Preventing Bad Plans by Bounding the Impact of Cardinality Estimation Errors

Guido Moerkotte
University of Mannheim
Mannheim, Germany
moerkotte@informatik.uni-mannheim.de

Thomas Neumann
Max Planck Institute for Informatics
Saarbrücken, Germany
neumann@mpi-inf.mpg.de

Gabriele Steidl
University of Mannheim
Mannheim, Germany
steidl@math.uni-mannheim.de

ABSTRACT

Query optimizers rely on accurate estimations of the sizes of intermediate results. Wrong size estimations can lead to overly expensive execution plans. We first define the *q-error* to measure deviations of size estimates from actual sizes. The q-error enables the derivation of two important results: (1) We provide bounds such that if the q-error is smaller than this bound, the query optimizer constructs an optimal plan. (2) If the q-error is bounded by a number q, we show that the cost of the produced plan is at most a factor of q^4 worse than the optimal plan. Motivated by these findings, we next show how to find the best approximation under the q-error. These techniques can then be used to build synopsis for size estimates. Finally, we give some experimental results where we apply the developed techniques.

1. INTRODUCTION

Query optimization relies on accurate cost calculations; inaccurate cost calculations may lead to (very) bad plans. Cost calculations require two kinds of estimations: First, size [1] estimations for intermediate results, and second, cost estimations for algebraic operators like joins. Size estimations are the input to the actual cost estimation functions. Whereas cost functions for algebraic operators, e.g. I/O cost estimations for different joins, are very accurate [4], typically less than three percent off the true execution times, size estimations tend to be more error prone. Clearly, this jeopardizes the accuracy of the cost functions. On the other hand, it is impossible to design 100 percent accurate size estimations in all cases. The importance of accurate size estimations is underlined by investigations undertaken by Reddy and Haritsa [15], who showed that commercial database systems are indeed very sensitive to slight changes in selectivity estimates. In Sec. 3, we underpin this experimental finding with theoretical results.

Given this database lore, some fundamental (and, from a practical point of view, urgent) questions arise. The first question is:

Q1 If we have to live with inaccurate size estimations, what is the best way to measure size estimation errors?

Why is this question fundamental? The reason is that if this question has been answered correctly, other fundamental questions can be answered as well. These are:

Q2a How can we minimize error propagation?
This question is fundamental since it is well-known that errors propagate exponentially through joins [7].

Q2b Are there bounds for size estimation errors such that if the size estimation error stays within these bounds the generated plan will still be optimal? And if such bounds exist, what do they look like?

Q2c Assume the optimal plan for some query is P, and the plan produced due to size estimation errors is $\hat{P}$. If a bound for the size estimation error is known, does this limit the cost of $\hat{P}$ compared to P? That is, can we somehow translate a size estimation error bound into a bound on the deviation of the true execution costs of $\hat{P}$ compared to those of P?

If we have positive answers to queries Q2 a-c and are thus convinced that our error metric is the one of choice, we immediately will ask:

Q3 How can we minimize size estimation errors under the devised metric?

None of these questions has been answered satisfactorily in previous literature, in particular, the state of the art in selectivity estimation offers no bounds for the quality of the resulting plans. We will address these questions one after the other in this paper, starting with a suitable error metric, called *q-error* (Section 2, answering Q1). The motivation for the definition follows in Sec. 3 by answering questions Q2a-c. The answers will suggest that minimizing the q-error is crucial. Thus, we develop an algorithm that computes optimal approximations under the q-error in Section 4 (answering Q3). Finally, we show how the best approximation under the q-error can be used to derive cardinality estimates by applying them to single and multiple buckets (Sec. 5). The resulting synopses are very accurate and allow us to derive bounds for the costs of constructed plans, something that was out of reach for previous approaches. We conclude the

[1] The term size is used to denote (1) the size of a relation in number of pages or (2) the size of a relation in terms of its cardinality. They are easily convertable into each other.

VLDB '09, August 24-28, 2009, Lyon, France

paper with a discussion of related (Section 6) and future work (Section 7). Due to space restrictions, we will not give all proofs (see [12]).

2. DEFINING THE Q-ERROR

We now define the q-error, which is multiplicative in nature. We first look at the approximation problem formulation of selectivity estimation, and then study different error metrics. We will discuss the rational behind the q-error in Sec. 3.

Let R be a relation, A one of its attributes, and $\{x_1, \ldots, x_m\} = \Pi_A(R)$ the set of distinct values of A. Then, the frequency density is a set of pairs (x_i, f_i) with $f_i = |\sigma_{A=x_i}(R)|$, $1 \leq i \leq m$.

The task is to approximate this set of pairs by a function $\hat{f}$. Then, $\hat{f}(x_i) =: \hat{f}_i$ gives the estimate $\hat{f}_i$ of f_i. Typically, $\hat{f}$ is a (piecewise) polynomial of degree 0 or 1.

The approximation function is then used to estimate the result cardinalities of various algebraic expressions. For example, the result cardinality of $\sigma_{A=c}(R)$ can then be calculated as $\hat{f}(x_i)$, and the result cardinality of $\sigma_{c_1 \leq A \leq c_2}(R)$ can be calculated as

$$\sum_{c_1 \leq x_i \leq c_2} \hat{f}(x_i).$$

More formulas for other algebraic expressions exist but are not in the focus of the paper. Instead, we are interested in (1) how to measure the error, i.e. the deviation of $\hat{f}_i$ from f_i, and (2) how to find the best approximation under the new error measure.

Typically, norms are used to define the error. Therefore, the correct values are organized into a vector $\vec{b} = (f_1, \ldots, f_m)^{\mathrm{T}} \in \mathbb{R}^m$ and the estimates into a vector $\hat{\vec{b}} = (\hat{f}_1, \ldots, \hat{f}_m)^{\mathrm{T}} \in \mathbb{R}^m$. Well-known l_p error metrics are based on l_p norms as in

$$||b - \hat{b}||_p,$$

where $1 \leq p \leq \infty$ and the most common norms are

$$\begin{aligned} ||z||_2 &= \sqrt{(z_1)^2 + \ldots + (z_m)^2} \\ ||z||_\infty &= \max_{i=1}^{m} |z_i| \end{aligned}$$

for $z = (z_1, \ldots, z_m)^{\mathrm{T}} \in \mathbb{R}^m$. While the l_2 error does not give bounds on estimates, l_∞ does. Define $\Delta = ||b - \hat{b}||_\infty$. Then

$$f_i - \Delta \leq \hat{f}_i \leq f_i + \Delta.$$

The absolute error bounds are not really useful. Consider an example where $\Delta = 500$ and $f_1 = 1000$ and $f_2 = 10,000$. Although the absolute error is the same for both cases, the error produced for f_1 is much worse as it amounts to a factor of two.

We need an error metrics, which is multiplicative in nature. Thus, we first define for $z \in \mathbb{R}$

$$||z||_Q = \begin{cases} \infty & \text{if } z \leq 0 \\ 1/z & \text{if } 0 < z \leq 1 \\ z & \text{if } 1 \leq z \end{cases}$$

Note that for $z > 0$, we have $||z||_Q = \max(z, 1/z)$. Now, for $z \in R^m$, we define

$$||z||_Q = \max_{i=1}^{m} ||z_i||_Q. \quad (1)$$

We denote $||\cdot||_Q$ by l_q. However, be careful: l_q is *not* a norm. Subadditivity (triangular inequality) is the only one of the three properties required by a norm, which is satisfied by l_q.

Let $\vec{a}$ and $\vec{b}$ be two vectors in R^m where $b_i > 0$. Define $\vec{a}/\vec{b} = \frac{\vec{a}}{\vec{b}} = (a_1/b_1, \ldots, a_n/b_n)^{\mathrm{T}}$. Then, we define the *q-error* of an estimation $\hat{b}$ of b as

$$||\hat{b}/b||_Q.$$

As l_∞, l_q produces valid, symmetric bounds for individual estimates. Define $q = ||\hat{b}/b||_Q$. Then,

$$(1/q) f_i \leq \hat{f}_i \leq q f_i.$$

Note that the error bounds are symmetric and multiplicative. The latter feature is very important, as we will see in the next section. Let us illustrate the error bounds by a small example. Assume $q = 2$ and the estimate is 1000. Then, we know that the true value lies between 500 and 2000.

Let us give another example to compare the q-error with other error metrics. Consider the data points (1,20), (2,10), and (3,60). We first consider two approximations by a single number. In histograms, the average frequency is used as an approximation of the frequencies occurring in a bucket [14]. It is well-known that the average minimizes the l_2 error. Let $\hat{f}_{30}$ be a function always returning 30, which happens to be the average of $\{20, 10, 60\}$. For a given set of numbers Y, define the *q-middle* as $\sqrt{\min Y * \max Y}$. The q-middle of 10, 20, 60 is $\sqrt{600}$. Under l_q, the q-middle provides the best approximation by a single number. Thus, the q-middle minimizes the q-error. The q-error of the q-middle can be directly calculated and equals $(q/\min Y) = (\max Y)/q$, if q is the q-error of Y, or even simpler as $\sqrt{\max Y / \min Y}$. Let $\hat{f}_{\sqrt{600}}$ be the constant function returning $\sqrt{600}$. Next, we consider linear functions to approximate the three data points. Further, denote by $\hat{f}_2$, $\hat{f}_\infty$, and $\hat{f}_q$ the best approximation of the data points by a linear function $a_0 + a_1 x$ under l_2, l_∞, and l_q. Then, the following table gives the coefficients a_i of the approximation functions and their error under l_2, l_∞, and l_q.

	$\hat{f}_{30}$	$\hat{f}_{\sqrt{600}}$	$\hat{f}_2$	$\hat{f}_q$	$\hat{f}_\infty$
a_0	30	$\sqrt{600}$	-6	0	-15
a_1	0	0.0	17	10	20
l_2	20	33.5	14.0	17	15
l_∞	30	27.6	18.0	30	15
l_q	3	2.4	2.8	2	4

The table shows us the expected: different error metrics result in different best approximations and different approximations result in different errors. The question is which metric is the best for query optimization purposes.

3. WHY Q?

After defining the q-error, we now give strong evidence why it is superior to other error metrics for selecivity estimation purposes. We will use different cost functions C, which assign costs to plans P. For example, we will use different join cost functions to account for different implementations of the join operator. All these cost functions take the sizes of the inputs to the join operator as parameters. Thus, we have to state precisely the size estimation we use to calculate costs. Hence, we define the following. For a given plan P and a cost

function C, $C(P)$ denotes the calculated costs for the plan P if the true intermediate result sizes are used and $\hat{C}(P)$ denotes the costs if they are calculated using estimates for the intermediate result sizes.

Sometimes, we require that the cost function has the ASI property (see [5, 10]). For convenience, the appendix contains the definition of the ASI property. At other times, we will use the real cost functions developed by Haas et al. [4]. Since the latter have occurrences of $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$, and we need the cost functions to be continous, we eliminate all occurrences of the these symbols. The error introduced is small. Further, real cost functions require that a certain amount of memory is given to each individual join. To eliminate these variables, we introduce the *fixed memory allocation scheme* assumption. It states that every join (in every plan) receives the same amount of memory. Further, if the memory is partitioned into different parts (e.g. input and output buffers), we further assume that this partitioning scheme is fixed and the same applies to all joins. Let us denote by C_{SMJ} the so modified cost function for the sort merge join, and by C_{GHJ} the so modified cost function for the Grace hash join as specified in [4]. These cost functions are used in theorems. In the experiments, the original cost functions with optimal memory allocation schemes are used.

3.1 Minimizing Error Propagation

The purpose of this section is to demonstrate the multiplicative nature of error propagation. Nothing here is new or surprising. Just the conclusion we draw is new. Let us assume that the purpose of our approximation is to estimate the output cardinalities of selections on relations R_i, i.e. $\sigma_{p_i}(R_i)$ for $i = 1, \ldots, n$. The results of these cardinality estimations are then used to find the optimal order of subsequent joins. More specifically, assume we have to find the optimal query execution plan for the following expression:

$$\sigma_{p_1}(R_1) \bowtie \ldots \bowtie \sigma_{p_n}(R_n), \tag{2}$$

where we here intentionally left out all the join predicates. Ioanidis and Christodoulakis pointed out that errors propagate exponentially through joins [7]. Denote by s_i the cardinality of $\sigma_{p_i}(R_i)$ and by $\hat{s}_i$ its estimate. Further assume that independence holds. We can write s_i as $f_i|R_i|$, where f_i is the selectivity of p_i. Denote by $f_{i,j}$ the selectivity of the join predicate between R_i and R_j, if it exists. Otherwise, we define $f_{i,j} = 1$. Due to the independence assumption, the result cardinality of joining a subset $x \subseteq \{R_1, \ldots, R_n\}$ is

$$s_x = (\prod_{R_i \in x} f_i)(\prod_{R_i, R_j \in x} f_{i,j})(\prod_{R_i \in x} |R_i|)$$

Denote by $\hat{f}_i$ the estimate for the selectivity of p_i and assume that the join selectivities have been estimated correctly (which, of course, is difficult in practice). Then the estimated cardinality of the result of joining the relations in x is

$$\begin{aligned}
\hat{s}_x &= (\prod_{R_i \in x} \hat{f}_i)(\prod_{R_i, R_j \in x} f_{i,j})(\prod_{R_i \in x} |R_i|) \\
&= (\prod_{R_i \in x} f_i/f_i)(\prod_{R_i \in x} \hat{f}_i)(\prod_{R_i, R_j \in x} f_{i,j})(\prod_{R_i \in x} |R_i|) \\
&= (\prod_{R_i \in x} \hat{f}_i/f_i)(\prod_{R_i \in x} f_i)(\prod_{R_i, R_j \in x} f_{i,j})(\prod_{R_i \in x} |R_i|) \\
&= (\prod_{R_i \in x} \hat{f}_i/f_i)s_x,
\end{aligned}$$

where some i belong to the category with $\hat{f}_i/f_i < 1$ and others to the one with $\hat{f}_i/f_i > 1$. Remember that during dynamic programming, all subsets of relations are considered. Especially those subsets occur in which all relations belong only to one category. Hence, building on the cancellation of errors by mixing them from different categories is not a real option. Instead, we should minimize

$$\prod_{R_i \in x} \max\{f_i/\hat{f}_i, \hat{f}_i/f_i\} = \prod_{R_i \in x} ||\frac{\hat{f}_i}{f_i}||_Q$$

in order to minimize errors and error propagation. This product can be minimized by minimizing each of its factors. Thus, we can draw the following conclusion: If we want to minimize error propagation, we have to minimize the multiplicative error $||\hat{f}_i/f_i||_Q$.

3.2 Bounds for Q Which Guarantee Plan Optimality

We now derive bounds on the q-error such that if these bounds are met, the plan produced by the query optimizer using size estimates instead of the correct sizes still has minimal costs. Let us consider again the join expression given in 2. Denote by f_i the correct selectivity of σ_{p_i} and by $\hat{f}_i$ its estimate. If the plan generator uses the correct cardinalities, it produces the optimal plan. Given the estimates, it might produce another plan. The question is, how far the cardinality estimates can deviate from the true cardinalities without affecting the optimality of the resulting plan. More formally, denote by P the optimal plan under the correct cardinalities f and by $\hat{P}$ the optimal plan under the estimates $\hat{f}$. Then, we can restate the above question to whether there exists a condition on $\hat{f}$ such that if this condition holds then $C(\hat{P}) = C(P)$.

Let us slide in a reminder on query graphs (see [19]). Queries can be mapped to undirected graphs, called query graphs, as follows. The nodes in a query graph are the relations referenced in the query. For every join predicate between relations R and S in the query, the query graph contains an edge between R and S. A query is called *acyclic* if its query graph is acyclic. A query in relations $R_1, \ldots, R_n$ is called a *chain query* if its query graph is a chain, i.e. the edges are (R_i, R_{i+1}) for $1 \leq i < n$. A query in relations $R_0, \ldots, R_n$ is called a *star query*, for all relations R_i, $1 \leq i \leq n$, there is an edge between R_0 and R_i, and these are the only edges. R_0 is called the *center relation* and the R_i, $1 \leq i \leq n$ are called the *satellite relations*.

First consider the following problem. Let a star query and a cost function, which has the ASI property [5], be given. The task is then to find an optimal left-deep tree not containing any cross products. An immediate consequence of the ASI

property is that the optimal join order for a given star query is the one which starts with the center relation followed by the satellite relations sorted according to their rank. This fact can be used to prove the following theorem.

THEOREM 3.1. *Let C be a cost function with ASI property. For a given star query in n relations, let P be the optimal left-deep plan without cross products under C and $\hat{P}$ the optimal left-deep plan without cross products under $\hat{C}$. If for all $1 \leq k \leq n$ and $r_i := f_{0,i}|R_i|$*

$$||\frac{f_k}{\hat{f}_k}||_Q < \min_{i \neq j} \sqrt{||\frac{f_i r_i}{f_j r_j}||_Q} \quad =: q,$$

then $C(\hat{P}) = C(P)$.

The condition of the theorem implies the weaker condition that for all $1 \leq i, j \leq n$, $i \neq j$

$$||\frac{\hat{f}_i}{f_i}||_Q \; ||\frac{\hat{f}_j}{f_j}||_Q < ||\frac{f_i r_i}{f_j r_j}||_Q \quad =: q_{i,j}$$

which is already sufficient for the proof. Indeed, we show in the technical report that if this condition is true, then the relative order of the ranks of the relations remains untouched, which in turn implies that the optimal plan remains the same (if there are no ties). For later use, define $q := min_{i,j} q_{i,j}$.

Whenever there are star queries, chain queries are typically not far away.

THEOREM 3.2. *Let C be a cost function with ASI property. For a given chain query in n relations, let P be the optimal left-deep plan without cross products under C and $\hat{P}$ be the optimal left-deep plan without cross products under $\hat{C}$. If for all $1 \leq k \leq n$*

$$||\frac{\hat{f}_k}{f_k}||_Q < \min_{i \neq j-1} \sqrt{||\frac{f_i f_{i,i+1}|R_i|}{f_j f_{j,j-1}|R_j|}||_Q} \quad =: q,$$

then $C(\hat{P}) = C(P)$.

Again, the proof makes use of the fact that this condition implies that the relative order according to the rank of the relations remains untouched (see technical report for details).

For tree queries, things are a little more complex. In the following theorem, we assume that $R_{i'}$ is a relation connected to R_i, which we denote by $R_{i'} - R_i$.

THEOREM 3.3. *Let C be a cost function with ASI property. For a given acyclic query in n relations, let P be the optimal left-deep plan without cross products under C, and $\hat{P}$ be the optimal left-deep plan without cross products under $\hat{C}$. If for all $1 \leq k \leq n$*

$$||\frac{\hat{f}_k}{f_k}||_Q < \min_{i \neq j, R_{i'} - R_i, R_{j'} - R_j} \sqrt{||\frac{f_i f_{i,i'}|R_i|}{f_j f_{j,j'}|R_j|}||_Q} \quad =: q,$$

then $C(\hat{P}) = C(P)$.

Again, the proof makes use of the fact that this condition implies that the relative order according to the rank of the relations remains untouched (see technical report for details).

For a real cost function, we needed strong assumptions to prove a similar theorem.

THEOREM 3.4. *Assume independence holds and all join selectivities are equal. For a given star query in n relations, let P be the optimal left-deep plan without cross products under C_{SMJ}, and $\hat{P}$ be the optimal left-deep plan without cross products under $\hat{C}_{SMJ}$. If for all $1 \leq i, j \leq j$, $i \neq j$*

$$||\frac{\hat{f}_i}{f_i}||_Q \; ||\frac{\hat{f}_j}{f_j}||_Q < ||\frac{f_i|R_i}{f_j|R_j|}||_Q \quad =: q^2,$$

then $C_{SMJ}(\hat{P}) = C_{SMJ}(P)$.

The proof of this theorem is very tedious, as many cases have to be considered. Please see the technical report for details.

In all theorems, a bound q was established such that if $||\hat{f}_i/f_i||_Q < q$, this estimation error does not affect the optimality of the plan if the costs are calculated using the estimates of the intermediate result sizes instead of the true intermediate result sizes. Thus, we strongly believe that $||\hat{f}_i/f_i||_Q$ is well-suited to measure size estimation errors.

However, in the theorems, we needed some assumptions and idealized cost functions. So, let us give an example using the real cost function of the Grace hash join [4]. Now, we do not use a fixed memory allocation scheme but calculate the optimal one using the method given in [4]. Although this cost function is not covered by the above theorems, the central claim still remains valid. To see this, consider a chain query in three relations and a star query with three satellite relations. We have successively increased the q-error $q = \max_{i=1,3} ||\hat{f}_i/f_i||_Q$ for estimating the size of $\sigma_{p_1}(R_1)$ and $\sigma_{p_3}(R_3)$ (in both queries). The theorems would claim that the costs of the best plan produced under size estimation errors equal the costs of the best plan as long as the error does not reach a certain threshold. This is also the case here. Figure 1 shows the increasing q-error on the estimations on the x-axis, and the cost ratio of the plan produced by the query optimizer under size estimation errors and the cost of the best plan ($C(\hat{P})/C(P)$) on the y-axis. We see that for both queries the costs of the optimal plan remain the same as long as the q-error is limited by 1.9 ($q \leq 1.9$). Beyond that, the optimal plan $\hat{P}$ under $\hat{C}$ has higher costs than the optimal plan P under C. For the star query and $q = 2.0$ or higher, $C(\hat{P})$ is more than eleven times (11.11) higher than $\hat{C}(P)$, which is very bad. For the chain query and $q = 2.1$ or higher, $C(\hat{P})$ is more than eight times higher than $\hat{C}(P)$, which is also very bad. For $q = 2.0$, the optimal plan under $\hat{C}$ is about three times more expensive than the optimal plan under C. We can conclude that bounding q matters because $C(\hat{P})/C(P)$ may jump badly. Increasing q further than 2.1 does not change the plan anymore.

For the star query, we see that

$$C(\hat{P})/C(P) = 11.11 = 2^{3.46} = q^{3.46}$$

and might ask whether there is a bound such that

$$C(\hat{P})/C(P) \leq q^e$$

for some e. Such a bound indeed exists, as we will see in the next subsection.

In case the reader wishes to verify the results, the following table summarizes the parameters used:

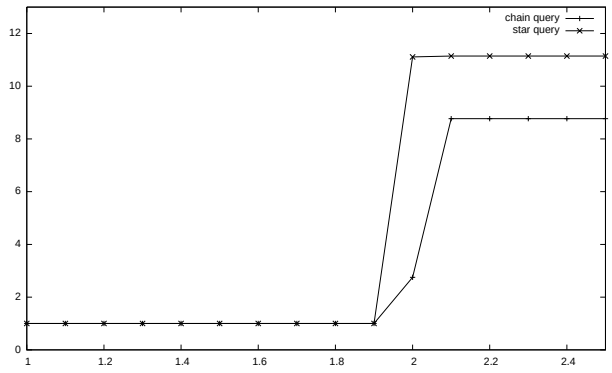

Figure 1: q **and** $C(\hat{P})/C(P)$

	chain query	star query
s_i	$2^{i-1} * 10111$	$2^{i-1}1000$
$f_{i,j}$	0.0042	0.02
buffer size	m/2m	m/10m/50m
m	1111	100
$\hat{s}_0$	–	s_0
$\hat{s}_1$	$qf_1\|R_1\|$	$qf_1\|R_1\|$
$\hat{s}_2$	s_2	s_2
$\hat{s}_3$	$1/qf_3\|R_3\|$	$1/qf_3\|R_3\|$

where sizes are given in number of pages. The buffer sizes are given for the first/second(/third) join. There are three more constants in the cost formulas. They describe disk characteristics and are chosen as in Table 1 in [4]. The values are $T_s = 9.5$, $T_x = 2.6$, $T_L = 8.3$.

3.3 Cost Bounds Implied by Q

As we have seen, wrong cardinality estimates can lead to plans $\hat{P}$, which are much worse than the best plan P. Now, we show how to translate bounds on $||\hat{s}/s||_Q$ into bounds on $C(\hat{P})/C(P)$, thus answering question Q2c.

THEOREM 3.5. *Let $C = C_{SMJ}$ or $C = C_{GHJ}$. For a given query in n relations, let P be the optimal plan under C, and $\hat{P}$ be the optimal plan under $\hat{C}$. Then*

$$C(\hat{P}) \leq q^4 C(P),$$

where q is defined as

$$q = \max_{x \subseteq X} ||\hat{s}_x/s_x||_Q,$$

with X beeing the set of relations to be joined. That is, q is the maximum estimation error taken over all intermediate results.

PROOF. We sketch the proof of this theorem. For two relations R_1 and R_2, denote by s_i the size of relation R_i. The cost function for the sort merge join developed by Haas et al. [4] can be expressed as a polynomial of degree two in variables s_1 and s_2 (see appendix). Further, all coefficients of this polynomial are positive. Denote this polynomial by $g_{\text{SMJ}}(s_1, s_2)$. For any $q > 1$, we have $g_{\text{SMJ}}(qs_1, qs_2) \leq q^2 g_{\text{SMJ}}(s_1, s_2)$. Let P be a plan whose join operators are all sort merge joins. Then, the total costs $C(P)$ are calculated by the sum over the costs for each sort merge join in the plan. The costs of every sort merge join are a polynomial of degree two with two variables. These variables are the input sizes of its two arguments. Thus, the overall cost function can be expressed as a polynomial of degree two, with coefficients greater than or equal to zero, and in variables s_x for all subsets x of $X = \{R_1, \ldots, R_n\}$, if these are the relations to be joined in P. It follows that

$$C(\hat{P}) \leq q^2 \hat{C}(\hat{P})$$

and

$$\hat{C}(P) \leq q^2 C(P),$$

if q is defined as in the theorem.

Since $\hat{P}$ is the optimal plan under $\hat{C}$, we have

$$\hat{C}(\hat{P}) \leq \hat{C}(P).$$

Concatenating the last three inequalities yields

$$\begin{aligned} C(\hat{P}) &\leq q^2 \hat{C}(\hat{P}) \\ &\leq q^2 \hat{C}(P) \\ &\leq q^4 C(P) \end{aligned}$$

The proof for C_{SMJ} follows the same line of reasoning. □

Note that we did not need any assumption on how the intermediate result sizes where derived. Especially, we made no use of the independence assumption.

It might be surprising to see that the cost deviation factor q^4 does not depend on n. Intuitively, one might expect this factor to grow with n. However, note that q limits the estimation error for all intermediate result sizes. Thus, q tends to increase if n increases, as errors propagate through joins (see Sec. 3.1).

Consider again the table at the end of Sec.2. The bound for $C(\hat{P})/C(P)$ improves from $3^4 = 81$ over $2.5^4 \approx 39$ to $2^4 = 16$ if we move from approximating the values by their average, over q-middle to a linear function. Remember that the frequencies in a bucket in a histogram are approximated by their average.

4. BEST APPROXIMATION UNDER L_Q

Since the q-error has such a large impact on plan optimality, we now develop an algorithm for finding the best approximation under l_q. This section makes heavy use of math. Therefore, we split it into two parts to make it more accessible: Section 4.1 derives the characteristics of an optimal solution, and Section 4.2 shows how to construct the optimal solution. Note that although the math is somewhat involved, the final algorithm is not that complex, it is shown in Figure 2. But as the derivation of the algorithm is somewhat hard to follow without mathematical background, we sketch a more intuitive derivation of the algorithm in Section 4.3. It can be read instead of the proofs to understand the algorithm.

4.1 Characterization of a Solution

Let (x_i, b_i) for $1 \leq i \leq m$ be a set of points with $b_i > 0$, which we want to approximate by a linear combination of a given set of functions Φ_j, $1 \leq j \leq n$. We measure the deviation by applying l_q. That is, we want to find the coefficients a_j such that the function

$$\hat{f}(x) = \sum_{j=1}^{n} a_j \Phi_j(x)$$

minimizes

$$\max_{i=1,\ldots,m} \max \left\{ \frac{b_i}{\hat{f}(x_i)}, \frac{\hat{f}(x_i)}{b_i} \right\}.$$

Let $A \in \mathbb{R}^{m\times n}$ be a matrix where $m > n$, and $\vec{b} = (b_1, \ldots, b_m)^{\mathrm{T}}$ be a vector in $\mathbb{R}^m$ where $b_i > 0$. Then we can state the problem as

$$\text{find } \vec{a} \in \mathbb{R}^n \text{ that minimizes } ||A\vec{a}/\vec{b}||_Q \tag{3}$$

under the constraint that $\alpha_i^{\mathrm{T}} > 0$, $1 \le i \le m$, for all row vectors α_i of A.

Alternatively, we can modify A by 'dividing' it by $\vec{b}$. Let $\vec{b} = (b_1, \ldots, b_m)^{\mathrm{T}}$ be a vector in $\mathbb{R}^m$. Define $\mathrm{diag}(\vec{b})$ to be the $m \times m$ diagonal matrix which contains the b_i in its diagonal and is zero outside the diagonal. For vectors $\vec{b}$ with $b_i > 0$, we can define $\vec{b}^{-1} = (1/b_1, \ldots, 1/b_m)^{\mathrm{T}}$.

Using these notations, we can define

$$A' = \mathrm{diag}(\vec{b}^{-1})A$$

In the special case of univariate polynomial approximation with $\hat{f}(x) = a_1 + a_2 x + \ldots + a_n x^{n-1}$, the matrix A' has the form

$$A' = \begin{pmatrix} 1/b_1 & x_1/b_1 & \ldots & x_1^{n-1}/b_1 \\ 1/b_2 & x_2/b_2 & \ldots & x_2^{n-1}/b_2 \\ \vdots & \vdots & \ldots & \vdots \\ 1/b_m & x_m/b_m & \ldots & x_m^{n-1}/b_m \end{pmatrix}. \tag{4}$$

We can solve Problem 3 if we can solve

$$\text{find } \vec{a} \in \mathbb{R}^n \text{that minimizes} ||A\vec{a}||_Q. \tag{5}$$

The following proposition ensures that a solution to this general problem exists. Further, since $||A\vec{a}||_Q$ is convex, the minimum is a global one.

PROPOSITION 4.1. *Let $A \in \mathbb{R}^{m,n}$ such that $\mathcal{R}(A) \cap \mathbb{R}^m_{>0} \neq \emptyset$. Then $||A \cdot ||_Q$ attains its minimum.*

PROOF. Since $\mathcal{R}(A) \cap \mathbb{R}^m_{>0} \neq \emptyset$, there exists $a \in \mathbb{R}$ such that $\mathrm{lev}_a Q(A\cdot)$ is nonempty. Moreover, then $\mathrm{lev}_a Q(A\cdot)$ is obviously compact. Now it is well-known, see [1, p. 14], that a lower semicontinuous function which has a nonempty, compact level set for some $a \in \mathbb{R}$ attains its minimum. □

Note that l_q is subadditive and convex. Further, it is lower semi-continuous (see also [16, p. 52]). However, it is not strictly convex. Remember that strict convexity typically implies the uniqueness of a solution to approximation problems. Later on, we will show that under certain conditions uniqueness still holds although l_q is not strictly convex.

We need some more notation. Let $A \in \mathbb{R}^{m,n}$. We denote by $\mathcal{R}(A) = \{A\vec{a} \mid \vec{a} \in \mathbb{R}^n\}$ the *range* of A and by $\mathcal{N}(A) = \{\vec{a} \in \mathbb{R}^n \mid A\vec{a} = 0\}$ the *nullspace* of A.

Problem (5) can be rewritten as the following constrained minimization problem:

$$\min_{(\vec{a},q)\in\mathbb{R}^n\times\mathbb{R}} q \qquad \text{subject to} \qquad \frac{1}{q} \le A\vec{a} \le q \quad \text{and} \quad q \ge 1. \tag{6}$$

The Lagrangian of (6) is given by

$$\begin{aligned} L(\vec{a}, q, \lambda^+, \lambda^-, \mu) \;:=\; & q \\ & -(\lambda^+)^{\mathrm{T}}(q - A\vec{a}) \\ & -(\lambda^-)^{\mathrm{T}}(A\vec{a} - \frac{1}{q}) \\ & \mu(q-1). \end{aligned}$$

Assume that $\mathcal{R}(A) \cap \mathbb{R}^m_{>0} \neq \emptyset$. Then the set $\{(\vec{a}, q) : \frac{1}{q} \le A\vec{a} \le q \text{ and } q \ge 1\}$ is non-empty and closed, and there exists $(\vec{a}, q)$ for which we have strong inequality in all conditions. Then the following *Karush-Kuhn-Tucker* conditions are necessary and sufficient for $(\hat{\vec{a}}, \hat{q})$ to be a minimizer of (6), see, e.g., [18, p. 62]: there exist $\hat{\lambda}^+, \hat{\lambda}^- \in \mathbb{R}^m_{\ge 0}$ and $\hat{\mu} \ge 0$ such that

$$\nabla_{\vec{a}} L(\hat{\vec{a}}, \hat{q}, \hat{\lambda}^+, \hat{\lambda}^-, \hat{\mu}) \;=\; A^{\mathrm{T}}\lambda^+ - A^{\mathrm{T}}\lambda^- \;=\; 0 \tag{7}$$

$$\begin{aligned} \frac{\partial}{\partial q} L(\hat{\vec{a}}, \hat{q}, \hat{\lambda}^+, \hat{\lambda}^-, \hat{\mu}) \;=\; & 1 - \sum_{i=1}^m \hat{\lambda}_i^+ \\ & -\frac{1}{q^2}\sum_{i=1}^m \hat{\lambda}_i^- - \mu \;=\; 0 \end{aligned} \tag{8}$$

and for $i = 1, \ldots, m$,

$$\hat{\lambda}_i^+ \left(\hat{a} - (A\hat{\vec{a}})_i\right) \;=\; 0, \tag{9}$$

$$\hat{\lambda}_i^- \left((A\hat{\vec{a}})_i - \frac{1}{\hat{q}}\right) \;=\; 0, \tag{10}$$

$$\hat{\mu}(\hat{q} - 1) \;=\; 0.$$

Assume that $1_m \notin \mathcal{R}(A)$, where 1_m is the vector with all components 1. Then $\hat{q} > 1$ and consequently $\hat{\mu} = 0$. Furthermore, it is clear that $\hat{\lambda}_i^+$ and $\hat{\lambda}_i^-$ cannot both be positive because the conditions $\hat{q} = (A\hat{\vec{a}})_i$ and $\frac{1}{\hat{q}} = (A\hat{\vec{a}})_i$ cannot be fulfilled at the same time, since $\hat{q} > 1$.

Setting $\hat{\lambda} := \hat{\lambda}^+ - \hat{\lambda}^-$, we can summarize our findings (7) - (10) in the following theorem.

THEOREM 4.1. *Let $A \in \mathbb{R}^{m,n}$ such that $\mathcal{R}(A) \cap \mathbb{R}^m_{>0} \neq \emptyset$ and $1_m \notin \mathcal{R}(A)$. Then $(\hat{\vec{a}}, \hat{q})$ solves (6) if and only if there exists $\hat{\lambda} \in \mathbb{R}^m$ such that*

i) $A^T\hat{\lambda} = 0$.

ii) $q = q \sum_{\hat{\lambda}_i > 0} \hat{\lambda}_i + \frac{1}{q} \sum_{\hat{\lambda}_i < 0} \hat{\lambda}_i$.

iii) $\hat{\lambda}_i = 0$ *if* $\frac{1}{\hat{q}} < (A\hat{\vec{a}})_i < q$.

iv) if $\hat{\lambda}_i > 0$ *then* $(A\hat{\vec{a}})_i = \hat{q}$ *and if* $\hat{\lambda}_i < 0$ *then* $(A\hat{\vec{a}})_i = 1/\hat{q}$.

PROOF. Follows from equations (7) - (10). □

Remark. We see that $1 < \hat{q} = (A\hat{\vec{a}})_i$ implies

$$\mathrm{sign}\left((A\hat{\vec{a}})_i - 1\right) = 1$$

and that $1 > 1/\hat{q} = (A\hat{\vec{a}})_i$ implies

$$\mathrm{sign}\left((A\hat{\vec{a}})_i - 1\right) = -1$$

and hence $\hat{\lambda}_i \left((A\hat{\vec{a}})_i - 1\right) \ge 0$. For our approximation problem (3), this means that the residuum $\hat{f}(x_i) - b_i$ fulfills $\hat{\lambda}_i \left(\hat{f}(x_i) - b_i\right) \ge 0$.

Under certain conditions, problem (5) has a unique solution which can be simply characterized. Let us start with some straightforward considerations in this direction. If $\mathcal{N}(A) \neq \{\vec{0}\}$, then we have for any minimizer $\hat{\tilde{a}}$ of $||A \cdot ||_Q$ that $\hat{\tilde{a}} + \beta$, $\beta \in \mathcal{N}(A)$ is also a minimizer. In particular, we have that $\mathcal{N}(A) \neq \{\vec{0}\}$ if

- $m < n$,
- $m \geq n$ and A is not of full range, i.e., $\mathrm{rank}(A) < n$.

In these cases, we cannot have a unique minimizer. Further note that if $1_m \in \mathcal{R}(A)$, then the minimum of $||A \cdot ||_Q$ is 1, and the set of minimizers is given by

$$A^+ 1_m + \mathcal{N}(A),$$

where A^+ denotes the Moore-Penrose inverse of A.

In the following, we restrict our attention to the case $m > n$ and $\mathrm{rank}(A) = n$. The following proposition considers $(n+1, n)$–matrices.

PROPOSITION 4.2. *Let $A \in \mathbb{R}^{n+1,n}$ such that $\mathcal{R}(A) \cap \mathbb{R}^{n+1}_{>0} \neq \emptyset$, $1_m \notin \mathcal{R}(A)$ and $\mathrm{rank}(A) = n$. Then $||A \cdot ||_Q$ has a unique minimizer if and only if the Lagrange multipliers $\hat{\lambda}_i$, $i = 1, \ldots, n+1$ are not zero.*

PROOF. 1. W.l.o.g. let the first n rows of A be linearly independent. Let $x_i := (A\alpha)_i$, $i = 1, \ldots, n+1$. Further, let $\tilde{A} := A|_{\{1,\ldots,n\}}$ be the restriction of A to its first n rows and $\tilde{x} := (x_1, \ldots, x_n)^{\mathrm{T}}$. By assumption, there exists $\tilde{A}^{-1}$ and we obtain that $\alpha = \tilde{A}^{-1}\tilde{x}$ and

$$x_{n+1} = a_{n+1}^{\mathrm{T}} \alpha = a_{n+1}^{\mathrm{T}} \tilde{A}^{-1} \tilde{x} = r^{\mathrm{T}}\, \tilde{x}, \qquad r^{\mathrm{T}} := a_{n+1}^{\mathrm{T}} \tilde{A}^{-1},$$

where a_{n+1}^{T} denotes the $(n+1)$-st row of A. On the other hand, we see by Theorem 4.1 i) that

$$\begin{aligned} (\hat{\lambda}_1, \ldots, \hat{\lambda}_n)\tilde{A} &= -\lambda_{n+1} a_{n+1}^{\mathrm{T}}, \\ (\hat{\lambda}_1, \ldots, \hat{\lambda}_n) &= -\hat{\lambda}_{n+1} a_{n+1}^{\mathrm{T}} \tilde{A}^{-1} = -\hat{\lambda}_{n+1} r^{\mathrm{T}}. \quad (11) \end{aligned}$$

By Theorem 4.1 ii), we have that $\hat{\lambda}_{n+1} \neq 0$.

2. Assume now that $\hat{\lambda}_i \neq 0$ for all $i = 1, \ldots, n$. Then r has only non-zero components. Now minimizing $Q(A\cdot)$ is equivalent to finding the minimizer of

$$\max\left\{Q(\tilde{x}), q(r^{\mathrm{T}}\, \tilde{x})\right\}.$$

Now $\mathrm{lev}_a Q(\tilde{x})$ are the cubes with $(n-1)$–faces parallel to the coordinate planes. On the other hand, the strip

$$\frac{1}{a} \leq r^{\mathrm{T}}\, \tilde{x} \leq a, \qquad a > 1, \qquad (12)$$

with axis $r^{\mathrm{T}}\, \tilde{x} = 1$ is not parallel to a 1–face of the cube, i.e., the unit vectors e_i, $i = 1, \ldots, n$ are not tangents of the hyperplane $r^{\mathrm{T}}\, \tilde{x} = 1$, since this would imply that $r^{\mathrm{T}}\, e_i = 0$, which is not possible since r has no zero component. But then the minimum $\hat{a}$ of $Q(A\cdot)$ is given by the smallest number a such that one of the hyperplanes $r^{\mathrm{T}}\, \tilde{x} = a$ or $r^{\mathrm{T}}\, \tilde{x} = 1/a$ touches a vertex of $\mathrm{lev}_a Q(\tilde{x})$. This condition uniquely determines $\hat{a}$ and the vertex $v_{\hat{a}}$ and we obtain finally the unique minimizer $\hat{\alpha} = \tilde{A}^{-1} v_{\hat{a}}$.

3. Conversely assume that at least one $\hat{\lambda}_i$ is equal to zero. Then r has a zero component and the strip (12) is parallel to some k-face, $k > 0$, of $\mathrm{lev}_a Q(\tilde{x})$ and touches this k-face for the optimal $\hat{a}$. Hence, the solution cannot be unique. □

By $\mathrm{spark}(A)$, we denote the smallest number of rows of A which are linearly dependent. In other words, any $\mathrm{spark}(A) - 1$ rows of A are linearly independent. For the 'spark' notation we also refer to [2].

Examples. 1. We obtain for the matrix

$$A := \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \qquad \mathrm{rg}(A) = 3,\ \mathrm{spark}(A) = 3.$$

The matrix (m,n)–matrix A in (4) is the product of the diagonal matrix $\mathrm{diag}\,(1/b_i)_{i=1}^m$ with positive diagonal entries and a Vandermonde matrix. Hence, it can easily be seen that $\mathrm{spark}(A) = n+1$. If an (m,n)–matrix A has $\mathrm{spark}(A) = n+1$, then A fulfills the Haar condition.

Proposition 4.2 can be reformulated as follows:

COROLLARY 4.2. *Let $A \in \mathbb{R}^{n+1,n}$ such that $\mathcal{R}(A) \cap \mathbb{R}^{n+1}_{>0} \neq \emptyset$ and $1_m \notin \mathcal{R}(A)$. Then $||A \cdot ||_Q$ has a unique minimizer if and only if* $\mathrm{spark}(A) = n+1$.

PROOF. Follow the lines of the proof of Proposition 4.2 and use that all components of r are nonzero if and only if $\mathrm{spark}(A) = n+1$. □

The result can be generalized by the following theorem.

THEOREM 4.3. *Let $A \in \mathbb{R}^{m,n}$ such that $\mathcal{R}(A) \cap \mathbb{R}^m_{>0} \neq \emptyset$. Suppose that* $\mathrm{spark}(A) = n+1$. *Then $||A \cdot ||_Q$ has a unique minimizer which is determined by $n+1$ rows of A, i.e., there exists an index set $J \subset \{1, \ldots, m\}$ of cardinality $|J| = n+1$ such that $||A \cdot ||_Q$ and $||A|_J \cdot ||_Q$ have the same minimum and the same minimizer. Here, $A|_J$ denotes the restriction of A to the rows which are contained in the index set J. We call such an index set J an* extremal set.

PROOF. Let $I := \{i : \hat{\lambda}_i \neq 0\}$, where $\hat{\lambda}_i$ are the Lagrange multipliers from Theorem 4.1. Then, by Theorem 4.1 iv), we have for $i \in I$ that $(A\hat{\alpha})_i$ equals $\hat{a}$ or $1/\hat{a}$. Assume that $|I| \leq n$. By Theorem 4.1 i), this implies $A^{\mathrm{T}}\hat{\lambda} = A|_I^{\mathrm{T}}\hat{\lambda} = 0$, which is not possible because $\mathrm{spark}(A) = n+1$. Thus, $|I| \geq n+1$. But then there exists an index set $J \subset I$ of cardinality $|J| = n+1$ such that $(A|_J\hat{\alpha})_j = \hat{a}$ or $(A|_J\hat{\alpha})_j = 1/\hat{a}$ for all $j \in J$ and whence $\hat{\alpha}$ is the minimizer of $Q(A|_J\,\cdot)$, which is unique by Proposition 4.2. □

Of course the condition $\mathrm{spark}(A) = n+1$ is not necessary for $||A \cdot ||_Q$ to have a unique minimizer as the following example shows.

Example. The matrices

$$A := \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & \frac{1}{2} \\ -4 & 2 \end{pmatrix}, \qquad \text{and} \qquad A := \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -4 & 4 \\ -1 & 1 \end{pmatrix}$$

have both $\mathrm{spark}(A) = 2$. By some following considerations, we obtain for both problems that the minimum of $||A \cdot ||_Q$ is $\hat{q} = 2$. However, in the first problem the minimizer is uniquely determined by $\hat{\tilde{a}} = (\frac{1}{2}, 2)^{\mathrm{T}}$, while the whole line $c(\frac{1}{2}, 1)^{\mathrm{T}} + (1-c)(\frac{3}{2}, 2)^{\mathrm{T}}$, $c \in [0,1]$ minimizes the functional in the second case. For $(\frac{1}{2}, 1)^{\mathrm{T}}$, we have $\mathrm{sign}(\hat{\lambda}_1, \hat{\lambda}_2, \hat{\lambda}_3, \hat{\lambda}_3) = (-1, 0, 1, -1)$, while the pattern is $(0, 1, 1, -1)$ for $(\frac{3}{2}, 2)^{\mathrm{T}}$ and $(0, 0, 1, -1)$ within the line bounded by these points.

By Theorem 4.3, a method for finding the minimizer of $||A \cdot ||_Q$ would be to compute the unique minimizers of the

$\binom{m}{n+1}$ subproblems $||A|_J \cdot ||_Q$ for all index sets J of cardinality $n+1$, and to take the largest minimum $\hat{a}$ and the corresponding $\hat{\tilde{a}}$ as minimizer of the original problem. For our line problem, there exist $\binom{m}{3} = \mathcal{O}(m^3)$ of these subproblems. Below, we give another algorithm, which is also based on Theorem 4.3, but ensures that the value a enlarges for each new choice of the subset J. Since there is only a finite number of such subsets, we must reach a stage where no further increase is possible and J is an extremal set. In normed spaces, such methods are known as *ascent methods*, see [20].

4.2 Approximation Algorithm

Now, we suggest a detailed algorithm (see Fig. 2) for minimizing $||A \cdot ||_Q$, where we restrict our attention to the line problem

$$\max_{i=1,\ldots,m} \max \left\{ \frac{b_i}{\beta + \alpha x_i}, \frac{\beta + \alpha x_i}{b_i} \right\}. \tag{13}$$

i.e., to the matrix A in (4) with $n = 2$.

COROLLARY 4.4. *Let (x_i, b_i), $i = 1, 2, 3$ be given points with pairwise distinct $x_i \in \mathbb{R}$ and positive b_i, $i = 1, 2, 3$. Then the minimum $\hat{q}$ and the minimizer $\hat{\tilde{a}} \in \mathbb{R}^2$ of (13) are given by $\hat{q} = ||\hat{q}_1||_Q$ and*

$$\begin{pmatrix} \hat{\beta} \\ \hat{\alpha} \end{pmatrix} = \frac{1}{x_2 - x_1} \begin{pmatrix} x_2 & -x_1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} b_1\,\hat{q}_1 \\ b_2\,\hat{q}_2 \end{pmatrix},$$

where

$$\hat{q}_1 := \begin{cases} \sqrt{\frac{r_2}{1-r_1}} & \text{if } r_1 < 0 \text{ and } r_2 > 0, \\ \sqrt{\frac{1-r_2}{r_1}} & \text{if } r_1 > 0 \text{ and } r_2 < 0, \\ \sqrt{\frac{1}{r_1+r_2}} & \text{if } r_1 > 0 \text{ and } r_2 > 0, \end{cases} \tag{14}$$

$$\hat{q}_2 := \begin{cases} 1/\hat{q}_1 & \text{if } \frac{r_1}{r_2} < 0, \\ \hat{x}_1 & \text{if } \frac{r_1}{r_2} > 0 \end{cases}$$

and

$$r_1 := \frac{b_1(x_2 - x_3)}{b_3(x_2 - x_1)}, \quad r_2 := \frac{b_2(x_3 - x_1)}{b_3(x_2 - x_1)}.$$

PROOF. Following the proof of Proposition 4.2, we set

$$x_i := \frac{1}{b_i}(\alpha_0 + \alpha_1 p_i), \quad i = 1, 2, 3,$$

and obtain with $\tilde{x} = \tilde{A}\alpha$, where

$$\tilde{A} := \begin{pmatrix} 1/b_1 & p_1/b_1 \\ 1/b_2 & p_2/b_2 \end{pmatrix}, \tilde{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

$$\tilde{A}^{-1} = \frac{1}{p_2 - p_1} \begin{pmatrix} p_2 b_1 & -p_1 b_2 \\ -b_1 & b_2 \end{pmatrix},$$

that

$$x_3 = (1/b_3, p_3)\,\alpha = (1/b_3, p_3)\,\tilde{A}^{-1}\,\tilde{x} = r_1 x_1 + r_2 x_2.$$

It is easy to check that

- $r_1 < 0$, $r_2 > 0$ if $p_1 < p_2 < p_3$ or $p_3 < p_2 < p_1$,
- $r_1 > 0$, $r_2 > 0$ if $p_1 < p_3 < p_2$ or $p_2 < p_3 < p_1$,
- $r_1 > 0$, $r_2 < 0$ if $p_2 < p_1 < p_3$ or $p_3 < p_1 < p_2$.

We restrict our attention to the case $r_1 < 0$ and $r_2 > 0$. The other cases can be handled in a similar way. It remains to minimize $\max\{Q(\tilde{x}), q(r_1 x_1 + r_2 x_2)\}$. The level sets $\mathrm{lev}_a Q(\tilde{x})$ are the cubes with $(n-1)$–faces parallel to the coordinate planes. The set $\{\tilde{x} : q(r_1 x_1 + r_2 x_2) \le a\}$ is given by the intersection of the positive quadrant with the strip

$$-\frac{r_1}{r_2}\,x_1 + \frac{1}{a r_2} \le x_2 \le -\frac{r_1}{r_2}\,x_1 + \frac{a}{r_2}, \qquad a \ge 1 \tag{15}$$

around the axis $x_2 = -\frac{r_1}{r_2}\,x_1 + \frac{1}{r_2}$. Note that $-\frac{r_1}{r_2}$ is positive and that the strip-axis meets the x_2-axis in $\frac{1}{r_2} > 0$.

If $r_1 + r_2 > 1$, then the axis of the strip lies below $(1, 1)$ and the minimum of (13) is characterized by the smallest number $a > 1$ such that the upper strip boundary meets the corner $(a, 1/a)$ of $\mathrm{lev}_a Q(\tilde{x})$, i.e.,

$$\frac{1}{a} = -\frac{r_1}{r_2}\,a + \frac{a}{r_2} \quad \Rightarrow \quad a^2 = \frac{r_2}{1 - r_1} \qquad \text{and}$$

$$\begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} = \tilde{A}^{-1} \begin{pmatrix} a \\ 1/a \end{pmatrix}.$$

If $r_1 + r_2 = 1$, then the axis of the strip contains $(1, 1)$. Thus, $a = 1$ and $\hat{\alpha} = \tilde{A}^{-1}(1, 1)^{\mathrm{T}}$.

If $r_1 + r_2 < 1$, then the axis of the strip lies above $(1, 1)$ and the minimum of (13) is characterized by the smallest number $a > 1$ such that the lower strip boundary meets the corner $(1/a, a)$, i.e.,

$$a = -\frac{r_1}{r_2}\,\frac{1}{a} + \frac{1}{a r_2} \qquad \Rightarrow a^2 = \frac{1 - r_1}{r_2} \qquad \text{and}$$

$$\begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} = \tilde{A}^{-1} \begin{pmatrix} 1/a \\ a \end{pmatrix}.$$

This completes the proof. □

Remark. If the points are ordered, i.e., $x_1 < x_2 < x_3$ (or alternatively in descending order), then either $A\hat{\tilde{a}} = (\hat{q}, 1/\hat{q}, \hat{q})^{\mathrm{T}}$ or $A\hat{\tilde{a}} = (1/\hat{q}, \hat{q}, 1/\hat{q})^{\mathrm{T}}$. This means that $\hat{\lambda}$ in Theorem 4.1 has alternating signs. In other words, the points $f(x_1)$, $f(x_3)$ lie above b_1, b_3 and $f(x_2)$ lies below b_2 or conversely.

Later, we will show that the alternating sign condition is true for general best polynomial approximation with respect to l_q.

PROPOSITION 4.3. *The algorithm shown in Fig. 2 computes the line $f(x) = \hat{\beta} + \hat{\alpha}x$, which minimizes* (13).

PROOF. The algorithm computes in 1 - 2 of each step the minimum $\hat{a}_j$ and the parameter $\hat{x}_1$ of the unique Q-minimizing line determined by the points (p_{i_1}, b_{i_1}), (p_{i_2}, b_{i_2}), (p_j, b_j). By Theorem 4.3, this is also the minimizing line for all points if a in step 3 fulfills $a \le \hat{a}_j$. In the other case, there exists an index k such that $q\,(r_{1,k}\hat{x}_1 + r_{2,k}/\hat{x}_1) > \hat{a}_j$. Now $(\hat{x}_1, 1/\hat{x}_1)$ is the corner where the strip $S_j := \{(x_1, x_2) : \frac{1}{\hat{a}_j} \le r_{1,j}x_1 + r_{2,j}x_2 \le \hat{a}_j\}$ touches the cube $\mathrm{lev}_{\hat{a}_j} Q(x_1, x_2)$. Moreover, the strips S_j and $S_k := \{(x_1, x_2) : \frac{1}{\hat{a}_j} \le r_{1,k}x_1 + r_{2,k}x_2 \le \hat{a}_j\}$ cannot be parallel because $\mathrm{spark}(A) = 3$. Thus, $S_j \cap S_k$ is a parallelogram which has no common point with $\mathrm{lev}_{\hat{a}_j} Q(x_1, x_2)$. This means for $(x_1, x_2) \in S_j \cap S_k$ that either $x_1 \notin [1/\hat{a}_j, \hat{a}_j]$ or $x_2 \notin [1/\hat{a}_j, \hat{a}_j]$ or both. Thus, at least one of the minimizing lines determined by p_{i_1}, p_l, p_k or p_{i_2}, p_l, p_k must have a larger minimizer than $\hat{a}_j$. Hence, the minimum of

Algorithm. (Best line approximation with respect to l_q)
Input: (x_i, b_i), $i = 1, \ldots, m$ of pairwise distinct points $x_i \in \mathbb{R}$ and $b_i > 0$
Set $i_1 := 1$, $i_2 := 2$ and *stopsignal* $:= -1$.

While *stopsignal* $= -1$ do

1. For $i = 1, \ldots, m$; $i \neq i_1, i_2$ compute
$$r_{1,i} := \frac{b_{i_1}(x_{i_2} - x_i)}{b_i(x_{i_2} - x_{i_1})}, \quad r_{2,i} := \frac{b_{i_2}(x_i - x_{i_1})}{b_i(x_{i_2} - x_{i_1})}.$$
2. Compute $\hat{q}_j = \max_i \{||\hat{x}_1(r_{1,i}, r_{2,i})||_Q\}$ by (14). Let $j \neq i_1, i_2$ be an index, where the maximum is attained and $\hat{x}_1 = \hat{x}_1(r_{1,j}, r_{2,j})$.
3. Compute $q := \max_i \{||r_{1,i}\hat{x}_1 + r_{2,i}\hat{x}_2||_Q\}$.
Let k be an index, where the maximum is attained.
4. If $q \leq \hat{q}_j$ then *stopsignal* $= 1$ and $\hat{q} = \hat{q}_j$,
$$\begin{pmatrix} \hat{\beta} \\ \hat{\alpha} \end{pmatrix} = \frac{1}{x_{i_2} - x_{i_1}} \begin{pmatrix} x_{i_2} & -x_{i_1} \\ -1 & 1 \end{pmatrix} \begin{pmatrix} b_{i_1}\hat{q}_1 \\ b_{i_2}/\hat{q}_1 \end{pmatrix},$$
otherwise set $i_1 := j$ and $i_2 := k$, and return to 1.

Figure 2: Algorithm finding best linear approximation under l_q.

the line computed in the next step of the algorithm is strictly larger than the previous one. Now the number of triples of points is finite such that this procedure terminates. □

Remark. Alternatively, one can deal with ordered points $b_1 < b_2 < b_3$, which restricts the effort in (14) to $\hat{q}_1 = \frac{r_2}{1-r_1}$, but requires an ascending ordering of the points x_{i_1}, x_{i_2}, x_j in each step of the algorithm.

Finally, we want to generalize the remark on the signs of the Lagrange multipliers given after Corollary 4.4. Therefore, we need the notion of Chebyshev set.

DEFINITION 4.5. *Let X be a closed interval of R. A set of continous functions $\Phi_1(x), \ldots, \Phi_n(x)$, $\Phi_i : X \to R$, is called a* Chebyshev set *if every non-trivial linear combination of these functions has at most $n-1$ zeros in X.*

The set of polynomials $\Phi_i(x) = x^{i-1}$, $i = 1, \ldots, n$ forms a Chebyshev set. Thus, for polynomials (and all other Chebyshev sets), we have the following theorem, which guarantees that residues have alternating signs.

THEOREM 4.6. *Let $\Phi_i : I \to \mathbb{R}$, $i = 1, \ldots, n$ be a Chebyshev set and let $x_1 < \ldots < x_{n+1}$ be points in I. Then, for*
$$\Phi := (\Phi_j(x_i))_{i,j=1}^{n+1,n},$$
the Lagrange multipliers $\hat{\lambda}_i$, $i = 1, \ldots, n+1$ corresponding to the minimizer of $||\Phi \cdot ||_Q$ have alternating signs.

PROOF. Let $\varphi_i^{\mathrm{T}} = (\phi_1(p_i), \ldots, \phi_n(p_i))$ be the i-th row of Φ. By Theorem 4.1 i) we obtain that
$$\Phi_{n+1}^{\mathrm{T}}(\hat{\lambda}_1, \ldots, \hat{\lambda}_n)^{\mathrm{T}} = -\hat{\lambda}_{n+1}\varphi_{n+1}.$$
Then, it follows by Cramer's rule that
$$\begin{aligned} \hat{\lambda}_i &= \frac{1}{\det \Phi_{n+1}} \det(\varphi_1, \ldots, \varphi_{i-1}, -\hat{\lambda}_{n+1}\varphi_{n+1}, \varphi_{i+1}, \ldots, \varphi_n). \\ &= -\hat{\lambda}_{n+1} \frac{(-1)^{n-i} \det \Phi_i}{\det \Phi_{n+1}}, \end{aligned}$$
where the factor $(-1)^{n-i}$ appears since we need $n-i$ column shifts to move φ_{n+1} to the last position. Finally, we conclude by using the theoreom from [20, p. 55] that $\hat{\lambda}_i = (-1)^{n-i+1}\hat{\lambda}_{n+1}$. □

For our polynomial approximation problem
$$\mathrm{argmin}_{\vec{a} \in \mathbb{R}^n} ||A\vec{a}||_Q$$
with $A \in \mathbb{R}^{n+1,n}$ defined by (4) and ordered points $x_1 < \ldots < x_{n+1}$, we see that $A = \mathrm{diag}(1/b_i)_{i=1}^{n+1}\Phi$, where Φ is the matrix belonging to the Chebyshev set $\Phi_i(x) = x^{i-1}$. Since the b_i are positive, we obtain immediately that the Lagrange multipliers $\hat{\lambda}_i$ have alternating signs. Again, this means that $\hat{f}(x_i) - b_i$ has alternating signs.

One can also ask for an exponential function
$$\hat{f} = e^{\sum_{j=1}^n \alpha_j \phi_j},$$
which best fits a set of given points (p_i, b_i), $i = 1, \ldots, m$ with pairwise distinct $p_i \in \mathbb{R}^d$ and $b_i > 0$, $i = 1, \ldots, m$ in the sense of (1). Note that $\hat{f} > 0$ by definition. Since the ln function increases strictly monotonically, this is equivalent to minimizing
$$\begin{aligned} &\ln\left(\max_{i=1,\ldots,m} \max\left\{\frac{b_i}{\hat{f}(p_i)}, \frac{\hat{f}(p_i)}{b_i}\right\}\right) \\ = &\max_{i=1,\ldots,m} \max\{\ln b_i - \ln \hat{f}(p_i), \ln \hat{f}(p_i) - \ln b_i\} \\ = &\max_{i=1,\ldots,m} |\ln b_i - \sum_{j=1}^n \alpha_j \phi_j(p_i)| \\ = &\|(\ln b_i)_{i=1}^m - \Phi\,\alpha\|_\infty. \end{aligned}$$
Thus, it remains to find the best function $\sum_{j=1}^n \alpha_j \phi_j(p_i)$ with respect to the ℓ_∞ norm. This problem was treated in various papers, see [20], and can be solved, e.g., by a linear program
$$\min_{(\alpha,a) \in \mathbb{R}^n \times \mathbb{R}} a \text{ subject to } -a \leq \Phi\,\alpha - (\ln b_i)_{i=1}^m \leq a \wedge a \geq 0.$$
In our examples in Section 5, we are using $\hat{f}(x) = e^{\alpha_0 + \alpha_1 x}$.

4.3 Intuition behind the Algorithm

As the formal derivation of the algorithm in the previous sections is somewhat hard to follow for those not versed in approximation theory, we present the intuition behind the algorithm. Note that this necessarily leaves out the mathematical details and some arguments are hand-waving.

If we have only one or two data points, the approximation problem is trivial, as we can fit a linear function through one or two data points. For three data points, we can find the best approximation analytically by solving a system of equations (Corollary 4.4). For more than three data points, we can solve the problem iteratively as follows. First, we pick three arbitrary data points and solve the system of equations. This gives us a current linear approximation. Clearly, its q-error for the three points is a lower bound on the overall q-error for all data points. We now steadily increase the

q-error of the approximation by exchanging one of the three data points by another one whose deviation from the current linear approximation is maximal. We take the one with the maximal deviation to accellerate convergence.

The real algorithm in Figure 2 is a bit more complex, but the intuition is the same as with the simple algorithm sketched above: We pick two data points i_1 and i_2, examine all other data points j as a possible third point in step 2, and then find the maximum derviation k from the optimal approximation of these points in step 3, choosing them as new base points. We repeat this process until the error no longer increases (step 4).

5. EXAMPLES

We now give examples how the choice of approximation affects the q-error.

5.1 Exact Match Queries

We consider a relation R containing author information and having an attribute A which contains the number of citations of papers of an author. The data are taken from a 2006 citeseer instance of the 10.000 most cited authors. Here, we restrict our attention to those authors who have between 256 and 512 citations. This is an arbitrary choice, which has the advantage that it is small enough to be shown in a figure. From R, we calculate the set of points

$$\{(x_i, f_i) \mid x_i \in \Pi_A(R), f_i = |\sigma_{A=x_i}(R)|\}$$

which we then approximate by

- **Avg** the average of the frequencies f_i, as done in histogram buckets [14],
- **Qmiddle** the q-middle of the frequencies f_i,
- **LinL2** a linear function $\beta + \alpha x$ minimizing the l_2 error, as proposed in [9],
- **LinQ** a linear function $\beta + \alpha x$ minimizing the q-error, and
- **ExpQ** an exponential function $e^{\beta+\alpha x}$ minimizing the q-error.

Fig. 3 contains the original points plotted with impulses and three approximations. The q-errors and their powers by four of the approximations are given in the following table.

approximation	q-error	q-error^4
Avg	3.51	151
Qmiddle	2.77	59
LinL2	2.92	72
LinQ	2.16	22
ExpQ	2.11	20

Using the average, as done to approximate the frequencies in a histogram's bucket, generates very large errors. Replacing it by the q-middle already reduces the maximum q-error by 0.8. Further, note the 4th power of the q-error, which is the error bound for the plan costs if the plan generator errs due to size estimation errors, which with a higher q-error becomes much more likely (see Sec.3). The difference between optimizing under l_2, an approach suggested in [9], and l_q is about 0.8, which is also quite notable if we consider the results of Sec. 3. Also, adding one more number, which adds a storage overhead of less than 30%, by using ExpQ instead of the q-middle reduces the q-error by 0.66. This also reduces the factor the costs of the produced plan are possibly higher than the costs of the best plan from 59 to 20. The best approximations by a polynomial of degree 3/4/5 have q-errors of 1.97/1.96/1.94, resp.

5.2 Range Queries

Let us first take a closer look at the errors. Since the q-error does not indicate over- or underestimation, we introduce the p-error, which is nicer to plot:

$$p(f_i, \hat{f}_i) = \begin{cases} (f_i/\hat{f}_i) - 1 & \hat{f}_i \leq f_i \\ -(\hat{f}_i/f_i) + 1 & \hat{f}_i > f_i \end{cases}$$

for a true value f_i and its estimate $\hat{f}_i$. Adding/subtracting 1 eliminates otherwise wasted space. It should be clear that minimizing l_q is equivalent to minimizing under the absolute value of p.

LinQ and ExpQ can not only be used to estimate the size of $\sigma_{A=c}(R)$, but also for range queries of the form $\sigma_{a\leq A\leq b}$. The correct size estimate for a range query can be calculated as

$$f(a, b) = \sum_{a\leq x_i\leq b} f_i$$

Replacing f_i by $\hat{f}(x_i)$ results in estimates $\hat{f}(a, b)$ for $f(a, b)$. This approach works well for LinL2. However, as ExpQ underestimates in most cases (see Fig. 3), we have to adjust it in order to guarantee an average error of zero. In order to do so, define for a given window size w $W(w) = \{x_i | x \geq \min, x_i + w \leq \max\}$, where min and max are the minimum and maximum of all x_i. Then, we can define the adjustment function

$$g(w) = 1/k_w \sum_{x_i\in W(w)} f_i/\hat{f}(x_i)$$

where $k_w = |W(w)|$. Then, the average error of

$$\sum_{a\leq x_i\leq b} \hat{f}(x_i) * g(b - a)$$

becomes zero. Of course, we cannot keep g since it is as big as the original data. Instead, we approximate $(w, g(w))$ by a linear function. Let $\hat{g}$ be the best approximation of $(w, g(w))$ under l_∞. Then, for a given approximation $\hat{f}(x)$ of the original data, we define the *adjusted* approximation

$$\hat{f}(a, b) = \sum_{a\leq x_i\leq b} \hat{f}(x_i) * \hat{g}(b - a).$$

The following plots show for a given window size w (x-axis) the minimum and maximum p-error (y-axis) taken over all elements in $W(w)$. Let us call such a representation *p-cone*. Fig. 4 shows the p-cone for Avg, LinL2, and ExpQ. We observe that for small to medium window sizes, there is a significant improvement of the q-error if the adjusted ExpQ approximation is used instead of the more common average or l_2-based LinL2 approximation [9]. Note that the LinL2 cone looks 'smaller' for large window sizes because we show the (signed) p-error and LinL2 always underestimates large windows.

5.3 Piecewise Approximation

For selectivity estimation purposes, we usually want to construct the best synopsis with a given space budget. We can construct a synopsis of arbitrary size by using piecewise approximation, i.e., partitioning the data into a suitable number of buckets and then approximating each bucket individually. The main problem is finding the right bucket boundaries. However, we can find them easily by using the characteristics of the q-error: We know that by increasing a

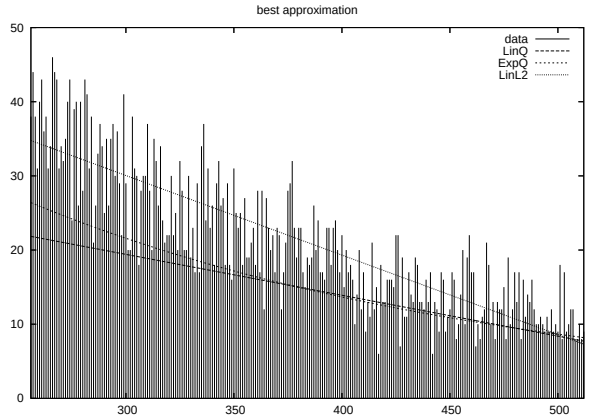

Figure 3: Original data and approximations

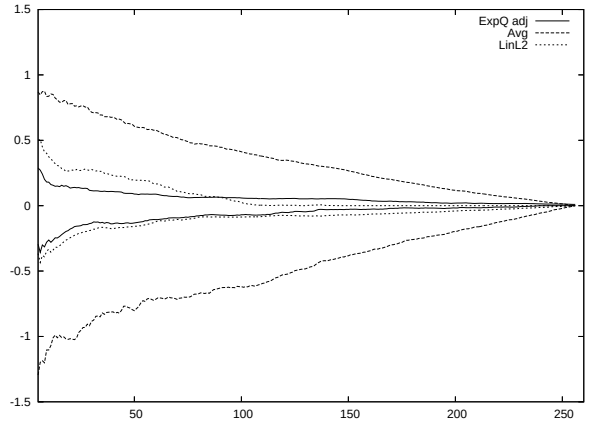

Figure 4: P-cone for range queries

bucket (i.e., adding more data points), the maximum q-error will increase montonically. Therefore, if we fix the desired maximum error, we can greedily build bucket boundaries using binary search such that we get maximal bucket sizes within the given error constraint. By using binary search over the error constraint, we can find the minimum error that results in the desired number of buckets.

The result of such a piecewise approximation with a given space budget is shown in Fig. 5. It shows the approximation of the CDF of a large data set with continues values (TF*IDF*PageRank scores from the TREC-12 Web Track benchmark (http://trec.nist.gov/) of all documents containing the word *public*) using 320 bytes of space. We constructed the *LinQ* approximation as described above, the *LinL2* approximation using the DP algorithm from [9], and the *Avg* approximation by building an equi-depth histogram. The LinQ approximatioin is very accurate over the whole domain (note the logarithmic scale), while the other approaches perform much poorer and have some data points with very large estimation errors.

We included a study of the maximum q-error for different histogram types using the same data set in Fig. 6. The *piecewise LinQ* approximation was discussed above, *V-Optimal* were proposed in [8], the *Wavelet* histograms we used were introduced in [11], the *Sampling* technique is described in [17], and the *Koenig* histograms (effectively piecewise LinL2) were introduced in [9]. The piecewise LinQ approximations have maximum q-errors that are orders of magnitude better than these of the other approaches, which effectively means

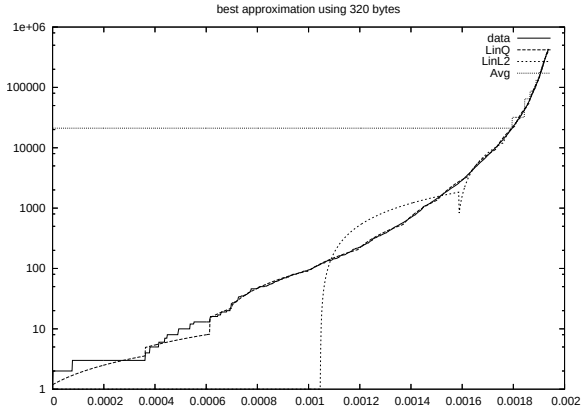

Figure 5: Original data and three piecewise approximations with a space budget of 320 bytes

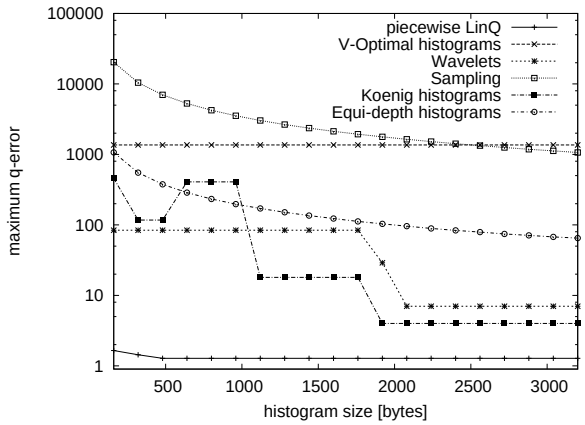

Figure 6: Approximation errors for different histogram types

that they can result in size estimations that are easily orders of magnitude off.

6. RELATED WORK

Selectivity estimation is an important and well studied field. A comprehensive overview of the field is given in [6]. Most of these techniques are based on histograms. They suffer from two severe problems: they (1) make no error guarantees and (2) give no hint at the resulting plan quality. More precisely, the influence of the errors occurring on plan optimality and costs has not been studied. One of the first papers to study histograms with error guarantees is [8]. However, it constructs classical bucket histograms, which are often not as accurate as histograms based upon function approximation and, worse, it concentrates on l_2 errors. A more accurate approach is proposed in [9], where the authors use piecewise linear functions as histograms and use least squares fitting to minimize, again, l_2 errors. Minimizing l_2 is very popular, also among other approaches like wavelet-based ones [11]. More recent work has studied relative and absolute errors as error metrics (e.g., [3, 13]). These errors can be connected more directly to the quality of the resulting execution plans. Still, these techniques give no guarantees for execution plan quality, and we are not aware of any other work that gives such guarantees.

7. CONCLUSION

For size estimations, the q-error is a much more meaningful error metric than other metrics like relative, l_2 or l_∞ errors. The reason is that there is a direct connection between the q-error and plan optimality and plan costs: Bounding the q-error of size estimations suitably guarantees plan optimality, and if these bounds cannot be met, the produced plan cannot have costs higher than q^4 times the costs of the optimal plan, where q is the q-error of the size estimations. Both results would have been impossible without using the q-error. While this motivates its use, there have not existed any means of minimizing it. Thus, we developed algorithms for constructing the best linear approximation under l_q and the best piecewise approximation.

With this paper, we laid the foundations of size estimations minimizing l_q. Many areas remain for future work: size estimations minizing the q-error for joins and projections, updates, and query feedback. More challenging will be the development of algorithms producing the best approximation under l_q in the multi-dimensional case. This will be very useful to deal with correlations, which is the next challenge we will undertake.

8. REFERENCES

[1] J. F. Bonnans and A. Shapiro. *Pertubation Analysis of Optimization Problems*. Springer, 2000.

[2] D. Donoho and M. Elad. Optimally sparse representation in general (non-orthogonal) dictionaries via ℓ_1 minimization. *Proc. of the National Academy of Sciences*, 100(5), 2003.

[3] M. N. Garofalakis and A. Kumar. Wavelet synopses for general error metrics. *ACM Trans. Database Syst.*, 30(4), 2005.

[4] L. Haas, M. Carey, M. Livny, and A. Shukla. Seeking the truth about ad hoc join costs. *VLDB Journal*, 6(3), 1997.

[5] T. Ibaraki and T. Kameda. Optimal nesting for computing n-relational joins. *ACM Trans. Database Syst.*, 9(3), 1984.

[6] Y. E. Ioannidis. The history of histograms (abridged). In *VLDB*, 2003.

[7] Y. E. Ioannidis and S. Christodoulakis. On the propagation of errors in the size of join results. In *SIGMOD*, 1991.

[8] H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. C. Sevcik, and T. Suel. Optimal histograms with quality guarantees. In *VLDB*, 1998.

[9] A. C. König and G. Weikum. Combining histograms and parametric curve fitting for feedback-driven query result-size estimation. In *VLDB*, 1999.

[10] R. Krishnamurthy, H. Boral, and C. Zaniolo. Optimization of nonrecursive queries. In *VLDB*, 1986.

[11] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *SIGMOD*, 1998.

[12] G. Moerkotte. Best approximation under a convex paranorm. Technical Report MA-08-07, University of Mannheim, 2008.

[13] T. Neumann and S. Michel. Smooth interpolating histograms with error guarantees. In *BNCOD*, 2008.

[14] V. Poosola, Y. Ioannidis, P. Haas, and E. Shekita. Improved histograms for selectivity estimates of range predicates. In *SIGMOD*, 1996.

[15] N. Reddy and J. R. Haritsa. Analyzing plan diagrams of database query optimizers. In *VLDB*, 2005.

[16] R. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[17] D. W. Scott. *Multivariate Density Estimation: Theory, practice, and visualization*. Wiley, 1992.

[18] P. Spellucci. *Numerische Verfahren der Nichtlinearen Optimierung*. Birkhäuser, 1993.

[19] J. Ullman. *Database and Knowledge Base Systems*, volume Volume 1. Computer Science Press, 1989.

[20] G. Watson. *Approximation Theory and Numerical Methods*. Addison-Wesley, 1980.

APPENDIX

A. ASI PROPERTY

The context in which the ASI property is defined can be sketched as follows [5, 10]. Only queries whose query graph is a tree are considered. By picking an arbitrary relation and pointing away the edges from this relation, we derive a *precedence graph*. For every relation, there exists exactly one precedence graph. Only left-deep plans are considered. That is, the right argument of every join operator in a plan must be a base relation. Every left-deep plan corresponds uniquely to a sequence of relations and vice versa. The ASI property is defined as follows.

DEFINITION A.1. *Let A and B be two sequences and V and U two non-empty sequences. We say that a cost function C has the* adjacent sequence interchange property *(ASI property) if and only if there exists a function T and a rank function defined for sequences S as*

$$rank(S) = \frac{T(S) - 1}{C(S)}$$

such that for non-empty sequences $S = AUVB$ the following holds:

$$C(AUVB) \leq C(AVUB) \iff rank(U) \leq rank(V)$$

if $AUVB$ and $AVUB$ satisfy the precedence constraints imposed by a given precedence graph.

B. COST OF SORT MERGE JOIN

We briefly present the cost function for the sort merge join as developed by Haas et al. [4]. Thereby, we use their notation. Due to space reasons, we cannot give a full explanation. The main point here is to show that the cost function is a polynomial of degree two with positive coefficients. Let R_1 and R_2 be two relations to be joined and s_1 and s_2 are their sizes.

Haas et al. introduce the following abbreviations

$$\begin{aligned} \text{WS} &= M - I - O \\ \text{RL} &= (2\text{WS})/F = 2(M - I - O)/F \\ \text{NR}_i &= s_i/\text{RL} \\ \text{MPR} &= M/(\text{NR}_1 + \text{NR}_2) = M\text{RL}/(s_1 + s_2) \end{aligned}$$

where M (buffer size), I (input buffer size), O (output buffer size) are constants derived from the memory allocation scheme. F (fudge factor) is a constant. WS stands

for working set, RL for run length, NR for number of runs, MPR for memory per run in the merge phase. The overall I/O costs are calculated by

$$N_s T_s + N_{IO} T_{\mathrm{IO}} + N_x T_x$$

where T_s denotes seek time, T_{IO} rotational delay, T_x transfer time, N_s the number of seeks, N_{IO} the number of I/O operations, and N_x the number of pages to be transferred. In the following, we first give the original formulation by Haas et al. and then in the next line the transformed version:

$$\begin{aligned}
N_x &= 3(s_1 + s_2) \\
N_s &= 4 + s_1/MPR + s_2/\mathrm{MPR} \\
&= 4 + (M\mathrm{RL})(s_1 + s_2)^2 \\
N_{IO} &= s_1/I + s_1/O + s_2/I + s_2/O + s_1/\mathrm{MPR} + s_2/\mathrm{MPR} \\
&= (I+O)/(IO)(s_1 + s_2) + (M\mathrm{RL})^{-1}(s_1 + s_2)^2
\end{aligned}$$

From the transformed part we can now see that the C is a polynomial of degree two in variables s_1 and s_2 with positive coefficients only.