to the constraints $0 \leq y_i \leq 1$ for all i, and $E[|\mathtt{WS}_A|] = \sum_i y_i \leq B$. ∎

We can readily adapt the combinatorial solution of the previous section to solve this minimization problem. Details are in the full paper. We will refer to this algorithm as the MinRelBias algorithm. Intuitively, this approach has more in common with traditional wavelet synopses, in that we either retain or discard a coefficient as is. However, due to the randomization and our choice of optimization metric, we still avoid the four pitfalls of traditional wavelet synopses outlined in Section 1. In fact, as shown in Section 4, the MinRelBias algorithm produces probabilistic wavelet synopses that yield significantly more accurate answers than traditional wavelet synopses, and often outperform our other approaches.

3.6 Extension to Multi-Dimensional Wavelets

In this section, we sketch the key ideas for extending our randomized rounding approach to multi-dimensional data. As observed in [4], a nice feature of the nonstandard Haar decomposition is that the multi-dimensional wavelet-transform array W_A can be computed in one pass over the (suitably-ordered) data array, with coefficients computed bottom up from the leaves of the error tree. Unfortunately, this pass may use far more than N_z space, where N_z is the number of non-zeros in the data array, because W_A can have far more than N_z non-zeros. Thus, as in [19], we perform coefficient thresholding during the computation of W_A, to ensure that we retain at most N_z (non-zero) coefficients. We developed a new thresholding technique that performs this initial thresholding *without introducing any reconstruction bias* (previous schemes did not limit the bias they introduced). Namely, in the one pass over the data array, we retain all coefficients computed until we reach our limit of N_z coefficients. At that point, we need to create room for more coefficients, so we select a threshold λ such that for say 20% of the coefficients c_i computed thus far, $|c_i| \leq \lambda/2$. Then we flip coins, rounding each such c_i up to λ (or -λ, if $c_i < 0$) with probability $\frac{|c_i|}{\lambda}$ and down to zero with probability $1 - \frac{|c_i|}{\lambda}$. We expect to discard at least half of these coefficients, reducing the space by between 10% and 20%. Moreover, this does *not* introduce any reconstruction bias, because the expected value for each coefficient is still c_i. We then continue with the pass over the data array, repeating this process whenever we accumulate N_z coefficients. This results in a probabilistic wavelet synopsis $\texttt{WS}_A$ with no more than N_z coefficients (and at least $\frac{4}{5}N_z$ coefficients).

The above captures the main idea, but the actual thresholding process we use is more complicated. First, we use $|c_i^*|$, the magnitude of the normalized coefficients, to select the pool of coefficients subject to coin flips. Second, each newly-computed coefficient that would have been subjected to coin flips under the most recent thresholding, is subjected to this thresholding. Thus at any point, all coefficients have been treated equally, independent of when they were computed. Third, each thresholding step with a new λ accounts for the previous thresholding steps in order to maintain unbiased answers and equal treatment. Fourth, since all coefficients are treated equally, we need not store the rounded value for retained coefficients, only the original value. This property is used in the phase described next.

As before, our goal is to have only $B \lll N_z$ coefficients. This we accomplish by applying our MinRelVar scheme to select the y_i, and then flipping coins to reduce $\texttt{WS}_A$ to B coefficients. We adapt the algorithm to account for the variance already incurred by rounded up values; this variance can be computed because we have the original c_i. (We cannot account for the variance due to rounded down values, due to our space limit of N_z.) More significantly, we extend MinRelVar to the multi-dimensional case. We argue that a modified dynamic programming formulation still holds for the multi-dimensional case, and that the running time for the quantized version is $O(N_z \mathtt{q}^2 B 2^d)$, where $\mathtt{q}$ is the quantization parameter, B is the space bound, and d is the number of dimensions. This is the first wavelet-based compression technique for multi-dimensional data that provably enables unbiased reconstruction of data values and unbiased answers to range queries.

i	0	1	2	5	11	22	45
c_i	204	-6	-4	20	4	-1	0
y_i	1	$\frac{2}{3}$	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{6}$	–
λ_i	204	-9	-8	20	8	-6	$\perp$
coins	S	S	F	S	S	F	–
`WS`	204	-9	–	20	8	–	–

Overall Algorithm:
Step 1. Select y_i's using the desired optimization algorithm (e.g., MinRelVar). Defines each $\lambda_i = \frac{c_i}{y_i}$ when using rounding (shown), or $\lambda_i = c_i$ when not using rounding.
Step 2. Flip coins, which defines the synopsis `WS`.
Steps 3+. Answer queries from `WS` by applying properties (P1)-(P2). E.g., $\hat{d}_{26} = 204 - 9 - 20 + 8 = 183$. (Note: the actual value for $d_{26} = 186$).

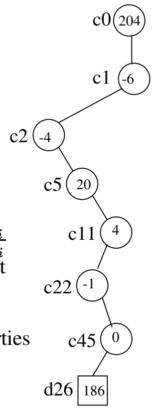

Figure 3: Summary of the approach, with an example (error bounds not shown).

3.7 Summary of the Approach

Figure 3 summarizes the steps for constructing and using our probabilistic wavelet synopses. For brevity, we show example values for c_i, λ_i, etc. only for a single path in an example error tree. In Step 1, we select the y_i using any of the optimization algorithms (e.g., MinL2, MinRelVar or MinRelBias). This defines the $\lambda_i = \frac{c_i}{y_i}$ when using rounding, or $\lambda_i = c_i$ when not using rounding. In Step 2, we randomly either retain or discard each λ_i, according to the probability y_i. An example outcome is shown (labeled "coins"). This results in the probabilistic wavelet synopsis $\{(0, 204), (1, -9), (5, 20), (11, 8), \ldots\}$. We perform several (e.g., 5) trials of coin flips, and select the synopsis that minimizes the mean relative error. In Steps 3+, the synopsis is used to answer queries. For point queries, property (P1) is used. For range sums or averages, property (P2) is used. More generally, we can apply any of the previous techniques [4, 19] for answering queries from wavelet synopses, using our probabilistic wavelet synopses. Furthermore, we can exploit the upper bounds on standard error or bias guaranteed by our synopses to provide strong probabilistic error guarantees on individual query answers [7].

4. EXPERIMENTAL STUDY

In this section, we present the results of an empirical study we conducted using the novel techniques developed in this paper for building probabilistic wavelet synopses. The objective of this study is to verify the effectiveness of our probabilistic synopsis techniques in reducing the data-reconstruction bias compared to conventional, deterministic thresholding based on normalized coefficient values. To this end, we have experimented with different synthetic and real-world data sets. The major findings of our study can be summarized as follows:

- **More Consistent, Low-Error Data Reconstruction.** By exploiting our randomized thresholding strategies, probabilistic wavelet synopses can enable a more consistent, lower-error approximation of data values; the result is a significantly smaller value of mean relative error in data reconstruction across the entire underlying data domain.
- **Improved Quality Guarantees for Individual Data Values.** Besides reducing the overall relative error, our probabilistic wavelet synopses can also significantly reduce the maximum (and other percentiles) of relative-error values at

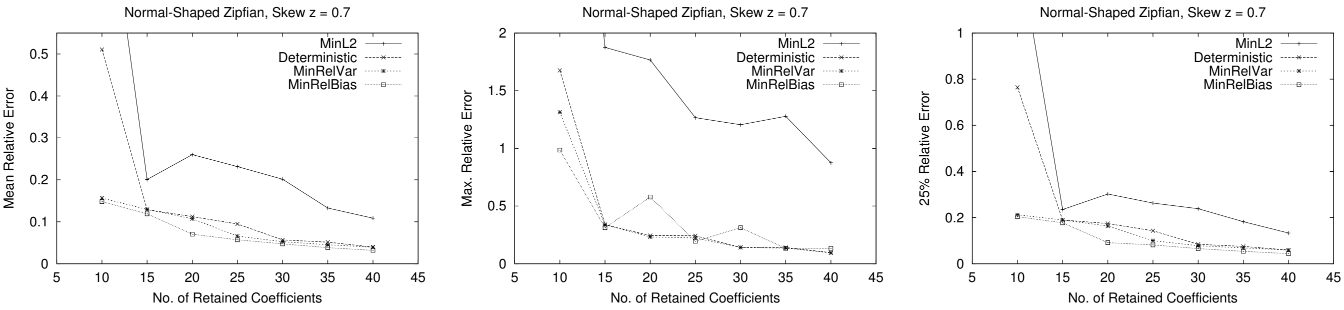

Figure 4: Approximation error for "Normal" Zipfian permutation with data skew $z = 0.7$: (a) Mean relative error. (b) Maximum relative error. (c) 25-percentile relative error.

individual data points; thus, for a given amount of synopsis space, they can offer tighter quality guarantees for reconstructed values than conventional deterministic wavelets.

As our results show, the most significant improvements occur with smaller synopses, larger skew, or noisy data. All experiments reported in this section were performed on a Sun Ultra-250 machine with 1 GB of main memory running Solaris 2.7.

4.1 Testbed and Methodology

Techniques and Parameter Settings. Our experimental study compares the conventional, deterministic thresholding scheme for Haar wavelet coefficients (i.e., maintaining the largest coefficients in absolute normalized value) with our three probabilistic thresholding schemes MinL2, MinRelVar, and MinRelBias, designed to minimize the expected L^2 error, the maximum normalized standard error, and the maximum normalized bias, respectively.[7] For MinRelVar, we used the optimization described in Section 3.4, as it improved the accuracy of our answers.

For the normalizing methods (i.e., MinRelVar and MinRelBias), we determined a setting for the quantization parameter $\mathtt{q}$ used in our dynamic-programming solution by comparing our algorithms to a continuous mathematical-optimization solver on some small example data sets. Our results showed that our algorithms quickly converged to the optimal continuous solution even for relatively small values of the quantization parameter $\mathtt{q}$. We decided to use a value of $\mathtt{q} = 10$ for our experimental runs, as we found that value to give good accuracy as well as reasonable running times for our dynamic-programming algorithms. For each input data set, we determined the value of the sanity bound $\mathtt{S}$ as the 10-percentile value in the data (i.e., 90% of the data points had values greater than $\mathtt{S}$). Finally, we experimented with different values for the perturbation parameter Δ, and determined that $\Delta = \min\{0.01, \frac{\mathtt{S}}{100}\}$ was a good choice.

Synthetic-Data Generation. We ran our techniques against several different one-dimensional synthetic data distributions, generated as follows. First, a Zipfian data generator was used to produce Zipfian frequencies for various levels of skew (controlled by the z parameter of the Zipfian), numbers of distinct values, and total frequency values (i.e., data-tuple counts). We varied the z parameter between 0.3 (low skew) to 2.0 (high skew), the distinct values between 128 and 512, and the tuple count between $100,000$ and $500,000$. Next, a permutation step was applied on the generated Zipfian frequencies to order them over the data domain; we experimented with four different permutation techniques: (1) *"NoPerm"* basically leaves the ordering as specified by the Zipfian data generator, i.e., smaller values have higher frequencies; (2) *"Normal"* permutes the frequencies to resemble a bell-shaped normal distribution, with the higher (lower) frequencies at the center (resp., ends) of the domain; (3) *"PipeOrgan"* permutes the frequencies in a "pipe-organ"-like arrangement, with higher (lower) frequencies at the two ends (resp., center) of the data domain; and, (4) *"Random"* permutes the frequencies in a random manner over the domain.

Approximation-Error Metrics. We consider three metrics to gauge the accuracy of the different wavelet-synopsis techniques. Let d_i ($\hat{d}_i$) denote the i^{th} accurate (resp., reconstructed) value in the domain, and let $\mathtt{S}$ be the specified sanity bound for the approximation. The *maximum relative error* in the reconstruction is $\max_i\{\frac{|\hat{d}_i - d_i|}{\max\{d_i, \mathtt{S}\}}\}$ The *mean relative error* is $\frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{d}_i - d_i|}{\max\{d_i, \mathtt{S}\}}$. The *25-percentile relative error* is an upper bound on the relative error of 75% of the data values in the domain. The maximum relative error can be returned to the user as a *guaranteed-error bound* for the reconstruction of any individual data value, and our MinRelVar and MinRelBias techniques are designed to help minimize this error. However, it is based on only the largest error, and hence it provides a less informative comparison than the other two metrics. Thus we will primarily use the mean relative error for the comparisons in this section (the 25-percentile relative error results are similar [7]).

4.2 Results – Synthetic Data Sets

We present some of our experimental results with synthetic data sets for different frequency permutations and settings of Zipfian skew. The numbers shown in this section were obtained using a data domain of 256 distinct values and a tuple count of $200,000$; we observed similar results for other parameter settings. After computing the y_i's for the MinL2, MinRelVar, and MinRelBias schemes, five trials of the coin flippings using different random seeds were performed, and the synopsis was selected that gave the least value for the observed mean relative error.

Relative-Error Numbers: Mean, Maximum, and 25-Percentile. Figure 4 depicts the numbers obtained by all four techniques for mean, maximum, and 25-percentile relative error on a "Normal" Zipfian data set with skew $z = 0.7$. Clearly, even for this moderate value of data skew, our MinRelVar and MinRelBias algorithms are able to guarantee better relative-error reconstruction for data values than deterministic, with the difference being especially evident for space-constrained synopses. More specifically, for 10 retained coefficients, both our methods give an over 200% improvement in mean relative error over deterministic, reducing it from over 0.5 to about 0.15. Remember that with 256 distinct values, 10 coefficients represent an approximately 4%-space synopsis of the full distribution. As the space for the synopsis is increased, the three methods

[7] To the best of our knowledge, there are no known *deterministic* thresholding schemes that optimize for relative-error metrics. Furthermore, our dynamic-programming techniques do not directly extend to a deterministic setting; for example, the error guarantees of our schemes rely on assigning fractional storage, $y_i \in (0, 1]$, to non-zero coefficients and then flipping coins to obtain $y_i \in \{0, 1\}$.

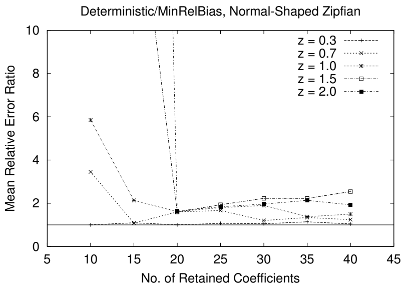

Figure 5: Effect of Skew: Ratio of mean relative error between deterministic synopses and MinRelBias synopses.

converge to the same relative error numbers. Similar trends can be seen for the maximum and 25-percentile relative errors. With respect to our MinL2 technique, our results show that it can result in large relative errors (even worse than those of deterministic); this is to be expected, since MinL2 does not explicitly optimize for the maximum normalized bias or variance in the reconstruction of data values. Due to its somewhat erratic behavior, MinL2 is omitted from our discussion in the remainder of this section. The trends are similar for higher values of the skew parameter z, with the relative difference between deterministic and our MinRelBias and MinRelVar strategies becoming even more pronounced [7].

Effect of Data Skew. Figure 5 depicts the ratio between the mean relative error values obtained by deterministic and those obtained by our MinRelBias technique for "Normal" Zipfian distributions with varying values of the skew parameter z. The trends are similar for our MinRelVar technique [7]. Obviously, the relative-error benefits obtained by our probabilistic techniques increase explosively as a function of the skew in the underlying data, with error ratios being way off the chart for $z = 1.5, 2.0$ and 10–15 coefficient synopses. For lower values of z, our probabilistic synopses offer more moderate benefits or match the relative-error performance of the deterministic scheme. Note that, even for higher synopsis sizes (e.g., 35–40 coefficients), our techniques can offer error improvements as high as 100% over deterministic thresholding.

Effect of Permutation Strategy. Figure 6 depicts the mean relative error ratio between deterministic and our MinRelBias synopses as a function of the data skew parameter z for 15-coefficient synopses and for each of the four permutation strategies tested in our experiments. The trends are similar for our MinRelVar technique [7]. Clearly, the "Normal" and "PipeOrgan" frequency arrangements (the two curves are indistinguishable in the figure) are the ones reaping the largest error benefits from our strategies, with the improvement increasing explosively for higher data skew; e.g., for $z = 2.0$, both MinRelBias and MinRelVar reduce the mean relative error by over $8,000\%$ with respect to deterministic. The error improvements for the non-permuted Zipfian ("NoPerm") are not as spectacular, but still are as high as 100–150$\%$ for more skewed distributions. Finally, for the "Random" data set, our methods seem to closely match (in most cases) the performance of deterministic thresholding and, as a consequence, offer little (if any) benefit in terms of relative error. We should note, however, that by randomly permuting Zipfian frequencies, "Random" results in a highly-irregular data set over which *any* small synopsis based on Haar wavelets is doomed to give poor approximations; thus, our results are not entirely surprising.

4.3 Results – Real-World Data Sets

To explore how our techniques performed on real-world data, we used the *Cover Type* data set from the National Forest Service, down-loaded from U.C. Irvine [11]. There are $581,012$ tuples in the data set; each tuple has 54 attributes (elevation, slope, distance to highway, forest cover type, etc). Most of these attributes have low cardinality, but there are 10 quantitative attributes, each with cardinality 256 or higher. We ran our techniques on a number of these attributes, and we report representative results on two of the attributes: "hillshade3pm" (`CovType-HS3`) and "aspect" (`CovType-A`). These two attributes test our algorithms under widely different distributions. `CovType-HS3` measures a hillshade index (from 0 to 255) at 3pm on the summer solstice. Its histogram (the input to the synopsis techniques) is shown in Figure 7(a); as can be seen from the figure, the distribution is bell-shaped and relatively smooth. `CovType-A` measures the aspect in degrees azimuth, ranging from 0 to 359. Its histogram is shown in Figure 7(b); the distribution is more uniformly spread, with a pipe-organ-style fluctuation and considerable peaks of noise.

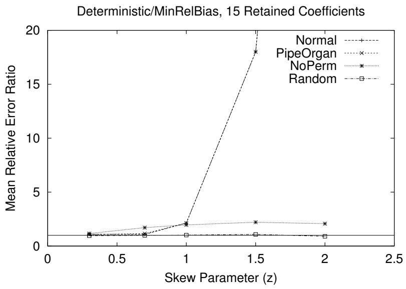

Figure 6: Effect of Permutation Strategy: Ratio of mean relative error between deterministic and MinRelBias synopses.

Figures 7(c,d) depict the ratio of mean relative errors between deterministic thresholding and our MinRelBias and MinRelVar schemes as the number of retained coefficients is varied. Clearly, our probabilistic wavelet synopses offer very substantial accuracy benefits over conventional deterministic synopses for both of the real-life data sets used in our tests. Further, our results show that for both `CovType-HS3` and `CovType-A` the mean relative error numbers for the deterministic scheme improve very slowly as more space is given to the synopsis; thus, the relative performance of our schemes actually *improves* as the number of coefficients increases.

5. RELATED WORK

Wavelets have a long history of successes in the signal and image processing arena [12, 18] and, recently, they have also found their way into data-management applications. Matias et al. [14] first proposed the use of Haar-wavelet coefficients as synopses for accurately estimating the selectivities of range queries. Vitter and Wang [19] described I/O-efficient algorithms for building multi-dimensional Haar wavelets from large relational data sets and show that a small set of wavelet coefficients can efficiently provide accurate approximate answers to range aggregates over OLAP cubes. Chakrabarti et al. [4] demonstrated the effectiveness of Haar wavelets as a general-purpose approximate query processing tool by designing efficient algorithms that can process complex relational queries (with joins, selections, etc.) entirely in the wavelet-coefficient domain. Matias et al. [15] considered the problem of on-line maintenance for coefficient synopses and propose a probabilistic-counting technique that approximately maintains the largest normalized-value coefficients in the presence of updates. (We are currently working on how to dynamically maintain our probabilistic wavelet synopses.) Gilbert et al. [9] proposed algorithms for building approximate one-dimensional Haar-wavelet synopses over numeric data streams. All the above papers rely on conventional, deterministic

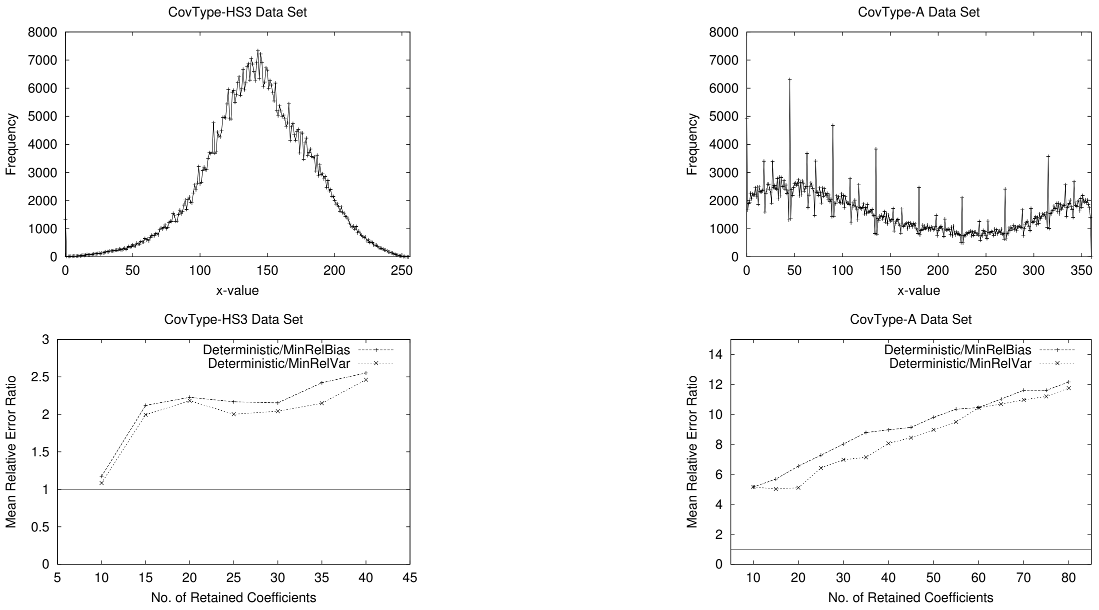

Figure 7: (a,b) `CovType-HS3` and `CovType-A` data sets. (c,d) Ratio of mean relative error between deterministic and probabilistic wavelet synopses for `CovType-HS3` and `CovType-A` data.

thresholding schemes that typically decide the significance of a coefficient based on its absolute normalized value; as demonstrated in this paper, such schemes can result in large relative errors.

There is a rich literature on m-term approximations using wavelets (m is the number of coefficients in the synopsis). Previous related work has studied dynamic-programming style approaches, deterministic thresholding to minimize a desired L^p metric, and bounds on worst case error [5]. We are not aware of work addressing relative errors with sanity bounds, arguably the most important scenario for approximate query processing in databases. Anastassiou and Yu (e.g., [2, 3]) have written a series of papers on the topic of *probabilistic wavelet approximation*. However, these papers are unrelated to our approach, as they actually study the mathematical properties of certain wavelet operators for approximating continuous monotone functions and probability distributions.

To the best of our knowledge, and after consulting several wavelet experts (e.g., [6]), it seems that our approach of probabilistically rounding and selecting coefficients has not been previously studied. To our knowledge, our approach is the first to provide unbiased reconstruction for individual data values and value ranges.

6. CONCLUSIONS

This paper has introduced *probabilistic wavelet synopses*, the first wavelet-based data reduction technique that provably enables unbiased data reconstruction, with error guarantees on individual approximate answers. We have described a number of novel techniques for tuning our scheme to minimize desired error metrics, as well as extensions to multi-dimensional data. Experimental results on real-world and synthetic data sets demonstrate that probabilistic wavelet synopses significantly reduce approximation relative error compared with the previous deterministic approach, in most cases by over a factor of 2, and up to a factor of 80 for highly skewed data. Therefore, we recommend their use in approximate query processing systems.

7. REFERENCES

[1] S. Acharya, P.B. Gibbons, V. Poosala, and S. Ramaswamy. "Join Synopses for Approximate Query Answering". In *Proc. of the 1999 ACM SIGMOD Intl. Conf. on Management of Data*, May 1999.

[2] G.A. Anastassiou and X.M. Yu. "Monotone and probabilistic wavelet approximation". *Stoch. Analysis and Applications*, 10(3), 1992.

[3] G.A. Anastassiou and X.M. Yu. "Probabilistic discrete wavelet approximation". *Num. Func. Anal. & Opt.*, 1992.

[4] K. Chakrabarti, M. Garofalakis, R. Rastogi, and K. Shim. "Approximate Query Processing Using Wavelets". In *Proc. of the 26th Intl. Conf. on Very Large Data Bases*, September 2000.

[5] R. A. DeVore. "Nonlinear Approximation". *Acta Numerica*, 7, 1998.

[6] David Donoho. Personal communication, March 2001.

[7] M. Garofalakis and P.B. Gibbons. "Wavelet Synopses with Error Guarantees". Bell Labs Tech. Memorandum, December 2001.

[8] David M. Gay. Personal Communication, February 2001.

[9] A.C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M.J. Strauss. "Surfing Wavelets on Streams: One-pass Summaries for Approximate Aggregate Queries". In *Proc. of the 27th Intl. Conf. on Very Large Data Bases*, September 2001.

[10] P.J. Haas and A.N. Swami. "Sequential Sampling Procedures for Query Size Estimation". In *Proc. of the 1992 ACM SIGMOD Intl. Conf. on Management of Data*, June 1992.

[11] Information and Computer Science, University of California at Irvine. ftp://ftp.ics.uci.edu/pub/machine-learning-databases, 2000.

[12] B. Jawerth and W. Sweldens. "An Overview of Wavelet Based Multiresolution Analyses". *SIAM Review*, 36(3), 1994.

[13] Y. Matias. *"Highly Parallel Randomized Algorithmics"*. PhD thesis, Tel Aviv University, 1992.

[14] Y. Matias, J.S. Vitter, and M. Wang. "Wavelet-Based Histograms for Selectivity Estimation". In *Proc. of the 1998 ACM SIGMOD Intl. Conf. on Management of Data*, June 1998.

[15] Y. Matias, J.S. Vitter, and M. Wang. "Dynamic Maintenance of Wavelet-Based Histograms". In *Proc. of the 26th Intl. Conf. on Very Large Data Bases*, September 2000.

[16] R. Motwani and P. Raghavan. *"Randomized Algorithms"*. Cambridge University Press, 1995.

[17] P. Raghavan and C.D. Thompson. "Randomized Rounding: A Technique for Provably Good Algorithms and Algorithmic Proofs". *Combinatorica*, 7(4), 1987.

[18] E.J. Stollnitz, T.D. DeRose, and D.H. Salesin. *"Wavelets for Computer Graphics"*. Morgan Kaufmann Publishers, Inc., 1996.

[19] J.S. Vitter and M. Wang. "Approximate Computation of Multidimensional Aggregates of Sparse Data Using Wavelets". In *Proc. of the 1999 ACM SIGMOD Intl. Conf. on Management of Data*, May 1999.