

# Nonuniform Compression in Databases with Haar Wavelet

S. Chen <sup>\*</sup>      A. Nucci <sup>†</sup>

## Abstract

*Data synopsis is a lossy compressed representation of data stored into databases that helps the query optimizer to speed up the query process, e.g. time to retrieve the data from the database. An efficient data synopsis must provide accurate information about the distribution of data to the query optimizer at any point in time. Due to the fact that some data will be queried more often than others, a good data synopsis should consider the use of nonuniform accuracy, e.g. provide better approximation of data that are queried the most. Although, the generation of data synopsis is a critical step to achieve a good approximation of the initial data representation, data synopsis must be updated over time when dealing with time varying data. In this paper, we introduce new Haar wavelet synopses for nonuniform accuracy and time-varying data that can be generated in linear time and space, and updated in sublinear time. The efficiency of our new data synopses is validated against other linear methods by using both synthetic and real data sets.*

## 1 Introduction

How to efficiently query data is a fundamental problem in databases. Estimating the cost of complex queries, reflecting CPU time, memory usage and I/O operations, requires a detailed knowledge of how data are distributed and stored into database tables. In practice, any database system maintains a concise lossy compressed data structure, called *data synopsis*, that approximates the data distribution at any point in time. The query optimizer will use this data summary to decide how a query is executed in order to retrieve the requested data at the minimal cost in terms of overall processing overhead. Besides accuracy, the cost of generating and updating synopses is a major concern, since the cost of optimization may overwhelm its benefits if the cost of synopsis is too large.

According to the characteristics of the environment at which data synopses are applied, data synopses might be classified as (i) *Uniform vs. Nonuniform* and (ii) *Static vs. Dynamic*.

Data synopses applied to data that require the same quality of approximation are known as “uniform”. All data subset will be represented by the same *weight*, that reflects the fact that any data subset must be treated in the exact same way as the others. Scenarios in which at least one data subset is required to be known with a better quality of approximation than others, e.g. for example because it might be invoked in the query process more often than the others, are known as “nonuniform”. In this case, the weight associated to each data subset will be different; the larger is the value of its weight the higher is the quality of its approximation for the data subset.

---

<sup>\*</sup>Dept. of Computer Science, Rutgers University, [suche@cs.rutgers.edu](mailto:suche@cs.rutgers.edu)

<sup>†</sup>Narus, Inc. [anucci@narus.com](mailto:anucci@narus.com)

At the same time, data synopses can be defined for “static” or “dynamic” data structures. When data and weights do not change over time, it is said to be “static”. In this context, it is important to generate a good data synopsis for the data on hands. When data or weights change over time, the scenario is said to be “dynamic”. Dynamic scenarios are challenging to be managed. In this case, data synopses should be well generated and updated over time in order to provide approximated answers with high accuracy and thus provide meaningful information to the query optimizer at any point in time.

There are two types of data synopses that can be used to answer database queries: *Point-wise approximation* and *Range-sum approximation*. The Point-wise approximation is defined for single data point query while the Range-sum approximation is defined for data intervals query, e.g. set of data points. As a consequence, Point-wise approximation can be seen as a special case of Range-sum approximation when the data interval collapses into one single data point.

In this paper, we introduce new wavelet data synopses that can be generated in linear time and updated in sublinear time for dynamic-data and nonuniform weights. Our major contributions are summarized in the following. **First**, we propose two linear algorithms for the Point-wise approximation problem, called *2-step* and *M-step*. Results clearly show how the proposed algorithms outperform weighted wavelet algorithm [10] on approximation error on both synthetic and real data sets with only  $O(B^3)$  extra running time. **Second**, we propose a linear algorithm for the Range-sum approximation problem called *Data-mapping*. To the best of our knowledge, this new algorithm represents the first linear algorithm ever proposed in literature for arbitrary weights. For  $n$  weights and  $B$  compression space, we show that both time and space complexity is  $O(n^2 + B^3)$ , which is linear to the weight size  $O(n^2)$ . **Third**, all the algorithms we proposed in this paper are the first ones that focus not only on the generation of the data synopsis but also on its update overtime. For dynamic data and weights, the time complexity of synopsis updates is  $O(\log(n) + B^3)$  for point-wise approximation and  $O(n + B^3)$  for range-sum approximation.

The remainder of this paper is structured as following. Section 2 provides the mathematical definition of Point-wise and Range-sum approximations and introduces the related work. In Sections 3 and 4 we present two new algorithms for generating Point-wise approximation, e.g. *2-step* and *M-step*, and a new algorithm for generating Range-sum approximation, e.g. *Data-mapping*. In Section 5 we validate the performance achieved by these algorithms using both synthetic and real-data, while Section 6 concludes the paper.

## 2 Problem Statement and Previous work

Considerable work is available in literature for the Point-wise data synopses generation when using histogram data synopses [1, 2, 6, 8, 9, 14]. However, most recently, several papers [3, 5, 10, 11, 12, 13] remark the effectiveness of using wavelet decomposition in reducing large amount of data to compact sets of wavelet coefficients, termed *wavelet synopses*. Wavelet synopses has been proved to provide fast and reasonably accurate approximate answers to queries. Among wavelet synopses, the Haar wavelet synopsis has been found to be the most interesting one to be used in database due to its simple structure.

In this paper we focus on Haar wavelet synopsis applied to dynamic data and nonuniform weights for both Point-wise and Range-sum approximations. In the following we provide the mathematical definition of the two problems.

**Problem 1 (Point-wise Approximation)** Let  $A_{1 \times n}$  be a generic data vector of dimension  $n$  and  $\Pi_{1 \times n}$  be the weight vector of dimension  $n$  that reflects the approximation quality of each data point of  $A$ , e.g. to each  $A[i]$  is associated a weight  $\Pi[i]$ . The weight vector is normalized between  $[0, 1]$ , e.g.  $\sum_{i=1}^n \Pi[i] = 1$ . Let  $\Psi$  be the set of  $n$  candidate wavelets and  $B$  be the allowed number of wavelets to be used for the data synopsis. Let  $D$  be the set of coefficient associated to the  $B$  chosen wavelets, e.g.  $D[i]$  is associated to  $\Psi[i]$ .

The Point-wise approximation problem can be stated as to identify the optimal subset of  $B$  wavelets from  $\Psi$  and their associated coefficients  $D$ , in order to minimize the weighted Point-wise approximation error defined as:

$$\epsilon^{(P)}(A) = \sum_{i=1}^n \Pi[i] (A[i] - \hat{A}[i])^2 \quad (1)$$

where  $\hat{A}$  represents the wavelet approximation of data set  $A$ , e.g.  $\hat{A} = \sum_{i=1}^B D[i] \Psi[i]$

**Problem 2 (Range-sum Approximation)** Let  $A_{1 \times n}$  be a generic data vector of dimension  $n$ , and let  $A(i, j) = \sum_{k=i}^j A[k]$ , with  $i \leq j$ , represent an additive function that operates on all elements of data vector  $A$  from  $A[i]$  to  $A[j]$ . Let  $\Pi_{n \times n}$  be the weight matrix of dimension  $n \times n$  such that  $\Pi[i, j] = 0 \forall i \geq j$ . Each element  $\Pi[i, j]$  represents the weight associated to  $A(i, j)$ . The weight matrix  $\Pi$  is normalized between  $[0, 1]$ , e.g.  $\sum_{i=1}^n \sum_{j=1}^n \Pi[i, j] = 1$ .

The Range-sum approximation problem can be stated as to identify the optimal subset of  $B$  wavelets from  $\Psi$  and their associated coefficients  $D$ , in order to minimize the weighted Range-sum approximation error defined as:

$$\epsilon^{(R)}(A) = \sum_{i=1}^n \sum_{j=1}^n \Pi[i, j] (A(i, j) - \hat{A}(i, j))^2 \quad (2)$$

where  $\hat{A}(i, j)$  represents a generic linear function of the wavelet approximation of  $A(i, j)$ , e.g.  $\hat{A}(i, j) = f(\sum_{i=1}^B D[i] \Psi[i])$ .

Most of the wavelet synopses are generated under the assumption of uniform weights for both Point-wise and Range-sum approximations [3][5][11]. For Point-wise approximation, the Parseval's theorem provides a solution that applies to all orthonormal data transforms, i.e., the best approximation is achieved by largest coefficients. For range-sum approximation, [11] presents an optimal solution on Haar wavelet synopsis.

The methods that study the more general case of nonuniform weights scenario result to be either suboptimal in terms of approximation errors [10] or too expensive in terms of running time [4]. Matias and Urieli in [10] provided the first linear algorithm, called **Weighted-wavelet** (W-wav), able to preserve Parseval's orthonormal condition by using a smart combination of wavelets and weights. As a result, their method provides the best synopses on weighted Haar bases, when choosing the largest coefficients as synopses. Unfortunately, this approach provides approximation errors that do not decrease monotonically with respect to the compression space  $B$  and the outcome approximation error may not be bounded. In Figure 1 we show with an example of what just stated. Data  $A$  is extracted from an exponential distribution. We define approximation error with  $B = 0$  as  $\epsilon_0 = \sum_i \Pi[i] A[i]^2$ , i.e. assume all data values are 0. The approximation error of W-wav with  $B = 0$  is  $\epsilon_0 = 623.49$ . With more compression space  $B = 2$ , the error increases to 686.55. Their approximation is far from the **ideal approximation** (see Section 3), and worse than our **2-step method** (see Section 3) that reduces the error to 306.95 with  $B = 2$ .

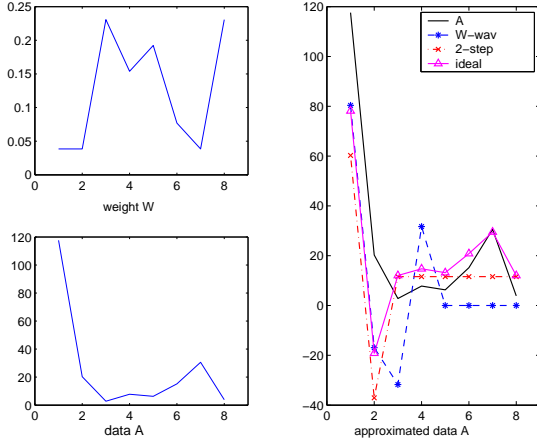


Figure 1: W-wav algorithm is not optimal

This problem exists not only for exponential data sets, but for all data sets that are characterized by only a few very large values.

Guha and Harb in [4] studied this problem for different  $L_p$  norm errors. They proved that the best coefficients can be found by searching a bounded region that is specified by the largest absolute value in data  $A$ . The accuracy of the solution is strictly related to the cardinality of the search space. The larger the search space is, the more accurate the result will be at the cost of a higher running time. The time complexity reaches  $O(n^3)$  for  $L_2$  error, and the space complexity is super linear to  $n$ . As a consequence, the complexity of their approach results to be too high for synopses used for databases.

In [7], the author studied a special case of this problem where the weights are assumed to be organized into  $k$  intervals, and a unique weight value is associated to each interval. The running time is  $O(nkB^2 \log(n))$ . When  $k = n$ , this case collapses to the Point-wise approximation problem, with a quadratic running time  $O(n^2B^2)$  in terms of  $n$ .

All previous work has been focused only on the generation of data synopses, but ignoring the importance of a more realistic scenario in which both data and weights can change over time.

### 3 Nonuniform point-wise approximation problem

In order to approach this problem, let us define three variables that come to play into the problem (see Figure 2): (i) the data vector  $A$ , (ii) the wavelet vectors  $\Psi$  and (iii) the weight vector  $\Pi$ . For uniform weights, the data vector  $A$  is mapped to the wavelet vector  $\Psi$  in order to generate data synopses. For nonuniform weights, the mapping is arbitrary. The  $W$ -wav algorithm combines the weight vector with the wavelet vector by stretching wavelets vertically. The new wavelet basis is thus weight-specific. Data synopsis obtained when considering a specific weight vector might be sub-optimal for a different weight vector. As a result, the wavelet basis has to be recomputed every time a weight change is experienced, leading to large cost in database management. A different approach to solve this problem is to combine the weight vector with the data vector. The intuition behind this approach comes from a good understanding of the error function.

Given the data vector  $A$  and the weight vector  $\Pi$ , the error function (Equation 1) can be formulated as:

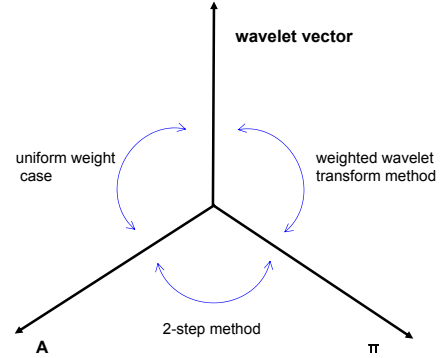


Figure 2: Methods

$$\epsilon^{(P)}(A) = \sum_{i=1}^n (\sqrt{\Pi[i]}A[i] - \sqrt{\Pi[i]}\hat{A}[i])^2 = \sum_{i=1}^n (A_W[i] - \hat{A}_W[i])^2 \quad (3)$$

where  $\hat{A}$  represents the approximation of  $A$ , while  $A_W[i] = \sqrt{\Pi[i]}A[i]$ .

With this simple manipulation, the error function has been reduced to a uniform-weight case. As a result, the best approximation  $\hat{A}^*$  of  $A$  can be solved from  $\hat{A}^*[i] = \hat{A}_W^*[i]/\sqrt{\Pi[i]}$ , where the optimal approximation  $\hat{A}_W^*$  of  $A_W$  exists according to Parseval's theorem. We refer to this method as **ideal approximation**. Unfortunately this method is impractical because it requires the knowledge of the weight value  $\Pi[i]$  for each point, and thus inadmissible because the compression space  $B$  is not large enough to store  $\Pi$ . One might consider to introduce an approximation of  $\Pi$  but this will lead to a strong sub-optimality.

In this paper we propose two algorithms, called *2-step* and *M-step* that are based on the intuition to first select wavelets according to the optimal error  $\epsilon^*$ , and then optimize their coefficients on  $\hat{A}$ .

**2-step algorithm** The name of this algorithm comes from its 2-step mechanism adopted to derive the data synopsis. In *Step A*, we define the weighted data  $A_W$  to be a point-wise product of  $A$  and  $\sqrt{\Pi}$ , e.g.  $A_W[i] = A[i]\sqrt{\Pi[i]}$ . The algorithm selects a set of wavelets with the largest  $B$  coefficients from the wavelet transform of  $A_W$ . *Step B* computes the best coefficients  $D[i]$  for the chosen wavelets by solving the partial differential equation of the function described in Equation (1), e.g.  $\frac{\partial \epsilon}{\partial D[k]} = 0, \forall k \in [1, B]$ .

The nice characteristics of the proposed algorithm is related to the fact that its approximation error is bounded by  $|\hat{A}[i] - \hat{A}^*[i]| = |\hat{A}_W[i](1 - \frac{1}{\sqrt{\Pi[i]}})|$  after the first iteration, and it will be reduced significantly at the second iteration.

The time complexity of the *2-step* algorithm is  $O(n)$  for the computation of the wavelet transform and  $O(n+B^3)$  for the computation of best coefficients [7]. The space complexity is  $O(n)$  for wavelet transform, and  $O(B^2)$  for coefficients selection. We want to remark to the reader that the  $O(n)$  space can be reused in the second step since the algorithm requires to keep the index of the  $B$  chosen wavelets only. As a consequence, the total time required for execution is expressed by  $O(n+B^3)$  while the space required is  $O(\max\{n, B^2\})$ . Furthermore, its complexity becomes linear to the data size  $n$  when  $B \ll n$ , e.g. typical of database applications.

**M-step Algorithm** The *M-step* algorithm represents an improvement of the *2-step* algorithm based on the observation that the error obtained by the *2-step* algorithm can be further reduced down by selecting new wavelets at each iteration. The selection of the new wavelets is carried out in order to minimize the difference between the approximated data and the original data, termed **residue** and denoted as  $R_i$  where  $i$  represents the  $i$ th-iteration. The algorithm starts by setting the initial residue  $R_0 = A_W$ , and the chosen wavelet set to be the empty set. At each iteration  $i$ , the algorithm computes the wavelet transform for the current residue  $R_i$ , chooses the wavelet  $\Psi_k$  with the largest coefficient and that does not belong to the chosen set. The new wavelet  $\Psi_k$  is added to the chosen set. The algorithm computes the new coefficient vector  $D$  for all chosen wavelets by solving  $\frac{\partial \epsilon^{(P)}}{\partial D[k]} = 0$ . At this point, the new residue  $R_{i+1}$  can be computed as  $R_{i+1} = A_W - \sum_{k=1}^i D[k]\Psi[k]$ , and the algorithm is ready to enter into the next iteration  $i+1$ . The *M-step* algorithm continues

until the cardinality of chosen wavelet set is equal to  $B$ .

At each step of this algorithm, every data point of the residue can change due to the new added wavelet. As a result, the algorithm needs to recompute the wavelet transform for every  $R_i$ . After  $B$  steps, the running time adds up to  $O(nB + B^4)$ . The storage space is  $O(\max\{n+B, B^2\})$ , because the algorithm needs to remember up to  $B$  chosen coefficients.

A variation of the  $M$ -step algorithm is to choose  $I$  wavelets together at each step. This reduces the running time to  $O(nB/I + B^4/I)$  at the cost of a degradation in accuracy. When  $I = B$ , this algorithm collapses to the *2-step* algorithm.

A general comment about the two algorithms proposed in this section is related to their property of having the approximation error decreasing monotonously. If a “bad” wavelet is chosen at any iteration, the next step can always ignore its negative effects by setting its coefficient to 0. In Section 5 we show how the two algorithms are able to efficiently reduce their estimation errors compared to *W-wav* algorithm.

## 4 Nonuniform Range-sums approximation problem

The same idea presented in the previous section for the Point-wise approximation, can be easily generalized in the case of the Range-sum queries. For a data set  $A$  with length  $n$ , there are  $\frac{n(n+1)}{2}$  number of intervals and weights. A **Naive** method to solve the Range-sum approximation problem is to generate a new data set  $A^{(new)}$ , in which every data point  $A(i, j)$  represents an interval extracted from the original data  $A$  and computed as  $A^{(new)}(i, j) = \sum_{k=i}^j A[k]$ . As a consequence, it is straightforward to write the error function of the new data  $A^{(New)}$  as  $\epsilon^{(P)}(A^{(New)}) = \sum_{i,j} \Pi[i, j] (A^{(New)} - \widehat{A^{(New)}})^2$ , and thus find the wavelet synopsis using methods in point-wise approximation case.

Although the idea is simple, the complexity of this approach is high because the length of  $A^{(New)}$  is  $O(n^2)$  and thus largely exceeds the synopsis space  $B$ .

Because the complexity is determined by the number of intervals constituting the original data  $A$  and the weights associated to each interval, in this Section we propose a new algorithm, named *Data-mapping*, that generate a new data vectors of size  $n$ . To the best of our knowledge, this algorithm is the first linear algorithm for arbitrary weights proposed in literature as now.

**Data-mapping algorithm** The *Data-mapping* algorithm transforms the original Range-sum approximation problem into a simpler Point-wise approximation problem by introducing a simple data and weight transformation. Given the original data  $A$ , we define a new data  $A^{(New)}$  obtained as the partial sum of  $A$ , e.g.  $A^{(New)}[i] = \sum_{k=1}^i A[k]$  with  $A^{(New)}[0] = 0$ . By using the new data  $A^{(New)}$ , we can re-write  $A(i, j)$  in Equation (2) as the difference between  $A^{(New)}[j]$  and  $A^{(New)}[i]$ . By substituting the new expression of  $A(i, j)$  as a function of  $A^{(New)}$ , we can obtain the expression of the new weights associated to the new data vector, e.g.  $\Pi^{(New)}[i] = \sum_{k=1}^i \Pi[k, i] + \sum_{k=i+1}^n \Pi[i+1, k]$ .

In order to minimize the error function and thus obtain the best wavelet coefficients  $D$ , we calculate the derivative equation of the error function regard to  $D$  and set it to 0, e.g.  $\frac{\partial \epsilon^{(R)}(A^{(New)})}{\partial D} = 0$ . As a consequence, the problem ends into solving a set of  $B$  linear equations of the form (for the general wavelet coefficient vector  $D[k]$ ):

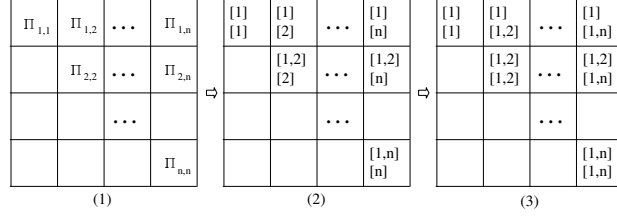


Figure 3: Example of generation of a prefix sum table for  $p_{k,l}$

$$\begin{aligned}
& D[1] \sum_{i,j} \Pi[i,j] (\psi_1[j] - \psi_1[i-1]) (\psi_k[j] - \psi_k[i-1]) + \dots + \\
& D[B] \sum_{i,j} \Pi[i,j] (\psi_B[j] - \psi_B[i-1]) (\psi_k[j] - \psi_k[i-1]) \\
= & \sum_{i,j} \Pi[i,j] (A^{(New)}[j] - A^{(New)}[i-1]) (\psi_k[j] - \psi_k[i-1]) \tag{4}
\end{aligned}$$

If the  $B$  linear equations are solved one by one, the complexity of the procedure ends up to be  $O(n^2B^2 + B^3)$ . In order to reduce the time complexity of this step, we propose to organize the Equations (4) in matrix notation of the form  $PD = Q$  and then construct a prefix sum table for  $\Pi[i,j]$  that helps to compute the weights for any interval  $(i,j)$  in constant time. The overall complexity ends up to be  $O(n^2 + B^3)$  due to  $O(B^3)$  time required to generate the matrix  $P$  ( $O(B^2)$ ) and  $Q$  ( $O(B)$ ),  $O(\frac{n(n+1)}{2})$  to generate the prefix sum table and  $O(B^3)$  to solve equation  $PD = Q$ . Due to lack of space, we do not provide the full proof of the above, but only highlight a few observations that sustain our claims. First, the matrix  $P$  can be computed in  $O(B^2)$  due to two major observations: (i) there is only a constant number of intervals, in which  $p_{kl}$  can be computed in constant time from  $\sum_{i \in I, j \in J} \Pi_{i,j}$ ; and (ii) the computation of  $\sum_{i \in I, j \in J} \Pi_{i,j}$  can be carried out in constant time with the help of a prefix sum table. The prefix sum table for  $\Pi_{i,j}$  can be easily generated by adding every row of the table  $\Pi_{i,j}$  to its next and then adding every column to its next right column (Figure 3). At the end of this process, the data is very well organized and easy to be used for the computation of the  $\sum_{i \in I, j \in J} \Pi_{i,j}$  for any interval  $I$  and  $J$ .

Similarly, the matrix  $Q$  can be computed in  $O(B)$  time after constructing a prefix sum table of similar form for  $\Pi[i,j]A(i,j)$ .

Major strength of the Data-mapping algorithm is its capability to severely reduce the complexity of the original problem. Indeed, it requires  $O(n^2)$  time and space to compute the new data set and  $O(n^2 + B^3)$  time and  $O(n^2 + B^2)$  space to compute wavelets and coefficients. As a consequence, the total running time is bounded by  $O(n^2 + B^3)$ , while its total space is bounded by  $O(n^2 + B^2)$ .

**Tracking dynamic changes of weights and data** As we discussed in the introduction, it is very important to keep the data synopses accurate over time when dealing with dynamic data and weights. For Point-wise approximation, when a data point or a weight value changes, only up to  $\log(n)$  wavelets will change. So the updating time is  $O(\log(n) + B^3)$  with  $O(B^3)$  time to find new coefficients. For Range-sum queries, the dominant part of the cost is associated to the prefix sum table construction time  $O(n^2)$ . We design an incremental method to keep the overall cost below  $O(n + B^3)$  while requiring an extra  $O(n)$  space to store updates. We omit the proof due to lack of space.

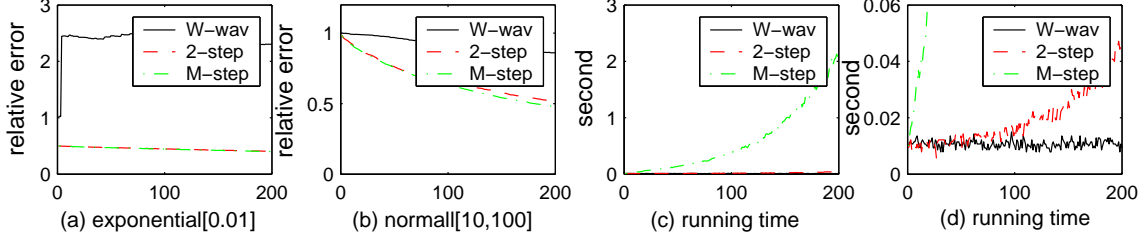


Figure 4: Accuracy and efficiency

## 5 Experiments

In this section, we show the accuracy and efficiency of our method on different data sets. We use the relative error defined as  $RE = \frac{\epsilon_B}{\epsilon_0} = \frac{\sum_i \Pi[i](A[i] - \hat{A}[i])^2}{\sum_i \Pi[i](A[i])^2}$  for point-wise approximation, and  $RE = \frac{\epsilon_B}{\epsilon_0} = \frac{\sum_{i,j} \Pi_{i,j}(A^{[i,j]} - \hat{A}^{[i,j]})^2}{\sum_{i,j} \Pi_{i,j}(A^{[i,j]})^2}$  for range-sum approximation, where  $\epsilon_B$  is the absolute approximation error with  $B$  buckets. The experiment was done by using a Linux machine with 2.80GHz processor and 2047 MB memory.

### 5.1 Point-wise queries

In this section, we compare the 2-step, M-step and W-wav algorithms using both synthetic and real data sets. In these experiments, we set  $I = 1$  for the M-step method. For other  $I$  values, the approximation error is between error of 2-step, and error of M-step with  $I = 1$ .

**Synthetic Data** The synthetic data set contains 4096 data points, and they are extracted from a normal, an exponential and a uniform distribution. The weights are extracted from a zipf distribution with  $\alpha = 0.2, 0.5$  and  $0.8$ . In the following we compare the 2-step and M-step algorithms for Point-wise approximation against the W-wav algorithm in terms of *accuracy, efficiency, time* and *skewness*.

**Accuracy** Accuracy results are shown in Figure 4(a) when the data set is extracted from an exponential distribution with parameter  $\lambda = 0.01$ . Major consideration to make is related to the shape of these curves: the error for W-wav jumps to  $2.5\epsilon_0$  and remains there even after  $B = 200$  while both the 2-step and M-step keep always the error under  $0.5\epsilon_0$ . The reason of such behavior is related to the fact that when a data set contains very large values, W-wav takes more wavelets from this region than necessary, while the 2-step and M-step algorithms can set 0 as coefficients for the “unwanted” wavelets.

**Efficiency** Efficiency results are shown in Figure 4(b) when the data set is extracted from a normal distribution with mean=10 and variance=100. Notice here the 2-step and M-step algorithms are capable to reduce the error to  $0.5\epsilon_0$  around  $B = 200$ , while the error of W-wav is still as high as  $0.85\epsilon_0$ . This is caused by cancellation among overlapped wavelets using the original coefficients. The 2-step and M-step algorithms can lower this side-effect by finding the best coefficient for each wavelet.

**Time** In Figures 4(c) 4(d) we show the running time of the three algorithms M-step method requires a very large running time. It reaches 2 second when  $B = 200$  (figure 4(c)). With the figure at a smaller scale (figure 4(d)), the running time for W-wav is  $O(n)$ , and is constant for all  $B$ . Running time for the 2-step is bounded by  $O(n + B^3)$ . For  $B^3 < n$ , that is  $B < 16$ , there is no difference of running time between the 2-step and W-wav algorithms. Even when  $B$  reaches 200,  $200^3 \gg 4096$ , the extra time for the 2-step is only 0.05 second.

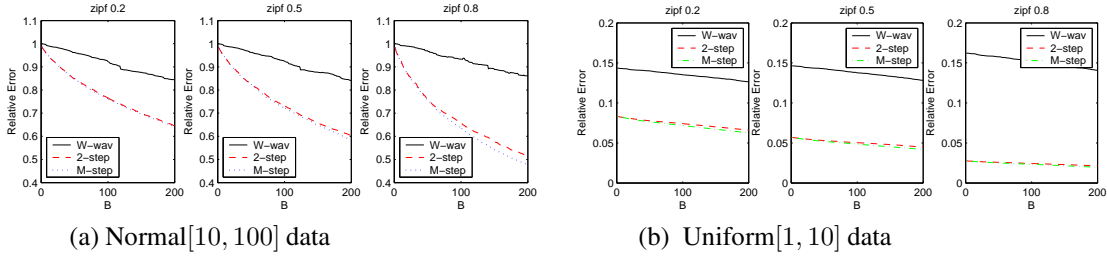


Figure 5: Skewness

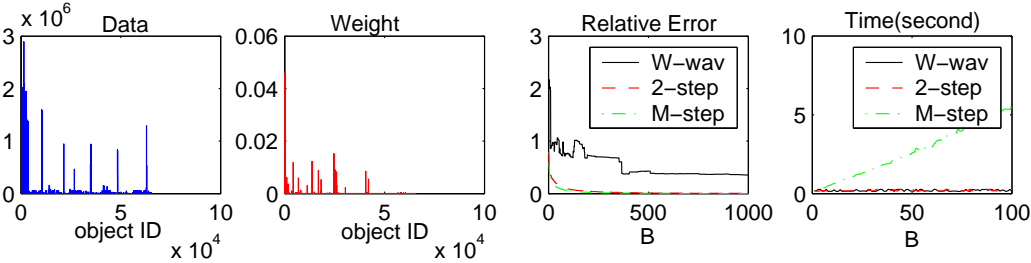


Figure 6: Relative error compare for world cup data

**Skewness** Last, in Figures 9(a) and 9(b) we compare the three algorithms while changing the weight distributions (zipf 0.2, 0.5 and 0.8). The data set is extracted from normal and uniform distributions. Important to notice here how the performance of the W-way method decreases as weights become more skewed, e.g. from zipf 0.2 to zipf 0.8. This characteristic can be explained because W-way method combines weights with wavelets. The more skewed the weights are, the higher will be the probability that wavelets with heavy weights will be used. On the other hands, both the 2-step and M-step algorithms can ignore those exaggerated wavelets by setting their coefficients to 0.

**Real Data** In order to validate the above performance metrics on more realistic data, we use a data set collected from the 66th day of the World Cup 1998 [15]. We chose this data set because it represents the set with the largest number of points. In this experiments, the data represents the query subject, which may be a web page or a picture, while the data value represents the max response size of the subject. Weights corresponds to the number of queries for each data point after normalization. In this case as well, we observe the same dynamics as before for the relative error; the W-way method provides a relative error above  $2\epsilon_0$  at first, then it is slowly reduced to  $0.35\epsilon_0$  at  $B = 600$  and remains there through  $B = 1000$  (Figure 6). The 2-step algorithm reduces its error to  $0.35\epsilon_0$  with  $B = 13$  while the M-step reduces its error to  $0.35\epsilon_0$  with only  $B = 9$ . The running time difference between the 2-step method and the W-way method is very small when  $B < 100$ .

## 5.2 Range-Sum Approximation

In this section we compare our proposed algorithm, the data mapping (**Data**) method, the **Naive** method (Section 4), and a different version of our data-mapping method when a simple additive weight is applied (**Data2**). Figure 7 summaries the approaches that the three different methods use to compute their weights. The *Naive* method produces relative errors in the range 40 – 90 while *Data* pushes it down below 1 (Figure 8). The reason of this dynamic is due to the fact that  $B$  is too small for data set with  $O(n^2)$  length. In Figure 9(a) we compare the *Data* and *Data2* methods in terms of their accuracy. We highlight how

Naive
$X = \{A^{[0,0]}, A^{[0,1]}, \dots, A^{[n-1,n-1]}\}$
Data
$W^S[i] = \sum_{k=1}^i W_{k,i} + \sum_{k=i+1}^n W_{i+1,k}$
Data2
$W^S[k] = \sum_{1 < i < j < k} W_{i,j}$

Figure 7: Different Weights

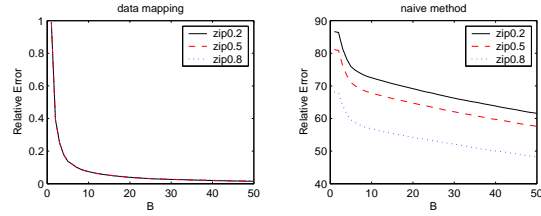


Figure 8: normal data [10, 100]

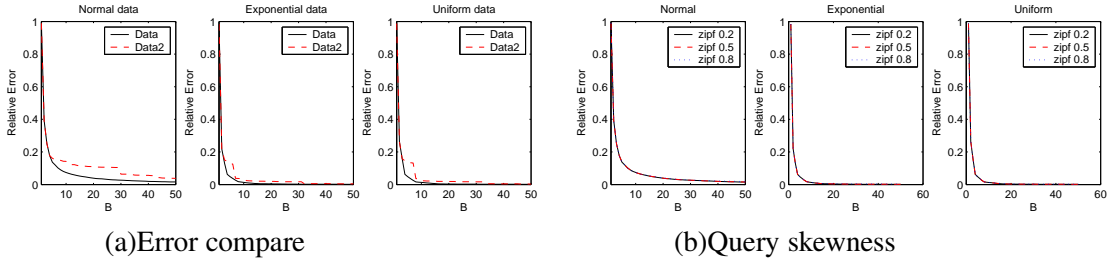


Figure 9: Range-sum approximation

*Data* performs better than *Data2* over different data sets, because the weights computed by *Data2* are not as accurate as the weights computed by *Data* that are derived directly from the error functions. In Figure 9(b) we show relative error of *Data* method as a function of skewness of weights (zipf 0.2, 0.5 and 0.8). As a consequence, the skewness of the queries does not affect the algorithm accuracy because *Data* method sums certain  $W[i, j]$ s together to compute a new weight. This summation cancels out the skewness effect.

## 6 Conclusion

We studied nonuniform approximation for both point-wise and range-sum queries. Although the approximation is one dimension, it can be easily generalized to multi-dimensions, because the methods we used to choose wavelets and coefficients are not restricted by dimensionality. The wavelets are selected from the optimal solution for weighted data, however they are not optimal with respect to the original data. To overcome the above problem, we add a second step to optimize their coefficients for the original data. This takes extra  $O(B^3)$  time, but improves accuracy significantly. How to find the optimal wavelets for original data is still an open problem.

## References

- [1] A. Aboulnaga and S. Chaudhuri. Self-tuning Histograms: Building Histograms Without Looking at Data. SIGMOD (1999)
- [2] N. Bruno, S. Chaudhuri, L. Gravano. STHoles: A Multidimensional Workload-Aware Histogram. SIGMOD (2001)
- [3] K. Chakrabarti, M. Garofalakis, R. Rastogi, and K. Shim. Approximate query processing using wavelets. VLDB Journal, Vol. 10, No. 2-3, (2001) 199-223.
- [4] S. Guha, B. Harb. Wavelet Synopsis for Data Streams: Minimizing Non-Euclidean Error. KDD(2005).
- [5] M. Garofalakis and A. Kumar. Deterministic Wavelet Thresholding for Maximum-Error Metrics. PODS(2004) 166-176.
- [6] N. Koudas, S. Muthukrishnan, D. Srivastava. Optimal histograms for hierarchical range queries. PODS(2000).
- [7] S. Muthukrishnan. Subquadratic Algorithms for Workload-Aware Haar Wavelet Synopses. FSTTCS(2005)
- [8] S. Muthukrishnan, M. Strauss. Rangesum histograms SODA(2003)
- [9] S. Muthukrishnan, M. Strauss, X. Zheng. Workload-Optimal Histograms on Streams ESA(2005)
- [10] Y. Matias, D. Urieli. Optimal workload-based weighted wavelet synopses, ICDT(2005)
- [11] Y. Matias, D. Urieli. Optimal wavelet synopses for Range-Sum Queries. <http://theory.stanford.edu/~matias/papers.html> (2004)
- [12] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-Based Histograms for Selectivity Estimation, SIGMOD(1998)
- [13] Y. Matias, J. S. Vitter, and M. Wang. Dynamic Maintenance of Wavelet-Based Histograms, VLDB(2000)
- [14] N. Thaper, S. Guha, P. Indyk, N. Koudas. Dynamic multidimensional histograms, SIGMOD(2002)
- [15] <http://ita.ee.lbl.gov/html/contrib/WorldCup.html>