On Linear-Spline Based Histograms

Qing Zhang and Xuemin Lin

School of Computer Science and Engineering, University of New South Wales
Sydney, NSW 2052, Australia {qzhang,lxue}@cse.unsw.edu.au

Abstract. Approximation is a very effective paradigm to speed up query processing in large databases. One popular approximation mechanism is data size reduction. There are three reduction techniques: sampling, histograms, and wavelets. Histogram techniques are supported by many commercial database systems, and have been shown very effective for approximately processing aggregation queries. In this paper, we will investigate the optimal models for building histograms based on linear spline techniques. We will firstly propose several novel models. Secondly, we will present efficient algorithms to achieve these proposed optimal models. Our experiment results showed that our new techniques can greatly improve the approximation accuracy comparing to the existing techniques.

1 Introduction

Traditional query processing has focused on generating exact answers in a way that seeks to minimize response time and maximize throughput. However, in many applications it may be too expensive for the DBMS to produce exact answers. For example, when user issues a complex query to a data warehousing, generating an exact answer may take hours or even days due to the costs of computation and disk I/O required. Sometimes a network or disk storage failure may cause a part of data not accessible; this makes exact answers impossible. Another example is that in a decision support system, an early response by approximate answers is especially helpful because the user can quickly determine a direction to drill down the data. Clearly, approximation is a good alternative in those applications. The quality of an approximate processing is measured by two conflicting parameters: efficiency and accuracy. A "good" approximate query processing usually means a good trade-off between efficiency and accuracy.

Database queries have two forms - *aggregation* and *non-aggregation*. An aggregation query returns a numeric value; for instance, COUNT, SUM, AVG, etc. A non-aggregation query returns a set of tuples from a database tables; for example, JOIN. Approximate processing of aggregation queries has recently attracted a great deal of attention. Most research results are based on a *data size reduction* paradigm. Three techniques [4] have been developed, such as *sampling, histogram* and *wavelet*. *Sampling* [1,18,5] is a popular data size reduction technique which takes a small portion of data as representative. To reduce approximation errors caused by applications of sampling techniques to data with big distribution skew, *wavelet* techniques were firstly applied by the authors

X. Meng, J. Su, and Y. Wang (Eds.): WAIM 2002, LNCS 2419, pp. 354–366, 2002.

in [14] to approximate query processing. The basic idea is to compress data by "important" wavelet *coefficients* [6]. Another popular technique is based on "histograms", which were originally used in commercial database systems to capture attribute value distribution statistics for query optimizers.

Among these three techniques, *histogram* is the most popular data reduction technique for approximately processing aggregation queries due to the following two reasons. Nowadays many commercial DBMS such as DB2, Informix, Ingres, Microsoft SQL Server, Sybase, etc. have been already using histogram techniques. Therefore, any new histogram techniques may be immediately accommodated by these database management systems. Secondly, the histogram technique is naturally suited to estimating aggregation queries. The basic idea of histogram technique is to partition original data into certain number of "intervals" ("buckets"). The key issues in histogram techniques are: how to partition the original data, what to store in each bucket, and how to estimate the result of an aggregation query for a given histogram. Many histogram techniques have been developed in [7,8,9,10,13,17].

To minimize approximation errors in a histogram, "linear-spline" [12] techniques have been proposed in combining with the least-square [19] method. It has been shown [12] that this technique out-performed those "conventional" histogram techniques [7,8,9,13,17]. To compliment the work in [12], in this paper we will propose a novel optimization model for generating linear-spline based histograms. This is the first contribution of the paper. The second contribution of the paper is that we use linear-splines to interpolate a "summation" distribution. Thirdly, we present a dynamic programming based paradigm for generating optimal histograms according to the proposed models in this paper, respectively. Our experiments reported that the new proposed optimization models in the paper out-perform the existing techniques by 2 to 20 times regarding the accuracy, subject to types of aggregation, data distributions, and data reduction degrees.

The rest of the paper is organized as follows. Section 2 presents the relevant definitions, an overview on the existing histogram techniques, and a motivation of our research in the paper. Section 3 presents details of our new optimal models for generating linear-spline based histograms, as well as their variations. Section 4 presents an efficient paradigm based on a dynamic programming technique to compute optimal histograms. Section 5 rep/setfigure orts our experiment results. This is followed by a conclusion.

2 Preliminary

Given a relation R and an attribute X of R, the *domain* D of X is the set of all possible values of X, and a finite set V $(\subseteq D)$ denotes the distinct values of X in an *instance* of R. Let V be ordered; that is $V = \{v_i : 1 \leq i \leq d\}$ where $v_i < v_j$ if $i < j$. An instance of R restricted to T of X is denoted by T, and can be represented as follows. $T = \{(v_1, f_1), (v_2, f_2), \ldots, (v_d, f_d)\}$. Here, each distinct v_i is called a *value* of T; and f_i represents the occurrence of v_i in T, which is called *frequency* of v_i. Note that in this paper, we call T *data distribution* or *data set*; that is, in this paper data set and data distribution will be used as synonymy.

A *histogram* on data set T is constructed by:

- partitioning T into β ($\geq$1) disjoint called *buckets* (or *intervals*) - $\{B_i : 1 \leq i \leq \beta\}$, such that each value in B_i is smaller than that in B_j if $i < j$, then
- approximately representing the frequencies and values in each bucket.

The *width* of a bucket is $v_j - v_i$ where v_j and v_i are respectively the maximal value and minimum value in the bucket.

2.1 Existing Histogram Techniques

In a histogram, a given data distribution in each bucket needs to be approximated in some fashion to minimize the information to be stored. In histogram techniques, the distribution of values in a bucket usually takes the *uniform spread assumption*; that is, the values are assumed to locate at equal distance from each other [17]. Consequently, we only need to store the minimum and maximum values in a bucket together with the number of values in this bucket; the other values can be approximately derived according to this assumption.

In the existing histogram techniques, the central focus is to closely match a given frequency distribution. In a *conventional* histogram model, a frequency distribution $\{(v_i, f_i), ..., (v_j, f_j)\}$ in a bucket has been approximately represented by a *constant* - the average frequency $\overline{f_{i,j}}$ where $\overline{f_{i,j}} = \frac{f_i + f_{i+1} + ... + f_j}{j-i+1}$. The existing techniques for generating conventional histograms may be summarized below according to different partition models:

- *Equi-width* [13]: bucket widths equal each other.
- *Equi-sum* [15,13]: the sum of the frequencies in each bucket is the same.
- *Maxdiff* [17]: the data distribution is partitioned such that the differences of the frequencies between adjacent boundaries are maximized.
- *V-optimal* [8,7,9]: Partition data such that $\sum_{j=1}^{\beta} \sum_{k=1}^{n_j} (\overline{f_j} - f_{j,k})^2$ is minimized, where β is the number of buckets, n_j is the number of entries in the jth bucket, $\overline{f_j}$ is the average frequency of jth bucket, and $f_{j,k}$ is the kth frequency of jth bucket.

Experiment results suggest that in most applications, Maxdiff and V-optimal out-perform Equi-width and Equi-sum.

To match closely a given frequency distribution, the authors in [12] proposed to use a line with an arbitrary slope to replace "horizontal" line in the conventional histogram model. For example, with respect to the data distribution in Figure 1, the line in Figure 1(b) is much closer to the original data distribution than that in Figure 1(a) where the horizontal line corresponds to the average frequency in a *conventional* histogram. Further, it is well-known that an application of the *least-square technique* [19] will minimize the errors in matching by a linear model. These motivated the development of *Linear-Spline histogram with Least Square method*[12] (LSLS):

- *LSLS*: In this model, the frequencies in each bucket are approximated by a linear function $l_j(v) = a_j * v + b_j$ where j represents the jth bucket. The goal

is to find a histogram with β buckets, such that $\sum_{j=1}^{\beta} \sum_{k=1}^{n_j} (l_j(v_{j,k}) - f_{j,k})^2$ is minimized where n_j is the number of entries in the jth bucket and $v_{j,k}$, $f_{j,k}$ are the kth *value* and *frequency* of jth bucket. Note that in this model, for each bucket B_j, the least square method is used to fix the variables a_j and b_j.

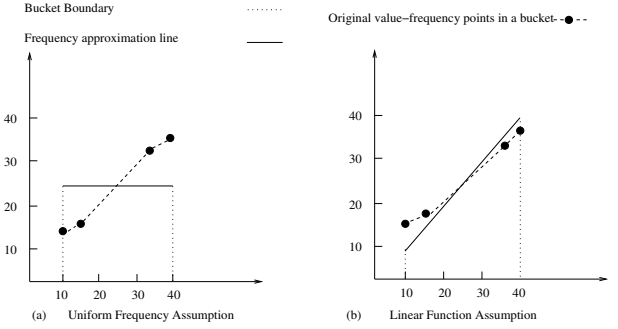


Fig. 1. Uniform Frequency VS. Linear Function

2.2 *Least-Square*: Best Alternative?

Clearly, the LSLS model can simulate an arbitrary data distribution more closely than a conventional data model does; this is because that a horizontal line is a special member in the family of linear splines. Further, LSLS usually provides the closest matching within linear models. However, it should be noted that a given data distribution does not always follow the uniform spread distribution for its values' distribution. Consequently, LSLS may not generally bring the best approximate solutions; we will show this in our experiments. In fact, the LSLS model cannot generally guarantee the following two properties:

P1: the approximation on total frequencies in a bucket of a histogram is the same as that in the original data set.

P2: the approximation on summation of all the values from the data set restricted to a bucket are the same as that (i.e., $\sum_i v_i * f_i$) in the original data set restricted to the bucket.

Note that in contrast, any conventional model has the property P1. For example, suppose that a bucket holds the following data distribution:

$$\{(10, 25), (20, 45), (50, 105), (60, 125), (70, 145)\}$$

Figure 2(a) and Figure 2(b) illustrate the information stored in a conventional model and in LSLS, respectively.

The table below summarises the results by querying the original data, the conventional histogram, and the LSLS histogram. This motivates our research.

	Lowest Value	Highest Value	Number of Values	Average Frequency
Conventional Histogram	10	70	5	89

(a) Conventional model

	Lowest Value	Highest Value	Number of Values	Slope a	Interception b
LSLS Histogram	10	70	5	2	5

(b) LSLS model

Fig. 2. Bucket Representations

Count(X) with $10 \leq X \leq 70$		Sum(X) with $10 \leq X \leq 70$	
Exact answer	445	Exact answer	24050
conventional histogram	445	*conventional histogram*	17800
LSLS histogram	425	*LSLS histogram*	21500

3 New Models of Linear Spline Histogram

Suppose that a data distribution $T = \{(v_1, f_1), (v_2, f_2), \ldots, (v_d, f_d)\}$ is given, which is partitioned into β buckets $\{B_k : 1 \leq k \leq \beta\}$. In each bucket B_k, a linear spline, $l_k(x) = a_k * x + b_k$, is used to approximate the corresponding data distribution. The total variances of the approximation by linear splines are: $\sum_{k=1}^{\beta} \sum_{v_i \in B_k} (l_k(v_i) - f_i)^2$.

Below, we show two new ways to determine a_k and b_k corresponding to each bucket B_k.

3.1 New Linear-Spline Models

In this subsection, we present the following two new models.

LSCG: Linear Spline histogram with Count Guaranteed. Suppose that a bucket B_k is given. A linear function $l'_k(x) = a'_k * x + b'_k$ is used to approximate the data distribution $\{(v_{i_k}, f_{i_k}), (v_{i_k+1}, f_{i_k+1}), ..., (v_{i_k+j_k}, f_{i_k+j_k})\}$ in B_k. In this model, we will first enforce that the total frequency approximately calculated from B_k in the histogram is the same as the original one. Then, the least-square method will be used. That is, we first enforce the following equation:

$$\sum_{m=0}^{j_k} (a'_k * v'_{i_k+m} + b'_k) = \sum_{m=0}^{j_k} f_{i_k+m} \tag{1}$$

Here, v'_{i_k+m} represents the $(m+1)$th smallest distinct value in B_k with respect to a histogram. According to the uniform spread assumption, v'_{i_k+m} should be calculated in (2). Note that this v'_{i_k+m} will be used in approximate query processing against the histogram instead of v_{i_k+m} (if $m \neq 0$ or $m \neq j_k$) in the original distribution.

$$v'_{i_k+m} = v_{i_k} + m * \frac{v_{i_k+j_k} - v_{i_k}}{j_k} \tag{2}$$

From (1) and (2), we can derive:

$$b'_k = \overline{f_k} - \frac{v_{i_k} + v_{i_k+j_k}}{2} * a'_k \tag{3}$$

Here, $\overline{f_k}$ is the average frequency in B_k. Next we use the least-square method to determine a'_k. That is:

$$\frac{d(\sum_{m=0}^{j_k}(a'_k * v_{i_k+m} + \overline{f_k} - \frac{v_{i_k}+v_{j_k}}{2} * a'_k - f_{i_k+m})^2)}{d(a'_k)} = 0 \tag{4}$$

Note that in (4), we use (3) to replace b'_k first.

Thus we derive:

$$a'_k = \frac{12 * \sum_{m=0}^{j_k}(f_{i_k+m} * (m+1)) - 6 * (j_k+2) * \sum_{m=0}^{j_k} f_{i_k+m}}{(v_{i_k+j_k} - v_{i_k}) * (j_k+1)(j_k+2)} \tag{5}$$

Note that in (5), the denominator equals zero if and only if $v_{i_k} = v_{i_k+j_k}$ (i.e., B_k contains only one distinct value v_{i_k}). Consequently, we assign that $a'_k = 0$ and $b'_k = f_{i_k}$ if B_k contains only one distinct value.

LSCSG: Linear Spline Histogram with Count and Sum Guaranteed. As with **LSCG**, a linear function $l''_k(x) = a''_k * x + b''_k$ is used to approximate the data distribution in bucket B_k. However, this time a''_k and b''_k will be chosen to make the total frequency and total summation approximately calculated over B_k in the histogram are the same as those from the original data set, respectively. This requires that a''_k and b''_k satisfy the following linear equations:

$$\begin{cases} \sum_{m=0}^{j_k}(a''_k * v'_{i_k+m} + b''_k) = \sum_{m=0}^{j_k} f_{i_k+m} & (a) \\ \sum_{m=0}^{j_k}(v'_{i_k+m} * (a''_k * v'_{i_k+m} + b''_k)) = \sum_{m=0}^{j_k}(f_{i_k+m} * v_{i_k+m}) & (b) \end{cases} \tag{6}$$

Similarly, here:

$$v'_{i_k+m} = v_{i_k} + m * \frac{v_{i_k+j_k} - v_{i_k}}{j_k}$$

Solving equation (a) and (b) of *(6)*, we get a''_k and b''_k as follows.

$$\begin{cases} a''_k = \frac{\sum_{m=0}^{j_k}(f_{i_k+m} * v_{i_k+m}) * (j_k+1) - \sum_{m=0}^{j} f_{i_k+m} * \sum_{m=0}^{j_k} v'_{i_k+m}}{(m+1) * \sum_{m=0}^{j_k}(v'_{i_k+m})^2 - \sum_{m=0}^{j_k} v'_{i_k+m} * \sum_{m=0}^{j_k} v'_{i_k+m}} & (a) \\ b''_k = \frac{\sum_{m=0}^{j_k} f_{i_k+m} * \sum_{m=0}^{j_k}(v'_{i_k+m})^2 - \sum_{m=0}^{j_k}(f_{i_k+m} * v_{i_k+m}) * \sum_{m=0}^{j_k} v'_{i_k+m}}{(j_k+1) * \sum_{m=0}^{j_k}(v'_{i_k+m})^2 - \sum_{m=0}^{j_k} v'_{i_k+m} * \sum_{m=0}^{j_k} v'_{i_k+m}} & (b) \end{cases} \tag{7}$$

It can be immediately verified that the denominators in (7) equals zero if and only if $v_{i_k} = v_{i_k+j_k}$ (i.e., B_k contains only one distinct value). Consequently, we assign that $a''_k = 0$ and $b''_k = f_{i_k}$ if B_k contains only one distinct value.

It may be worth to point out that if a given value distribution follows the uniform spread assumption, then both LSCSG and LSCG are equivalent to LSLS.

3.2 Matching Summation

Suppose that $T = \{(v_1, f_1), ..., (v_d, f_d)\}$ is a data distribution. Clearly, the COUNT query for $v_i \leq X \leq v_j$ equals:

$$\sum_{v_i \leq v_m \leq v_j} f_m.$$

The SUM query for $v_i \leq X \leq v_j$ equals:

$$\sum_{v_i \leq v_m \leq v_j} (v_m * f_m).$$

Note that all the existing histogram techniques, including our LSCG and LSCSG, approximately match the frequency distribution in each bucket. However, our initial experiments showed that it is also a good idea to approximately match a "summation" distribution - $v_m * f_m$ $(1 \leq m \leq d)$. Next we present our new models based on this idea.

LSSCG: Linear Spline histogram for Summation with Count Guaranteed. For a bucket B_k, let $l_k(x) = a_k * v_t + b_k$. We aim to find a partition to minimize (8):

$$\sum_{k=1}^{\beta} \sum_{v_i \in B_k} (l_k(v_i) * v_i - f_i * v_i)^2 \tag{8}$$

It should be noted that a_k and b_k are determined in the same way in B_k as those in LSCG.

LSSCSG: Linear Spline histogram for Summation with Count and Sum Guaranteed. In this model, we aim to find a partition to minimize (9):

$$\sum_{k=1}^{\beta} \sum_{v_i \in B_k} (l_k(v_i) * v_i - f_i * v_i)^2 \tag{9}$$

Here, $l_k(x) = a_k * v_t + b_k$ is a linear function in B_k, and a_k and b_k are determined in the same way as in LSCSG.

4 Building Histograms

Suppose that $T = \{(v_1, f_1), (v_2, f_2), ..., (v_d, f_d)\}$ is a data distribution. In this section, we will present an efficient paradigm to solve LSCG, LSCSG, LSSCG, LSSCSG. The paradigm is based on a dynamic programming technique [2].

4.1 A Dynamic Programming Paradigm

Note that the optimal goal functions P in LSCG, LSCSG, LSSCG, LSSCSG may be represented by the following uniform form.

$$P = \sum_{k=1}^{\beta} \sum_{v_i \in B_k} V_i \tag{10}$$

Here, β represents the number of buckets, v_i represents a value. Note that the variance V_i in each model is as below.

$$V_i = \begin{cases} (l_k(v_i) - f_i)^2 & if\ LSCG\ or\ LSCSG \\ (l_k(v_i) * v_i - f_i * v_i)^2 & if\ LSSCG\ or\ LSSCSG \end{cases} \tag{11}$$

Our paradigm follows the framework in [2,10,11,12]. Let $P^*(X, Y)$ represent the optimal result of using Y buckets to partition the first X values of T. Let $P[a, b]$ denote the bucket containing the consecutive $\{(v_a, f_a), ..., (v_b, f_b)\}$ in T. Then below is the crucial formula for our paradigm.

$$P^*(d, \beta) = \min_{1 \le j \le d-1} \{P^*(j, \beta - 1) + P[j + 1, d]\}$$

Thus, in order to calculate $P^*(d, \beta)$, we must calculate $P^*(i, k)$ for $1 \le i \le d$ and $1 \le k < \beta$. After storing all these intermediate results, we can finally get the optimal value by comparing different grouping plans. The time complexity of this dynamic programming paradigm is $O(\beta * d^2)$ for a given data distribution T.

4.2 Matching Area

As noted in [17], two data pairs with similar frequencies but large value difference may be possibly grouped in one bucket according to the existing models. To resolve this, another partition parameter - *area* - may be useful.

Given a data distribution $T = \{(v_1, f_1), (v_2, f_2), ..., (v_d, f_d)\}$. The *spread* s_i of v_i (for $1 \le i \le d$) is defined as $v_{i+1} - v_i$; we make $s_d = 1$. The *area* a_i of v_i is defined as $f_i * s_i$ for $1 \le i \le d$.

Clearly, the area parameter may be adopted by the models: LSLS, LSCG, and LSCSG. We name the corresponding models "LSLS(area)", "LSCG(area)", and "LSCSG(area)", respectively, which aim to minimize the following goal function.

$$\sum_{k=1}^{\beta} \sum_{v_i \in B_k} s_i^2 * (l_k(v_i) - f_i)^2 \tag{12}$$

Note that the difference among LSLS(area), LSCG(area), and LSCSG(area) is a different choice of a_k and b_k for l_k. In fact, a_k and b_k will be chosen in the same way as those in LSLS, LSCG, LSCSG, respectively.

Similarly, we name *LSSCG(area)* and *LSSCSG(area)*, respectively, for the modifications of LSSCG and LSSCSG using the parameter area; that is, to minimize the following goal function in LSSCG(area) and LSSCSG(area), respectively:

$$\sum_{k=1}^{\beta} \sum_{v_i \in B_k} s_i^2 * (l_k(v_i) * v_i - f_i * v_i)^2 \tag{13}$$

Again the difference between LSSCG(area) and LSSCSG(area) is a different choice of a_k and b_k in l_k; a_k and b_k will be calculated in the same way as those in LSSCG and LSSCSG, respectively.

Note a similar idea was proposed in V-optimal [17]. The paper [17] proposed to find a partition to minimize the following goal function.

$$\sum_{k=1}^{\beta} \sum_{v_i \in B_k} (\overline{f * s} - f_i * s_i)^2 \tag{14}$$

Here $\overline{f * s}$ is the average $f_i * s_i$ in bucket B_k. In our experiment, we generate the V-optimal and Maxdiff area-matching histograms based on the original idea in [17]; that is, based on (14).

It can be immediately verified that our dynamic programming based paradigm also works for finding the optimal solutions for (12) and (13).

5 Experiments

The data sets used in our experiments are synthesized *zipf* [20] data. In database 1, 10 tables have been generated, and each table has all together $10,000$ tuples but with only 101 different values from the domain $[0, 1000]$. Tables in database 1 have a *high* frequency. Database 2 also contains 10 generated tables, and each table contains $10,000$ tuples with 1001 different values from the domain $[0, 10000]$. The generation of each data set follows three steps below.

- *Generating Frequencies:* Different frequencies are generated according to *zipf law* and the *zipf* parameter $z = 1.0$. This means a medium skew frequency distribution.
- *Generating Values:* The spreads of values follow one distribution of *zipf_inc*, *zipf_dec*, *cusp_min*, *cusp_max* and *zipf_ran* [16]. The Zipf parameter $z = 1.0$.
- *Generating Data Distribution:* Frequencies are randomly assigned to different values.

Note that the number of buckets to be used in a histogram reflects a data reduction "degree". If the number of buckets equals the number of distinct values, then there is no data reduction, and consequently all histograms will produce the same result - the exact result.

In our experiments, for tables in database 1 we use different bucket numbers $10, 15, 20, 25, 30$, to produce corresponding histograms. For databases used in database 2, we use bucket numbers – $10, 20, 30, 40, 50$ to produce different histograms.

In our experiments, we build eight histograms respectively according to our eight new models:

LSCG, LSCSG, LSSCG, LSSCSG, LSCG(area), LSCSG(area), LSSCG(area), and LSSCSG(area).

For a comparison purpose, we also constructed the six histograms based on the following existing models:

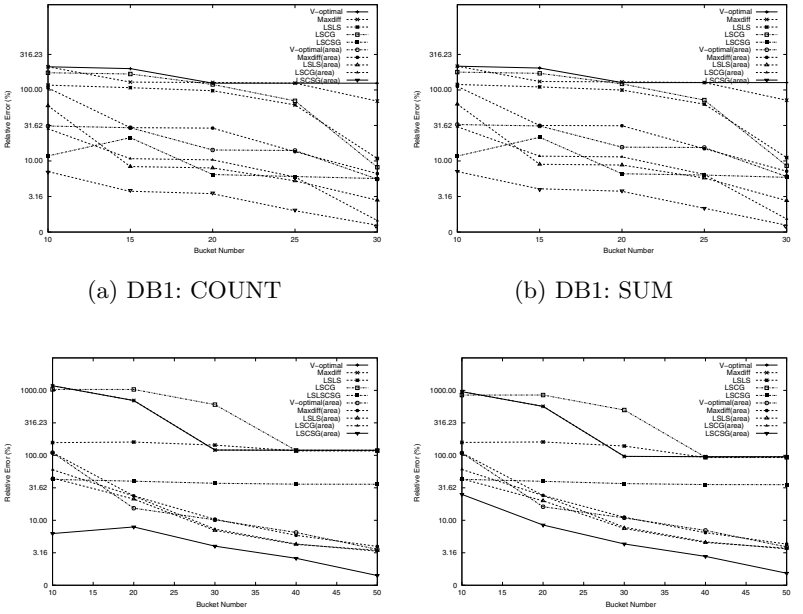
V-optimal, Maxdiff, LSLS, V-optimal(area), Maxdiff(area) and LSLS(area).

In contrast to conventional histograms, each bucket in a linear-spline based histogram needs to store one more column information as depicted in *Figure 2.2*. Note that in our experiment, we have enforced that every histogram occupies the same storage space. To achieve this, the available bucket number for a linear-spline histogram is reduced to 80% of that for a conventional histogram.

In our experiments, we targeted the 3 most popular aggregation operations, COUNT, SUM, and AVG. Since AVG is derived from a division between SUM and COUNT, we focused only on two types of range queries, COUNT and SUM. For each data set and each type of range queries, 1000 queries are randomly generated with the form:

$$\{a \leq X \leq b \mid a < b\}$$

Here a and b are randomly selected from the values' domains.



(a) DB1: COUNT (b) DB1: SUM

(c) DB2: COUNT (d) DB2: SUM

Fig. 3. Approximating aggregation query on Databases 1 and 2

Let S_i represent the actual result of query q_i and S_i' represent the approximately calculated result. The error metrics used to evaluate our histograms are:

- **relative error:** $e_i^{rel} = \frac{|S_i - S_i'|}{S_i}$
- **average relative error:** $\overline{e_N^{rel}} = \frac{\sum_{i=1}^{N} e_i^{rel}}{N}$ **,** where N represents the number of queries.

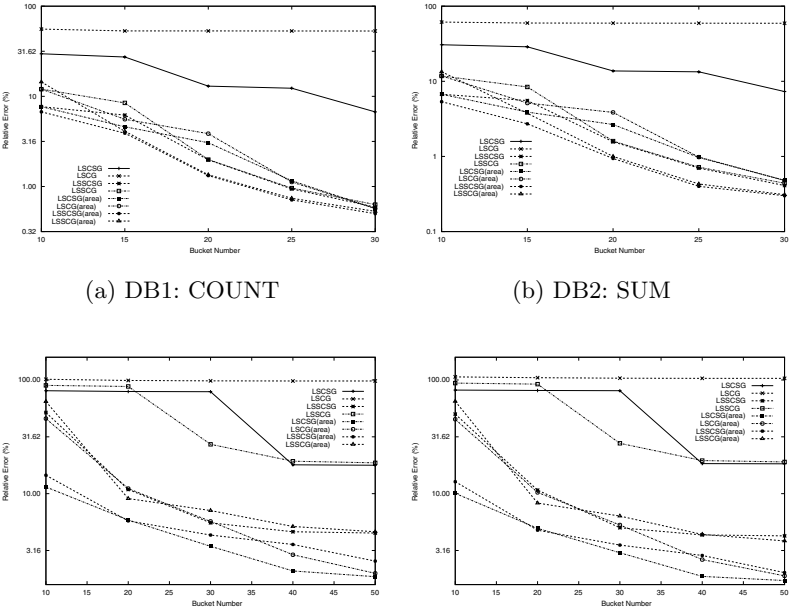
All the generated tables and histograms are stored in an *Oracle* DBMS and our experiments are done on a Pentium III 700 MHz CPU, 256MB memory computer with Linux 2.4.7.

Since the results do not vary significantly on different tables within a database, we only show some typical results here; that is, one table per data set.

Our first group of experiments are comparing the following ten histograms:

V-optimal, Maxdiff, LSLS, LSCG, LSCSG, V-optimal(area), Maxdiff(area), LSLS(area), LSCG(area), LSCSG(area).

Figure 3 provides the experiment results for database 1 and 2, respectively. From Figures 3(a) and 3(b), we can see that when original data distribution



(a) DB1: COUNT (b) DB2: SUM

(c) DB2: COUNT (d) DB2: SUM

Fig. 4. Approximating on Database 1 and 2 (with/without summation matching)

is "sharp" (i.e. database 1), our LSCSG(area) model out-performs the existing techniques (even modified by using the area parameter) by 5 - 20 times regarding accuracy subject to the number of buckets. Figures 3(c) and 3(d) suggest that when original data distribution is "smooth" (i.e. database 2), our LSCSG(area) model also out-performs the existing techniques by 2 - 15 times depending on the number of buckets used. Clearly, our experiments suggested that LSCSG(area) has the best performance.

Next, we examine if "summation-matching" may bring a further improvement. Our experiment suggested that it may further improve the accuracy if there are many repeated values; the experiment results are illustrated in Figure 4.

Note that due to the sapce limitation, a numerical illustration of our experiment results have been omited in the final version; the interested readers may refer our full paper for the details.

6 Conclusion

In this paper, we proposed several novel and effective optimal models for building linear-spline based histograms. By the proposed new models, the accuracy of approximating aggregation query has been greatly improved. Our experiments showed that the new models outperform the existing techniques by $2-20$ times depending on the degrees of data reduction and types of queries.

Very recently, the authors in [3] proposed a very effective post processing after a data distribution is partitioned. As a future study, we will investigate whether our new partitioning models will support this post processing better than the other existing models. We also plan to investigate the proposed new models in the paper against *streaming* data, and to find a way to speed up the algorithms for producing the optimal histograms.

Acknowledgement. The research was partially wupported by CMCRC seed grant. We would like to specially thank Ms. Xiaozheng Wang's kindly help with preparing figures and tables in this paper.

References

1. S. Acharya, P. B. Gibbons, and V. Poosala. Congressional samples for approximate answering of group-by queries. In *SIGMOD Conference*, pages 487–498, 2000.
2. R. E. Bellman. The theory of dynamic programming. *Bull. Amer Math Soc*, 60:503–516, 1954.
3. F. Buccafurri, L. Ponteri, D. Rosaci, and D. Sacca. Improving range query estimation on histograms. In *18th ICDE*, pages 628–638, 2002.
4. M. Garofalakis and P. B. Gibons. Approximate query processing: Taming the terabytes. VLDB, 2001.
5. P. B. Gibbons and Y. Matias. New sampling-based summary statistics for improving approximate query answers. pages 331–342, 1998.
6. A. Graps. An introduction to wavelets. *IEEE Computational Science and Engineering*, 2:50–61, Summer 1995.

7. Y. Ioannidis. Universality of serial histograms. In *Proceedings of the 19th Conference on Very Large Databases, Morgan Kaufman pubs. (Los Altos CA), Dublin*, 1993.
8. Y. E. Ioannidis and S. Christodoulakis. Optimal histograms for limiting worst-case error propagation in the size of the join results. *ACM Transactions on Database Systems*, 18(4):709–748, 1993.
9. Y. E. Ioannidis and V. Poosala. Balancing histogram optimality and practicality for query result size estimation. pages 233–244, 1995.
10. H. V. Jagadish, H. Jin, B. C. Ooi, and K.-L. Tan. Global optimization of histograms. In *SIGMOD Conference*, 2001.
11. H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. C. Sevcik, and T. Suel. Optimal histograms with quality guarantees. In *VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases, August 24-27, 1998, New York City, New York, USA*, 1998.
12. A. C. König and G. Weikum. Combining histograms and parametric curve fitting for feedback-driven query result-size estimation. In *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK*, 1999.
13. R. P. Kooi. *The optimization of queries in relational database.* PhD thesis, Case Western Reserver University, Sep 1980.
14. Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. pages 448–459, 1998.
15. G. Piatetsky-Shapiro and C. Connell. Accurate estimation of the number of tuples satisfying a condition. 14(2):256–276, 1984.
16. V. Poosala. *Histogram-Based Estimation Techniques in Database Systems.* PhD thesis, University of Wisconsin-Madison, 1997.
17. V. Poosala, P. J. Haas, Y. E. Ioannidis, and E. J. Shekita. Improved histograms for selectivity estimation of range predicates. In *SIGMOD'96*, pages 294–305, 1996.
18. J. S. Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, 1985.
19. D. D. wackerly, W. Mendenhall, and R. L. Scheaffer. *Mathematical Statistics with Application.* Duxbury Press, 1995.
20. G. K. Zipf. *Human behaviour and the principle of least effort.* Addison-Wesley, Reading, MA, 1949.