TRCS01-05

Applying the Golden Rule of Sampling for Query Estimation

Abstract. Query size estimation is crucial for many database system components. In particular, query optimizers need efficient and accurate query size estimation when deciding among alternative query plans. In this paper we propose a novel sampling technique based on the golden rule of sampling, introduced by von Neumann in 1947, for estimating range queries. The proposed technique randomly samples the frequency domain using the cumulative frequency distribution and yields good estimates without any a priori knowledge of the actual underlying distribution of spatial objects. We show experimentally that the proposed sampling technique gives smaller approximation error than the Min-Skew histogram based and wavelet based approaches for both synthetic and real datasets. Moreover, the proposed technique can be easily extended for higher dimensional datasets.

Key words. random sampling, cumulative frequency distribution, query estimation, range query

1 Introduction

Query size estimation plays an important role in the query optimizer of a database management system. Depending on the size of the intermediate results, a particularly query execution plan may be favored over another. A common approach in relational databases is to store, for each attribute, summary information that can be used to estimate the resulting size of a query. Random samples of tuples from the base relation of database can be used for selectivity estimation [HS92, HS95, GM98, HHW97, HH99, LNS90]. The AQUA system [GPA$^+$98, GM98, AGPR99] uses random samples of tuples for general purpose query result estimation. The idea is to create a down-sized copy of the original relation and run the queries against the down-sized copy, which is significantly smaller than the original relation, to get an estimate of the results. Recently Chaudhuri et al. [CMN99] have developed a theoretical framework for explaining the difficulties in implementing sampling as a primitive relational operator. We refer to the above techniques as *tuple sampling*. Instead of using a down-sized copy of the original relation for query estimation, other stored information (e.g., histograms) that summarizes the distribution of attribute values of a relation can be used for query result estimation. Such approaches further reduce the amount of auxiliary information maintained for estimation.

Data reduction techniques can be divided into two main categories: parametric and non-parametric. The parametric data reduction techniques try to use parameters, which are usually few, to describe the original data distributions. For example, the linear regression technique [BS98] employs the model fitting concept to best describe the original data distributions. Although the model fitting techniques can summarize data distributions with a few descriptive parameters (e.g., the z value in the standard normal distribution), in the cases that the data distribution does not follow any known distributions, or their linear combination, the model fitting techniques produce inferior results. Other parametric techniques employ transformation functions (e.g., SVD, wavelet, etc.) to extract a relatively small number of descriptive parameters (in the transformed domains) to summarize the original data distributions [KJF97, MVW98, MVW98, VW99].

The non-parametric techniques use different approaches to summarize data distributions. Most of the techniques in this category can be viewed as variations of tree based summarization; e.g., index trees [Gut84, Rob81, LJF94, BKK96, GG98], variations of histograms [Poo97, APR99, PG99, PIHS96, PI97, JKM+98], or cluster-based [EKX95, EKSX96, WYM97]. The idea is to divide the attribute domains into overlapping or non-overlapping regions where each region can be summarized separately with much less storage space. Other techniques in the non-parametric category employ variations of random sampling methods to summarize the original data distribution [HS92, HS95, GM98, HHW97, HH99, LNS90, GPA+98, AGPR99]. The basic idea is to create a down-sampled version of the original relation and run the queries against the down-sized copy, which is significantly smaller than the original relation, to get an estimate of the results.

One of the main problems of applying estimation techniques in a database management system is the unknown characteristic of the distribution of the data that will populate a given DBMS. In fact, in order for such a technique to be useful, the estimation technique should be effective for different kinds of data distributions. This is because a DBMS is used for a variety of applications resulting in a wide spectrum of data that populates the database. In this paper, we develop a new strategy to summarize the distribution of attribute values, the *golden estimator*, based on a simple observation (known as the golden rule of sampling) made by von Neumann [Eck87] for sampling cumulative probability distributions. The proposed method uses samples based on the frequency domain of the cumulative frequency distribution. The surprising observation is that such an approach results in samples that reflect the distribution without any a priori knowledge. We first develop the golden estimator for single attribute domains and then extend it to handle two dimensional domains which is important for spatial databases (this approach could then be applied to higher dimensions easily). We develop this approach in the discrete domain and conduct several experiments to compare it with other state of the art approaches. We demonstrate that in most cases, under the same space requirement, the golden estimator gives more accurate estimation for the result size of range queries.

This paper is organized as follows. In Section 2 we introduce our terminology and develop the golden estimator technique to estimate range queries in a single dimension space. In Section 3, we first show that the golden estimator is not applicable for spatial range query estimation without modification. We then present an extension of the golden estimator technique for multi-attribute domains that can be applied to spatial range query estimation. Detailed experiments and results are presented in Section 4. Section 5 concludes this paper.

2 The Golden Estimator for Single Attribute Domains

In this section, we develop the golden estimator for single attribute domains. Our approach uses random sampling over the distribution function of a relation instead of the tuple level as in the tuple sampling techniques. We begin by defining the range query estimation problem and then describe the golden rule of sampling [Eck87], which is invariant to the underlying data distribution. Unfortunately, this technique is applicable only in continuous domains. We therefore adapt this approach for discrete domains.

2.1 Preliminaries

Given a relation R with an attribute A, we are interested in estimating the size of a range query on attribute A. We adopt the notation develop in [PIHS96, MVW98] where information about the attribute domain is summarized and used in estimating the size of the query. We assume the discrete domain D of attribute A is $\{d_0, d_1, d_2, ..., d_{N-1}\}$ which contains all the possible values of A. A range query is specified using a range predicate of the form $a \leq A \leq b$, where a and b are constant values in the discrete domain D. Sampling is an efficient and frequently used technique for estimating query size. Two general approaches for sampling are used depending on whether the actual tuples or the domain are sampled. In tuple sampling, the tuples in the relation are sampled, and the resulting samples are taken as representatives for the actual relation and hence can be used to estimate sizes of queries. In domain sampling, the domain of the attribute of interest is sampled using a frequency distribution. The collected set of samples are represented by a histogram, which can then be used to estimate query sizes. In this paper we pursue the latter approach, namely, frequency distribution sampling and show that, in general, it outperforms tuple sampling in terms of accuracy of results.

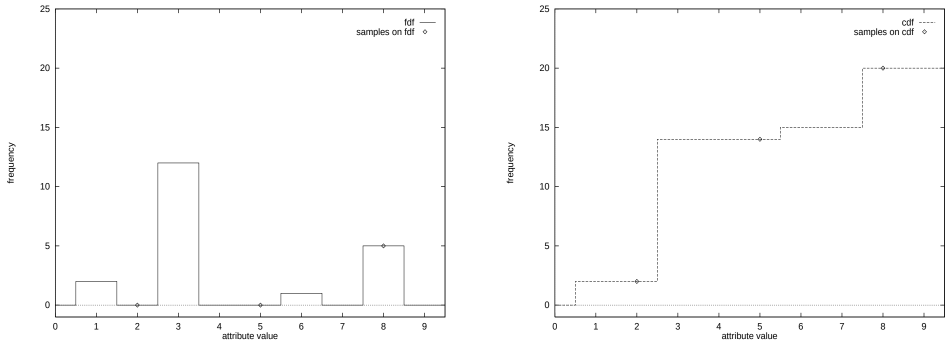
We now define some of the basic terms used throughout this paper. The *frequency* f_i of value d_i for attribute A is the number of tuples $t \in R$ with $t.A = d_i$. If no tuples t in R have $t.A = d_i$, then $f_i = 0$; i.e., zero frequency. The *frequency distribution function* (fdf) is an ordered set of value pairs $fdf = \{\langle d_0, f_0\rangle, \langle d_1, f_1\rangle, \langle d_2, f_2\rangle, ..., \langle d_{N-1}, f_{N-1}\rangle\}$. Given the fdf, the size of a range query $q(a \leq A \leq b)$ can be calculated as the number of tuples $t \in R$ with $a \leq t.A \leq b$, which is equal to $\sum_{i=a}^{b} f_i$. Much research has been devoted to obtaining accurate values for the fdf [HS95, Vit85, AD85, PIHS96], however good approximations usually require high storage, which becomes unrealistic to use the fdf to estimate query sizes. As a result, most commercial systems maintain histograms for estimating query sizes and recently there have been several histogram based techniques that have been proposed [PIHS96, PI97, Poo97]. In these approaches, the histogram is derived either from the actual fdf or, more realistically, based on a down-sized copy (often obtained using tuple sampling) with lesser accuracy.

In this paper, we will use sample pairs $\langle d_i, f_i \rangle$ to approximate the fdf (as with histogram derivations, we can either use the actual fdf or a down-sized copy). Our approach is based on the cumulative frequency distribution, which can be easily obtained from the frequency distribution function. The main challenge in query size estimation is to use a limited amount of storage to approximate the fdf. This approximation is then used to estimate the number of tuples that are in the range specified by the query.

2.2 Random Sampling in Databases

Random sampling is one of the most straightforward and simple techniques used for estimating values in a given distribution. In general, random sampling techniques work as follows. In the construction phase, random samples are selected from the domain under study, in our case, for example, the frequency distribution (fdf), and stored as value pairs $\langle d_i, f_i \rangle$ in a searchable structure (e.g., array, tree, etc.). Note that unlike tuple sampling, where actual tuples are sampled, in this approach the domain is sampled and hence some values may be zero. We are trying to estimate the entire distribution. In the query phase, when a range query $q(a \leq \; X \leq b)$ is presented we search the stored samples and compute the estimated result s' from the samples corresponding to the range $[a : b]$. The absolute difference between the actual result size s and s', $|s - s'|$ is referred to as the *absolute error*. The *relative error* E_{rel}, which evaluates the error as related to the result size is defined as:

$$E_{rel} = \frac{|s - s'|}{s} \times 100 \qquad (1)$$



(a) fdf (b) cdf

Fig. 1. Example fdf and cdf

Figure 1.(a) shows an example fdf. One naive approach is to construct the fdf by randomly sampling the attribute domain and when a query is issued, the sampled fdf value pairs are used to estimate the result size. For example, let the sample set taken be $S_{fdf} = \{\langle 2, 0 \rangle, \langle 5, 0 \rangle, \langle 8, 5 \rangle\}$, which is randomly sampled from the fdf of the attribute domain as shown in Figure 1.(a). Consider a range query $q(2 \leq X \leq 7)$. Since two samples in this query fall in the query range, namely $\langle 2, 0 \rangle$ and $\langle 5, 0 \rangle$, we can therefore

use these two samples to estimate the query result. If we assume a uniform value distribution (since we do not assume knowledge of the underlying distribution), we can assume that neighboring attributes have similar frequencies. We therefore estimate the frequencies of attribute values 3, 4, 6, and 7 are all zero because their neighboring attributes 2 and 5 have zero frequencies, which yields a query result estimate of zero. As can be verified from Figure 1.(a), the actual result to this query is 13. Hence the relative error $E_{rel} = |13 - 0|/13 \times 100 = 100\%$. In fact, several disadvantages can be observed when using randomly sampled sets from the fdf. First, as shown in Figure 1.(a), most of the frequencies are of a very small value, zeros or ones in most cases. The sample set is expected to have most of the value pairs with zero frequencies and thus will not contribute to the estimate. Second, to use the value pairs in the sample set to estimate the missing neighboring values is inappropriate because the uniformity distribution assumption does not hold in general; i.e., there may be no correlation among frequencies of neighboring attributes.

As was observed by Matias et al. [MVW98], a more appropriate approach is to randomly sample the cumulative distribution function (cdf) instead of the fdf. The cdf is the ordered set of value pairs $\{\langle d_0, c_0\rangle, \langle d_1, c_1\rangle, \langle d_2, c_2\rangle, ..., \langle d_{N-1}, c_{N-1}\rangle\}$ where c_i is the cumulative frequency of value d_i, which is the number of tuples $t \in R$ with $t.A \leq d_i$; i.e., $c_i = \sum_{j=1}^{i} f_j$. For a range query $q(a \leq A \leq b)$, the cdf can be used to compute the number of tuples as $c_b - c_a$. The intuition behind this is that each sample of the cdf $\langle d_i, c_i\rangle$ contains the sum of all frequencies of the fdf that is smaller than d_i; i.e., even if the random sample misses some attribute values with large frequencies, the missing information can still be gathered in a random sample with larger d_i. Furthermore, since the cdf is a monotonically increasing function, it is easier to have better approximations than using the fdf. From a query cost point of view, it is worth noting that the worst case complexity of executing a range query using the cdf is $O(1)$ whereas using the fdf is $O(n)$. This property was exploited in the context of multidimensional data cubes by Ho et al. [HAMS97] to ensure constant query cost.

To show how to apply randomly sampled cdf for range query estimations, Figure 1.(b) shows the cdf for the underlying fdf of Figure 1.(a). Let us take the same three samples as before but from the cdf, $S_{cdf} = \{\langle 2, 2\rangle, \langle 5, 14\rangle, \langle 8, 20\rangle\}$. The estimated query result of $q(2 \leq X \leq 7)$ is $c'_7 - c'_2$, where c'_2 and c'_7 are estimates of the actual values c_2 and c_7, respectively. Because the cdf with attribute value 2 is contained in the sample set, c'_2 can be determined exactly as $c'_2 = c_2 = 2$. c'_7 can be estimated in many ways. The simplest way to estimate the value of c'_7 is to assign it to the nearest sample point in the sample set S_{cdf}, which is $c_8 = 20$. The final estimation is now $c'_7 - c'_2 = 20 - 2 = 18$. The relative error is $E_{rel} = |13 - 18|/13 \times 100 = 38.5\%$, which is smaller then the previous estimate using the random sample set of fdf. But choosing the nearest neighbor point $\langle d_i, c_i\rangle$ in the sample set to represent the query point c'_j in some cases may not be accurate. In this example, to estimate c'_7 we used the nearest neighbor $c_8 = 20$ in the sample set S_{cdf} but the actual c_7 is 15 and thus the error is 5.

One can reduce the error of the estimation further by using a more elaborate method, namely, *interpolation*. For example, to estimate c'_i, we take the two nearest neighbor samples $\langle d_{low}, c_{low} \rangle$ and $\langle d_{high}, c_{high} \rangle$ from the sample set S_{cdf} such that d_{low} is the largest value smaller than or equal to d_i and d_{high} is the smallest value greater than or equal to d_i. To interpolate c_i, we use

$$c'_i = c_{low} + (c_{high} - c_{low}) \times \frac{d_i - d_{low}}{d_{high} - d_{low}} \tag{2}$$

In the above example, to estimate c'_7 we take two samples from S_{cdf} $\langle d_{low}, c_{low} \rangle = \langle 5, 14 \rangle$ and $\langle d_{high}, c_{high} \rangle = \langle 8, 20 \rangle$. Using Equation 2, $c'_7 = 14 + (20 - 14) \times (7 - 5)/(8 - 5) = 18$. As a result, by using the interpolation method together with the random sampling of the cdf, the E_{rel} reduces to $|13 - 16|/13 \times 100 = 23\%$.

Random sampling of the cdf gives better estimation than sampling the fdf in the above example because sampling the fdf has the potential of choosing sample sets that contain less information if the distribution of the fdf value pairs is sparse with respect to the attribute domain. Sampling the cdf instead of the fdf can partially remedy this problem. However, several problems still remain with regard to estimation based on random sampling. For example, with the above cdf sample set, query $q(0 \leq X \leq 3)$ will result in an estimated answer of zero because $\langle d_2, c_2 \rangle$ is the nearest sample to both d_0 and d_3. The error can be greatly reduced if the sample set contains the sample $\langle 3, 14 \rangle$. The above example illustrates one of the main problem with random sampling, namely, that sampling is performed in a random manner whereas the distribution of the underlying data may not be uniform. Hence the sample set may not capture the necessary information in terms of the underlying data distribution. Also note that sampling on the cdf does not always give better estimates than sampling on fdf. This is especially true when the underlying distribution has near uniform or clusters of uniform distributions. What is needed is a method that samples the data in a manner that mirrors the underlying distribution.

One approach to achieve this is to sample more values with higher frequencies than values with lower frequencies; i.e., samples should not be taken randomly by attribute values but according to their corresponding frequencies. However, in general, such a sampling strategy is hard if the underlying distribution is not known in advance. Alternatively, the fdf values can be sorted by frequencies and then apply a biased sampling; i.e., more samples are taken from the more frequent values. In fact, Poosala et al. [IP95, PIHS96] have shown that in constructing histograms, the best results are attained when values are *sorted* and histogram buckets are chosen to favor more frequent values. In the following section we describe a simple approach that captures the essence of the underlying distribution without a priori knowledge of the distribution and without the need for sorting.

2.3 The Golden Rule for Random Sampling

We now propose to adapt a very efficient sampling mechanism for capturing the relevant information, irrespective of the underlying distributions. In a letter to Ulam in 1947, von Neumann suggested a sampling rule often referred to as "the golden rule of sampling" (Los Alamos Science, p.135, June 1987, [Eck87]). Since the rule is stated for continuous probability distributions, we now digress, and assume that the distribution function is a continuous probability function, and later show how to adapt the proposed technique to a discrete domain. The initial development is based on [Com95].

Consider a non-uniform probability distribution function $f(x)$. As discussed earlier, uniformly sampling such a distribution will not yield a set of random samples that effectively capture the essential information of $f(x)$. For example, if $f(x)$ has a Zipf distribution, sampling may yield most samples that do not carry any information or have zero values. By using some transformation of the fdf, it it possible to get better random sample sets; i.e., more samples with higher frequencies than samples with lower frequencies. Initially we assume a continuous domain where a *probability distribution function* (pdf) instead of a frequency distribution function is used to describe the data distribution. Let us assume that there exists a monotonically increasing transformation function $y(x)$, with $dy/dx > 0$ for all x. The transformation must conserve the probability that random variable x' occurring in dx about x must be the same as the probability of a random variable y' occurring in dy about y. More precisely, there must be a one-to-one relationship between x and y and y must exist. From the definition of the probability distribution function, $f(x)$ and $g(y)$ are

$$f(x)dx = prob(x \leq x' \leq x + dx) \tag{3}$$

$$g(y)dy = prob(y \leq y' \leq y + dy) \tag{4}$$

where $g(y)$ is the probability distribution of the transformation $y(x)$.

The transformation implies that these probabilities must be equal, thus we have

$$f(x)dx = g(y)dy \tag{5}$$

and we can solve $g(y)$:

$$g(y) = f(x)/[dy/dx] \tag{6}$$

One of the most important transformations that satisfies the above condition is the *cumulative distribution function* (*cdf*):

$$y(x) = F(x) = \int_{-\infty}^{x} f(x')dx' \tag{7}$$

In this case, we have $dy/dx = f(x)$ and more importantly,

$$g(y) = 1, 0 \leq y \leq 1 \tag{8}$$

Equation 8 shows that the cdf is uniformly distributed on [0,1], independently of the pdf $f(x)$. This result has significant impact on how we can sample from an arbitrary pdf. Because the random variable x and the cdf $F(x)$ are one-to-one correspondent, we can sample x by first sampling $y = F(x)$ and then use the reverse function to invert $F(x)$ to x, or $x = F^{-1}(y)$. And from Equation 8 we know the cdf is uniformly distributed on [0,1]. Therefore, by using a uniform random number generator $U[0,1]$ to generate a sample ξ from the cdf $F(x)$ we can have a sample about x by $x = F^{-1}(\xi)$. The golden rule of sampling [Eck87] can therefore be summarized as follows.

1. Given a pdf $f(x)$, generate the cdf $F(x)$.
2. Generate a random number y between $[0, c_{N-1}]$ (the range of the cdf is between 0 and c_{N-1}) then equate it to the cdf; i.e., $y = F(x)$.
3. Solve for x by inverting the cdf; i.e., $x = F^{-1}(y)$.

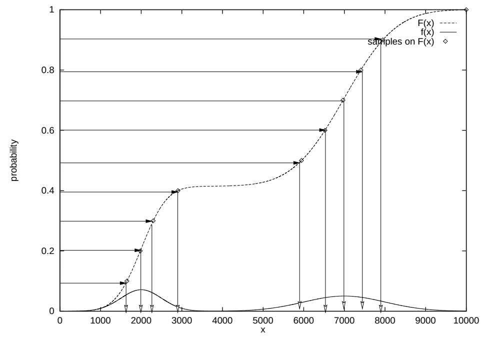


Fig. 2. Sampling using the inverse of the cdf

Let us illustrate the essence of the above rule using a simple example. Consider a function $f(x)$ as illustrated in Figure 2, which is not uniform. We generate the cdf, $F(x)$, and then take random uniformly distributed samples on the cdf (i.e., uniformly sample the y axis). Since the cdf is uniformly distributed, the sample values capture its distribution. Now for each random sample, we calculate its inverse with respect to $F(x)$; i.e., $F^{-1}(x)$. As the figure illustrates the x values chosen essentially capture the non-uniformity of the underlying function $f(x)$. In particular, more samples end up being taken in the parts of the $f(x)$ that have more information.

2.4 Adapting the Golden Rule to Discrete Domains

Unfortunately the golden rule holds for a continuous probability distribution function, whereas databases in general have discrete frequency distributions. The main problem that arises, if we try to use the golden rule in our domain, is that the cdf is not a 1-to-1 mapping. In fact, the cdf in a discrete domain can be viewed as a monotonically nondecreasing step function. Consider one such step as shown in Figure 3. Several random samples in the frequency domain, e.g., c', c'', c''' all correspond to the same value in the attribute domain, namely d_{i+1}. This arises due to the fact that $F(x)$ is not a 1-to-1 function with respect to $f(x)$ in a discrete domain. The main challenge now is to determine the appropriate sample when

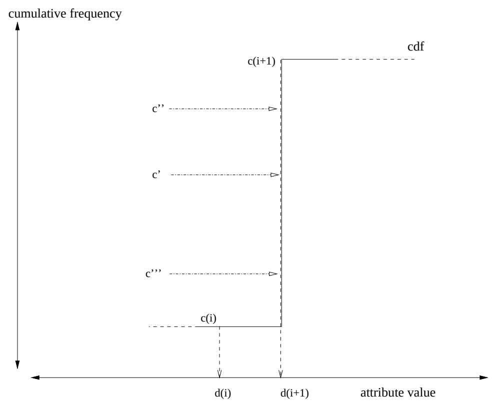


Fig. 3. Sampling in the discrete domain

we encounter such a situation. There are basically two possible sample pairs $\langle d_i, c_i \rangle$ and $\langle d_{i+1}, c_{i+1} \rangle$ as shown in Figure 3. If we choose $\langle d_i, c_i \rangle$, we refer to that method as *sampleLo*. Alternatively, if we choose $\langle d_{i+1}, c_{i+1} \rangle$, we call it *sampleHi*.

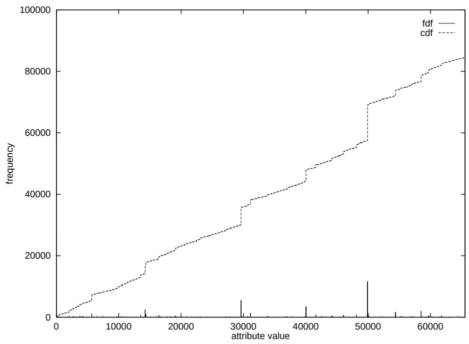
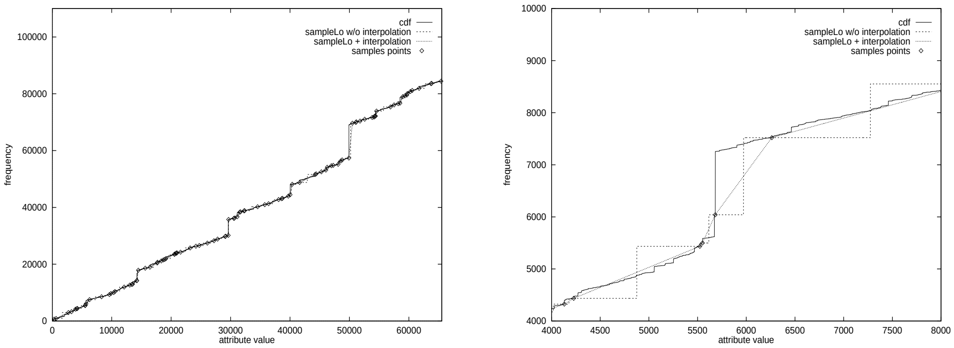


Fig. 4. fdf and cdf with Zipf distribution

We conducted a simple experiment to test the relative benefits of the different sampling approaches identified. The setting of this experiment is as follows. The domain of the attribute values is from 0 to $2^{16} - 1$. The number of tuples in the relation is 10^5. We employ the Zipf distribution in this experiment [Zip49, Poo97] with $z = 0.1$. Figure 4 shows an example fdf and cdf of one such attribute in the given relation. The vertical lines on the x-axis correspond to the frequencies associated with each attribute value and the dashed line is the resulting cumulative frequency distribution.

Figure 5.(a) shows the result of sampling the cdf with the *sampleLo* method (similar results can be obtained by using *sampleHi*). We take 100 samples from the cdf using the golden rule and depict them using the diamond shape. As discussed earlier, a query could either use the closest sample or interpolate between the two surrounding samples. We compare the approximation approaches both with and without interpolation. In Figure 5 and the subsequent figures in this section, we approximate the cdf as follows: for each attribute value, we use the samples to estimate the value of the cdf. For the purpose



(a) entire range (b) close up

Fig. 5. Approximation with sampleLo method with or without interpolation

of analyzing different sampling approaches, we concentrate on a detailed part of Figure 5.(a), namely the attribute range of [4000 : 8000] as shown in Figure 5.(b). The dashed line is the approximation of the cdf using the sampling rule *sampleLo*. It shows a step function because *sampleLo* uses the nearest neighbor in the sample set S_{cdf} to represent the values across the attribute value domain, hence a flat line region lies on both sides of each sample point. The dotted line represents the effect of using interpolation to answer queries based on the *sampleLo* method. In theory, the more the approximated cdf is similar to the original cdf, the more accurate the range query estimation would be. One can see how the interpolation method results in a smoother approximation, which is closer to the original cdf. However, both techniques, *sampleLo* with and without interpolation do not approximate sharp changes in the cdf very well. Recall that sharp changes are frequent in Zipf distribution. As Figure 5.(b) illustrates for the sudden change near 5700, *sampleLo* with interpolation conservatively approximates the sudden change with a gradual linear interpolation, which may give rise to higher errors.

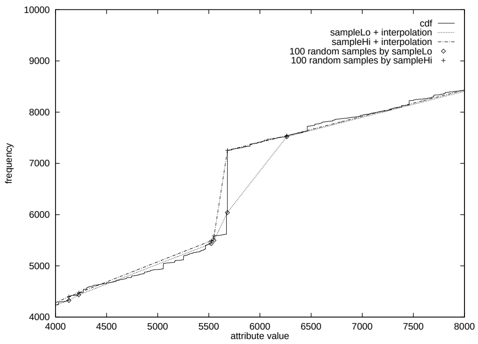


Fig. 6. Estimations using sampleLo and sampleHi sampling techniques

One reason for the large estimation error when using either the *sampleLo* or *sampleHi* method is that a single sample is taken at the sharp changing edges of the cdf. If only one sample is taken, then the estimation error between this sample and the next one is still large due to the large frequency difference. In fact, as depicted in Figure 6, neither *sampleLo* nor *sampleHi* produces good estimates if the query points fall between the sample at the sharp edge and the next neighboring sample even with interpolation.

One simple modification of the sampling method is when the first sample c' with $d_i \leq d' \leq d_{i+1}$ is taken, insert $\langle d_i, c_i \rangle$ into the sample set S_{cdf} as in *sampleLo*; if another sample c'' occurs in the same range with $d_i \leq d'' \leq d_{i+1}$, insert the other sample $\langle d_{i+1}, c_{i+1} \rangle$ into the sample set S_{cdf} as in *sampleHi* (see Figure 3). The intuition is that if the frequency f_i of d_i is large enough, there is a greater chance that the cdf range from c_i to c_{i+1} will be sampled more than once. If that is the case, we should take both $\langle d_i, c_i \rangle$ and $\langle d_{i+1}, c_{i+1} \rangle$ in order to best approximate the original cdf. However, if there is yet another sample c''' such that $d_i \leq d''' \leq d_{i+1}$, then since we have already taken both $\langle d_i, c_i \rangle$ and $\langle d_{i+1}, c_{i+1} \rangle$, no new information can be gained by taking the same sample again, we simply discard c''' and replace it by drawing another random sample. We refer to this modified sampling method as the *golden estimator*.

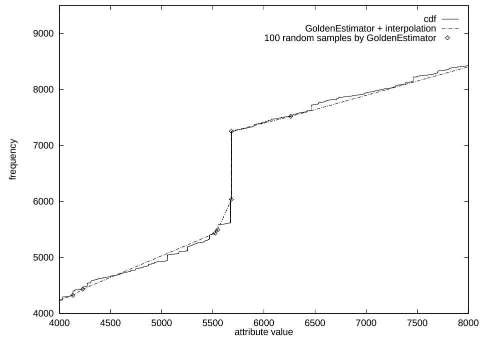


Fig. 7. Estimations using the new golden estimator techniques

In Figure 7 we show the result of using the golden estimator for the same distribution used earlier. The golden estimator curve tracks the cdf curve closely. Furthermore, note how the sharp changing edge near 5700 is well approximated using golden estimator.

Figure 8 depicts the algorithm that is used for drawing random samples based on the golden estimator method. The function has two input parameters: number of samples and the cdf of the target attribute value tuples in a relation. The main loop of the algorithm generates a random number s between 0 and the maximum value of the cdf. We then identify $\langle x_i, c_i \rangle$ in the cdf such that x_i is the largest value for which $s \geq c_i$ and $\langle x_{i+1}, c_{i+1} \rangle$ such that x_{i+1} is the smallest value for which $s \leq c_{i+1}$. Depending on the samples accumulated so far, first $\langle x_i, c_i \rangle$ is included and if that already exist and $\langle x_{i+1}, c_{i+1} \rangle$ is distinct then $\langle x_{i+1}, c_{i+1} \rangle$ is added to the sample set. The algorithm terminates after the specified number of samples are drawn.

```
PROCEDURE GoldenEstimator (noOfSampleNeeded, cdf)
BEGIN
    sampleSet = NULL; (* keep samples *)
    sampleCount = 0; (* keep track of the number of samples taken *)
    s = 0; (* random sample *)
    xLo = 0; (* low attribute value *)
    xHi = 0; (* high attribute value *)
    WHILE sampleCount ≤ noOfSampleNeeded
        s = pick a random number between 0 and c_{n-1} in cdf;
        xLo = Max(x_i), where c_i ≤ s, 0 ≤ i ≤ n − 1;
        xHi = Min(x_i), where c_i ≥ s, 0 ≤ i ≤ n − 1;
        IF ⟨xLo, c_xLo⟩ ∉ sampleSet THEN
            add ⟨xLo, c_xLo⟩ to sampleSet;
            sampleCount + +;
        ELSE
            IF ⟨xHi, c_xHi⟩ ∉ sampleSet THEN
                add ⟨xHi, c_xHi⟩ to sampleSet;
                sampleCount + +;
            END; (* IF *)
        END; (* IF *)
    END; (* WHILE *)
    RETURN sampleSet;
END; (* GoldenEstimator *)
```

Fig. 8. Golden Estimator Sampling Algorithm

```
PROCEDURE queryEstimate (a, b, sampleSet) (* query = q(a ≤ X ≤ b) *)
BEGIN
    c'_a, c'_b; (* estimates of c_a and c_b *)
    dLo, dHi; (* low/high attribute value *)
    cLo, cHi; (* low/high cumulative value *)
    (* find nearest samples that lies on either side of a*)
    cLo = Max(c_i), dLo = Max(d_i), where d_i ≤ a, ⟨d_i, c_i⟩ ∈ sampleSet;
    cHi = Min(c_i), dHi = Min(d_i), where d_i ≥ a, ⟨d_i, c_i⟩ ∈ sampleSet;
    (* interpolate c_a using Eq 2 *)
    c'_a = clo + (cHi − Clo) ∗ (a − dLo)/(dHi − dLo);
    (* similarly for b*)
    cLo = Max(c_i), dLo = Max(d_i), where d_i ≤ b, ⟨d_i, c_i⟩ ∈ sampleSet;
    cHi = Min(c_i), dHi = Min(d_i), where d_i ≥ b, ⟨d_i, c_i⟩ ∈ sampleSet;
    c'_b = clo + (cHi − Clo) ∗ (b − dLo)/(dHi − dLo);
    RETURN c'_b − c'_a;
END; (* queryEstimate *)
```

Fig. 9. The Golden Estimator Query Estimation Algorithm

Figure 9 illustrates the algorithm that is used to estimate the answer for a given range query $[a : b]$ from the set of random samples. As shown, for each end point a and b we find the corresponding estimates for cdf as c'_a and c'_b. c'_a is the interpolated value from the nearest random samples that lie on either side of a in the random sample set. Similarly for c'_b. The query result is estimated to be the difference $c'_b - c'_a$.

3 The Golden Estimator for Spatial Domains

The golden estimator described in Section 2 is applicable only for a single attribute domain. We now extend the golden estimator for multi-attribute domains. For convenient we concentrate on range queries for two attributes only. This case is of particular importance in spatial databases. Furthermore, simple extensions to more attributes are not difficult.

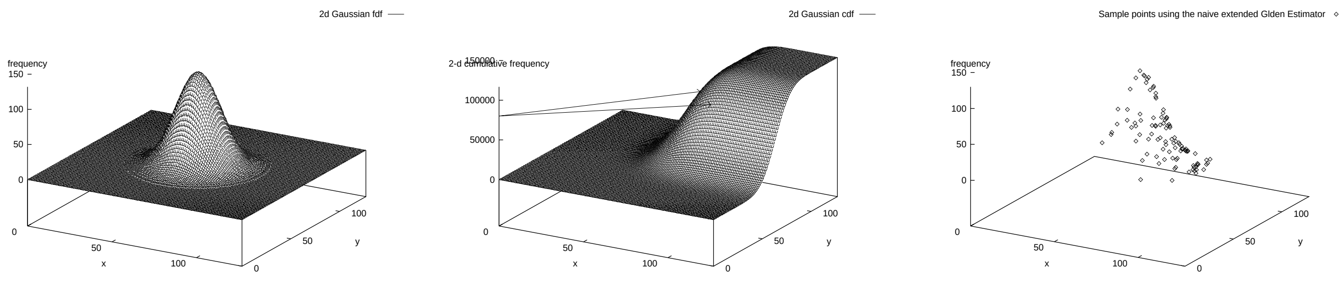
3.1 Preliminaries for Multi-attribute Domains

Given a relation R with attributes X and Y, we are interested in estimating the size of a range query on attribute X and Y. We assume the discrete domains D_x and D_y of spatial attributes X and Y are $\{x_0, x_1, x_2, ..., x_{N-1}\}$ and $\{y_0, y_1, y_2, ..., y_{M-1}\}$, respectively. A range query is specified using a range predicate of the form $(a \leq X \leq b \quad AND \quad c \leq Y \leq d)$, where a and b are constant values in the discrete domain D_x and c and d are constant values in the discrete domain D_y. The *frequency* $f_{(x,y)}$ of value pair $d_{(x,y)}$ for attributes X and Y is the number of tuples $t \in R$ with $t.X = x \ \ and \ \ t.Y = y$. If no tuples t in R have $t.X = x \ \ and \ \ t.Y = y$, then $f_{(x,y)} = 0$; i.e., zero frequency. The *frequency distribution function* (fdf) is a set of value pairs $fdf = \{\langle d_{(0,0)}, f_{(0,0)}\rangle, ..., \langle d_{(i,j)}, f_{(i,j)}\rangle, ..., \langle d_{(N,M)}, f_{(N,M)}\rangle\}$. The cumulative distribution function (cdf) is the ordered set of value pairs $\{\langle d_{(0,0)}, c_{(0,0)}\rangle, ..., \langle d_{(i,j)}, c_{(i,j)}\rangle, ..., \langle d_{(N,M)}, c_{(N,M)}\rangle\}$ where $c_{(i,j)}$ is the cumulative frequency of value $d_{(i,j)}$, which is the number of tuples $t \in R$ with $t.X \leq x \ \ and \ \ t.Y \leq y$; i.e., $c_{(i,j)} = \sum_{x=0}^{i} \sum_{y=0}^{j} f_{(x,y)}$.

In the following sections, we will use sample pairs $\langle d_{(i,j)}, c_{(i,j)}\rangle$ to approximate the fdf. Our approach is based on the cumulative frequency distribution, which can be easily obtained from the frequency distribution function. As before, the main challenge in query size estimation is to use a limited amount of storage to approximate the fdf. This approximation is then used to estimate the number of tuples that are in the range specified by the query.

3.2 Naive Approach

In Section 2 we demonstrated how to use the cdf to sample a single attribute data distribution. In this section we show a direct extension of the golden estimator for multi-attribute data distributions. For simplicity, we discuss an extension of the golden estimator in two dimensions. Consider the two



(a) (b) (c)

Fig. 10. Sampling a 2-d Gaussian distribution

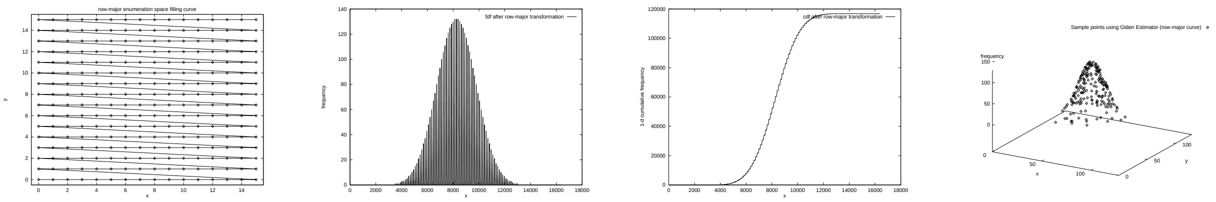
dimensional cdf of a two dimensional fdf $f(x_1, x_2)$.

$$y(x_1, x_2) = F(x_1, x_2) = \int_{-\infty}^{x_1} \int_{-\infty}^{x_2} f(x'_1, x'_2) dx'_1 dx'_2 \tag{9}$$

At first, we may think that the two dimensional cdf, as defined in Equation 9, can be used as the two dimensional transformation function F in a similar manner as the golden estimator technique discussed in the previous section by replacing the $[dy/dx]$ in Equation 6 with the Jacobian of F.

Figure 10.(a) depicts a synthetic two dimensional fdf, which is a 2-d Gaussian distribution. Figure 10.(b) depicts the cdf of the distribution in Figure 10.(a). According to the golden estimator, a sample is taken by randomly selecting a value in the domain of F, which is the z axis in Figure 10.(b), then applying the inverse transform F^{-1} to get a sample in the original domain. In Figure 10.(b), a domain value $80,102$ of F is chosen and the inverse F^{-1} mapping is applied in the same manner as discussed in Figure 2.(b). However, there are two such points, $[70, 85]$ and $[85, 70]$, in the cdf with values 80,102 as depicted. At this moment, let us solve this one-to-many mapping problem by randomly selecting one of the many possible mappings. The result is a sample point $[70, 85]$ in the original domain. Figure 10.(c) shows 100 randomly selected samples plot in the original fdf domain. Note how the sample set keeps the shape of the original fdf in Figure 10.(a). However, more samples are taken toward the opposite direction to the origin; i.e., the sample set is biased not only by the fdf distribution but also by the position in space. A proper sample set should only be biased by the fdf distribution.

From the above example, we see two problems. First, in Equation 6, replacing $[dy/dx]$ with the Jacobian of F will not result in $g(y) = 1, 0 \leq y \leq 1$, as in Equation 8. This indicates that the values of the two dimensional cdf are not uniformly distributed. Second, the two dimensional cdf in Equation 9 is not a one-to-one mapping function of the fdf (as in the example of Figure 10.(b)). The above problems prevent the use of a uniform random number generator $U[0, 1]$ to generate a sample ξ from two dimensional cdf F, then use the inverse function F^{-1} to convert to a sample in the two



(a) (b) (c) (d)

Fig. 11. 2-d sampling with the golden estimator and row-major transformation

dimensional fdf. We conclude that the two dimensional cdf cannot be used by the golden estimator technique to capture the underlying two dimensional fdf without further modifications.

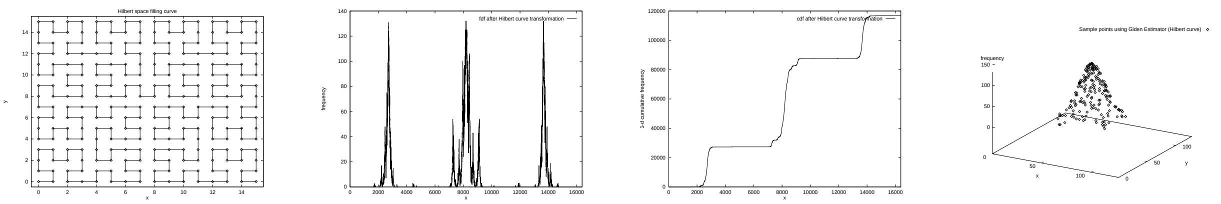
3.3 Extending the Golden Estimator Technique for Multiple Attributes

Although the results in the previous section seem to discourage the application of the golden estimator in a two dimensional domain, we propose modifications to get around the above problems by using an alternative transformation. Our intuition is that, since the golden estimator works in single attribute spaces, we can transform a two dimensional space into a single attribute space, then apply the golden estimator in the transformed one dimensional domain. The question then arises is whether this approach will result in a sample set that characterizes the frequency distribution of the original spatial data. An important requirement is that the transformation must be a one-to-one mapping. Fortunately, it can be shown that there always exists a one-to-one mapping from a multiple dimensional space to a one dimensional space. This indicates that the golden estimator can indeed be applied to two dimensional spaces.

We now focus our discussion on two dimensional discrete spaces. First, we discuss how to find a proper transformation, which is a one-to-one mapping, from two dimensional space to one dimensional space, and vice versa. There exist many space filling curves which transform from a multi-dimensional space to a single dimensional space; e.g., the row-wise enumeration (row-major ordering), the z-ordering [OM84], Hilbert curve [Jag90], Grey ordering [Fal86], etc. Any of these space filling curve functions can be applied as the multi-dimension to one dimension transformation.

To apply the golden estimator in spatial database, we use the following steps to generate sample sets that can capture the original data distribution without a priori information.

1. For multi-dimensional dataset fdf^2, we apply multi-dimensional to one dimensional transformation $F_{2d \to 1d}$ on fdf^2 to get one dimensional fdf fdf^1.
2. Generate the one dimensional cdf (i.e.; $F(x)$) cdf^1 from fdf^1.
3. Take a uniformly distributed sample set S^1_{cdf} from the frequency domain of cdf^1 (i.e.; uniformly sample the y axis).



(a) (b) (c) (d)

Fig. 12. 2-d sampling with the golden estimator and Hilbert curve transformation

4. For each random sample in S^1_{cdf}, we calculate its inverse with respect to $F(x)$ (i.e., $F^{-1}(x)$) and get sample set S^1_{fdf}. S^1_{fdf} captures the distribution of fdf^1.
5. For each sample in set S^1_{fdf}, apply the inverse transformation $F^{-1}_{2d\to 1d}$on the one dimensional sample set S^1_{fdf} to get multi-dimensional sample set S^2_{fdf} with respect to fdf^2. S^2_{fdf} captures the distribution of fdf^2.
6. For each sample in multi-dimensional sample set S^2_{fdf}, compute the multi-dimensional cumulative frequency of each sample and store it in the final cumulative frequency sample set S^2_{cdf} for query estimation processing.

We refer to the above technique as the *multi-attribute golden estimator* technique. Note that in Step 3 and Step 4 we use the same algorithm of the golden estimator without any modifications.

We now present examples of how the multi-attribute golden estimator technique works on the same synthetic spatial dataset depicted in Figure10.(a). Figure 11.(a) shows how the row-major enumeration function fills a two dimensional space with a one dimensional space-filling curve. Figure 11.(b) shows the one dimensional fdf after applying the row-major enumeration transformation function, on the fdf in Figure 10.(a). Each spike in Figure 11.(b) is actually a 1-d Gaussian distribution curve created by slicing in the x direction of the 2-d Gaussian distribution in Figure 10.(a). Figure 11.(c) shows the corresponding cdf of the fdf in Figure 11.(b) on which we applied the golden estimator to collect the one dimensional sample set. Figure 11.(d) shows the final two dimensional sample set in the original fdf domain after applying the inverse transformation of the row-major enumeration. Note how the sample set in Figure 11.(d) captures the original fdf in Figure 10.(a). However, unlike the results in Section 3.2, the sample set captures the fdf distribution and is not biased by the position in space.

Although the above example clearly shows the feasibility of the multi-attribute golden estimator on spatial domains, it raises another question. Do different multi-dimension to one dimension transformation functions $F_{md\to 1d}$ result in different sample sets? Moreover, is there an optimal $F_{md\to 1d}$ for the two dimensional golden estimator technique? We answer these questions by giving experimental results on a very different space filling curve function, the Hilbert curve [Jag90]. Figure 12.(a) shows how the Hilbert curve function fills a two dimensional space with a one dimensional space-filling curve. Figure 12.(b) and Figure 12.(c) show the one dimensional fdf and cdf after the Hilbert curve transformation, respectively. Note the significant differences between the fdf and cdf in Figure 11 and Figure 12 with different $F_{md \to 1d}$ applied. Figure 12.(d) shows the final two dimensional sample set in the original fdf domain after applying the inverse transformation of the Hilbert curve function. Despite the significant differences in the transformed one dimensional cdf and fdf, the Hilbert curve function and the row-major enumeration function produce almost identical two dimensional sample sets. This result supports that, without loss of generality, any of the space filling curve functions can be applied as the multi-dimension to one dimension transformation function $F_{md \to 1d}$. This is due to the unique property of the golden estimator in a single attribute space, which can always capture the one dimensional fdf^1 irrespective of the characteristics of the fdf^1. And since the inverse transformation function $F^{-1}_{md \to 1d}$ is a one-to-one mapping function from the transformed one dimensional space back to the original multi-dimensional space, the resulting multi-dimensional sample set S^m_{fdf} can always capture the the multi-dimensional fdf^m irrespective of the characteristics of the fdf^m. Although we experimentally show that any of the space filling curve functions can be applied in the proposed technique, a formal proof of the proposed theorem is still remains an open problem.

4 Performance Evaluation

In this section we conduct experiments to compare the performance of the Golden Estimator with current estimation methods in single attribute and spatial domains.

4.1 Performance Evaluation of the Golden Estimator for Single Attribute Domains

We first conduct a set of experiments to compare the performance of the golden estimator with other estimation methods in a single attribute domain. We choose two of the most recent and competitive methods namely the histogram method MaxDiff(V,F) proposed by Poosala et al. [PIHS96] and the wavelet based approach proposed by Matias et al. [MVW98]. We also compare the performance with the *tuple sampling* technique. In general, the tuple sampling techniques work as follows: A random sample of tuples is taken from the relation a priori or during the query processing phase. Let sample set S be taken from the relation R. When a range query Q is presented, if the subset t of S satisfied Q, the selectivity is estimated as $\frac{|R| \times |t|}{|S|}$.

Experimental Setting We consider a relation R with 10^5 tuples. The attribute A, on which range queries are specified, is an integer with range from 0 to $2^{16} - 1$. Since the Zipf distribution has been shown to best describe data in the real world, we use the Zipf distribution for most of the experiments. To ensure that there is no correlation between attribute values and their frequencies, we randomly assign the Zipf frequencies to the attribute values. We use the algorithm described in [Zip49] for generating the Zipf distribution.

In the experiments we explore the performance of the different methods on query sets with different selectivities. The selectivity sel of query $q(a \leq X \leq b)$ is defined as $sel = (b-a)/N$, which is the percentage of the query range to the entire attribute domain. We varied the selectivity experiment in steps of 0.01 from 0.01 to 0.8; i.e., an experiment was conducted for selectivity 1%, 2%, ..., 80% of the attribute domain. For each such selectivity, a query set of 5000 random range queries was generated. The error measure used to compare the different methods is the average relative error for the query set Q; i.e., if we assume $Q = \{q_1, q_2, ..., q_m\}$, then

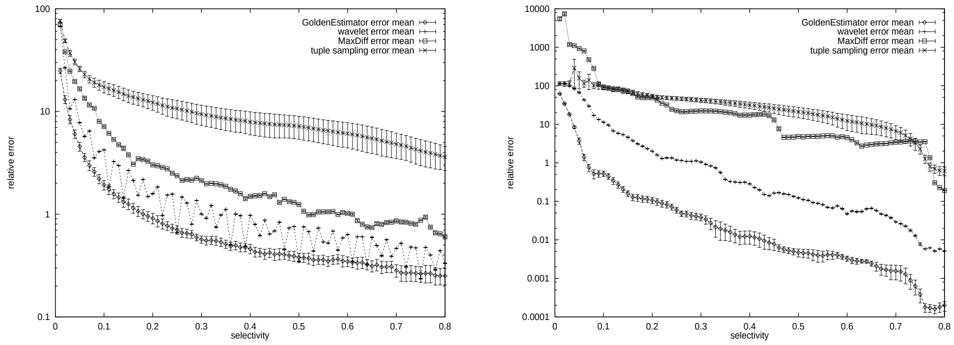
$$\overline{E}_{rel}(Q) = \frac{\sum_{i=1}^{m} E_{rel}(q_i)}{m} \tag{10}$$

A problem that has been observed [MVW98, PIHS96] with this equation is that when the actual result size is zero, the relative error E_{rel} becomes infinity. For computational reasons, in such cases, we readjust E_{rel} from infinity to 100%. Also note that the E_{rel} is not limited to 100% and can exceed 100%. This is typically the case when the actual result size s is small as in Equation 1. In all figures, the relative error is depicted on the y-axis using a logarithmic scale.

We used the independent replication method to conduct our experiments. We repeated each experiment sufficient times to obtain 90% confidence intervals, some of which are shown in the figures included in this section.

Relative Accuracy of the Golden Estimator The first experiment compares tuple sampling, the wavelet method [VWI98], the MaxDiff histogram method [PIHS96], and the golden estimator using the same Zipf distribution; i.e., $z = 0.1$ (this represents a slightly skewed distribution where $z = 0$ refers to a uniform distribution). For each technique, we compare the performance assuming the space requirement for each are identical. We use 100 random samples for the random sampling technique, 100 wavelet coefficients for the wavelet technique, 100 histogram bins for the MaxDiff histogram technique, and 200 tuple samples for the tuple sampling technique. Note that in an actual implementation of the MaxDiff histogram, two or three numbers are actually needed for each bin to store information depending on whether there is more than one attribute value in the same bin. We choose to ignore such differences and assume that only two numbers are needed for each bin and thus assigned more bins to the MaxDiff method. Also note that for the tuple sampling technique, the sample of tuples is stored as an unordered set of domain values and thus can store up to twice as many samples as compared to the golden estimator technique.

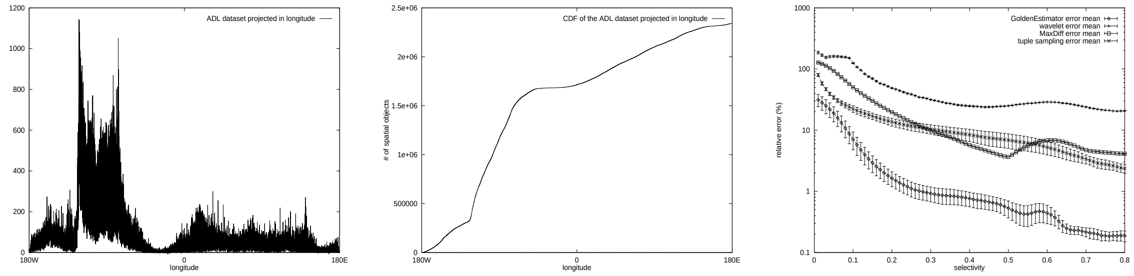
Figure 13.(a) shows the estimation error with the 90% confidence intervals. The graph clearly shows that the golden estimator results in smaller errors than the three other methods under all query selectivities. Since the confidence intervals shown are fairly tight, it also shows that taking the sample set randomly (which was done for independent replication) from the cdf has little effect on the estimation accuracy of golden estimator.



(a) z = 0.1 (b) z = 2.0

Fig. 13. Relative error using different estimation techniques

Figure 13.(b) shows the estimation error of the different methods when the underlying distribution is highly skewed. We use the same experimental setting as the previous experiment except the z value of the Zipf distribution is changed from 0.1 to 2.0; i.e., more skewed distribution. As the result shows the golden estimator has significantly less estimation errors than all other methods even with highly skewed data.



(a) ADL dataset projected in 1/100 degree longitude (b) CDF of (a) (c) relative error with ADL dataset (100 samples)

Fig. 14. Experiment results of the ADL dataset

The next dataset is a real dataset taken from the Alexandria Digital Library (ADL) [Smi96], which includes relatively large datasets derived from spatial domain. The dataset contains 2,339,771 digital spatial objects. Figure 14.(a) depicts the dataset with the centers of each spatial object projected on the longitude attribute with 1/100 degree precision; i.e., $36,000$ cells along longitude direction with each cell 1/100 degree. Note that the peaks in Figure 14.(a) generally fall in to the range between [40W:130W], which is roughly the America. It means that a big portion of the ADL collection is composed of spatial objects covering the America. Figure 14.(b) depicts the CDF of Figure 14.(b). Figure 14.(c) shows the average relative errors with the same experimental settings discussed in Section 4.1. This experiment establishes that the golden estimator still outperforms all other methods with real datasets.

The next experiment shows how different amounts of storage space will affect the estimation errors. We fix the selectivity to be 0.1 and $z = 0.1$. The storage space ranges from 20 to 2000 numbers, which is

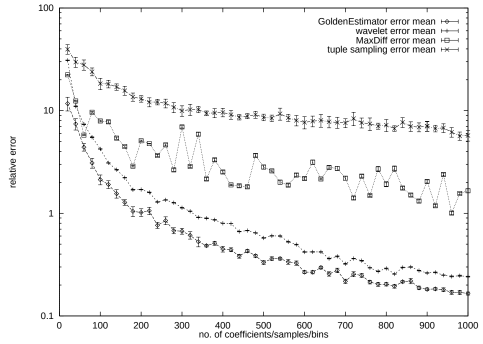


Fig. 15. Relative error with different storage space

10 to 1000 random samples, 10 to 1000 wavelet coefficients, 10 to 1000 MaxDiff histogram bins, or 20 to 2000 tuples for the tuple sampling techniques, respectively. Figure 15 shows that the golden estimator outperforms all other three approaches for different storage requirements. It also shows that both the wavelet and the golden estimator benefit from the increase in storage resulting in the reduction of overall relative error. The MaxDiff histogram method, on the other hand, does not exhibit the same degree of advantage with respect to the increase in storage.

We also study the performance of the golden estimator on different data distributions and the effect of distribution skewedness. Due to limited space, the experiment results are included in Appendix A.

4.2 Performance Evaluation of the Golden Estimator for Spatial Domains

In this section we conduct a set of experiments to compare the performance of the golden estimator with other estimation methods in spatial domain. We choose two of the most recent and competitive methods namely the Min-Skew histogram method proposed by Acharya et al. [APR99] and the wavelet based approach proposed by Matias et al. [MVW98]. The Min-Skew histogram technique is a non-parametric estimation technique that partitions the space into non-overlapping buckets with minimum sum of spatial-skews of all buckets. However, our initial experiments show that the partition process in the original Min-Skew histogram technique may stop before it generates enough buckets as desired. In such cases, the estimation accuracy of the Min-Skew histogram technique suffers considerably from not having enough histogram buckets. The problem is because under some data distributions no partition is possible for decreasing the total weighted variance of buckets; i.e., every possible partitions of buckets can only increase the the total weighted variance of buckets and the partition process stops [Poo00]. To overcome this problem, we define another weighted variance which is based on the domain area of each bucket, not the number of tuples in each bucket as suggested in [APR99], in our experiments. The modified Min-Skew histogram technique generates the desired number of buckets with minimum total weighted variance in all our experiments. By generating more buckets, the modified Min-Skew histogram technique reduces the estimation error of the original Min-Skew histogram technique. Hence we used the modified Min-Skew histogram technique for comparison in all our experiments. The wavelet technique is a parametric estimation technique that employs the wavelet transform to extract the most descriptive coefficients for the original data distributions. We also compare the performance with the *space random sampling* technique. In general, the space random sampling technique takes uniformly distributed samples from the cdf according to spatial position.

In the experiments we explore the performance of the different methods on query sets with different settings. All the queries in the query sets are 2-d *prefix* queries (also know as Type A query as defined in [VWI98]). In short, a 2-d prefix query q on relation R with spatial attributes X and Y is defined as

$$q_p(x \leq m \ \ \& \ \ y \leq n) = \sum_{i=x_0}^{x_m} \sum_{j=y_0}^{y_n} f_{(i,j)} \tag{11}$$

where $f_{(i,j)}$ is the frequency of tuples as described in Section 3.1. As we can see the prefix query is a special case of 2-d *range* type query (also know as Type B query as defined in [VWI98]), which is defined as

$$q_r(a \leq x \leq b \ \ \& \ \ c \leq y \leq d) = \sum_{i=x_a}^{x_b} \sum_{j=y_c}^{y_d} f_{(i,j)} \tag{12}$$

The reason for using prefix queries is that every range query can be translated into a linear combination of prefix queries. For example, a range query $q_r(a \leq x \leq b \ \& \ c \leq x \leq d)$ can be evaluated by four prefix queries in constant time as shown by Equation 13.

$$\begin{aligned} q_r(a \leq x \leq b \& c \leq y \leq d) = q_p(x \leq b \& y \leq d) - q_p(x \leq b \& y \leq c) - \\ q_p(x \leq a \& y \leq d) + q_p(x \leq a \& y \leq c) \end{aligned} \tag{13}$$

The same approach can be used for 2-d *delta* query (referred to Type C query as defined in [VWI98]) by replacing the values b and d in Equation 12 and 13 with $a + \triangle$ and $c + \triangle$, respectively. Note that the selectivity queries used in Section 4.1 are Type C queries with constant selectivity; i.e., $\triangle$. Similarly, other types of range query (e.g.; Type D and Type E query in [VWI98]) can also be evaluated by a linear combination of prefix queries. Due to space limitation, and since all other types of queries can be expressed in terms of prefix queries, in this section, we restrict our experiment results to prefix type queries for clarity.

Since we do not have a priori knowledge of the distributions of user queries, we use a form of biased query sets. The position of each prefix query q is biased with respect to the distribution of the spatial objects in R, as discussed in [APR99].

The first set of experiments compare the wavelet method [VWI98], the modified Min-Skew histogram method, and the golden estimator using synthetic datasets with different distributions. For each technique, we compare the performance assuming the space requirements for each are identical. For example, if we use 100 random samples for the golden estimator technique then we use 100 wavelet coefficients for the wavelet technique, 100 samples for space random sampling technique, and 50 histogram bins for

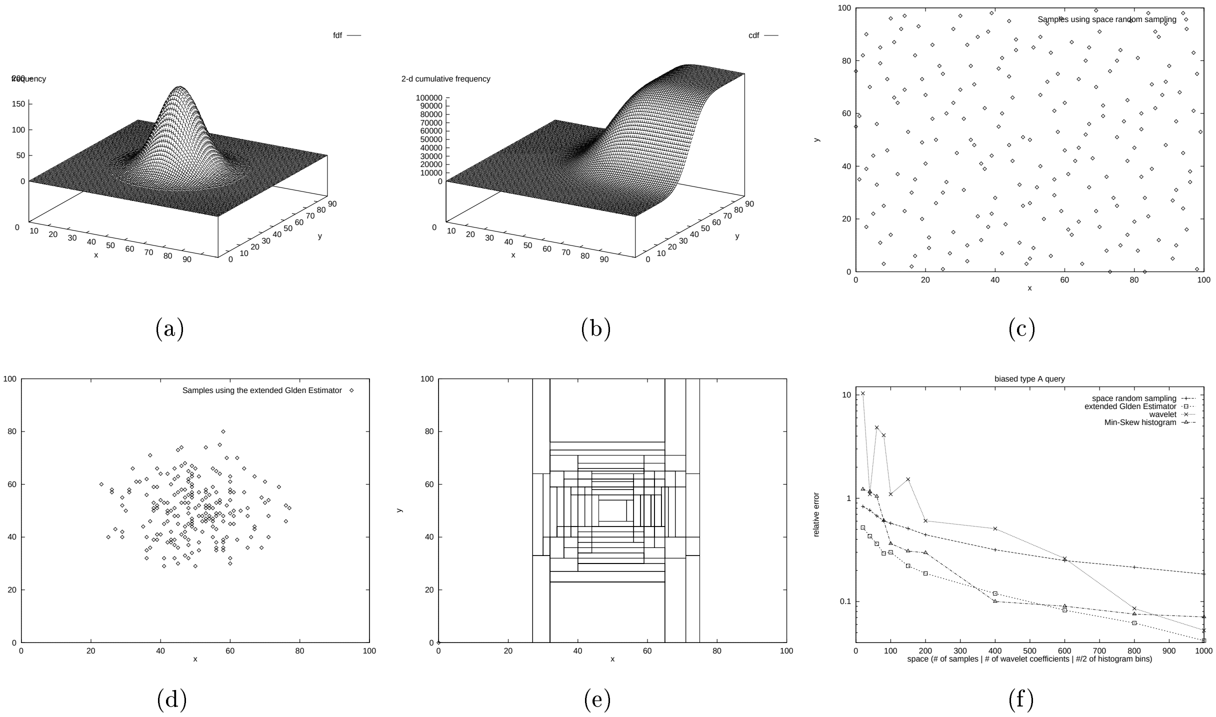


Fig. 16. Experiment results of the synthetic 2-d Gaussian distribution dataset

the modified Min-Skew histogram technique. The reason for having only half the number of histogram bins in the same storage space is because each histogram bin needs to store two points to determine the bin boundary plus other information to describe the data distribution; e.g., mean frequency of each bin.

The domains of the synthetic datasets are the same, 100 by 100 in two dimensions. The number of objects in spatial relation R is 400,000, with different spatial distributions. The first synthetic data set follows the 2-d Gaussian distribution ($\mu = 50$ and $\sigma = 10$). The fdf and cdf of the 2-d Gaussian distribution data set are depicted in Figure 16.(a) and Figure 16.(b), respectively. Figure 16.(c) depicts the sample set taken by the space random sampling technique from the 2-d Gaussian distribution datasets. Note that the sample sets are generated by a quasi-random number generator (to be more specific, Sobel sequence [PTVF93]) and not by a pseudo-random number generator. The intuition is from the Monte Carlo method, which suggests that the quasi-random number generator can generate sample sets with low discrepancy (a quantitative measure for the deviation from uniform distribution) [Nie92]. But the differences between using quasi-random number generator and the pseudo-random number generator is minimal in our study, which is in two dimensional spaces. Note that the sample sets are distributed with respect to the corresponding fdf. Figure 16.(e) shows the histogram buckets generated by the modified Min-Skew histogram technique [APR99], which consists of 100 histogram buckets. Figure 16.(f) shows the average relative error when applying different estimation techniques with different

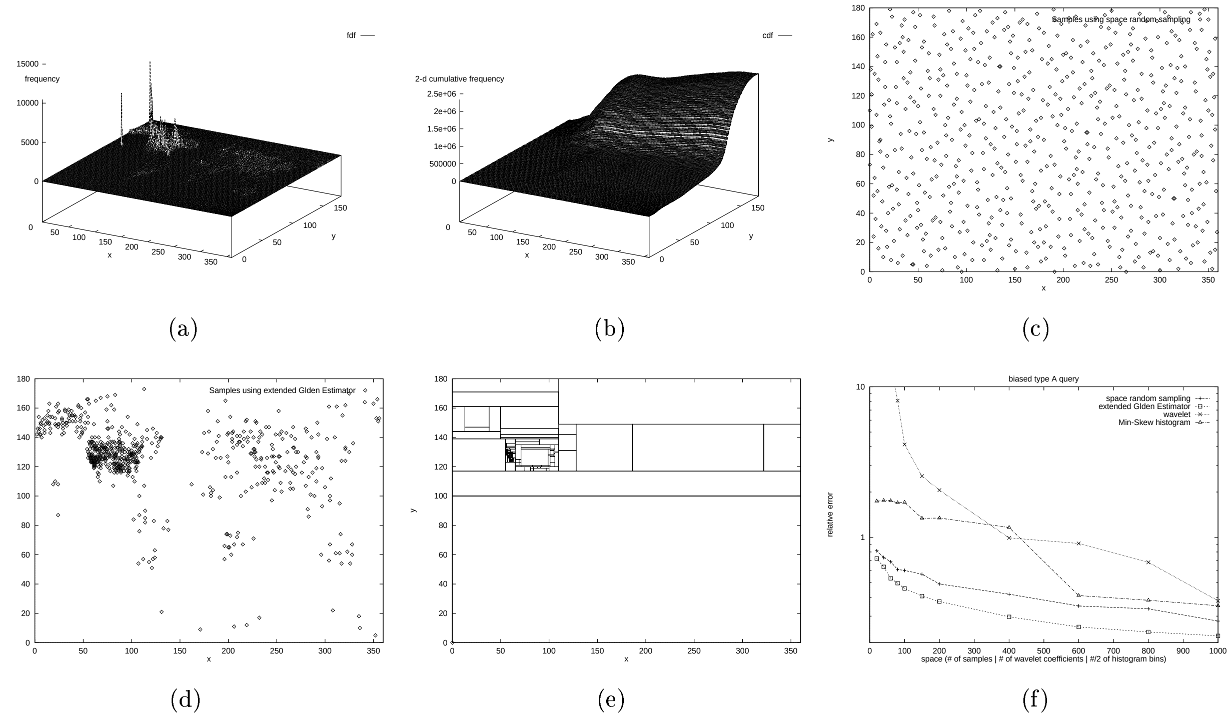


Fig. 17. Experiment results of the ADL dataset

storage space requirements. The results show that the golden estimator outperforms other estimation techniques by producing the smallest average relative errors under any storage space requirements with the 2-d Gaussian distribution.

The next data set is a real data set taken from the Alexandria Digital Library (ADL) [Smi96] as described earlier. We map the centers of the map objects into a 360 by 180 grid for estimating range queries. Figure 17.(a) and Figure 17.(b) depict the fdf and cdf of the ADL dataset, respectively.

Figure 17.(c) and Figure 17.(d) show the sample sets, each containing 600 sample points, taken by the space random sampling and the golden estimator techniques, respectively. Figure 17.(e) shows the histogram buckets generated by the Min-Skew technique, which consists of 100 histogram buckets, biased by the distribution of the ADL dataset. Figure 17.(f) shows the average relative error when applying different estimation techniques on the ADL dataset with different storage space requirements. As was the case before, the golden estimator has the least relative error compared to all other approaches.

The last real dataset is the *NJ Road* dataset taken from TIGER/Line Files [TIG97] which represents the road segments for the state of New Jersey. The dataset contains 524,486 road segment objects. We map the center of each road segment into a 100 by 100 grid for estimating range queries. Figure 18.(a) and Figure 18.(b) depict the fdf and cdf of the TIGER dataset, respectively. Figure 18.(c) and Figure 18.(d) show the sample sets, each containing 200 sample points, taken by the space random sampling and the golden estimator techniques, respectively. Figure 18.(e) shows the histogram buckets generated by

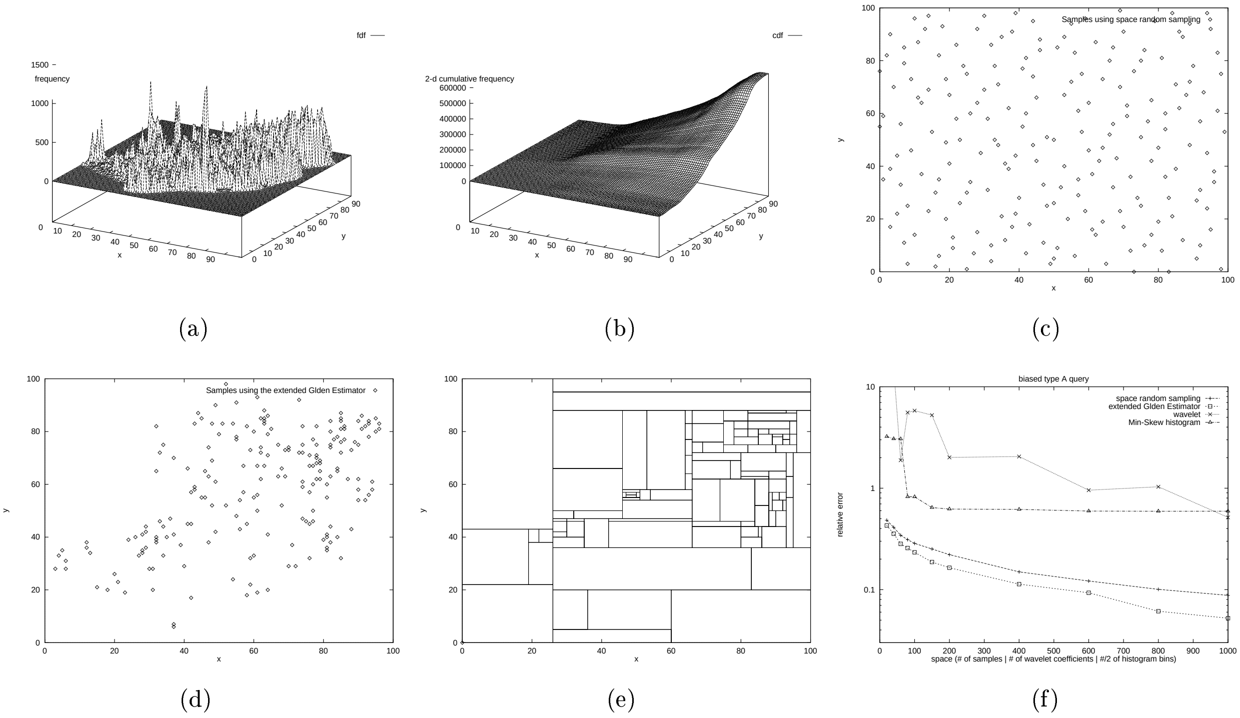


Fig. 18. Experiment results of the TIGER (NJ road segments) dataset

the Min-Skew technique consisting 100 histogram buckets, biased by the distribution of the TIGER dataset. Figure 18.(f) shows the average relative error when applying different estimation techniques on the TIGER dataset with different storage space requirements. The above results conclude that the golden estimator outperforms other estimation techniques for both synthetic and real datasets.

5 Discussion

Random sampling has long been proposed as a viable technique for query size estimation and aggregation in database management systems. Query size estimation is a critical component in DBMS for generating optimal query execution plans and feedbacks to the users. Commercially, most DBMSs employ histogram based techniques for approximation of data distributions. [APR99] pointed out that range query estimation problem in multi-dimensioal domains is very different from traditional range query estimation (i.e.; on single attribute). Therefore, new techniques other than the histogram based techniques in traditional DBMSs need to be developed.

In this paper we propose a new technique which can be used for summarizing data distributions in database. The proposed technique is based on the golden rule of sampling, proposed by von Neumann [Eck87], to randomly sample the frequency domain using the cumulative frequency distribution and yields good estimates in single attribute domains. By introducing space mapping functions, we show the proposed technique can be applied to estimate range queries in spatial domains as well.

Unlike the histogram approach, the proposed technique is based on random sampling and therefore is highly efficient both in terms of computational and storage cost. As has been stated earlier, it is mandatory for a summarization approach to work well with different data distributions if it has to be integrated in databases. This is because, databases are used for a variety of applications resulting in a very wide spectrum of data that eventually populates the databases. We conducted extensive experiments to establish the validity of the proposed technique by showing that it is an effective technique for a large variety of data distributions. We also demonstrated that the proposed technique produces less estimation errors when compared to other range query estimation techniques; i.e., MaxDiff histogram technique [PIHS96], Min-Skew histogram technique [APR99] and the wavelet based technique [VWI98, VWI98, VW99]. It should be noted that, however, the proposed technique has almost the same level of complexity as histograms whereas the wavelet based approach requires more complex computational machinery and sophistication. Although in this paper we only show that the proposed technique is suitable for range queries over single attribute and spatial attribute (two dimensional) domains, it can be easily extended for higher dimensional datasets. In this paper, we focus on estimation of range queries. Our future work will explore its applications for estimating other types of queries (join, intersection, exclusion, etc.) Also we plan to develop efficient techniques to address the problem of updating the fdf (based on the methods developed in [GMP97, GPA+98, GM98]) and cdf in a dynamic environment.

References

[AD85] H. Ahrens and U. Dieter. Sequential random sampling. *ACM Transaction Mathematical Software*, 11(2):157 – 169, June 1985.

[AGPR99] Swarup Acharya, Phillip B. Gibbons, Viswanath Poosala, and Sridhar Ramaswamy. Join synopses for approximate query answering. In *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadephia, Pennsylvania, USA*, pages 275–286. ACM Press, 1999.

[APR99] Swarup Acharya, Viswanath Poosala, and Sridhar Ramaswamy. Selectivity estimation in spatial databases. In *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadephia, Pennsylvania, USA*, pages 13–24. ACM Press, 1999.

[BKK96] Stefan Berchtold, Daniel A. Keim, and Hans-Peter Kriegel. The x-tree : An index structure for high-dimensional data. In *VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases, September 3-6, 1996, Mumbai (Bombay), India*, pages 28–39. Morgan Kaufmann, 1996.

[BS98] Daniel Barbara and Mark Sullivan. Quasi-Cubes: A Space-Efficient Way to Support Approximate Multidimensional Databases. Technical Report ISSE-TR-98-03, School of Information Technology and Engineering, George Mason University, Fairfax, Virginia, October 1998.

[CMN99] Surajit Chaudhuri, Rajeev Motwani, and Vivek Narasyya. On Random Sampling over Joins. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 263–274, Philadelphia, Pennsylvania, June 1999.

[Com95] Computational Science Education Project. Introduction to Monte Carlo Methods. *http://csep1.phy.ornl.gov/mc/mc.html*, 1995.

[Eck87] Roger Eckhardt. Stan Ulam, John Von Neumann, and the Monte Carlo Method. *Los Alamos Science*, (15, Spacial Issue):135, 1987.

[EKSX96] Martin Ester, Hans-Peter Kriegel, J. Sander, and Xiaowei Xu. Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231, 1996.

[EKX95] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification. In *Proceedings of the Advances in Spatial Databases, 4th International Symposium, SSD'95, Portland, Maine, USA*, pages 67–82, August 6-9 1995.

[Fal86] Christos Faloutsos. Multiattribute hashing using gray codes. In *Proceedings of the 1986 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 28-30, 1986*, pages 227–238. ACM Press, 1986.

[GG98] Volker Gaede and Oliver Günther. Multidimensional access methods. *Computing Surveys*, 30(2):170–231, 1998.

[GM98] Phillip B. Gibbons and Yossi Matias. New sampling-based summary statistics for improving approximate query answers. In *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 331–342. ACM Press, 1998.

[GMP97] Phillip B. Gibbons, Yossi Matias, and Viswanath Poosala. Fast incremental maintenance of approximate histograms. In *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 466–475. Morgan Kaufmann, 1997.

[GPA+98] P. B. Gibbons, V. Poosala, S. Acharya, Y. Matias Y. Bartal, S. Muthukrishnan, S. Ramaswamy, and T. Suel. AQUA: System and Techniques for Approximate Query Answering. Technical report, Bell Laboratories, Murray Hill, NJ, February 1998.

[Gut84] Antonin Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984*, pages 47–57. ACM Press, 1984.

[HAMS97] Ching-Tien Ho, Rakesh Agrawal, Nimrod Megiddo, and Ramakrishnan Srikant. Range Queries in OLAP Data Cubes. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, pages 73–88, Arizona, 1997.

[HH99] Peter J. Haas and Joseph M. Hellerstein. Ripple Joins for Online Aggregation. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 287–298, Phildelphia, Pennsylvania, June 1999.

[HHW97] Joseph M. Hellerstein, Peter J. Haas, and Helen J. Wang. Online Aggregation. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, pages 171–182, Arizona, 1997.

[HS92] Peter J. Haas and Arun N. Swami. Sequential sampling procedures for query size estimation. In *Proceedings of 1992 ACM SIGMOD international conference on Management of data*, pages 341–350, 1992.

[HS95] P. Haas and A. Swami. Sampling-based Selectivity for Joins Using Augmented Frequent Value Statistics. In *Proceedings of the 11th IEEE Conference on Data Engineering - ICDE 95*, pages 522–531, 1995.

[IP95] Yannis E. Ioannidis and Viswanath Poosala. Balancing Histogram Optimality and Practicality for Query Result Size Estimation. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 233–244, March 1995.

[Jag90] H. V. Jagadish. Linear clustering of objects with multiple atributes. In *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, Atlantic City, NJ, May 23-25, 1990*, pages 332–342. ACM Press, 1990.

[JKM+98] H.V. Jagadish, Nick Koudas, S. Muthukrishnan, Viswanath Poosala andKen Sevcik, and Torsten Suel. Optimal Histogram with Quality Guarantees. In *Proceedings of the 24th VLDB Conference - VLDB 98*, New York, USA, 1998.

[KJF97] Flip Korn, H. V. Jagadish, and Christos Faloutsos. Efficiently supporting ad hoc queries in large datasets of time sequences. In *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, pages 289–300. ACM Press, 1997.

[LJF94] King-Ip Lin, H. V. Jagadish, and Christos Faloutsos. The tv-tree: An index structure for high-dimensional data. *VLDB Journal*, 3(4):517–542, 1994.

[LNS90] Richard J. Lipton, Jeffrey F. Naughton, and Donovan A. Schneider. Practical selectivity estimation through adaptive sampling. In *Proceedings of 1990 ACM SIGMOD international conference on Management of data*, pages 1 – 11, 1990.

[MVW98] Yossi Matias, Jeffrey Scott Vitter, and Min Wang. Wavelet-Based Histograms for Selectivity Estimation. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 448–459, Seattle, 1998.

[Nie92] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* Society for Industrial and Applied Mathematics, 1992.

[OM84] Jack A. Orenstein and T. H. Merrett. A class of data structures for associative searching. In *Proceedings of the Third ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, April 2-4, 1984, Waterloo, Ontario, Canada*, pages 181–190. ACM, 1984.

[PG99] Viswanath Poosala and Venkatesh Ganti. Fast approximate answers to aggregate queries on a data cube. In *11th International Conference on Scientific and Statistical Database Management, Proceedings, Cleveland, Ohio, USA, 28-30 July, 1999*, pages 24–33. IEEE Computer Society, 1999.

[PI97] Viswanath Poosala and Yannis E. Ioannidis. Selectivity Estimation Without the Attribute Value Independence Assumption. In *Proceedings of the 23rd VLDB Conference - VLDB 97*, pages 486–495, Ahtens, Greece, 1997.

[PIHS96] V. Poosala, Y. E. Ioannidis, P. J. Haas, and E. Shekita. Improved Histograms for Selectivity Estimation of Range Predicates. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, Montreal, Canada, May 1996.

[Poo97] Viswanath Poosala. *Histogram-Based Estimation Techniques in Database Systems.* Ph. D. Dissertation, University of Wisconsin-Madison, 1997.

[Poo00] Viswanath Poosala. Private communication. 2000.

[PTVF93] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (Second Edition).* Cambrige University Press, 1993.

[Rob81] John T. Robinson. The k-d-b-tree: A search structure for large multidimensional dynamic indexes. In *Proceedings of the 1981 ACM SIGMOD International Conference on Management of Data, Ann Arbor, Michigan, April 29 - May 1, 1981*, pages 10–18. ACM Press, 1981.

[Smi96] T. R. Smith. A digital library for geographically referenced materials. *Computers*, 29(5):54–60, May 1996.

[TIG97] TIGER. 1997 TIGER/Line Files (machine-readable data files). Technical report, U.S. Bureau Of the Census, Washington, DC, 1997.

[Vit85] Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Transaction Mathematical Software*, 11(1):37 – 57, March 1985.

[VW99] Jeffrey Scott Vitter and Min Wang. Approximate Computation of Multidimensional Aggregates of Sparse Data Using Wavelets. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 193–204, Phildelphia, June 1999.

[VWI98] Jeffrey Scott Vitter, Min Wang, and Bala Iyer. Data Cube Approximation and Histograms via Wavelets. In *Proceedings of Seventh International Conference on Information and Knowledge Management - CIKM'98*, pages 96–104, Bethesda, Maryland, November 1998.

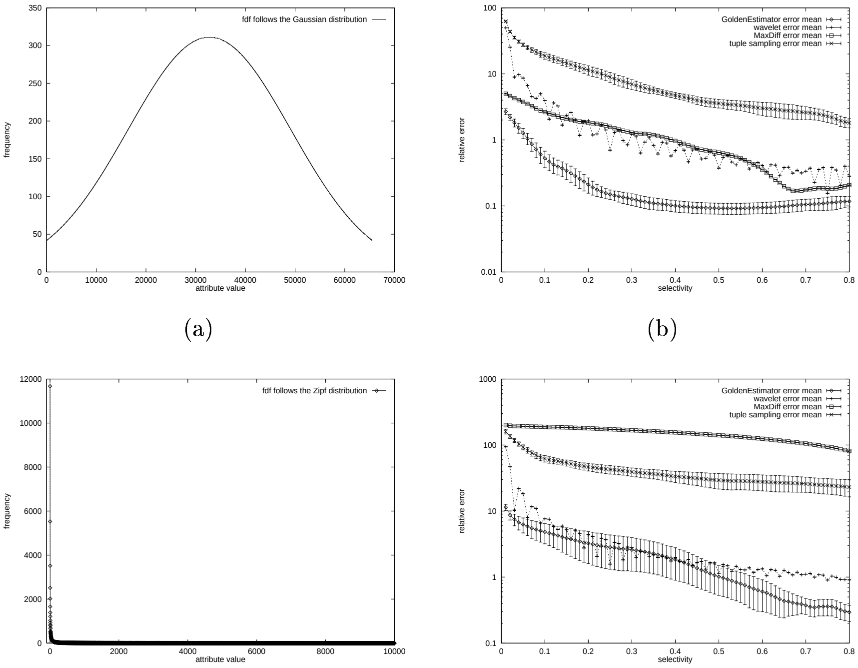
[WYM97] Wei Wang, Jiong Yang, and Richard R. Muntz. Sting: A statistical information grid approach to spatial data mining. In *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 186–195, 1997.

[Zip49] G. K. Zipf. *Human Behavior and the Principle of Least Effort.* Addison-Wesley, Reading, MA, 1949.

Appendix A: Other Performance Evaluation of the Golden Estimator

Comparison Using Other Data Distributions In the Section 4.1, we considered data with uncorrelated Zipf distribution. In this section, we explore the relative performance of the different estimation approaches using values that are correlated to each other under a variety of distributions. Such correlation may be particularly favorable for histogram based methods where the attribute domain is divided into b bins by some sort parameters under different criteria [Poo97]. Since a small amount of information is used to describe the frequencies of each attribute values in the same bin, intuitively it would seem that if the variance of values in a bin is small then the bin information will be more accurate to estimate each of the individual values. An extreme case is that all attribute values in the bin have the same frequency, then the bin keeps a single frequency value to describe all of the values without any error. The MaxDiff histogram method exploits this property by putting neighboring attribute values with large frequency differences into separate bins to reduce the average estimation error of each bin. However if neighboring attribute values have similar frequencies, histogram methods should achieve less estimation error with the same amount of storage.

The next set of experiments tests this hypothesis by comparing the performance of the MaxDiff histogram with the golden estimator, the wavelet based approach, and the tuple sampling technique. In



(a) (b)

(c) (d)

Fig. 19. Estimation error with different fdf distributions

particular, we use different data distributions where neighboring attribute values are correlated according to the distribution. Our first experiment uses a Gaussian distribution, as depicted in Figure 19.(a), with mean equal to one half of the domain size and standard deviation equal to one fourth of the domain size. Figure 19.(b) shows the result of this experiment. There are several interesting observations to make. First we note that the performance of the MaxDiff has improved when compared to the earlier uncorrelated data distributions of Section 4.1. In fact, when comparing Figure 13 and 19.(b) we note that the error has been reduced by almost an order of magnitude. This confirms our intuition that when attribute values are correlated, it is to the advantage for MaxDiff histogram method. Second we note that in fact the relative performance as compared with the result in Section 4.1, between MaxDiff and wavelet has changed. Both have relatively comparable error estimation for most of the range of selectivities. Finally, we note that the golden estimator still outperforms all other methods.

We now study data distributions that are particularly skewed with long correlated zero sequences. Such distributions are more appropriate for the wavelet based approach since it captures changes in data distribution in an elegant manner. We use a Zipf distribution with $z = 0.1$ as depicted in Figure 19.(c). Note that the values are correlated according to the Zipf distribution; i.e., smaller attribute values intentionally have higher frequencies as specified by the Zipf distribution (note that in real applications values are usually not correlated in this manner, although for some application this might be the case). The result is shown in Figure 19.(d). We first note that the performance of MaxDiff has not changed relative to the other methods. However, the error due to the wavelet based approach is now much better and in fact is comparable to the error of the golden estimator for selectivity queries in the range 0.1 to 0.5. It is, however, interesting to note that for low selectivity query, i.e., less than 4%, the golden estimator gives lower error estimation than the wavelet approach. Thus, for realistic query sizes (less then 10% selectivity), the golden estimator clearly provides better estimation.

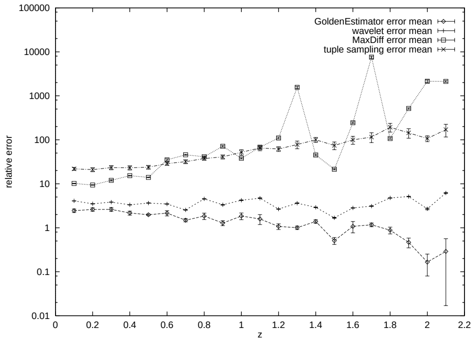


Fig. 20. Relative error with different z values of Zipf distribution

Effect of Distribution Skewedness In this section we study the effect on estimation accuracy due to the skewedness of the data distribution and different storage space. We start by showing how the skewedness of the underlying data distribution affects the estimation error. We use the same setting as the experiment in Section 4.1, i.e., uncorrelated Zipf, but fix the selectivity to be 0.1; i.e., 10% selectivity. The Zipf distribution z parameter ranges from 0.1 to 2.1, which is highly skewed. The result in Figure 20 shows that the golden estimator produces smaller relative error than the tuple sampling, the wavelet and the MaxDiff technique and the variance in estimation is relatively independent of skewedness of the underlying fdf.