Selectivity Estimation Without the Attribute Value Independence Assumption

Viswanath Poosala
poosala@research.bell-labs.com
Bell Laboratories
600, Mountain Avenue
Murray Hill, NJ 07974

Yannis E. Ioannidis*
yannis@cs.wisc.edu
Computer Sciences Department
University of Wisconsin
Madison, WI 53706

Abstract

The result size of a query that involves multiple attributes from the same relation depends on these attributes' *joint data distribution*, i.e., the frequencies of all combinations of attribute values. To simplify the estimation of that size, most commercial systems make the *attribute value independence assumption* and maintain statistics (typically histograms) on individual attributes only. In reality, this assumption is almost always wrong and the resulting estimations tend to be highly inaccurate. In this paper, we propose two main alternatives to effectively approximate (multi-dimensional) joint data distributions. (a) Using a *multi-dimensional histogram*, (b) Using the *Singular Value Decomposition* (SVD) technique from linear algebra. An extensive set of experiments demonstrates the advantages and disadvantages of the two approaches and the benefits of both compared to the independence assumption.

1 Introduction

There are several components in a database management system (DBMS) that require reasonably accurate estimates of the result sizes (or *selectivities*) of operators. Cost-based *query optimizers* use them to obtain estimates of the costs of subsequent operators and eventually of complete query execution plans. Also, *query profilers* use them to provide quick feedback to users as a means to detect some forms of semantic misconceptions before queries are actually executed. Selectivity estimation typically relies on some approximate knowledge of the database contents.

For a query involving a single attribute of a relation, its result size depends on the *data distribution* of that attribute in the database. Proposals to approximate single-attribute data distributions include histogram-based techniques [9] (the adoption of the *uniform distribution assumption* [18] being a special case of them), sampling [11], and parametric techniques [19]. The main advantages of histograms are that they incur almost no run-time overhead, they do not require the data to fit a probability distribution or a polynomial and, for *most* real-world databases, there exist histograms that produce low-error estimates while occupying reasonably small space (in the order of a few hundred bytes in a catalog). Hence, they are the most commonly used form of statistics in practice (e.g., they are used in DB2, Informix, Ingres, Microsoft, Oracle, Sybase), and have been studied quite extensively in the literature [4, 5, 6, 9, 13, 14, 16]. Our own earlier work has resulted in a taxonomy that includes both the old and several new classes of histograms, some of the latter being far more accurate than the former [16].

For a query involving two or more attributes of the same relation, its result size depends on the *joint data distribution* of those attributes, i.e., the frequencies of all combinations of attribute values in the database. Due to the multi-dimensional nature of these distributions and the large number of such attribute value combinations, direct approximation of joint distributions can be rather complex and expensive. In practice, most commercial DBMSs adopt the *attribute value independence assumption* [2, 18]. Under this assumption, the data distributions of individual attributes in a relation are *independent* of each other and the joint data distribution can be derived from the individual distributions (which are approximated by one-dimensional histograms).

Unfortunately, real-life data rarely satisfies the attribute value independence assumption. For instance, functional dependencies represent the exact opposite of the assumption. Moreover, there are intermediate situations as well. For example, it is natural for the `salary` attribute of

*Partially supported by the National Science Foundation under Grant IRI-9157368 (PYI Award) and by grants from DEC, IBM, HP, AT&T, Informix, and Oracle.

Proceedings of the 23rd VLDB Conference
Athens, Greece, 1997

the `Employee` relation to be 'strongly' dependent on the `age` attribute (i.e., higher/lower salaries mostly going to older/younger people). Making the attribute value independence assumption in these cases may result in very inaccurate approximations of joint data distributions and therefore inaccurate query result size estimations with devastating effects on a DBMS's performance [2]. We are aware of only one proposal to replace this assumption, which involved the construction of multi-dimensional equi-depth histograms [12]. But, in light of the new and far more accurate histogram classes, this proposal seems limited and the heuristic technique proposed for the partitioning of two-dimensional spaces is often ineffective.

Motivated by the above problems, we have investigated several ways to approximate joint data distributions in relatively accurate fashion. This paper contains the results of this effort and makes the following contributions:

1. All histograms in our earlier, one-dimensional, taxonomy [16] are generalized to multiple dimensions. The newer classes of histograms that we have introduced prove to be much more accurate in capturing joint data distributions than the traditional equi-depth histograms.
2. A novel technique is provided for partitioning a multi-dimensional space into a given number of partitions while satisfying various mathematical constraints. This technique is compared with a technique based on *Hilbert-numbering* and a generalization of the technique of Muralikrishna and DeWitt [12] and shown to result in significantly better multi-dimensional histograms.
3. The *Singular Value Decomposition (SVD)* technique from linear algebra [10] is introduced as a mechanism to approximate two-dimensional joint data distributions by a small number of individual data distributions.

An extensive set of experiments demonstrates the advantages and disadvantages of various approaches and their benefits compared to the independence assumption.

2 Problem Formulation

We provide definitions in the context of a set of n real- or integer-valued attributes X_i ($i = 1..n$) in a relation R. These definitions can be extended to non-numerical attributes by first converting values in their domains into floating point numbers.

2.1 Data Distributions

The *value set* $\mathcal{V}_i$ of attribute X_i is the set of values of X_i that are present in R. Let $\mathcal{V}_i = \{ v_i(k)\text{: } 1 \leq k \leq D_i \}$, where $v_i(k) < v_i(j)$ when $k < j$. The *spread* $s_i(k)$ of $v_i(k)$ is defined as $s_i(k) = v_i(k+1) - v_i(k)$, for $1 \leq i \leq D_i$. (We take $s_i(D_i) = 1$.) The *frequency* $f_i(k)$ of $v_i(k)$ is the number of tuples in R with $X_i = v_i(k)$. The *area* $a_i(k)$ of $v_i(k)$ is defined as $a_i(k) = f_i(k) \times s_i(k)$. The *data distribution* of X_i is the set of pairs $\mathcal{T}_i = \{ (v_i(1), f_i(1)), (v_i(2), f_i(2)), \ldots, (v_i(D_i), f_i(D_i)) \}$.

The *joint frequency* $f(k_1, .., k_n)$ of the value combination $< v_1(k_1), .., v_n(k_n) >$ is the number of tuples in R that contain $v_i(k_i)$ in attribute X_i, for all i. The *joint data distribution* $\mathcal{T}_{1,..,n}$ of $X_1, .., X_n$ is the entire set of *(value combination, joint frequency)* pairs. Often we refer to the individual data distributions of each of the attributes as their *marginal distributions*. A natural way to represent joint data distributions is using multi-dimensional frequency matrices (tensors). The frequency matrix $\mathcal{F}_{1,..,n}$ for X_i's is a $D_1 \times .. \times D_n$ matrix (tensor) whose $[k_1, .., k_n]$ entry is equal to $f(k_1, .., k_n)$. We refer to such matrices as *n-dimensional matrices* in this paper. We can similarly define one-dimensional frequency vectors corresponding to the marginal distributions of X_i's.

The joint frequency distribution can be visualized as a set of points in a multi-dimensional space, with each attribute corresponding to an axis. For each combination of attributes values that is present in the relation, there is a point in the space whose coordinates are equal to the attribute values. This is illustrated in Figure 1 for the two-dimensional case[1]. The numbers next to the points denote the joint frequencies of the corresponding attribute-value pairs.

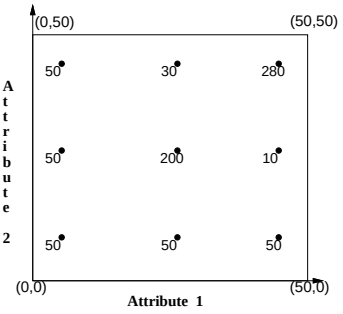

Figure 1: Multi-dimensional Data Distribution

Note that the joint data distribution is multi-dimensional, can be very large for a large database relation, and differs for each combination of attributes (of which there can be many in a database). Due to these complexities, it is impractical to store the entire joint data distribution of the relation and is considered expensive even to approximate it. In the next section, we define an important but rare characteristic of certain joint data distributions that makes them simpler to approximate.

2.2 Attribute Value Independence

Definition 2.1 A set of attributes X_i, $1 \leq i \leq n$ have

[1] Although our formulation and techniques are presented for the general, multi-dimensional case, for simplicity, all our examples are two-dimensional.

mutually *independent* data distributions if

$$\forall\, 1 \le i, j \le n,\ \forall\, 1 \le k, m \le D_i,\ 1 \le l, n \le D_j,\ \frac{f(\ldots, k, \ldots, l, \ldots)}{f(\ldots, k, \ldots, n, \ldots)} = \frac{f(\ldots, m, \ldots, l, \ldots)}{f(\ldots, m, \ldots, n, \ldots)}, \quad (1)$$

where k and m appear in the ith argument of f and l and n appear in the jth argument of f. In other words if the tuples of a relation are grouped based on their values in one of the attributes, the data distribution of the values in the other attribute within each group is identical up to a constant factor.

Let T be the number of tuples in relation R, $\mathcal{F}_i$ be the $1 \times \ldots \times D_i \times \ldots \times 1$ frequency vector of X_i and $\mathcal{F}_{1,..,n}$ be the joint frequency matrix of all X_i, $1 \le i \le n$. Then, Definition 2.1 implies that

$$\mathcal{F}_{1,..,n} = \frac{1}{T^{n-1}} \times \mathcal{F}_1 \times \ldots \times \mathcal{F}_n.^2 \quad (2)$$

These equations are illustrated in the following example.

Example 2.1 Let X_1 and X_2 contain three values each, with the following joint and marginal frequency matrices:

$$\mathcal{F}_{12} = \begin{pmatrix} 50 & 20 & 10 \\ 30 & 12 & 6 \\ 15 & 6 & 3 \end{pmatrix}, \mathcal{F}_1 = \begin{pmatrix} 80 \\ 48 \\ 24 \end{pmatrix},$$

$$\mathcal{F}_2 = \begin{pmatrix} 95 & 38 & 19 \end{pmatrix}.$$

One can easily verify that (1) holds. For instance, $f(2,1)/f(2,3) = 30/6 = f(3,1)/f(3,3) = 15/3$. Likewise, summing up all frequencies in any matrix yields $T = 152$, so (2) holds as well:

$$\begin{pmatrix} 50 & 20 & 10 \\ 30 & 12 & 6 \\ 15 & 6 & 3 \end{pmatrix} = \frac{1}{152} \times \begin{pmatrix} 80 \\ 48 \\ 24 \end{pmatrix} \times \begin{pmatrix} 95 & 38 & 19 \end{pmatrix}.$$

2.3 Query Result Size Estimation

In this paper, we mostly focus on queries containing predicates of the form $(P_1 \& .. \& P_n)$, where P_i is a selection on attribute X_i. The result size of such a query can be computed from the joint data distribution of the participating attributes as the sum of the frequencies of the attribute-value pairs that satisfy the query predicate. Any approximation to the joint frequency matrix would generate a corresponding approximation to the query result size as well. One-dimensional histograms are very common tools for single-attribute distribution approximation and are central to this paper, so they are introduced in the following section.

[2] here, $\times$ is overloaded to indicate both multiplication of multidimensional matrices and multiplication of a scalar with such a matrix.

3 Histograms

In this section, we define one-dimensional histograms and briefly describe a taxonomy presented in our earlier work [16]. Extensions to multi-dimensional histograms are defined later in the paper.

A *histogram* on an attribute X is constructed by using a *partitioning rule* to partition its data distribution into β (≥ 1) mutually disjoint subsets called *buckets* and approximating the frequencies and values in each bucket in some common fashion. In particular, the most effective approach for values is the *uniform spread* assumption [16], under which attribute values are assumed to be placed at equal intervals between the lowest and highest values in the bucket. Likewise, the most effective approach for frequencies is the *uniform frequency* assumption, under which the frequencies in a bucket are approximated by their average.

As examples, consider the well-known *equi-width* and *equi-depth* histograms. They both group contiguous ranges of attribute values into buckets but differ in the partitioning rule they employ. In an equi-width histogram, all buckets are assigned value ranges of equal length; in an equi-depth histogram, all buckets are assigned the same total number of tuples.

We have introduced several new classes of (one-dimensional) histograms with significant differences in their characteristics and accuracies. Our effort to understand all possibilities has generated a taxonomy that allows us to systematically deal with both the old and new histogram classes [16]. This taxonomy is based on four orthogonal characteristics that uniquely identify a histogram class and are described below.

Sort Parameter: This is a parameter whose value for each element in the data distribution is derived from the corresponding attribute value and frequencies. All histograms require that the sort parameter values in each bucket form a contiguous range that has no overlap with any other bucket. Attribute value (*V*), frequency (*F*), and area (*A*) are the proposed sort parameters.

Partition Class: This indicates any restrictions on the number of elements in buckets. Two important classes are *serial* – which place no restrictions, and *end-biased* – which requires at most one non-singleton bucket. These classes differ in their accuracy (highest for serial) and storage efficiency (highest for end-biased).

Source Parameter: It captures the property of the data distribution that is the most critical in an estimation problem and is used in conjunction with the next characteristic in identifying a unique partitioning. Spread (*S*), frequency (*F*), and area (*A*) are the most useful source parameters.

Partition Constraint: The partition constraint is a mathematical constraint on the source parameter that uniquely identifies a single histogram:
Equi-sum: In an *equi-sum* histogram, the sum of the source values in each bucket is approximately the same.

V-Optimal: Define the *variance* of a histogram to be the weighted sum of the variances of its source parameter in each of the buckets, with the weights being the number of attribute values grouped in the bucket. The *v-optimal* histogram on an attribute is the histogram with the least variance among all the histograms using the same number of buckets.
MaxDiff: In a *maxdiff* histogram, there is a bucket boundary between two source parameter values that are adjacent (in sort parameter order) if the difference between these values is one of the $\beta - 1$ largest differences.
Compressed: In a *compressed* histogram, the h highest source values are stored separately in h singleton buckets; the rest are partitioned as in an equi-sum histogram. We have chosen h to be the number of source values that (a) exceed the sum of all source values divided by the number of buckets and (b) can be accommodated in a histogram with β buckets.

By making different choices for each of these orthogonal histogram characteristics, one obtains different classes of histograms. Following [16], we will use *p(s,u)* to denote a histogram class with partition constraint *p*, sort parameter *s*, and source parameter *u*. Under this notation, for example, the equidepth and equiwidth histograms become equisum(V,F) and equisum(V,S) histograms, respectively.

Using the above framework for histograms, we now turn to the main theme of this paper, which is approximating joint data distributions.

4 Attribute Value Independence Assumption (AVI)

The *attribute value independence assumption* was introduced in the context of the System-R optimizer [18]. Under this assumption, all attributes are treated as if independent of each other (Definition 2.1), regardless of the actual data dependencies. The data distribution of each attribute is approximated separately using any of the one-dimensional histograms in the taxonomy presented above or any other technique(as in this paper).
Usage: Let the predicate P be of the form (P_1&..&P_n), where P_i is a selection on attribute X_i. Let H_i be the histogram on X_i and T be the relation cardinality. First, the estimated result size s_i of applying P_i on the relation based on H_i is calculated. Then, an estimate for the result size S of applying P on the relation can be obtained through

$$S = \frac{s_1 \times .. \times s_n}{T^{n-1}}, \quad (3)$$

which is a straightforward consequence of formula (2).
Comments: An advantage of this approach is that one can use good-quality one-dimensional histograms, which are inexpensive to compute, store, and maintain. The main disadvantage is that the assumption is almost always wrong, and therefore it results in approximate joint data distributions (and consequently query result sizes) that are very far from the actual ones.

Example 4.1 Consider the joint frequency matrix on the left:

$$\begin{pmatrix} 40 & 25 & 15 \\ 37 & 9 & 2 \\ 18 & 4 & 2 \end{pmatrix} \qquad \begin{pmatrix} 50 & 20 & 10 \\ 30 & 12 & 6 \\ 15 & 6 & 3 \end{pmatrix}$$

It is easy to verify that the marginal distributions of this matrix are the same as those in Example 2.1. Hence, their joint frequency matrix computed under the attribute value independence assumption is the one given in that example and repeated above (on the right) for clarity. The differences between the two matrices are obvious.

Next, we consider more accurate techniques that attempt to capture dependencies between the attributes.

5 Multi-Dimensional Histograms (MHIST)

A multi-dimensional histogram on a set of attributes is constructed by partitioning the joint data distribution into β mutually disjoint buckets and approximating the frequency and value sets in each bucket in a uniform manner as follows.

Values: The value domain is approximated by an extension of the uniform spread assumption (Section 3). Let the smallest and largest X_i values in bucket B be min_i and max_i respectively. Then, we can visualize the bucket as an n-dimensional rectangle with two extreme corners being $< min_1, .., min_n >$ and $< max_1, .., max_n >$. Let d_i be the number of distinct values in attribute X_i that are present in B. Let the k'th approximate value in dimension i (obtained by applying the uniform spread assumption along that dimension) be denoted by $v'_i(k)$. The actual data points in B are then approximated by all possible combinations $< v'_1(k_1), .., v'_n(k_n) >$, where $1 \leq k_i \leq d_i$.

Frequencies: All histograms make the *uniform frequency* assumption and approximate the frequencies in a bucket by their average. Thus, for example, if F is the sum of all frequencies in B, each approximate value $< v'_1(k_1), .., v'_n(k_n) >$ is associated with an approximate frequency equal to $F/(d_1 \times .. \times d_n)$.

Example 5.1 To illustrate the above approximations, consider the actual and approximate values and frequencies in a bucket shown in Figure 2. The average frequency of the bucket is obtained by dividing the sum of the frequencies (315) by the total number of approximate values inside the bucket ($4 \times 5 = 20$).

Buckets in multi-dimensional histograms need to keep the following information: number of tuples and for each dimension, the low and high values, as well as the number of distinct values in that dimension.

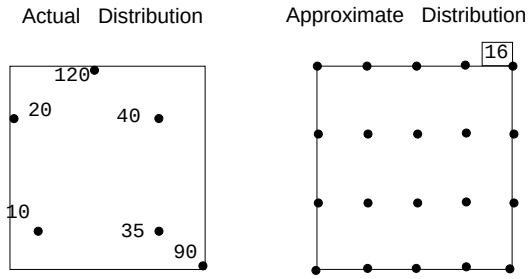

Figure 2: Approximations within a multi-dimensional bucket

Next, we identify several classes of multi-dimensional histograms based on different partitioning rules. The resulting histograms can be classified using the same parameters as in the one-dimensional taxonomy. While the partition class, source parameter, and partition constraint extend in a straightforward manner to the multi-dimensional case, multi-dimensional sort parameters introduce a serious "ordering" problem, which is described below.

Scalar sort parameters (i.e., F, A, or V_i) are totally ordered, so in those cases a bucket simply corresponds to a range of values and groups elements that are contiguous in the order of the sort parameter. A multidimensional sort parameter, e.g., combination of attribute values (*V*), is more difficult to handle because it requires finding arbitrary nonoverlapping regions in n-dimensional space, of which there is a very large number. We solve this issue using two separate techniques as follows.

Hilbert Numbering: One-dimensional histograms with V as the sort parameter were shown in our earlier work to be highly accurate mainly because they group physically nearer values into the same bucket and thus achieve a good approximation of the value domain. A well-known technique in spatial databases for capturing the proximity of multi-dimensional values in a linear order is to use a space-filling curve, such as the *Hilbert curve* [3, 7, 8]. We propose using the Hilbert numbering of attribute value combinations as a sort parameter (denoted by H) to order the data and thus once again reduce the problem to a single dimension.

This scheme (called HILBERT) is illustrated in Figure 3, which shows a MaxDiff(H,F) partitioning of Figure 1 into six buckets. Note that this technique may generate non-rectangular regions, so the corresponding buckets (which must be rectangular) may end up overlapping. By the very nature of any linear ordering of multi-dimensional data, it is often the case that two points that are adjacent to each other in the n-dimensional space may be distant in the linear ordering (this problem is much worse for higher dimensions). Hence, the resulting histograms may not be able to capture proximity in the value domain accurately.

Rectangular Partitioning: In the second class of techniques, the n-dimensional space is approximated *directly* by using non-overlapping n-dimensional rectangular regions computed via heuristics.

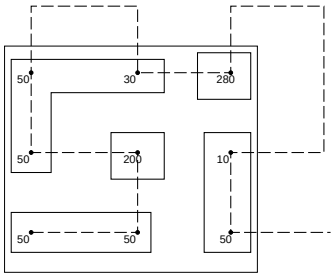

Figure 3: MaxDiff(H,F) (Hilbert) histograms

In order to motivate our technique, we first describe a generalization of the approach proposed in [12] for *equi-depth* partitioning of a two-dimensional data distribution $\mathcal{T}$ into β buckets. Our generalization (called *PHASED* in this paper) which extends their algorithm to other partition constraints, source parameters, and to higher dimensions is described below (the α_i's are nearly equal integers whose product is approximately β.).

Step 1: The n-dimensional space is partitioned along one of the dimensions, say X_1, into α_1 equi-depth buckets.

Step $i, i = 2..n$**:** In step i, each of the regions $\mathcal{T'}_j$ found in Step $i-1$ is partitioned along the attribute X_i into α_i regions. The resulting partitions in step n constitute the final buckets.

This algorithm has some drawbacks. First, it can generate a very limited set of histogram bucketizations and hence may miss better quality histograms. Second, since the order in which the dimensions are to be split is decided only once at the beginning and arbitrarily, this technique could result in poor partitionings. This is mainly because a good one-dimensional split along the marginal data distribution may still result in a very poor partitioning of the joint data distribution, especially when the joint data distribution is large.

Motivated by these limitations, we proposed a new technique (called *MHIST*) which at every step chooses and partitions the most "critical" attribute as described below. At every step, this algorithm deals with a set $\mathcal{P}$ of *partial joint data distributions* that are subsets of the entire joint data distribution. Initially, $\mathcal{P}$ contains just the entire joint data distribution. The following steps are repeated until the number of partial distributions in $\mathcal{P}$ equals the number of buckets available, at which point each of them forms a bucket in the histogram.

Step 1: First, from the set $\mathcal{P}$, we choose the distribution $\mathcal{T'}$ that contains an attribute X_i whose marginal distribution in $\mathcal{T'}$ is the *most in need of partitioning.* For the V-Optimal histograms, this means a marginal distribution that has the maximum variance of source parameter values; for the MaxDiff histogram, one with the largest difference in source values between adjacent values; and for the Equi-Sum, Compressed histograms, one with the largest sum of source values.

Step-2: Next, $\mathcal{T}'$ is split along X_i into a small number (p) of buckets. The resulting p new partial joint data distributions replace $\mathcal{T}'$ in set $\mathcal{P}$.

Clearly, different values of p may result in different histograms. The impact of p on histogram accuracy is studied in the experiments section. We refer to the MHIST technique using p-way splits as *MHIST-p*.

The two schemes (PHASED with $\alpha_1 = 2, \alpha_2 = 3$ and MHIST-2) are graphically illustrated in Figure 4, which shows a MaxDiff(V,F) partitioning of the space of Figure 1 into six buckets. The numbers on the dashed lines denote the order in which the corresponding splitting took place. Note that MHIST-2 avoids grouping highly different frequencies (which is the goal of MaxDiff), while due to its simplified bucketization scheme, PHASED uses up buckets for grouping equal frequencies and fails to result in an accurate MaxDiff histogram.

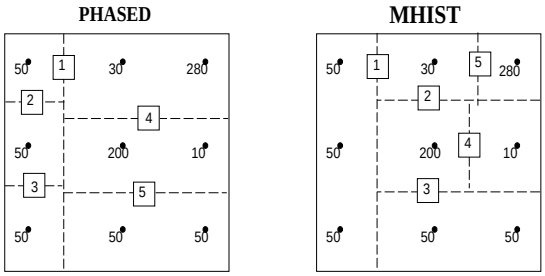

Figure 4: Two-dimensional MaxDiff(V,F) histograms

By picking a dimension based on its criticality to the partition constraint at *each* step (thus allowing the same dimension to be picked several times, for example), MHIST-2 often results in a desirable histogram. It is also clear that this algorithm can generate far more types of partitionings than the older approach. These comments are empirically verified in our experiments in Section 7.

Usage: Let the selection predicate P be of the form $(P_1 \& \ldots \& P_n)$, where P_i is a selection on attribute X_i. Assume that an n-dimensional histogram exists on the set of attributes $\{X_i\}$. In principle, P is directly applied to each histogram bucket and the (partial) result sizes are added to yield an estimate for the overall result size.

Comments: By trying to approximate the joint frequency distribution directly, multidimensional histograms have the potential of capturing attribute value dependencies with high accuracy. In fact, we have shown that under certain assumptions, the V-Optimal(F,F) histograms are *optimal* in any dimensionality, thus generalizing our earlier result for the one-dimensional case [6]. The main disadvantage of multidimensional histograms is that they are often quite expensive to construct. Also, for relation with several attributes, there is an exponential number of joint data distributions that one might want to directly approximate, so choosing among them is nontrivial.

6 Singular Value Decomposition (SVD)

6.1 Mathematical Background

The *transpose* of a matrix M is denoted by M^T. A square matrix with 0s in all its non-diagonal entries is called a *diagonal matrix*. Let J be an $M \times N$ matrix with $M \geq N$. A *singular value decomposition (SVD)* of J is any *factorization* of the form:

$$J = U\ D\ V^T, \tag{4}$$

where U is an $M \times N$ matrix, D is an $N \times N$ diagonal matrix, and V is an $N \times N$ matrix. The entries of U and V are all between -1 and 1. Several such factorizations are possible. It has been shown that there exist matrices U and V such that the diagonal elements of D are non-negative and in descending order. Assume that all instances of SVD in this paper have such a property. Let d_i be the ith diagonal entry in D. The quantities d_i are called the *singular values* of J, and the columns of U and V are called the left and right *singular vectors*, respectively. SVD is illustrated in the following example.

Example 6.1 Let $J = \begin{pmatrix} 100 & 10 \\ 4 & 7 \end{pmatrix}$. This matrix has the following SVD:

$$\begin{pmatrix} -0.99 & 0.05 \\ -0.05 & -0.99 \end{pmatrix} \begin{pmatrix} 100.61 & 0 \\ 0 & 6.56 \end{pmatrix} \begin{pmatrix} -0.99 & -0.10 \\ 0.10 & -0.99 \end{pmatrix}.$$

For a two-dimensional matrix M, let $R_M(i)$ be the horizontal vector corresponding to the ith row of M and $C_M(i)$ be the vertical vector corresponding to the ith column of M. It follows from (4) that J can be written in terms of several one-dimensional vectors [17]. That is,

$$J = \sum_{k=1}^{N} d_k\ C_U(k)\ R_V(k). \tag{5}$$

It follows that any two-dimensional matrix can be computed from its singular vectors and singular values. This observation motivates our usage of SVD in approximating joint data distributions, as described next.

6.2 Technique

Consider a joint data distribution $\mathcal{T}$ on two attributes X_1 and X_2, with value-set sizes of D_1, D_2 ($D_1 \geq D_2$), respectively. Let J be the corresponding joint frequency matrix. Then, $\mathcal{T}$ is approximated based on the following steps:

1. Compute the SVD of (a sample of) $J = U\ D\ V^T$ using well-known algorithms [17] for this purpose.

2. For some small number $k <= N$, store accurately the k highest singular values.

3. Construct one-dimensional histograms on the $2k$ row and column vectors corresponding to these terms. In principle, a different histogram from the taxonomy can be used for each vector, but in practice, it makes sense to use the same one for all of them.

The histograms constructed in step 3 can be plugged into formula (5) in place of the row and column vectors to obtain an approximation of J and consequently of the desired joint data distribution.

Elaborating briefly on step 2 above, it is clear that for high-cardinality attributes (high N), the matrix size will also be very high, making the approximation of all singular vectors a rather impractical solution. It turns out that when the attributes are highly dependent or nearly independent, the distribution of d_i values tends to be highly skewed (a few high and mostly very small values) [10]. As a result, by storing histograms for only the first k terms of the SVD, one can get a reasonably good approximation of the joint frequency matrix. We refer to a specific instance of SVD-based approximation using k terms as the SVD-k technique. Results from experiments showing the sufficiency of a small k, e.g., k=5, are presented in Section 7.

Usage: It can be easily shown that, one can use (5) to express the selectivity of a predicate $P1\&P2$ as the sum of k terms: the i'th term is $d_i c_i r_i$, where c_i and r_i are the selectivities of P_1 and P_2 computed from histograms on $C_U(i)$ and $R_V(i)$ respectively, and d_i is the i'th singular value.

Comments: This technique requires only one-dimensional histograms, which are quite inexpensive to compute and store. Unlike the attribute independence assumption, this technique makes a serious effort to accurately capture data dependencies, so its estimates should be more accurate. Its main disadvantage is that it can not be extended to higher dimensions (> 2) [1]. Like the multi-dimensional histograms case, systems employing SVD also require advance knowledge of important combinations of attributes.

7 Experimental Evaluation

In order to study the accuracy of various techniques in estimating the result sizes of multi-attribute predicates, we conducted several experiments over a testbed containing synthetic data and queries. Due to space limitations we our experiments on real-life and TPC-D data appear elsewhere [15].

7.1 Experiment Testbed

Techniques: The following techniques were studied: AVI (Section 4), HILBERT, PHASED, MHIST-p with p=1..10 (Section 5), and SVD-k with $k = 1..10$ (Section 6). The histograms required in these techniques were taken from the taxonomy. AVI and SVD-k require multiple one-dimensional histograms to be built, which can in general belong to different classes. For our experiments we assume that all histograms are taken from any single class in the taxonomy.

The sample size (s) for the histogram construction was 2000, 10000, or equal to the number of tuples (T) in the relation. Each technique was given the same storage space B, which ranged from 400 to 4000 bytes. All histograms in a given technique divide this space equally amongst them. The relative performance of various techniques was fairly constant over different values of sample size and storage space. Hence, we present the detailed results for the default values ($s = 2000$, $B = 800$), chosen because they are small enough to be practical and also lead to reasonably accurate estimates for the good-quality techniques.

Data Distributions: Several synthetic joint data distributions were generated by choosing different value domains and frequencies, as described below. The number of relation tuples was fixed at $1M$ (million) for the two-dimensional data and $5M$ for higher dimensions. The number of attribute values (D) in all n attributes was identical and varied from 50 to 200, and was chosen such that the total number of combinations does not exceed the number of tuples.

Frequency Sets: Several different types of dependencies were modeled using the joint frequency matrices. Since the observations remained the same in each case, we present a single broad class of joint frequency matrices in this paper. The motivation behind this distribution (called Z^n for n-dimensions) comes from the fact that when the attributes have dependencies between them, there will often be a few combinations of attribute values that occur much more frequently in the relation than others. We model this phenomenon by populating the joint frequency matrix from a Zipf distribution [20], and varying the level of dependency by means of the z parameter of the Zipf distribution. For higher values of the z parameter, there are a few very high frequencies implying strong dependence between the attributes. For small values of z (close to 0), all frequencies tend to be similar, implying that all attribute-value combinations are equally likely to appear in the relation (independence).

Value Sets: All attribute values were nonnegative integers, and spreads were generated according to several distributions given in [16]. In this paper we present the results for *cusp_min* distribution which consists of $D/2$ increasing spreads followed by $D/2$ decreasing spreads.

The following notation is used to represent various joint data distributions arising out of these combinations: $value\_set_0 ... value\_set_n.Z^n(D, z)$.

Queries and Error Formulas: All techniques were evaluated for range predicates of the form $(X_1 \leq a_1)\&..\&(X_n \leq a_n)$, where a_i is a constant in $\mathcal{D}_i$, the domain of X_i. The query set contains queries over all possible values in the joint value domain. For each query, we find two forms of error: error as a percentage of the result size (E_S) and error as a percentage of the input relation size (E_T). When considering a set of queries together, we compute the av-

erages of the above errors over the query set ($\bar{E}_S$ and $\bar{E}_T$, respectively). Since both error measures led to identical conclusions about the relative performance of the studied techniques, we only present results for the E_S error.

We first present the results of experiments on two-dimensional queries for all the techniques and then present the results for higher dimensions.

7.2 Effectiveness of Histograms

The relative performance of various histograms was fairly constant over a wide range of data and query sets. Hence, we present results from the $cusp\_min.cusp\_min.Z^2(50,1)$ data distribution. Table 5 contains the errors (as a percentage of the result size) of the techniques using various histograms. The horizontal line separates the consistently good-quality histograms from the poor quality histograms. There are two main conclusions to be drawn from this table. First, the most accurate histograms in each technique belong to the MaxDiff(V,A) or V-Optimal(V,A) classes[3]. This is because of the effectiveness of the three parameters: *A* (area) in capturing the skew in value and frequency domains, *MaxDiff* and *V-Optimal* in grouping only similar spreads/frequencies, and *V* in capturing the value domain. Since the MaxDiff(V,A) histograms are less expensive than the V-Optimal(V,A) histograms [16], all the techniques in the remaining experiments use the MaxDiff(V,A) histograms. Second, for most histograms, AVI performs the poorest while the other techniques perform significantly better. This observation is corroborated by the remaining experiments in this paper.

7.3 Effect of Dependencies

In this section, we study the effectiveness of the studied techniques in handling data dependencies, the main concern of this paper. Dependence is increased by increasing the z parameter of the Z^2 distribution.

Figures 7 and 8 depict the effects of k (number of SVD terms retained) and p (number of partitions in a split) parameters on the SVD-k and MHIST-p techniques, respectively, with dependency (z) on the x-axis and error ($\bar{E}_S$) on the y-axis. Finally, Figure 9 compares the accuracy of the best SVD and MHIST techniques thus identified, with the other techniques. The following conclusions can be drawn from these figures.

SVD-k (Figure 7): Among the SVD-k algorithms, SVD-5 has the best performance. For $k = 10$, although several terms are captured, the storage space allocated for each of them is smaller and the accuracy of approximating the most important terms in the distribution is low.

MHIST-p (Figure 8): Among the MHIST-p algorithms, MHIST-2 has the best performance. The reason stems from local-vs-global considerations. For the MaxDiff constraint, a high value of p results in partitioning the attribute with the largest difference between any neighboring source values at several points. While one of these partitions falls exactly at the largest difference, the remaining ones may be at points that exhibit much smaller differences than those found in other attributes, leaving fewer buckets to handle those. Clearly, this problem does not arise for $p = 2$ and is small for $p = 3$.

Based on these observations, we have chosen MHIST-2 and SVD-5 as the best representatives of the MHIST, SVD techniques and present experimental results only for them in the rest of the paper.

All techniques (Figure 9): It is clear that despite using high-quality one-dimensional histograms, the AVI technique results in very high errors. Among other techniques, PHASED and SVD-5 have nearly similar performances while MHIST-2 and HILBERT are noticeably better. Overall, MHIST-2 performs the best. Further analysis of each case showed the following: first, the quality of partitions obtained by MHIST-2 was significantly better than PHASED, and second, HILBERT incurs errors because of overlapping grouping of values and the unavoidable loss of proximity in linear ordering.

Interestingly, all techniques are more effective at handling low and high dependencies than intermediate levels ($z = 2$). For high values of skew, there are few frequencies that are very high and, therefore, captured accurately by the MaxDiff(V,A) histograms, while the rest of the frequencies are very low and grouping them into the remaining buckets causes small errors. For low values of skew, since frequencies are all nearly the same, any grouping of them is unlikely to result in high errors. For z around 2, there are several frequencies in the Zipf distribution that are dissimilar and significantly high, and hence cause high errors when grouped with each other.

7.4 Sensitivity Analysis

In this section we study the effect of storage space (B) allocated on the relative performance of various techniques. Figure 10 depicts the performance curves for these techniques on the *cusp_min.cusp_min.*Z^2*(2,2)* data distribution. The errors ($\bar{E}_S$) are shown on the y-axis and the space (B) on the x-axis.

It is clear from this figure that, all technique, except AVI, benefit significantly from increases in space. The effect of B on AVI is small because, the one-dimensional histograms AVI capture the marginal distributions nearly 100% accurately even for small amounts of space and any further increase in space does not affect their accuracy. Since the other techniques capture the joint data distribution more accurately as B increases, their errors decrease. At very large amounts of space all these techniques will have nearly 0 errors (most probably they will never become 0 because they are computed from a sample). The SVD

[3]The order of performance is in fact almost identical to our earlier results for single-attribute queries [16].

Histogram	% Error ($\bar{E}_S$)				
	AVI	SVD-5	PHASED	HILBERT	MHIST-2
Uniformity (System-R)	89.3	87.2	89.3	89.3	89.3
Equiwidth	72.3	54.2	52.8	57.5	52.8
Equidepth	68.2	44.4	40.6	47.5	40.6
Compressed(V,F)	59.1	26.4	22.3	24.7	17.9
MaxDiff(V,F)	57.4	24.8	21.9	22.7	14.9
V-Optimal-Serial(V,F)	52.5	23.7	21.1	22.8	13.2
Compressed(V,A)	46.2	21.7	17.3	14.8	9.7
MaxDiff(V,A)	43.2	17.4	17.2	12.1	6.6
V-Optimal-Serial(V,A)	41.5	16.5	16.2	12.0	6.4

Figure 5: Effect of histograms on the accuracy of various techniques

Technique	Time Taken (msec)	
	Average	Maximum
AVI	287	319
HILBERT	658	722
PHASED	679	745
MHIST-2	683	770
SVD-5	806	848

Figure 6: Construction costs

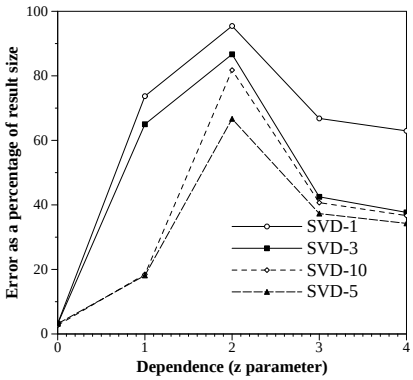

Figure 7: Effect of k on SVD-k

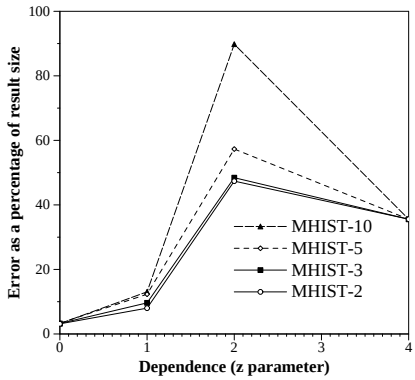

Figure 8: Effect of p on MHIST-p

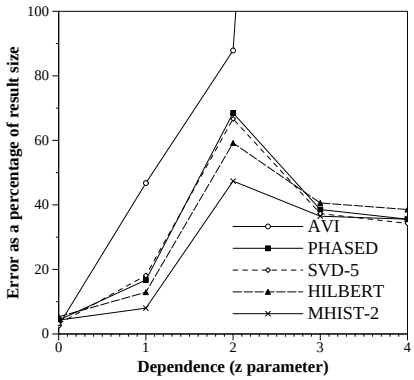

Figure 9: Effect of Dependence on all techniques

errors do not fall as low even for $B = 4000$ because SVD approximates only few of the terms in the SVD expansion. Over all, in an intermediate range of B, the techniques still retain their relative order of accuracy while converging towards each other.

7.5 Effect of dimensionality (n) on accuracy

In this section, we study the performance of various techniques for higher-dimensional queries ($n \geq 2$). Figure 11 contains the storage space on the x-axis and errors ($\bar{E}_S$) on the y-axis for the MHIST-2 technique for various dimensions. Figure 12 contains the errors for $n = 3$ for the MHIST-2, AVI and PHASED techniques. Note that, as n increases, the errors due to MHIST-2 increase, but by increasing storage space these errors can be effectively reduced. The increase in errors is sharper between 2 and 3 than 3 and 4 because, at higher dimensionalities, even with $5M$ tuples, the skew in the data distribution is limited due to the large number of attribute value combinations. Space has a similar effect on PHASED, but as in the earlier experiments, PHASED performs worse than MHIST-2. Interestingly, space does not seem to have any effect on AVI errors. This is because, the one-dimensional histograms in AVI were 100% accurate in capturing the value domains even at small storage spaces. Hence, the errors are all due to the complete lack of dependency information in AVI for any amount of space. The main conclusion is that one can use MHIST-2 histograms for higher dimensions simply by allocating more space.

7.6 Comparison of Construction Costs

Table 6 illustrates the difference in the construction costs of various techniques. It contains actual timings (in milliseconds) collected from running the corresponding algorithms on a SUN-SPARC10, for various techniques using 800 bytes of space. The times listed are averages over five runs of the computation program on a lightly loaded machine and do not include the time taken to compute the sample. A sample of 2000 tuples was used as the input. AVI incurs the least time because it only needs to compute two one-dimensional histograms. SVD-5 incurs the highest time because of the cost of SVD expansion (mainly) and computing 10 histograms. HILBERT, PHASED, and MHIST-2 are more expensive than AVI because of various intermediate computations (Hilbert numbers for HILBERT and several one-dimensional partitionings for PHASED, MHIST-2).

In conclusion, compared to the cost of collecting the sample from the large relation (required for all techniques) which could be in the order of seconds, these costs (which are all less than 1 second) are almost negligible and do not affect the practicality of any of these techniques.

8 Conclusions

In this paper, we have proposed several techniques based on multi-dimensional histograms and SVD as alternatives to the attribute value independence assumption. We have

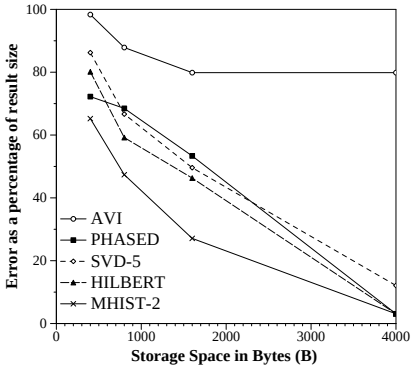

Figure 10: Effect of space for $n = 2$

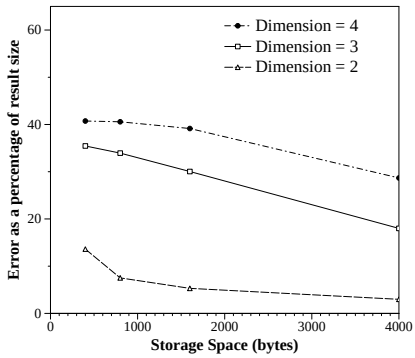

Figure 11: Effect of space and dimension on MHIST-2

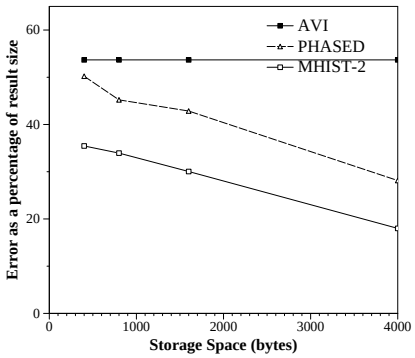

Figure 12: Effect of space for $n = 3$

conducted an extensive set of experiments to study the performance of various techniques and arrived at the following conclusions:

- The multi-dimensional MaxDiff(V,A) histograms computed using the MHIST algorithm are the most accurate among all techniques (including earlier approaches for multi-dimensional histograms).
- SVD- and HILBERT curve-based techniques are less accurate than the multi-dimensional histogams computed using MHIST-2. A positive characteristic of these two techniques is that they use one-dimensional histograms, which are already implemented in nearly all commercial systems.
- Traditional techniques making the attribute value independence assumption (as in nearly all commercial systems) incur very high errors in selectivity estimation for predicates on multiple attributes.

Overall, we believe that the attribute value independence assumption can be successfully abandoned in real-life systems and be replaced by multi-dimensional histograms computed using the MHIST technique. Based on the performance-cost trade-offs and the applicability of MHIST for arbitrary dimensions, we believe that it is the most appropriate technique for this purpose.

References

[1] M. W. Berry. Private communication, 1997. C.S dept, Univ. of Tennessee.

[2] S. Christodoulakis. Implications of certain assumptions in database performance evaluation. *ACM TODS*, 9(2):163–186, June 1984.

[3] J. G. Griffiths. An algorithm for displaying a class of space-filling curves. *Software - Practice and Experience*, 16(5):403–411, May 1984.

[4] Y. Ioannidis. Universality of serial histograms. *Proc. of the 19th Int. Conf. on Very Large Databases*, pages 256–267, December 1993.

[5] Y. Ioannidis and S. Christodoulakis. Optimal histograms for limiting worst-case error propagation in the size of join results. *ACM TODS*, 1993.

[6] Y. Ioannidis and V. Poosala. Balancing histogram optimality and practicality for query result size estimation. *Proc. of ACM SIGMOD Conf*, pages 233–244, May 1995.

[7] H. V. Jagadish. Linear clustering of objects with multiple attributes. *Proc. of ACM SIGMOD Conf*, pages 332–342, 1990.

[8] I. Kamel and C. Faloutsos. Hilbert R-trees: an imporived R-tree using fractals. *Proc. of the 20th Int. Conf. on Very Large Databases (VLDB), Santiago, Chile*, pages 500–509, 1994.

[9] R. P. Kooi. *The optimization of queries in relational databases*. PhD thesis, Case Western Reserver University, Sept 1980.

[10] S. Leach. Singular value decomposition - a primer. Unpublished Manuscript, Department of Computer Science, Brown University, Providence, RI, USA.

[11] R. J. Lipton, J. F. Naughton, and D. A. Schneider. Practical selectivity estimation through adaptive sampling. *Proc. of ACM SIGMOD Conf*, pages 1–11, May 1990.

[12] M. Muralikrishna and D. J. Dewitt. Equi-depth histograms for estimating selectivity factors for multi-dimensional queries. *Proc. of ACM SIGMOD Conf*, pages 28–36, 1988.

[13] B. Muthuswamy and L. Kerschberg. A DDSM for relational query optimization. *Proc. ACM Annual Conference*, Oct 1985.

[14] G. Piatetsky-Shapiro and C. Connell. Accurate estimation of the number of tuples satisfying a condition. *Proc. of ACM SIGMOD Conf*, pages 256–276, 1984.

[15] V. Poosala and Y. Ioannidis. Selectivity estimation without the attribute value independence assumption. Technical report, Bell Labs, 1997.

[16] V. Poosala, Y. Ioannidis, P. Haas, and E. Shekita. Improved histograms for selectivity estimation of range predicates. *Proc. of ACM SIGMOD Conf*, pages 294–305, June 1996.

[17] W. H. Press. *Numerical recipes in C: The art of scientific computing*. Cambridge University Press, 1988.

[18] P. Selinger, M. Astrahan, D. Chamberlin, R. Lorie, and T. Price. Access path selection in a relational database management system. *Proc. of ACM SIGMOD Conf*, pages 23–34, 1979.

[19] W. Sun, Y. Ling, N. Rishe, and Y. Deng. An instant and accurate size estimation method for joins and selections in a retrieval-intensive environment. *Proc. of ACM SIGMOD Conf*, pages 79–88, 1993.

[20] G. K. Zipf. *Human behaviour and the principle of least effort*. Addison-Wesley, Reading, MA, 1949.