

The Power-Method: A Comprehensive Estimation Technique for Multi-Dimensional Queries

Yufei Tao
Department of Computer Science
City University of Hong Kong
Tat Chee Avenue, Hong Kong
taoyf@cs.cityu.edu.hk

Christos Faloutsos
Department of Computer Science
Carnegie Mellon University
Pittsburgh, USA
christos@cs.cmu.edu

Dimitris Papadias
Department of Computer Science
HKUST
Clear Water Bay, Hong Kong
dimitris@cs.ust.hk

ABSTRACT

Existing estimation approaches for multi-dimensional databases often rely on the assumption that data distribution in a small region is uniform, which seldom holds in practice. Moreover, their applicability is limited to specific estimation tasks under certain distance metric. This paper develops the *Power-method*, a comprehensive technique applicable to a wide range of query optimization problems under various metrics. The Power-method eliminates the local uniformity assumption and is accurate even in scenarios where existing approaches completely fail. Furthermore, it performs estimation by evaluating only one simple formula with minimal computational overhead. Extensive experiments confirm that the Power-method outperforms previous techniques in terms of accuracy and applicability to various optimization scenarios.

1. INTRODUCTION

The most common query types in multi-dimensional (e.g., spatial, multimedia, time-series) databases (*DB*) can be classified into three categories:

- Given a query point q and a radius r , a *range-search* (RS) retrieves all points $o \in DB$ such that $dist(o, q) \leq r$, where $dist(o, q)$ denotes the distance between o and q .
- A *k nearest neighbor* (*kNN*) query returns the k closest data points o_1, o_2, \dots, o_k to a query q , or formally: given any point $o \in DB - \{o_1, o_2, \dots, o_k\}$, $dist(o_i, q) \leq dist(o, q)$ for all $1 \leq i \leq k$.
- A *regional distance (self-) join* (RDJ) retrieves all pairs of objects (in some *constrained region*) that are close to each other. Formally, it specifies a query point q , a constraint radius r , a distance threshold t , and returns $(o_1, o_2) \in DB \times DB$ such that $o_1 \neq o_2$, $dist(o_1, q) \leq r$, $dist(o_2, q) \leq r$, and $dist(o_1, o_2) \leq t$. The special case where $r = \infty$, corresponds to a *global distance join* (GDJ).

The result of any query depends on the underlying distance metric. The most widely used metric is the L_x norm, or formally,

given two points q, o whose coordinates on the i -th dimension ($1 \leq i \leq m$) are q_{yi} and o_{yi} respectively, their L_x distance is:

$$\begin{aligned} L_x(q, o) &= [\sum_{i=1}^m (q_{yi} - o_{yi})^x]^{1/x} && (\text{for } x \neq \infty) \\ L_x(q, o) &= \max_{i=1}^m (|q_{yi} - o_{yi}|) && (\text{for } x = \infty) \end{aligned}$$

For instance, (i) if L_∞ is assumed, RS corresponds to a *window query* centered at q with extent $2r$ on each dimension whereas, (ii) if L_2 is assumed, the query region corresponds to a circle centered at q with radius r .

1.1 Motivation

Efficient optimization of multi-dimensional queries requires accurate estimation of the following values [29, 7, 4]:

- *Query selectivity*, where the selectivity of a RS (RDJ) query is the ratio between the number of retrieved points (pairs) and the dataset cardinality (size of the cartesian product $DB \times DB$). The concept of selectivity does not apply to *kNN* queries where exactly k points are returned.
- *Query cost*, in terms of the number of disk accesses using a multi-dimensional index.

Existing estimation methods fall into two categories. *Local methods* produce a “tailored” estimate depending on the query’s concrete location, typically using a histogram [1, 17], which partitions the data space into (disjoint or overlapping) *buckets*. As explained later, histograms have several limitations. First, they assume that the data distribution in each bucket is uniform (i.e., *local uniformity assumption*), which is rarely true for real datasets. Second, estimation for queries under any metric other than L_∞ may require expensive evaluation time. Third, their applicability to *kNN* queries is questionable.

Global methods [15, 7] provide a single estimate corresponding to the average selectivity/cost of all queries, independently of their locations. Such techniques avoid the problems of histograms, but have a serious drawback: since queries at various locations may have different characteristics, their respective selectivity/cost can differ considerably from the average value.

In summary, currently there is not a comprehensive approach able to perform all estimation tasks (i.e., selectivity/cost prediction in any distance metric) effectively. Such an approach is highly desirable in practical systems, where it is important to have a single efficient method that applies to all cases, instead of multiple methods that are effective for certain sub-problems but inefficient/inapplicable to others.

1.2 Contributions

This paper develops the Power-method, a novel estimation technique which combines the advantages of both local and global methods, but avoids their deficiencies. The proposed method possesses the following attractive features:

- It *eliminates the local uniformity assumption* and, therefore, is accurate even in scenarios where existing techniques fail.
- It is the first *comprehensive* technique applicable to *selectivity and cost estimation for all the query types mentioned earlier*.
- It supports *all L_x metrics with small space requirements*.
- It performs any query estimation by evaluating only *one simple formula*; hence, its computational overhead is minimal.

Extensive experimentation proves that the Power-Method achieves accurate estimation (with average relative error below 20%) in circumstances where traditional methods completely fail. The rest of the paper is organized as follows. Section 2 reviews existing query estimation techniques. Section 3 introduces the novel concept of local power law, proves the related theorems, and illustrates its implementation in practice. Section 4 experimentally evaluates its performance, and Section 5 concludes the paper with directions for future work.

2. RELATED WORK

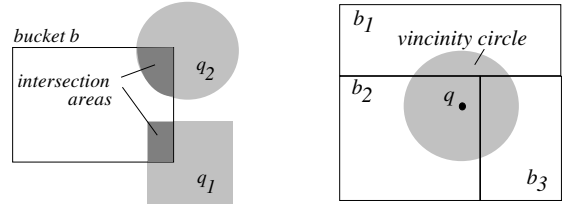
Most approaches for multi-dimensional selectivity estimation are based on histograms [22, 26, 1, 17, 19, 8], which construct a small number of rectangular buckets, and store for each bucket b the number¹ of points N_b in its extents. The *density* D_b of bucket b is defined as $D_b = N_b/a_b$, where a_b is the area of b . To estimate the selectivity of a RS (range search) query q , a histogram first computes for each bucket b , the intersection area a_{intr} (with q); assuming that the data in each bucket are uniform, the number of qualifying objects inside this bucket is then estimated as $D_b \cdot a_{intr}$. The final selectivity is obtained by summing the estimate from each intersecting bucket.

Theodoridis et al. [33] suggest that, under the local uniformity assumption, RDJ (regional distance join) selectivity estimation can be reduced to GDJ (global distance join) on uniform data, for which several cost models are available [2, 3, 33]. The idea is to apply these uniform models inside the query constrained region Q , based on the density of the bucket that covers Q . If Q intersects multiple buckets, their average density is used. The cost estimation of RS [33, 29] and RDJ [33] follows the same reasoning.

Application of histograms has been limited mainly to RS queries under the L_∞ metric, where the intersection area a_{intr} (between the bucket and the query) is a (hyper) rectangle (e.g., query q_1 in Figure 1a). For queries in the other metrics, however, the area computation is usually expensive. In Figure 1a, for instance, the L_2 RS query q_2 intersects bucket b into a rather irregular shape whose area a_{intr} is difficult to compute. To tackle this problem, Berchtold, et al. [5] suggest the *monte-carlo* method, which generates a set of α uniform points in the bucket, counts the number β of them in the query region, and estimates the a_{intr} as

¹ In case of overlapping buckets, a point in the intersection region of multiple buckets is assigned to a unique bucket.

$a_b \beta / \alpha$, where a_b is the bucket’s area. A large number of random points (which increases with the dimensionality) is necessary to obtain accurate estimation. Repeating this process for every (partially) intersecting bucket may lead to significant overhead.



(a) Irregular intersection area (b) Difficulty for k NN
Figure 1. Deficiencies of histograms

We are not aware of any local estimation method for k NN query cost. The main difficulty of applying histograms lies in the prediction of d_k (the distance from the query point to its k -th NN), which is the first step of the cost analysis. The value of d_k should be such that the vicinity circle centering at q with radius d_k covers expectedly k points (assuming uniformity in buckets). Finding such a circle requires non-trivial repetitive tuning (increasing-decreasing the radius, depending on how many points fall in the current “trial” circle). This is especially complicated if the circle intersects multiple buckets (the circle in Figure 1b intersects 3 buckets) and produces irregular intersection regions. To avoid this, [5, 4] apply their (uniform) k NN cost model, to non-uniform distributions *anyway*, and show that (surprisingly) sufficient accuracy *may* be achieved. In Section 3.3 we reveal the conditions that must be satisfied for this conjecture to hold in practice.

Other “local” multi-dimensional estimation techniques include *sampling* [25, 27, 12, 34, 6, 13, 18], *kernel estimation* [10], *single value decomposition* [26], *compressed histograms* [24, 20, 23, 35], *sketches* [32], *maximal independence* [14], *Euler formula* [19, 30, 21], etc. These methods, however, target specific cases (mostly RS selectivity under the L_∞ norm), and their extensions to other problems are unclear. Finally, among the “global” estimation methods, [7, 16] analyze selectivity estimation of RS and GDJ using datasets’ fractal dimensions, but their analysis does not address RDJ. Regarding average query costs, [15] studies RS queries, while k NN retrieval is discussed in [28, 4].

3. THE LOCAL POWER LAW

Section 3.1 explains the problems associated with the local uniformity assumption in histograms. Then, Section 3.2 describes the *local power law* (LPLaw) that overcomes these problems, and Section 3.3 solves all the estimation problems using the LPLaw. Finally, Section 3.4 elaborates the implementation of LPLaw in practice. Our analysis focuses on biased queries (i.e., the query distribution follows that of data), due to their practical importance [7, 4, 28, 34], while unbiased queries are discussed in Section 3.4.

3.1 Density trap

Histograms are based on the hypothesis that the data distribution in sufficiently small regions of space is piece-wise uniform. Thus, they compute and store the density of such regions. A main contribution of this work is to show that this, apparently reasonable, hypothesis is wrong for the vast majority of real

data, as well as the vast majority of perfect, Euclidean-geometry datasets. Consider a set of $N=1000$ 2D points along the major diagonal of a unit square and a query point q (Figure 2). What is the density in the vicinity of q , e.g., a square region centered at q ? The answer is surprising: *undefined!*

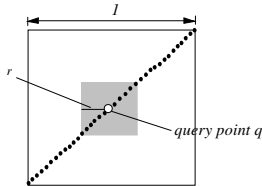


Figure 2. Density trap

The density changes *dramatically* with the radius r of the vicinity, diverging to infinity when r goes to zero (we call this phenomenon *density trap*)! This is so counter-intuitive that we give an arithmetic example. If the radius is $r=0.5$, the density is 1000. If the radius shrinks to $r=0.05$, then the density is 10000! Conversely, the density goes to zero, with growing radius! In fact it is easy to show that the density $D(r)$ for radius r is given by: $D(r)=N/(4r)$. Notice that the paradox is generated not by a mathematical oddity like the Hilbert curve or the Sierpinski triangle, but by a line, a perfectly behaving Euclidean object! Moreover, many real objects behave like lines or collections of lines (highway networks, rivers). Thus, the problem is real, and can not be downplayed as a mathematical oddity – it does appear in practice! The next question then is: if the local density around a point q is not well defined, what is the invariant that could help us describe somehow the neighborhood of q ? The following section provides the answer.

3.2 Assumptions, definitions, and properties

The resolution to the density trap comes from the concept of intrinsic dimensionality. The data points around the vicinity of point q in Figure 2 form a linear manifold – asking for their density is as unusual as asking for the area of a line. The invariant we hinted before is encapsulated in the concept of *Local Power Law* (LPLaw):

Assumption 3.1 (Local power law): Let p be a data point in a dataset. For certain range of r , the number $nb_p(r)$ of points with distances no more than r from p can be represented as:

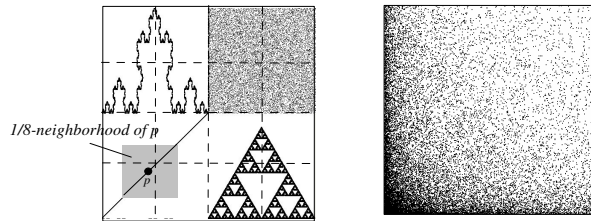
$$nb_p(r) = c_p \cdot r^{n_p}$$

where c_p and n_p are constants termed the *local constant* and *exponent*, respectively. ■

For convenience we refer to c_p and n_p collectively as the *coefficients* of the LPLaw for point p . Each LPLaw models the distribution around a *particular* data point (e.g., in Figure 2 with $c_q=2N$ and $n_q=1$). In general, however, the LPLaw of various points may differ in two aspects. *First, two points can have different local exponents, implying that the data correlation “characteristics” in their respective vicinity are different.* This is obvious in Figure 3a that mixes four distributions with different intrinsic dimensions. The local exponent of each point is determined by the intrinsic dimension of the region it lies in. *Second, two points may have similar local exponents but different constants, implying that the data “densities” in their respective vicinity are different.* Figure 3b illustrates the 2D independent Zipf distribution, where all points have local

exponents “2” (revealing the independence between the two dimensions), while those in denser areas have higher constants.

Given a point p in a m -dimensional dataset, we define the L_∞ ρ -neighborhood of p as the m -dimensional box centering at p with length 2ρ on each axis. Then, the LPLaw coefficients of p can be measured as follows.



(a) “Mixture” dataset (b) 2D Zipf
Figure 3. Non-fractal examples

Lemma 3.1 (LPLaw under L_∞): Given a data point p such that the data distribution in its L_∞ ρ -neighborhood is self-similar with intrinsic dimension d_p , then the LPLaw of p under the L_∞ metric is:

$$nb_{p_\infty}(r) = \left(\frac{N_{p_\infty}}{\rho^{d_p}} \right) r^{d_p}$$

where N_{p_∞} denotes the number of points in the L_∞ ρ -neighborhood of p . ■

Note that ρ can be any value in the range of r where the LPLaw holds. Similarly, the LPLaw of other distance metrics L_x ($x \neq \infty$) can be measured in the L_x ρ -neighborhood of a point p (i.e., a L_x m -dimensional sphere centering at p with radius ρ) while the following lemma provides a faster way to derive a L_x LPLaw from its L_∞ counterpart.

Lemma 3.2 (LPLaw under L_x): Given a m -dimensional data point p with L_∞ LPLaw $nb_{p_\infty}(r)=c_{p_\infty} \cdot r^{n_p}$, its L_x LPLaw is:

$$nb_{x_p}(r) = c_{p_\infty} \left\{ \frac{VolSphere_x(1)}{VolSphere_\infty(1)} \right\}^{n_p/m} r^{n_p}$$

where $nb_{x_p}(r)$ is the number of points within L_x distance r from p , and $VolSphere_x(1)$ is the volume of a m -dimensional L_x sphere with radius 1. ■

It is clear that the LPLaw of a point p under different distance metrics have the same local exponent n_p , and their local constants can be derived from each other using n_p . The intuitive explanation is that the ρ -neighborhood of a point under various metrics only affects how many points fall in the neighborhood (related to the neighborhood volume), but does not influence the way data are correlated (which is a data characteristic captured by the exponent).

The LPLaw is satisfied in many real datasets. We illustrate this using two real distributions (Figure 4): (i) SC dataset, which contains 36k points representing the coast line of Scandinavia, and (ii) the CA dataset, which contains 62k points representing locations in California. Figure 5a plots $nb_{x_p}(r)$ (i.e., the number of points within distance r to a point p in the L_∞ metric) as a function of r (in log-log scale) for the two points p_1, p_2 in Figure

4a. Similarly, Figure 5b illustrates the same information for the two points p_3, p_4 in Figure 4b. It is clear that $nb_{\infty p}(r)$ approximates a power law in all cases, and has various exponents (i.e., slopes of the fitting lines in Figure 5) for different points.

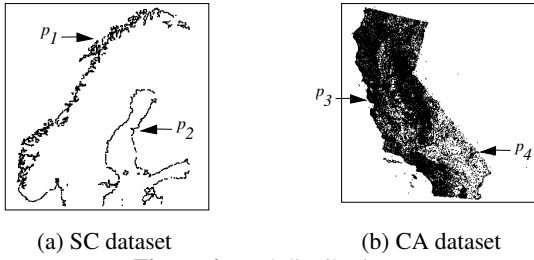


Figure 4. Real distributions

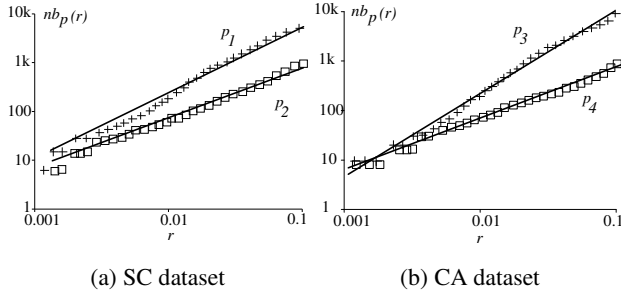


Figure 5. LPLaw in real data

Figure 6a (6b) demonstrates the local exponent (constant) distribution for SC. The values are measured using L_∞ 0.05-neighborhoods (i.e., a square with length 0.1 on each axis). Figures 6c, 6d illustrate the corresponding distributions for CA. Constants and exponents differ substantially in the data space (suggesting that a global law for the whole dataset would introduce inaccuracy), but are highly location-dependent (confirming the intuition behind LPLaw).

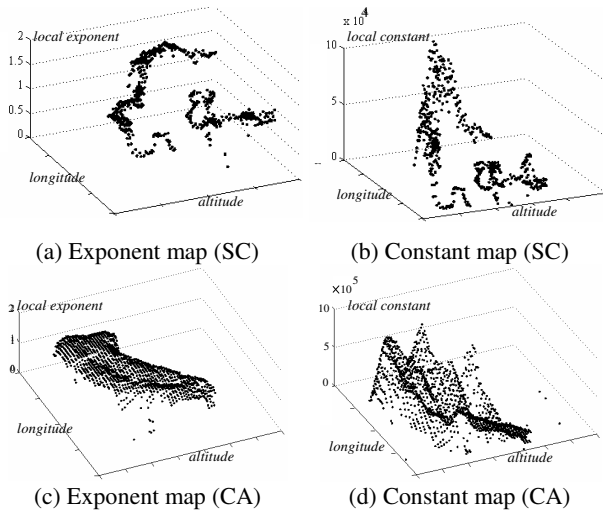


Figure 6. LPLaw coefficient distributions

3.3 Estimation using the LPLaw

In the sequel we apply LPLaw to solve all the estimation problems defined in Section 1, namely, the selectivity of RS, RDJ, and the query cost of RS, k NN, and RDJ.

Theorem 3.1 (RS selectivity): Given a RS query point q with radius r , the selectivity of q is:

$$Sel(q) = \frac{c_q \cdot r^{n_q}}{N}$$

where N is the cardinality of the dataset, and c_q, n_q are the local constant (under the corresponding distance metric) and exponent at location q , respectively.

Proof: Straightforward since the number $nb_q(r)$ of points retrieved by q satisfies the LPLaw $nb_q(r) = c_q \cdot r^{n_q}$. Hence the selectivity equals $c_q \cdot r^{n_q} / N$. ■

Theorem 3.2 (RDJ selectivity): Let q be a RDJ query with (i) constraint radius r , and (ii) distance threshold t . The selectivity of q is:

$$Sel(q) = \frac{c_q^2 (t \cdot r)^{n_q}}{(2N^2)}$$

where N is the cardinality of the dataset, c_q, n_q are the local constant (under the corresponding distance metric) and exponent at location q respectively.

Proof: (Sketch) Let N_r be the number of points in the constrained region (i.e., within distance r to q). By the LPLaw (Definition 3.1), $N_r = c_q \cdot r^{n_q}$. Applying the GDJ selectivity formula proposed in [7] inside the constrained region (after appropriate normalization), we obtain that the total number of qualifying pairs equals $N_r^2 \cdot (t/r)^{n_q} / 2$. Thus, Theorem 3.2 follows. ■

For query cost estimation we consider the R*-tree [11] due to its popularity and simplicity. Particularly, we measure the query cost in terms of the number of R-tree leaf accesses since, (i) this number dominates the total cost, and (ii) in practice non-leaf nodes are usually found in the buffer directly. Unlike the selectivity analysis, the derivation for the query cost results in different formulae for various distance metrics. In the sequel we provide the theorems for L_∞ as the results for the other metrics can be derived in a similar manner. Our discussion is based on the following lemma:

Lemma 3.3 (R-tree node extent): Let l be the length of a leaf MBR on each dimension; then: $l = 2(f/c_\infty)^{1/n}$, where c_∞, n are the L_∞ local constant and exponent at the centroid of the MBR respectively, and f is the average node fanout (i.e., number of entries in a node).

Theorem 3.3 (RS query cost): Given a RS query point q with radius r , the cost of q is:

$$Cost(q) = \frac{c_{q_\infty}}{f} \left[\left(\frac{f}{c_{q_\infty}} \right)^{1/n_q} + r \right]^{n_q}$$

where c_{q_∞}, n_q are the local constant (under metric L_∞) and exponent at q , and f is the average node fanout.

Proof: (Sketch) This problem can be reduced to the average query cost of a hypothetical point dataset with cardinality $c_{q_\infty} \cdot 2^{n_q}$ and intrinsic dimension n_q . As shown in [4], the L_∞ RS query

cost for such a dataset equals $[c_{q\infty}/(2^{n_q} \cdot f)] \cdot (l+2r)^{n_q}$, where l is the length of a leaf MBR on each axis (see Lemma 3.3). The theorem results from this equation, after necessary simplification. ■

Theorem 3.4 (kNN query cost): The cost of a k NN query q is:

$$Cost(q) = \frac{1}{f} \left[f^{1/n_q} + k^{1/n_q} \right]^{n_q}$$

where n_q is the local exponent at q and f the average node fanout. ■

Proof: Let d_k be the distance between the query point and its k -th NN. Then, by Definition 3.1, we have $k=c_{q\infty} \cdot (d_k)^{n_q}$, where $c_{q\infty}$ is the local constant at location q ; hence $d_k=(k/c_{q\infty})^{1/n_q}$. As proven in [5], the cost of a k NN query equals that of a RS query with radius d_k . As a result, the k NN cost can be represented as in Theorem 3.3, setting $r=d_k$. The formula in the theorem results from necessary simplification (which removes $c_{q\infty}$). ■

An important observation from the above theorem is that, the cost of a k NN query q is not affected by the local constant, but depends only on the local exponent. Thus, the conjecture of [5, 4], that the k NN cost is the same for *all* datasets (i.e., independently of the data density in the vicinity of q), *only holds for datasets with the same local exponent* (i.e., 2 if uniform models are applied).

Theorem 3.5 (RDJ query cost): Let q be a RDJ with (i) constraint radius r , and (ii) distance threshold t . The cost of q is: $Cost(q)=$

$$\frac{c_{q\infty}^2}{f^2} \left[\left(\frac{f}{c_{q\infty}} \right)^{1/n_q} + r \right]^{2n_q} \left[\frac{2(f/c_{q\infty})^{1/n_q} + t}{r} \right]^{n_q} + \frac{c_{q\infty}}{f} \left[\left(\frac{f}{c_{q\infty}} \right)^{1/n_q} + r \right]^{n_q}$$

where $c_{q\infty}$ and n_q are the local constant (under metric L_∞) and exponent at q , respectively.

Proof: (Sketch) Let τ be the number of nodes intersecting the constrained region. By Theorem 3.3 we have: $\tau=(c_{q\infty}/f)[(f/c_{q\infty})^{1/n_q} + r]^{n_q}$. The centroid distribution of these nodes' MBRs also has intrinsic dimension n_q . The query cost equals two times the number α of centroid pairs within distance $l+t$ (where l is the extent of a leaf) plus τ . Following the GDJ selectivity analysis [7], α can be derived as $\alpha=1/2[(l+t)/r]^{n_q}$. After substituting the corresponding variables, the theorem follows. ■

3.4 Implementation of the Power-method

The last section showed that all the estimation tasks can be performed by evaluating a single equation based on the LPLaw at the query location. Motivated by the fact that, points close in space usually have similar local constants and exponents (see Figure 6), we can pre-compute the LPLaw for a set of representative points, and perform the estimation using the LPLaw of a point close to q . Based on this idea, we select a set A of *anchor points* from the database (using any sampling technique [25, 27, 34]) and, for each anchor point, compute the LPLaw coefficients in its $L_\infty \rho$ -neighborhood (using Lemma 3.1). Given a biased query q , the estimation algorithm first finds the

point p in A nearest to q , and then obtains the estimates using the LPLaw of p .

It is worth mentioning that, this method differs considerably from a sampling method in the following ways. First, it can perform all the estimation tasks, while sampling has been limited to selectivity estimation. Second, even for selectivity prediction, it evaluates the LPLaw of a single anchor point, while sampling examines each point against the query conditions. Third, it is efficient independently of the data distribution, while it is a well-known problem that sampling is inaccurate for skewed data [12]. Fourth, it achieves satisfactory accuracy using a very small number (100 in our experiments) of anchors, while sampling requires a much larger fraction of the dataset [12].

The Power-method is optimized for biased queries. On the other hand, query optimization for unbiased queries is less important because, RS and RDJ queries in the non-populated regions usually return only a small number of objects and the query optimizer should employ an index structure (rather than sequential scan). Detecting whether a query q is biased is easy: we check if the query region covers any anchor point. The optimizer performs query estimation only if the answer is positive, using the LPLaw associated with the patch (anchor point) closest to q . As shown in the experiments, using this policy, our implementation maintains satisfactory performance even for un-biased queries.

On the other hand, a k NN query that is far away from data is most likely to be *meaningless* [9], because in this case the distance from q to its k -th NN is similar to that between q and its $(k+k')$ -th NN, where k' is a large constant, so that the k NNs returned are not necessarily "better" than the next k' NNs. As suggested in [9], such queries should be avoided, or the users should at least be warned about the significance of the result. Detecting meaningless queries can be achieved by computing the distance between the query and its nearest anchor point. If the distance is larger than certain threshold, then we judge the query to be meaningless.

4. EXPERIMENTS

This section compares experimentally the accuracy and the computational overhead of the Power-method with *minskew* [1], a benchmark histogram in the literature [34, 17, 8], and the "global method" that provides an average estimate using the fractal dimension [15, 7, 4]. For our experiments we use a PIII 1Ghz CPU and four real datasets: SC/UK that contain 36k/40k points representing the coast lines of Scandinavia/the United Kingdom, and CA/LB that include 62k/53k points corresponding to locations in California/Long Beach county [31] (Figure 4 shows the visualization of SC and CA). Our implementation of the Power-method (denoted as *power* in the sequel) randomly selects a set of anchor points from each dataset. To save space we partition the space into 64x64 cells, snap each anchor point to its closest cell corner, and represent its coordinates as those of the corner (i.e., using 12 bits). The LPLaw coefficients of each anchor point are computed using L_∞ 0.05-neighborhoods (see Lemma 3.1). The total number anchors is 100 so that the total memory consumption is 1K bytes.

For *minskew*, the bucket construction algorithm first obtains, for each cell in the 64x64 grid, the number (i.e., *frequency*) of data

points covered. Then, the cells are grouped into 140 buckets² (so that the size of *minskew* is also 1K) using a greedy algorithm that aims at minimizing a certain function. The bucket structure is rather sensitive to the dataset and the “optimization function” deployed; in many cases the algorithm terminates without producing enough buckets (the problem is also mentioned in [34]). Similar tuning problems exist for most multi-dimensional histograms such as *genhist* [17]. To alleviate the problem, we tested multiple optimization functions. The best overall performance was obtained by minimizing $\sum_{i=1-B}(v_i/n_i)$, (where n_i is the average frequency of cells in bucket b_i and v_i the variance of n_i) and we follow this approach in our implementation.

A query workload consists of 500 queries with the same parameters (unless specifically stated, the query distribution is biased). The workload estimation error is defined as: $\sum_i |act_i - est_i| / \sum_i act_i$ (same as [1]), where act_i (est_i) is the actual (estimated) selectivity/cost of the i -th query in the workload. We report the results for L_∞ and L_2 distance metrics due to their particular importance in practice. For *minskew* and metric L_2 , if the query region (of RS/RDJ) partially intersects a bucket, the intersection area is computed using the *monte-carlo* method with 1000 random points, which (based on our experiments using different numbers) leads to a reasonable tradeoff between estimation time and accuracy. We evaluate all estimation tasks discussed earlier, starting with selectivity prediction.

4.1 Selectivity estimation

Figure 7a shows the error of L_∞ RS selectivity estimation as a function of the query radius r (ranging from 0.01 to 0.1) using the SC dataset³. *Power* significantly outperforms the previous techniques. Note that *power* has the minimum error at $r=0.05$, because the LPLaw is obtained using 0.05-neighborhoods. As expected, the global power law yields considerable error due to the selectivity variation of individual queries. *Minskew* is even less accurate for small queries due to the violation of the local uniformity assumption. Its precision gradually improves for larger queries (which is consistent with previous studies [1]), because a large query covers more buckets completely, from which precise partial estimation can be obtained. Figure 7b shows the results for LB dataset under the L_2 metric, confirming similar observations.

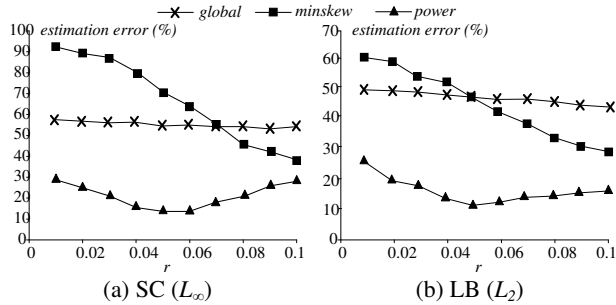


Figure 7. Error vs. query radius r (RS selectivity)

² The number of buckets (140) is larger than the number of anchor points, since each bucket stores a single value (frequency) instead of two (local coefficients).

³ Due to the space constraint, in each set of experiments we show the results of two datasets in L_∞ and L_2 metrics respectively.

To evaluate the estimation error of regional distance join selectivity, we fix the query constraint radius r to 0.05, and vary the distance threshold t from 0.001 to 0.01. Figure 8 shows the results for various datasets and metrics. There is no prediction by the global power law because, as mentioned in Section 2.2, the application of this method to RDJ is unclear (existing analysis [16] focuses only on global distance joins). *Power* is almost an order of magnitude more accurate than *minskew*. Since selectivity estimation is meaningless for k NN queries, in the sequel we proceed with cost (i.e., number of disk accesses) estimation.

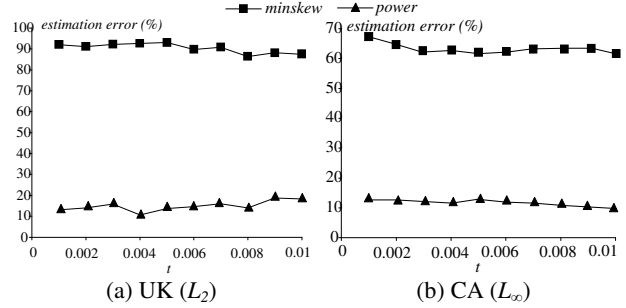


Figure 8. Error vs. distance threshold t (RDJ selectivity)

4.2 Query cost estimation

This section examines the accuracy of predicting the query costs of RS, k NN and RDJ queries. Figure 9 plots the error rate, as a function of query radius for RS queries. The relative performance of alternative methods (and the corresponding explanation) is similar to that in Figure 7.

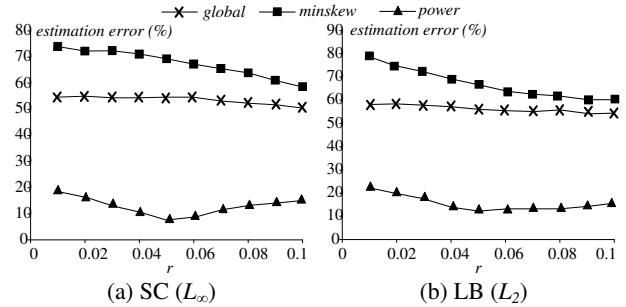


Figure 9. Error vs. query radius r (RS cost)

As mentioned in Section 2.1, there is no previous work on k NN cost estimation using histograms. Thus, we replace *minskew* with the cost model proposed in [5], which assumes local uniformity around the query’s location. Figure 10 compares this model with *power* and *global*, varying k from 1 to 100.

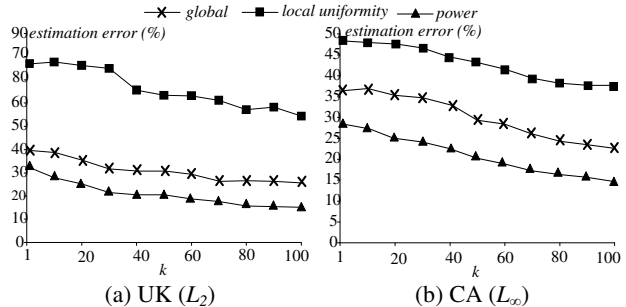


Figure 10. Error vs. k (k NN cost)

The local uniformity assumption leads to substantial error,

confirming the necessity of capturing local data correlation. *Global* has reasonable performance, because, according to Theorem 4.3, the cost of k NN queries is only affected by the local exponent at the query point. As a result, compared to other estimation problems, the variation of local constants does not introduce additional error.

The fact that the accuracy of *power* improves with k can be explained as follows. For small k , the distance from the query to the k -th NN is very short, and falls out of the range where LPLaw holds. As this distance increases with k , it is better modeled by the LPLaw, leading to more accurate prediction. Figure 11 compares the cost prediction of alternative methods for regional distance joins (query radius $r=0.05$), where the distance threshold t ranges from 0.001 to 0.01. The behavior is analogous to Figure 8, and the superiority of *power* is obvious (the global power law is again inapplicable).

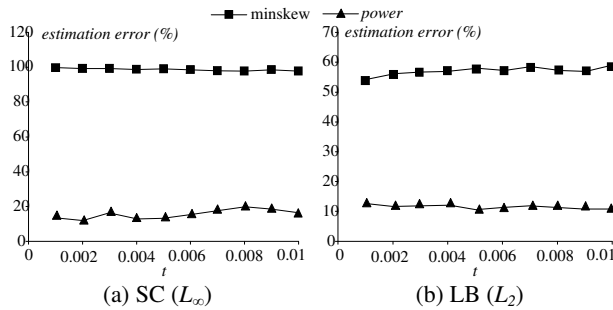


Figure 11. Error vs. dist threshold t (RDJ cost, $r=0.05$)

4.3 Estimation for unbiased queries

To study the efficiency of *power* for unbiased queries, we repeat the above experiments using workloads where query locations uniformly distribute in the data space. Figure 12 illustrates the results for RS selectivity prediction. Compared with biased workloads (Figure 7), the accuracy of all methods, except *global*, improves as many queries return empty results, in which case their estimation is trivial. The error of *global* increases because the average selectivity does not capture unbiased queries at all. Since the relative performance of all the methods is the same in the other experiments with uniform workloads, we omit them to avoid redundancy.

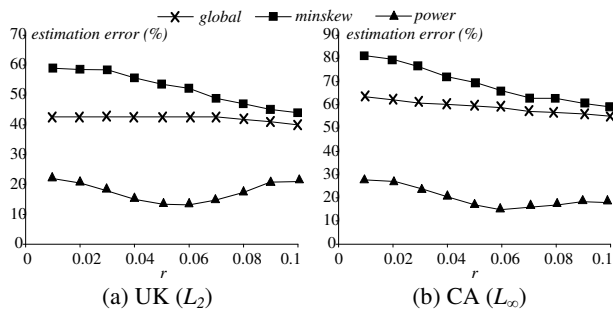


Figure 12. Results for unbiased queries (range search)

4.4 Pre-processing and estimation overhead

This section evaluates the computational overhead of the various methods. The global power law is omitted as its overhead is negligible. According to Table 1, *minskew* is slightly faster to construct than *power*. However, as discussed in Section 2.1, histograms (including *minskew*) may incur significant overhead

in general L_x metrics ($x \neq \infty$).

Table 1. Construction time (seconds)

Dataset	<i>Power</i>	<i>Minskew</i>
SC	18	15
UK	19	15
CA	38	32
LB	32	29

In order to verify this, Table 2 compares the estimation time for L_2 queries (for each method given the same radius r , the estimation time for L_2 RS/RDJ selectivity/cost estimation is similar). *The overhead of power is constant* because only one point is used to obtain the local coefficients, independently of the number of intersecting buckets. On the other hand, *minskew* is significantly more expensive and its overhead increases with the query radius, as more buckets partially intersect the query (for which the *monte-carlo* method must be performed).

Table 2. Estimation time for L_2 RS/RDJ queries (msec)

Query radius	<i>Power</i>	<i>Minskew</i>
0.01	0.5	3
0.03	0.5	5
0.05	0.5	9
0.07	0.5	15
0.09	0.5	25

5. CONCLUSION

The uniformity assumption is extremely easy to believe and analyze. Although it took many years before it was discredited, it is still lingering, hiding inside the “local uniformity assumption”, which is the basis underneath most multi-dimensional histograms. In this paper, not only we spot the “density trap” problem of the local uniformity assumption, but we also show how to resolve it, using a novel perspective, which leads to the concept of Local Power Law. The advantages of LPLaw are:

- It accurately models real datasets, and leads to single-formula estimation methods for selectivities of all the popular query types, for all the L_x distance metrics (Manhattan, Euclidean, etc.)
- It also leads to single-formula estimation methods for query I/O costs – something that no other published method can achieve.

We also propose a simple implementation of the LPLaw that is fast to initialize and run. Extensive experiments on several real datasets confirm the effectiveness, efficiency and flexibility of our techniques. Query optimization and data mining are related, both looking for methods to concisely describe a dataset. The parameters of LPLaw do exactly that: they use the local coefficients to describe the vicinity of a point. Thus, these coefficients are suitable for data mining and pattern recognition tasks. For example, in Figure 6a, data in the north-west part of Scandinavia have higher local exponents which, in retrospect, correspond to the Norwegian fjords (northwest Norway has LPLaw exponents in the range 1.3-1.5). This is actually a rule that can be used to detect outliers and extrapolate hidden/corrupted values: suppose that the very north part of Norway is not available – what can we say about it? Clearly, we can speculate that the points we are missing will have high

LPLaw exponents. We believe that the above examples are just the beginning. Exploiting LPLaws for discovering patterns in real datasets (rules, outliers, clusters) is a very promising area of research.

ACKNOWLEDGEMENTS

Yufei Tao and Dimitris Papadias were supported by grants HKUST 6197/03E and HKUST 6081/02E from Hong Kong RGC. Christos Faloutsos was supported by National Science Foundation under Grants No. IIS-9988876, IIS-0083148, IIS-0113089, IIS-0209107, IIS-0205224, by the Pennsylvania Infrastructure Technology Alliance (PITA) Grant No. 22-901-0001, and by the Defense Advanced Research Projects Agency under Contract No. N66001-00-1-8936.

REFERENCES

- [1] Acharya, S., Poosala, V., Ramaswamy, S. Selectivity Estimation in Spatial Databases. *SIGMOD*, 1999.
- [2] Aref, W., Samet, H. A Cost Model for Query Optimization Using R-Trees. *ACM GIS*, 1994.
- [3] An, N., Yang, Z., Sivasubramaniam, A. Selectivity Estimation for Spatial Joins. *ICDE*, 2001.
- [4] Bohm, C. A Cost Model for Query Processing in High Dimensional Data Spaces. *TODS*, 25(2): 129-178, 2000.
- [5] Berchtold, S., Bohm, C., Keim, D.A., Kriegel, H. A Cost Model for Nearest Neighbor Search in High-Dimensional Data Space. *PODS*, 1997.
- [6] Babcock, B., Chaudhuri, S., Das, G. Dynamic Sample Selection for Approximate Query Processing. *SIGMOD*, 2003.
- [7] Belussi, A., Faloutsos, C. Estimating the Selectivity of Spatial Queries Using the Correlation's Fractal Dimension. *VLDB*, 1995.
- [8] Bruno, N., Gravano, L., Chaudhuri, S. STHoles: A Workload Aware Multidimensional Histogram. *SIGMOD*, 2001.
- [9] Beyer, K., Goldstein, J., Ramakrishnan, R. When Is "Nearest Neighbor" Meaningful? *ICDT*, 1999.
- [10] Blohsfeld, B., Korus, D., Seeger, B. A Comparison of Selectivity Estimators for Range Queries on Metric Attributes. *SIGMOD*, 1999.
- [11] Beckmann, N., Kriegel, H., Schneider, R., Seeger, B. The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. *SIGMOD*, 1990.
- [12] Chaudhuri, S., Das, G., Datar, M., Motwani, R., Narasayya, V. Overcoming Limitations of Sampling for Aggregation Queries. *ICDE*, 2001.
- [13] Chaudhuri, S., Das, G., Narasayya. A Robust, Optimization-Based Approach for Approximate Answering of Aggregate Queries. *SIGMOD*, 2001.
- [14] Deshpande, A., Garofalakis, M., Rastogi, R. Independence Is Good: Dependency-Based Histogram Synopses for High-Dimensional Data. *SIGMOD*, 2001.
- [15] Faloutsos, C., Kamel, I. Beyond Uniformity and Independence, Analysis of R-trees Using the Concept of Fractal Dimension. *PODS*, 1994.
- [16] Faloutsos, C., Seeger, B., Traina, A., Traina, C. Spatial Join Selectivity Using Power Laws. *SIGMOD*, 2000.
- [17] Gunopulos, D., Kollios, G., Tsotras, V., Domeniconi, C. Approximate Multi-Dimensional Aggregate Range Queries over Real Attributes. *SIGMOD*, 2000.
- [18] Jermaine, C. Making Sampling Robust with APA. *VLDB*, 2003.
- [19] Jin, J., An, N., Sivasubramaniam, A. Analyzing Range Queries on Spatial Data. *ICDE*, 2000.
- [20] Lee, J., Kim, D., Chung, C. Multidimensional Selectivity estimation Using Compressed Histogram Information. *SIGMOD*, 1999.
- [21] Lin, X., Liu, Q., Yuan, Y., Zhou, X. Multiscale histograms: Summarizing topological relations in large spatial datasets. *VLDB*, 2003.
- [22] Muralikrishna, M., DeWitt, D. Equi-Depth Histograms for Estimating Selectivity Factors for Multi-Dimensional Queries. *SIGMOD*, 1988.
- [23] Mattias, Y., Vitter, J., Wang, M. Dynamic Maintenance of Wavelet-Based Histograms. *VLDB*, 2000.
- [24] Mattias, Y., Vitter, J., Wang, M. Wavelet-Based Histograms for Selectivity Estimation. *SIGMOD*, 1998.
- [25] Olken, F., Rotem, D. Random Sampling from Database Files: A Survey. *SSDBM*, 1990.
- [26] Poosala, Y., Ioannidis, Y. Selectivity Estimation without the Attribute Value Independence Assumption. *VLDB*, 1997.
- [27] Palmer, C., Faloutsos, C. Density Biased Sampling: An Improved Method for Data Mining and Clustering. *SIGMOD*, 2000.
- [28] Pagel, B., Korn, F., Faloutsos, C. Deflating the Dimensionality Curse using Multiple Fractal Dimensions. *ICDE*, 2000.
- [29] Pagel, B., Six, H., Toben, H., Widmayer, P. Towards an Analysis of Range Query Performance in Spatial Data Structures. *PODS*, 1993.
- [30] Sun, C., Agrawal, D., El Abbadi, A. Exploring Spatial Datasets with Histograms. *ICDE*, 2002.
- [31] [Http://www.census.gov/geo/www/tiger/](http://www.census.gov/geo/www/tiger/)
- [32] Thaper, N., Guha, S., Indyk, P., Koudas, N. Dynamic Multidimensional Histograms. *SIGMOD*, 2002.
- [33] Theodoridis, Y., Stefanakis, E., Sellis, T. Efficient Cost Models for Spatial Queries Using R-trees. *TKDE*, 12(1): 19-32, 2000.
- [34] Wu, Y., Agrawal, D., El Abbadi, A. Applying the Golden Rule of Sampling for Query Estimation. *SIGMOD*, 2001.
- [35] Wang, M., Vitter, J., Lim, L., Padmanabhan, S. Wavelet-Based Cost Estimation for Spatial Queries. *SSTD*, 2001.