UMIACS-TR-93-138 December 1993
CS-TR-3197

Adaptive Selectivity Estimation Using Query Feedback*

Chungmin Melvin Chen Nick Roussopoulos

Institute for Advanced Computer Studies and
Department of Computer Science
University of Maryland, College Park
E-mail: min@cs.umd.edu, nick@cs.umd.edu

Abstract

In this paper, we propose a novel approach for estimating the record selectivities of database queries. The real attribute value distribution is adaptively approximated by a curve-fitting function using a query feedback mechanism. This approach has the advantages of requiring no extra database access overhead for gathering statistics and of being able to continuously adapt the value distribution through queries and updates. Experimental results show that the estimation accuracy of this approach is comparable to traditional methods based on statistics gathering.

1 Introduction

In most database systems, the task of query optimization is to choose an efficient execution plan. Best plan selection requires accurate estimates of the *costs* of alternative plans. One of the most important factors that affects plan cost is *selectivity*, which is the number of tuples satisfying a given predicate. Therefore, in most cases, the accuracy of selectivity estimates directly affects the choice of best plan.

A study on error propagation [IC91] revealed that selectivity estimation errors can increase exponentially with the number of joins and thus affect the decisions in query optimization. Accurate selectivity estimation has become even more important in today's systems of much larger database sizes, possibly distributed over a LAN or a WAN. In such systems, the query plans are expected to diverge much more in cost due to the database size and the volume of data transmission. Therefore, accurate selectivity estimation is even more crucial.

*This research was sponsored partially by the National Science Foundation under grants IRI-9057573 and GDR-85-00108, and by NASA/USRA under contract 5555-09. This paper also appears in *Proceedings of the 1994 ACM-SIGMOD International Conference on Management of Data, 1994.*

The issue of selectivity estimation has attracted popular interest, and different methods have been proposed [MO79, Chr83b, Chr83a, PSC84, KK85, HOT88, Lyn88, MD88, LN90, SLRD93, Ioa93]. They can be categorized into four classes: the *non-parametric method*, the *parametric method*, *sampling*, and *curve fitting*. In the following paragraphs, we review the essential approaches for each of these four classes. A detailed survey of the first two classes can be found in [MCS88].

Non-Parametric Method Methods in this class maintain attribute value distributions using ad hoc data structures and algorithms. The most common method is the *histogram*, which divides an attribute domain into intervals and counts the number of tuples holding values which fall into each of the intervals. Variations of the histogram method can be found in [MO79, PSC84, MD88, Lyn88, Ioa93]. The histogram is simple, but tradeoff between the computation/storage overhead and the estimation accuracy must be considered. Satisfactory accuracy will not be reached until the domain is divided into a sufficient large number of small intervals. In addition to the histogram, a pattern recognition technique was used by [KK85] to construct discrete cells of distribution table, and [Lyn88] used a keyterm-oriented approach to keep counts of the most frequently queried attribute values.

Parametric Method Parametric methods approximate the actual distribution with a mathematical distribution function of a certain number of free statistical parameter(s) to be estimated (we call such a function a *model* function). Examples of the model function include the uniform, normal, Pearson family and Zipf distributions. In these methods, statistics must be collected, either by scanning through or by sampling from the relation, in order to estimate the free parameter(s). These methods usually require less storage overhead and provide more accurate estimation than non-parametric methods (if the model function fits the actual distribution). The disadvantage of this method is that the "shape" of the actual distribution must be known a priori in order to choose a suitable model function. Moreover, when the actual distribution is not shaped like any of the known model functions, any attempt to approximate the distribution by this method will be in vain. Contributions to research of parametric methods can be found in [S^+79, SB83, Fed84, Chr83b, Chr83a].

Curve Fitting In order to overcome the inflexibility of the parametric method, [LST83] and [SLRD93] used a general polynomial function and applied the criterion of least-square-error to approximate attribute value distribution. First, the relation is exhaustively scanned, and the number of occurrences of each attribute value is counted. These numbers are then used to compute the coefficients of the optimal polynomial that minimizes the sum of the squares of the estimation errors over all distinct attribute values. Polynomial approximation has been widely used in data analysis; however, care must be taken here to avoid the problem of oscillation (which may lead to negative values) and rounding error[1] (which may propagate and result in poor estimation when the degree of the

[1]The problem caused by rounding errors is usually termed a case of being *ill-conditioned*. This can always be avoided by representing the approximating polynomial with a more *numerically stable* basis. For example,

polynomial is high, say, more than 10).

Sampling The sampling method has recently been investigated for estimating the resulting sizes of queries. Sample tuples are taken from the relations, and queries are performed against these samples to collect the statistics. Sufficient samples must be examined before desired accuracy can be achieved. Variations of this method have been proposed in [HOT88, LN90, HS92]. Though the sampling method usually gives more accurate estimation than all other methods (suppose suffucient samples are taken), it is primarily used in answering statistical queries (such as `COUNT(...)`). In the context of query optimization where selectivity estimation is much more frequent, the cost of the sampling method is prohibitive and has essentially prevented its practical use.

Although accuracy is very important for selectivity estimates, the cost of obtaining such estimates must be confined if they are to be cost effective. In all the above methods, however, extra I/O accesses to the database are required for the very purpose of collecting statistics. This procedure might be expensive and, as suggested, should be done off-line or when the system is light-loaded. In a static database where updates are rare, this overhead is acceptable. However, in the presence of updates, the procedure must be re-run either periodically or whenever the updates exceed a given threshold. This process not only incurs more overhead, but also degrades the query optimizer before the out-dated statistics are refreshed.

In the following, we present a novel approach which approximates the attribute value distribution using query feedbacks and totally avoids the overhead of statistics collection. The idea is to use subsequent *query feedbacks* to "regress" the distribution gradually, in the hope that as queries proceed, the approximation will become more and more accurate. We say that the adaptive approximation "learns" from the query executions in the sense that it not only "remembers" and "recalls" the selectivities of repeating query predicates, but can also "infer" (predict) the selectivities of new query predicates. This approach is advantageous in the following respects:

- Efficiency — Unlike the previous methods, no off-line database scans or on-line sampling are needed to form the value distribution. Also, unlike all the other methods (except sampling [Wil91]), where the statistics collection and computation overhead is proportional to the relation size, the overhead of our method has a negligible cost in constant time for each query feedback, regardless of the relation size.

- Adaptation — The technique we use here adapts the approximating value distribution to queries and updates. None of the previous methods achieve this. They neither take into account query information when approximating the value distribution (only relations are scanned), nor continuously adjust the distribution to updates (re-computation is invoked only after the updates exceed a threshold).

The rest of this paper is organized as follows: Section 2 describes the adaptive selectivity estimator in detail. Section 3 presents some of our experimental results. Finally, conclusions are given in Section 4.

the Legendre polynomials are used as the basis in [LST83].

2 Adaptive Selectivity Estimation

In this section, we describe the implementation of an Adaptive Selectivity Estimator (ASE). At the heart of our approach is a technique called *recursive least-square-error* (RLSE), which is adopted to adjust the approximating distribution according to subsequent feedbacks. Before exploring the details, we first define some notations used throughout this paper.

Let A be an attribute of relation R, and let range $D = [d_{min}, d_{max}]$ be the *domain* of A. In this study, we consider only numerical domains (either discrete or continuous).[2] Let D' be the collection of all sub-ranges of D, and define $f_A : D' \rightarrow N$ as the actual distribution of A, i.e., for each sub-range $d \subseteq D$, $f_A(d) = |\{t \in R \ : \ t.A \in d\}|$ is the number of tuples in R whose values of attribute A belong to range d. Notice that the above notation is well-defined for both discrete and continuous cases. We denote a *selection query* $\sigma_{l \leq R.A \leq h}(R)$, where $l \leq h$, as $q = (l, h)$. The *selectivity* of query q, defined as $s = f_A([l, h])$, is the number of tuples in the query result. The *query feedback* from query q is then defined as $\zeta = (l, h, s)$.

2.1 Customizing RLSE for Query Feedback

The goal of our approach is to approximate f_A by an easily evaluated function f which is able to self-adjust from subsequent query feedbacks. Thus, given a sequence of queries $q_1, q_2, \ldots$, we can view f as a sequence $f_0, f_1, f_2, \ldots$ where f_{i-1} is used to estimate the selectivity of q_i, and, after q_i is optimized and executed, f_{i-1} is further adjusted into f_i using feedback ζ_i (which contains the actual selectivity, s_i, of query q_i obtained after the execution).

We use a general form $f(x) = \sum_{i=0}^{n} a_i \phi_i(x)$ as the underlying approximating function, where $\phi_i(x)$, $i = 0, \ldots, n$, are $n+1$ pre-chosen functions (called *model functions*), and a_i are coefficients to be adjusted from the query feedbacks. The corresponding *cumulative* distribution of $f(x)$ is given as $F(x) = \sum_{i=0}^{n} a_i \Phi_i(x)$, where $\Phi_i(x)$ is the indefinite integral of $\phi_i(x)$. Using this form of approximation, the estimated selectivity of query $q = (l, h)$, denoted by $\hat{s}$, is computed as:

$$\hat{s} = \int_{l}^{h+1} f(x)dx = F(h+1) - F(l) = \sum_{j=0}^{n} a_j [\Phi_j(h+1) - \Phi_j(l)].$$

Now suppose a sequence of query feedbacks $\zeta_1, \ldots, \zeta_m$, where $m \geq n$, have been collected. A reasonable criterion for tuning $f(x)$ is to find the optimal coefficients a_i that minimize the sum of the squares of the estimation errors (thus referred to as *least-square-error* (LSE)):

$$\sum_{i=1}^{m} (\hat{s}_i - s_i)^2 = \sum_{i=1}^{m} (\sum_{j=0}^{n} a_j [\Phi_j(h_i + 1) - \Phi_j(l_i)] - s_i)^2. \tag{1}$$

The above problem can be reformulated in linear algebra form as:

$$\textit{find the optimal } A \textit{ that minimize } ||X * A - Y||^2, \tag{2}$$

[2]Non-numerical domains can be mapped into numerical ones using certain mapping techniques. The mapping functions should be provided by the database creators who know the semantic meaning of the attributes.

where $||\cdot||^2$ denotes the sum of the squares of all elements in the vector, and

$$X = \begin{bmatrix} \Phi_0(h_1+1)-\Phi_0(l_1) & \dots & \Phi_n(h_1+1)-\Phi_n(l_1) \\ \Phi_0(h_2+1)-\Phi_0(l_2) & \dots & \Phi_n(h_2+1)-\Phi_n(l_2) \\ \dots & \dots & \dots \\ \Phi_0(h_m+1)-\Phi_0(l_m) & \dots & \Phi_n(h_m+1)-\Phi_n(l_m) \end{bmatrix}, Y = \begin{bmatrix} s_1 \\ s_2 \\ \dots \\ s_m \end{bmatrix}, A = \begin{bmatrix} a_0 \\ a_1 \\ \dots \\ a_n \end{bmatrix}. \quad (3)$$

Let X^t be the transpose of X, the solution to Eq. 2 is obtained as

$$A^* = (X^tX)^{-1}X^tY. \quad (4)$$

The above computation has the drawback that the space requirement of X and Y increases in proportional to the number of query feedbacks m, and each time a new query feedback is added, the whole thing must be re-computed. This concern can be relieved with some rearrangement of the above computation. Let $P = X^tX$ and $N = X^tY$. It is not hard to see that P is a $n \times n$ matrix and N is a $n \times 1$ vector—both of whose dimensions are independent from the number of feedbacks m. A more careful look into P and N shows that

$$P = X^tX = \sum_{i=1}^{m} X_i^tX_i, \quad N = \sum_{i=1}^{m} X_i^ts_i, \quad (5)$$

where X_i is the ith row of X, and X_i^t its transpose. Now, let $\zeta_1, \zeta_2, \dots, \zeta_i, \dots$ be a sequence of query feedbacks, and A_i^* be the optimal coefficients of $f(x)$ corresponding to the first i feedbacks. According to Eq. 4 and 5 we have

$$A_i^* = P_i^{-1}N_i, \text{ for } i = n+1, n+2, \dots \text{ where} \quad (6)$$

$$P_i = P_{i-1} + X_i^tX_i, \quad N_i = N_{i-1} + X_i^ts_i, \text{ for } i = 1, 2, \dots, \quad (7)$$

with initial condition $P_0 = N_0 = 0$. Note that for $i \leq n$, P_i^{-1} dose not exist and thus a default distribution (e.g., uniformity) must be used temporarily. Later in this context, we will relax this restriction. Also notice that by using Eqs. 6 and 7, only two constant size arrays, P and N, need to be maintained.

The above equations can be further transformed into another form where the expensive matrix inversion P_i^{-1} need not be explicitly computed. [You84] derived the following recursive formulas, referred to as *Recursive Least-Square-Error* (RLSE), from Eqs. 6 and 7 :

$$\begin{aligned} A_i^* &= A_{i-1}^* - G_iX_i^t(X_iA_{i-1}^* - s_i), & (8) \\ G_i &= G_{i-1} - G_{i-1}X_i^t(1 + X_iG_{i-1}X_i^t)^{-1}X_iG_{i-1}, & (9) \end{aligned}$$

for $i = 1, 2, \dots$, while A_0 and G_0 can be of any arbitrary values. In this expression, no explicit matrix inverse operation is needed, and only an $n \times n$ matrix G (called a *gain matrix*) needs to be maintained (actually, $G = P^{-1}$). The computation complexity is in the order of $O(n^2)$. Since n is a pre-chosen small integer, the computation overhead per query feedback is small and is considered constant, regardless of the relation size. The initial values G_0 and A_0 may affect the convergence rate of A_i^* and, thus, the rate at which f_i converges to f_A. We describe later in this section how to initialize G_0 and A_0 with appropriate values. It is interesting to see that the computation of A_i^* resembles the technique of *stochastic approximation* [AG67], in the sense that A_i^* is adjusted from A_{i-1}^* by subtracting a *correction term* which is the product of the estimation error $(X_i A_{i-1}^* - s_i)$ and the *gain* value $G_i X_i^t$. Because of their simplicity and efficiency in both space requirement and computation, Eqs. 8 and 9 were adopted in the ASE.

Accommodating Update Adaptiveness (with Weighted LSE)

The RLSE can be further generalized to accommodate adaptability to updates. We accomplish this by associating different *weights* with the query feedbacks so that the outdated feedbacks can be suppressed by assigning smaller weights to them. In Eq. 1, we now associate an *importance weight* β_i to the estimation error of the ith query, and a *fading weight* α_i to the estimation errors of all the preceding queries. That is, instead of minimizing Eq. 1, we now want to minimize:

$$\sum_{i=1}^{m}[(\prod_{j=i+1}^{m} \alpha_j) \cdot \beta_i \cdot (\hat{s}_i - s_i)]^2. \tag{10}$$

The recursive solution to the above is similar to Eqs. 8 and 9 (see Appendix A for derivation detail):

$$\begin{aligned} A_i^* &= A_{i-1}^* - \beta_i^2 G_i X_i^t (X_i A_{i-1}^* - s_i), & (11) \\ G_i &= (\frac{1}{\alpha_i})^2 G_{i-1} - (\frac{\beta_i}{\alpha_i})^2 G_{i-1} X_i^t (\alpha_i^2 + \beta_i^2 X_i G_{i-1} X_i^t)^{-1} X_i G_{i-1}, & (12) \end{aligned}$$

for $i = 1, 2, \ldots$ Intuitively, β_is determine the "importance" of individual feedbacks; α_is determine the "forgetting" rate of previous feedbacks. Note that Eqs. 8 and 9 offer a special case of Eqs. 11 and 12 with $\alpha_i = \beta_i = 1$, for all i. Apparently, different weights affect the adaptation behavior of the approximating function. As an innovation, we consider only fixed-value weights. We set $\beta_i = \alpha_i = 1$ for all i, except that α_i is assigned another positive number less than 1 if ζ_i is the first feedback after update. The smaller the α_i, the more the knowledge from previous feedbacks is to be forgotten. Note that we cannot set $\alpha_i = 0$, because it appears as a denominator in Eq. 12. Nonetheless, the same effect (of discarding all previous knowledge) can be achieved by assigning an extremely small number to α_i. Experiments with different values of α_i are given in the next section.

Initializing A_0 and G_0

The initial values of G_0 and A_0 must be determined before the recursive formulas in Eqs. 11 and 12 can be used. Theoretically, arbitrary initial values can be used for G_0 and A_0 [You84], though they differ greatly in convergence rates. To speed up convergence, we compute $G_0(= P_0^{-1})$ and A_0^* using Eqs. 4 and 5 by substituting the following $(n+1)$ *manual* feedbacks into Eq. 3:

$$l_i = h_i = d_{min} + (i-1) * \frac{(d_{max} - d_{min})}{(n-1)}, \quad s_i = \frac{|R|}{(d_{max} - d_{min})}, \quad i = 1 \ldots n \tag{13}$$

$$l_{n+1} = d_{min}, \; h_{n+1} = d_{max}, \qquad s_{n+1} = |R|, \tag{14}$$

where $|R|$ denotes the number of tuples in relation R. The intention here is to force ASE to begin with a uniform distribution (enforced by Eq. 13), and to keep knowledge of the relation cardinality in the gain matrix (enforced by Eq. 14).

Choosing the Model Functions

The remaining problem now is to choose the model functions $\phi_i(x)$. The polynomial function is a good candidate due to its generality and simplicity and has been used in [LST83] and [SLRD93]. We adopted polynomials of degree 6 throughout our experiments, i.e., the approximating function is of the form $f(x) = \sum_{i=0}^{6} a_i x^i$. Whereas polynomials of higher degrees have the pontential problem of being ill-conditioned, polynomials of lower degrees might not be flexible enough to fit the variety of actual distributions. Therefore, our choice of degree 6 is a compromise between these concerns [3]. Another interesting class of functions is the *spline* functions [dB78], which are piecewise polynomial functions. Splines have many advantages over polynomials in the aspects of adaptability and numerical stability. However, they are more complex in computation and particularly in representation. We are currently investigating this approach and will not discuss it here.

A practical problem of polynomials is the negative values which are undesired in distribution approximations. This poses no problem so long as the negative values occur only outside the attribute domain, or so long as the resulting estimated selectivity of the query of interest is still positive (even if some negative values do occur within the domain). If a negative selectivity is ever estimated for a query, we simply use zero instead (and note that if the error is large, it will be tuned through feedback). Finally, we summarize ASE in the following description.

Variables

f: a polynomial of degree 6; F: the indefinite integral of f;

A: the (adaptable) coefficients of f; G: the gain matrix;

Initialization

Use the manual feedbacks listed in Eqs. 13 and 14 to compute the initial values for A and G from Eqs. 3, 4 and 5.

Selectivity Estimation

The selectivity of query $q_i = (l_i, h_i)$ is estimated as $F(h_i + 1) - F(l_i)$; if it is negative, simply return 0.

[3] In our experiments using degree 6, the "ill-conditioned" problem did not arise. However, for higher degrees we might need to use another basis (such as Legendre polynomials or B-splines) since the basis of $x^i, i = 1, \ldots, n$ is in general ill-conditioned for large values of n.

Feedback and Adaptation

After the execution of q_i, get feedback $\zeta_i = (l_i, h_i, s_i)$ where s_i is the actual selectivity of q_i obtained from execution. If q_i is the first query after the latest update, set the fading weight α_i to a positive number less than 1. Use ζ_i to adjust A and G, as shown in Eqs. 11 and 12.

Comparison with [SLRD93]

Sun, Ling, Rishe, and Deng proposed in [SLRD93] a method of approximating the attribute distribution using a polynomial with the criterion of least-square-error. While both their method and ours use polynomial approximations, there are several differences between the two methods. First, their approach is *static* in the sense it is necessary to scan the database and count the frequencies of distinct attribute values, and, once computed, the approximating distribution remains unchanged until the next re-computation. Our method is *dynamic* and depends only on query feedbacks, with no access to the database. For a relation which is large and/or is updated regularly, the overhead of collecting or refreshing the statistics can be very expensive. Our approach totally avoids such overhead. Besides, in an environment where queries exhibit highly temporal or spatial locality on certain attribute ranges, ASE's dynamic adaptation to queries will perhaps be of greater benefit. Finally, ASE's adaptiveness to updates not only eliminates the overhead of statistics re-collection, but also provides a more graceful performance degradation for selectivity estimations through a query session interleaved with updates.

2.2 An Example

We use an example to demonstrate how the ASE works by using successive query feedbacks to approximate the data distribution. The experimental data is from a movie database, courtesy of Dr. Wiederhold of Stanford University, which records 3424 movies produced during the years 1890–1989. Figure 1 snapshots the evolution of the approximating distribution for a sequence of query feedbacks. The queries are listed in the table, where $[l_i, h_i]$ denotes the range of the ith query, and $\hat{s}_i$ and s_i denote the selectivities estimated (by ASE) before and obtained after the query execution respectively. In each frame, the curve of the approximating distribution f_i, drawn in solid line, is compared to the real distribution, drawn in discrete points. In frame 1, uniform distribution is assumed at the very beginning, as no queries have been issued. Note that knowledge of the relation cardinality (3424 tuples) has been implicitly enforced in the initial approximating distribution f_0, using the initialization scheme explained in the previous section. After the execution of two queries, as shown in frame 2, the approximating curve becomes closer to the actual distribution. However, f_2 is relatively inaccurate for attribute ranges outside $[1925, 1966]$ which have not been queried yet (and, thus, no distribution information is yet known). The third query and its feedback $\zeta_3 = (1904, 1939, 890)$ tunes f_2 into f_3 with better accuracy for range $[1904, 1939]$. It is worth mentioning that at the same time, f_3 improves

query sequence	1	2	3	4	5	6	7	8	9
$[l_i, h_i]$	[1935,1966]	[1925,1950]	[1904,1939]	[1890,1923]	[1908,1913]	[1948,1989]	[1957,1980]	[1964,1989]	[1916,1981]
$\hat{s_i}$	1073	1138	1248	567	2	1956	1103	1041	3173
s_i	1872	1399	890	136	14	2033	1130	1134	3045

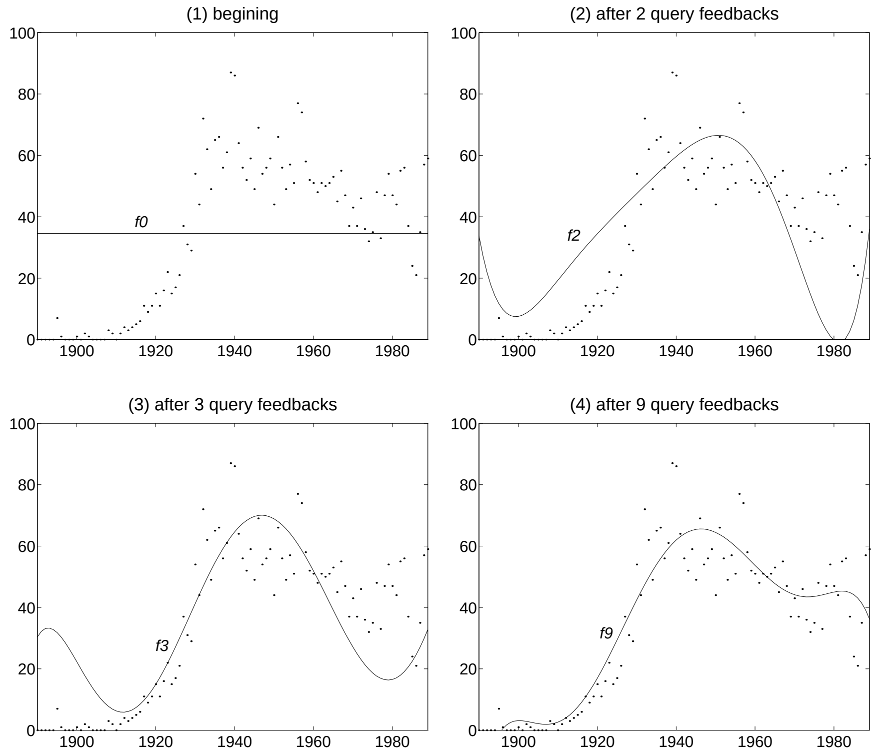

Figure 1: Adaptation Dynamics of ASE — an Example

the distribution of years greater than 1966, though no queries against this range have ever been posed. This is attributed to ASE's ability to infer and properly shape the unknown ranges using knowledge about the relation cardinality and distribution information obtained from queries on other attribute ranges. Subsequently, frame 4 shows the curve after nine query feedbacks, by which time the approximation has become even closer to the real distribution.

3 Experimental Results

A comprehensive set of experiments was performed to evaluate the ASE. We ran the experiments using the mathematics package MAPLE, developed by the Symbolic Computation Group of the University of Waterloo; MAPLE was chosen for its provision of immediate access to matrix operations and random number generators. We experimented also with the method proposed in [SLRD93] (referred to as SLR in what follows) for comparisons whenever appropriate. The selectivity estimation errors and the adaptation dynamics of ASE were observed and graphed for demonstration. However, to interpret and compare the estimation errors correctly, both *absolute error* and *relative error* are presented; they are calculated as:

$$\text{abs. err.} = \frac{|\hat{s} - s|}{|R|} \times 100, \quad \text{rlt. err.} = \frac{|\hat{s} - s|}{s} \times 100,$$

where $\hat{s}$ and s are the estimated and actual query result sizes, respectively; $|R|$ is the cardinality of the queried relation. Our reason for using both is that neither one alone can provide evidence of good or poor estimation in all cases. For example, a 200% relative error for a query of selectivity of 1 tuple by no means represents a poor estimate; in fact, it is the stringent selectivity (of 1 tuple) that causes such an exaggerated relative error. It must be pointed out that we do not compare the computation overhead since our method, which costs only negligible CPU time for query feedback computation, is definitely superior to all other methods which require extra database accesses (either off-line or on-line) for statistics gathering or sampling.

Both real and synthetic data were used in the experiments. The use of real data validates the usefulness of our method in practice (as has been demonstrated in the example); the use of synthetic data allows systematic evaluation of ASE under diverse data and query distributions. Throughout the experimentation, only selection queries were considered. Each query is represented as a range $[x - \delta/2, x + \delta/2]$, where $d_{min} \leq x \leq d_{max}$, $0 \leq \delta \leq d_{max} - d_{min}$. In this paper, we report only results from those experiments where x and δ are generated randomly from their respective domains using a random number generator. Experimental results regarding the impacts of different distributions of x and δ on the convergence rate of ASE are prepared in a more detailed version of this paper.

Three sets of experimental results are presented here. The first set shows the adaptability of ASE to various data distributions. The second set shows how ASE adapts to *query locality*, in the sense that it provides more accurate selectivity estimates for the attribute sub-ranges which are queried most. In the last set, we demonstrate ASE's elegant adaptation through database updates which require no overhead for database re-scan and statistics re-computation.

3.1 Adaptation to Various Distributions

To observe ASE's adaptability to various data distributions, synthetic data generated from each of the following four customized distributions were tested: normal distribution, chi-square distribution, the F distribution, and a "bi-modal" distribution.[4] The notations and customized parameters of each distribution are described in Tables 1 and 2. For each data distribution, three random query streams (each of which contains 50 queries) were run for both ASE and SLR.

Table 3 lists the average error per query of ASE and SLR under each data distribution. In order to achieve a fair comparison between ASE and SLR, the average errors, which exclude the first 10 queries of each query stream (during which ASE is still in its "learning" stage), are also calculated for comparison. The first set of columns shows that ASE is slightly inferior to SLR

[4] We do not present the results of uniform distribution since the ASE assumes uniform distribution from the very beginning.

Notations	Meaning
$N(\mu,\sigma)$	Normal distribution with mean μ, standard deviation σ
$\chi^2(n)$	chi-square distribution with n degrees of freedom
$F(m,n)$	F distribution with m and n degrees of freedom for numerator and denominator
$B(u_1,\sigma_1,u_2,\sigma_2)$	a bi-modal distribution which is an overlap of $N(u_1,\sigma_1)$ and $N(u_2,\sigma_2)$

Table 1: Distribution Notations

Distribution	$[d_{min}, d_{max}]$	cardinality
$N(200,150)$	$[-150, 550]$	10,000
$\chi^2(10)$	$[0, 1200]$	20,000
$F(10,4)$	$[0, 800]$	10,000
$B(250,150,450,50)$	$[-150, 550]$	12,500

Table 2: Customized Experiment Parameters

	1st - 50th queries				10th - 50th queries			
	ASE		SLR		ASE		SLR	
	abs. err.	rlt. err.	abs. err.	rlt. err.	abs. err.	rlt. err.	abs. err.	rlt. err.
N	0.73	4.43	0.16	2.4	0.16	3.66	0.16	2.73
χ^2	1.36	13.0	0.33	8.0	0.33	8.36	0.40	8.93
F	2.2	28.6	1.7	28.2	1.10	15.3	1.76	30.1
B	1.40	8.75	0.60	3.08	0.80	5.11	0.60	3.13

Table 3: Average Errors in Various Data Distributions

in estimation accuracy; however, the second set of columns shows that after ASE converges (after 10 queries), its accuracy is very comparable to that of SLR. Figures 2 through 5 depict the corresponding dynamics of ASE and SLR for one of the query streams under each data set. In the figures to the left marked (a), curve g corresponds to the approximating distribution computed from SLR; f_i denotes the adaptive approximating distribution from ASE after i query feedbacks. Figures (b) compare the estimation errors of ASE and SLR by plotting them along with the query streams. The adaptiveness of ASE can be clearly observed from the decreasing trend of errors as queries proceed. The occasionally high relative errors of ASE are either caused by stringently small selectivities (as evidenced by the high relative errors of SLR for the same queries), or are indications of the moments where feedbacks take place for the first time on the queried ranges. However, as can be seen from all the figures, after sufficient query feedbacks have covered the whole attribute domain, ASE converges the approximating distribution to a stable curve and provides estimations with constantly small errors.

3.2 Adaptiveness of ASE to Query Locality

No matter what method is used to estimate the data distribution, the computation *capacity* of the method is always limited (e.g., the number of intervals in a histogram, the degree of a polynomial). It is not uncommon for the distribution to be estimated to be too detailed to be modeled by the limited capacity. Therefore, we believe that instead of approximating the overall distribution evenly, the limited capacity should be used to approximate more accurately

	Queried Range	Locality
LowQL	Jan. 1948 – Dec. 1978	Low
MedQL	Jan. 1948 – June 1960	Medial
HighQL	Jan. 1948 – Jan. 1953	High

Table 4: Three Levels of Query Localities

	ASE		SLR	
	abs. err.	rlt. err.	abs. err.	rlt. err
LowQL	1.1	5.6	0.93	5.0
MedQL	0.33	6.3	0.66	10.6
HighQL	0.086	12.8	0.14	21.3

Table 5: Average Errors in Different Query Localities

the local distribution of a rather narrow attribute sub-range which imposes either a temporal or spatial query locality. ASE inherits this merit: the more query feedbacks obtained from a local area, the more accurate the resulting approximating distribution for this area.

An "event" database which contains 431,258 records of events during 1948-1978 was used in this experiment. Three levels of query localities, as outlined in Table 4, were designed to compare ASE and SLR. For each level of locality, three random query streams (each of which contains 50 queries) were tested for both ASE and SLR. Table 5 summarizes the average errors for the 10th to 50th queries (we excluded the first 10 queries during which ASE has not yet converged). The curves of the approximating functions and the estimation errors of ASE and SLR are graphed for comparison, according to the three levels of localities, in Figures 6 , 7, and 8. It can be seen both from the tables and figures that ASE and SLR behave almost the same for low locality, but that as locality increases, ASE turns out to be better. This is because ASE is computed dynamically according to the query feedbacks and thus implicitly takes into account the query locality; in contrast, SLR is statically computed from the underlying data.

3.3 Adaptiveness to Updates (ASE Performance under Updates)

In this section, we show the elegant adaptation of ASE to updates. The normal distribution data from Section 3.1 is used again. Table 6 briefs the characteristics of three different update workloads to be interleaved with the query streams (more details about the update workloads are given in Appendix B). Orthogonal to the update loads are three versions of ASE, namely, $ASE_{0.01}$, $ASE_{0.1}$, and $ASE_{0.5}$, with different fading weights (as indicated in the subscripts). For each update workload, three query streams (each of which contains 40 selection queries interleaved with updates) are generated, and each of them is tested with all three fading weights. Table 7 tabulates the average errors; Figures 9, 10, and 11 correspond to the adaptation dynamics of ASE in the three different update loads. The corresponding curves for the three fading weights are grouped and graphed in each figure.

It can be seen from the figures that ASE adapts elegantly to all update loads. For example, in Figure 9.b, the errors go up over a few queries after the 10th query where update occurs,

	updates occur at	no. of total tuples updated	change of distribution shape	update transition
LOAD1	11	4,500	local, big increase	in batch
LOAD2	11, 17, 23, 29	9,000	global, slightly increase	gradual
LOAD3	11, 17, 23, 29	9,000	global, drastic	gradual

Table 6: Characteristics of Three Update Workloads

Update Workload	$\mathrm{ASE}_{\alpha=0.01}$		$\mathrm{ASE}_{\alpha=0.1}$		$\mathrm{ASE}_{\alpha=0.5}$	
	abs. err.	rlt. err.	abs. err.	rlt. err.	abs. err.	rlt. err.
LOAD1	3.38	16.7	3.58	25.7	4.71	30.0
LOAD2	3.35	22.2	2.66	17.2	2.59	15.9
LOAD3	5.58	31.0	4.19	21.3	4.24	21.6

Table 7: Average Errors in Different Update Workloads

and then decline back to a stable low level. This adaptation can also be observed in Figure 9.a, where frames 2 through 4 show the adaptation of the approximating curves to the local distribution change at interval $[-50, 250]$. It is interesting to note from Table 7 that $\mathrm{ASE}_{0.01}$, $\mathrm{ASE}_{0.1}$, and $\mathrm{ASE}_{0.5}$ are respectively the best in update loads LOAD1, LOAD3, and LOAD2. This is no surprise since in LOAD1, a vast amount of update is done at once and thus it is advantageous to forget previous feedbacks and rely mainly on new ones. Therefore, the smallest fading weight $\mathrm{ASE}_{0.01}$ (which forgets previous feedbacks to the greatest extent) outperforms the other two in this case. Similarly, in LOAD2, the shape of the distribution does not change too much during successive updates, and thus $\mathrm{ASE}_{0.5}$ benefits the most by using old knowledge during transition. Finally, in LOAD3, where the distribution shape changes greatly through gradual updates, the use of $\mathrm{ASE}_{0.1}$ offers a compromise between the two extremes.

4 Conclusions

In this paper, we have presented a new approach for selectivity estimation. Capitalizing on the technique of recursive weighted least-square-error, we devised an adaptive selectivity estimator which uses query feedbacks to approximate the actual attribute distribution and to provide efficient and accurate estimations. The most significant advantage of this approach over traditional methods is that it incurs no extra cost for gathering database statistics. Furthermore, it adapts better to updates and query localities.

We hope this study will inspire a new direction for data knowledge acquisition, especially in systems where statistics gathering is cost prohibitive because of large data sizes (such as tertiary databases). The adaptive selectivity estimator can be further improved in several ways and explored in several directions. First, we will refine the feedback mechanism so that adaptation will stop after the approximating distribution converges and will be triggered after updates. We would also like to extend this work to complex queries which involve compound predicates or joins. Lastly, mathematical analysis of ASE is desired in order to give deeper insight into its performance behavior under diverse query distributions and into its theoretical limits.

Acknowledgements

The authors would like to thank Dr. Gio Wiederhold for providing us with a copy of the movie database.

References

[AG67] A.E. Albert and L.A. Gardner. *Stochastic Approximation and Nonlinear Regression.* M.I.T. Press, Cambridge, Massachusetts, 1967.

[Chr83a] S. Christodoulakis. Estimating block transfers and join sizes. In *Proceeding of 1983 ACM-SIGMOD Intl. Conf. on Management of Data*, pages 40–54, New York, 1983.

[Chr83b] S. Christodoulakis. Estimating record selectivities. *Inf. Syst.*, 8(2):105–115, 1983.

[dB78] C. de Boor. *A practical guide to splines.* Springer-Verlag, New York, 1978.

[Fed84] J. Fedorowicz. Database evaluation using multiple regression techniques. In *Proceedings of the ACM-SIGMOD Intl. Conf. on Management of Data*, pages 70–76, Boston, MA, 1984.

[HOT88] W. Hou, G. Ozsoyoglu, and B. K. Taneja. Statistical estimators for relational algebra expressions. In *Proceedings of the ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, pages 276–287, 1988.

[HS92] P. Haas and A. Swami. Sequential sampling procedures for query size estimation. In *Proceedings of the ACM-SIGMOD Intl. Conf. on Management of Data*, pages 341–350, San Diego, CA, 1992.

[IC91] Y.E. Ioannidid and S. Christodoulakis. On the propagation of errors in the size of join results. In *Proceedings of the ACM-SIGMOD Intl. Conf. on Management of Data*, pages 268–277, Denver, Colorado, 1991.

[Ioa93] Y.E. Ioannidis. Universality of serial histograms. In *Proceedings of the 19th VLDB Conference*, Dublin, Ireland, 1993.

[KK85] N. Kamel and R. King. A method of data distribution based on texture analysis. In *Proceedings of the ACM-SIGMOD Intl. Conf. on Management of Data*, pages 319–325, Austin, Texas, 1985.

[LN90] R. J. Lipton and J. F. Naughton. Practical selectivity estimation through adaptive sampling. In *Proceedings of the ACM SIGMOD Intl. Conf. on Management of Data*, pages 1–11, Atlantic City, NJ, 1990.

[LST83] E. Lefons, A. Silvestri, and F. Tangorra. An analytic approach to statistical databases. In *Proceedings of the 9th VLDB Conference*, 1983.

[Lyn88] C. A. Lynch. Selectivity estimation and query optimization in large databases with highly skewed distributions of column values. In *Proceedings of the 14th VLDB Conference*, pages 240–251, Los Angeles, CA, 1988.

[MCS88] M.V. Mannino, P. Chu, and T. Sager. Statistical profile estimation in database systems. *ACM Computing Surveys*, 20(3), 1988.

[MD88] M. Muralikrishma and D. DeWitt. Equi-depth histograms for estimating selectivity factors for multi-dimensional queries. In *Proceedings of the ACM-SIGMOD Conf. on Management of Data*, pages 28–36, Chicago, Illinois, 1988.

[MO79] T.H. Merrett and E. Otoo. Distribution models of relations. In *Proceedings of the 5th VLDB Conference*, pages 418–425, Rio De Janero, Brazil, 1979.

[PSC84] G. Piatetsky-Shapiro and C. Connell. Accurate estimation of the number of tuples satisfying a condition. In *Proceedings of the ACM-SIGMOD Intl. Conf. on Management of Data*, pages 256–275, Boston, MA, 1984.

[S+79] P. G. Selinger et al. Access path selection in a relational database management system. In *Procs. of ACM-SIGMOD*, pages 23–34, 1979.

[SB83] W. Samson and A. Bendell. Rank order distriburions and secondary key indexing. In *Proceedings of the 2nd Intl. Conf. on Databases*, Cambridge, England, 1983.

[SLRD93] W. Sun, Y. Ling, N. Rishe, and Y. Deng. An instant and accurate size estimation method for joins and selection in a retrieval-intensive environment. In *Proceedings of the ACM-SIGMOD Intl. Conf. on Management of Data*, pages 79–88, Washington, DC, 1993.

[Wil91] D.E. Willard. Optimal sample cost residues for differential database batch query problems. *Journal of the ACM*, 38(1):104–119, 1991.

[You84] P. Young. *Recursive estimation and time-series analysis*. Springer-Verlag, New York, 1984.

A Recursive Solution for Weighted LSE

Using the notations defined in Section 2, for each query feedback $\zeta_i = (l_i, h_i, s_i)$, let

$$X_i = \left[\ \Phi_0(h_i+1) - \Phi_0(l_i) \quad \Phi_1(h_i+1) - \Phi_1(l_i) \quad \ldots \quad \Phi_n(h_i+1) - \Phi_n(l_i)\ \right]$$

Now suppose m query feedbacks ζ_i, $i = 1, 2, \ldots, m$ are given. It is not hard to see that Eq. 10 can be rearranged and expressed as

$$||\mathcal{X}^m * A - \mathcal{Y}^m||^2, \tag{15}$$

where $\mathcal{X}^m$ is a $m \times n$ weighted matrix, $\mathcal{Y}^m$ is a $m \times 1$ weighted vector, and the respective ith rows of $\mathcal{X}^m$ and $\mathcal{Y}^m$ are defined as

$$\mathcal{X}_i^m = (\prod_{j=i+1}^{m} \alpha_j) \cdot \beta_i \cdot X_i, \quad \mathcal{Y}_i^m = (\prod_{j=i+1}^{m} \alpha_j) \cdot \beta_i \cdot s_i \tag{16}$$

According to Eqs. 7 and 16, the optimal values of A that minimize Eq. 15 is computed as

$$\begin{aligned} A^*_m &= G_m N_m && (17)\\ G_m &= [\sum_{i=1}^{m}(\mathcal{X}_i^m)^t\mathcal{X}_i^m]^{-1} = [\alpha_m^2\sum_{i=1}^{m-1}(\mathcal{X}_i^{m-1})^t\mathcal{X}_i^{m-1} + \beta_m^2 X_m^t X_m]^{-1} && (18)\\ N_m &= \sum_{i=1}^{m}(\mathcal{X}_i^m)^t\mathcal{Y}_i^m = \alpha_m^2\sum_{i=1}^{m-1}(\mathcal{X}_i^{m-1})^t\mathcal{Y}_i^{m-1} + \beta_m^2 X_m^t s_m && (19)\end{aligned}$$

First, we derive the recursive formula for G_m. From the above equations we have

$$G_m^{-1} = \alpha_m^2 G_{m-1}^{-1} + \beta_m^2 X_m^t X_m \qquad (20)$$

$$N_m = \alpha_m^2 N_{m-1} + \beta_m^2 X_m^t s_m \qquad (21)$$

Do $G_m\times$ (Eq.20) $\times G_{m-1}$, we obtain

$$G_{m-1} = \alpha_m^2 G_m + \beta_m^2 G_m X_m^t X_m G_{m-1} \qquad (22)$$

Multiply the above equation by X_m^t, and we get

$$G_{m-1}X_m^t = G_m X_m^t[\alpha_m^2 + \beta_m^2 X_m G_{m-1} X_m^t] \qquad (23)$$

Rearrange Eq.23 and multiply it by $X_m G_{m-1}$,

$$G_{m-1}X_m^t[\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t]^{-1}X_m G_{m-1} = G_m X_m^t X_m G_{m-1} \qquad (24)$$

Finally, substitute Eq.24 into Eq.22 and rearrange, we obtain the recursive formula for G_m:

$$\underline{G_m = (\frac{1}{\alpha_m})^2 G_{m-1} - (\frac{\beta_m}{\alpha_m})^2 G_{m-1}X_m^t[\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t]^{-1}X_m G_{m-1}} \qquad (25)$$

Now we can derive the recursive formula for A^*. Substitute Eqs. 21 and 25 into Eq. 17 and let $\Delta = [\alpha_m^2 + \beta_m^2 X_m G_{m-1} X_m^t]^{-1}$, we obtain

$$\begin{aligned} A^*_m &= G_{m-1}N_{m-1} - \beta_m^2 G_{m-1}X_m^t\Delta X_m G_{m-1}N_{m-1}\\ &\quad +(\frac{\beta_m}{\alpha_m})^2 G_{m-1}X_m^t s_m - (\frac{\beta_m^4}{\alpha_m^2})G_{m-1}X_m^t\Delta X_m G_{m-1}X_m^t s_m\\ &= A^*_{m-1} - G_{m-1}X_m^t\Delta[\beta_m^2 X_m A^*_{m-1} - (\frac{\beta_m}{\alpha_m})^2(\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t)s_m\\ &\quad +(\frac{\beta_m^4}{\alpha_m^2})X_m G_{m-1}X_m^t s_m]\\ &= A^*_{m-1} - G_{m-1}X_m^t\Delta(\beta_m^2 X_m A^*_{m-1} - \beta_m^2 s_m)\\ &= A^*_{m-1} - \beta_m^2 G_{m-1}X_m^t\Delta(X_m A^*_{m-1} - s_m) && (26)\end{aligned}$$

The term $G_{m-1}X_m^t\Delta$ in the above expression can be further simplified as

$$\begin{aligned} & G_{m-1}X_m^t[\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t]^{-1}\\ =\ & [G_m G_m^{-1}]G_{m-1}X_m^t[\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t]^{-1}\\ & \text{now substitue } \mathrm{G_m^{-1}} \text{ above with Eq. 20}\\ =\ & G_m[\alpha_m^2 G_{m-1}^{-1} + \beta_m^2 X_m^t X_m]G_{m-1}X_m^t[\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t]^{-1}\\ =\ & G_m[\alpha_m^2 X_m^t + \beta_m^2 X_m^t X_m G_{m-1}X_m^t][\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t]^{-1}\\ =\ & G_m X_m^t[\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t][\alpha_m^2 + \beta_m^2 X_m G_{m-1}X_m^t]^{-1}\\ =\ & G_m X_m^t && (27)\end{aligned}$$

Therefore, from Eq.26 and 27 we get the recursive formula for A^*:

$$\underline{A_m^* = A_{m-1}^* - \beta_m^2 G_m X_m^t (X_m A_{m-1}^* - s_m)} \tag{28}$$

B Update Workload Specifications

In our experiments, an update query is simulated by its effect on the value distribution of the attribute of interest. An update query is specified by five parameters:

$$(i, N, D, [min, max], prob_{INS}),$$

where i means this update takes place immediately before the ith query in the query stream. N is the number of tuples updated (either inserted or deleted). Each tuple's attribute value is randomly generted from range $[min, max]$ according to a distribution D. A tuple is inserted with probability $prob_{INS}$ or deleted with probability $1 - prob_{INS}$. Three different update workloads are tested, each of which is interleaved with another 40 random selection queries. The three update workloads are specified in the following, with $U(x, y)$ denotes the uniform distribution among range $[x, y]$, $N(\mu, \sigma)$ the normal distribution with mean μ and standard deviation σ.

LOAD1:	$(11, 4500, U(-50, 250), [-50, 250], 1.0)$	
LOAD2:	$(11, 2250, U(-150, 550), [-150, 550], 0.75)$,	$(17, 2250, U(-150, 550), [-150, 550], 0.75)$,
	$(23, 2250, U(-150, 550), [-150, 550], 0.75)$,	$(29, 2250, U(-150, 550), [-150, 550], 0.75)$
LOAD3:	$(11, 3000, N(-63, 50), [-150, 25], 0.9)$,	$(17, 1500, N(112, 40), [25, 200], 0.1)$,
	$(23, 2250, N(290, 60), [200, 375], 1.0)$,	$(29, 2250, N(455, 50), [375, 550], 0.4)$

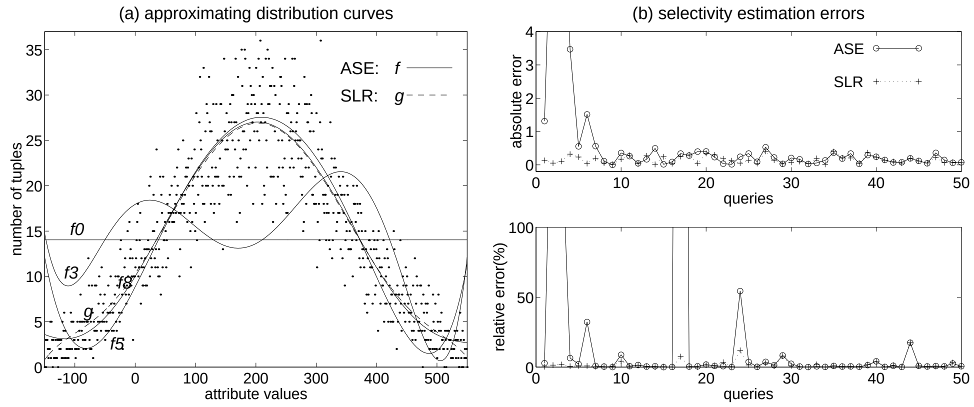

Figure 2: Normal Distribution

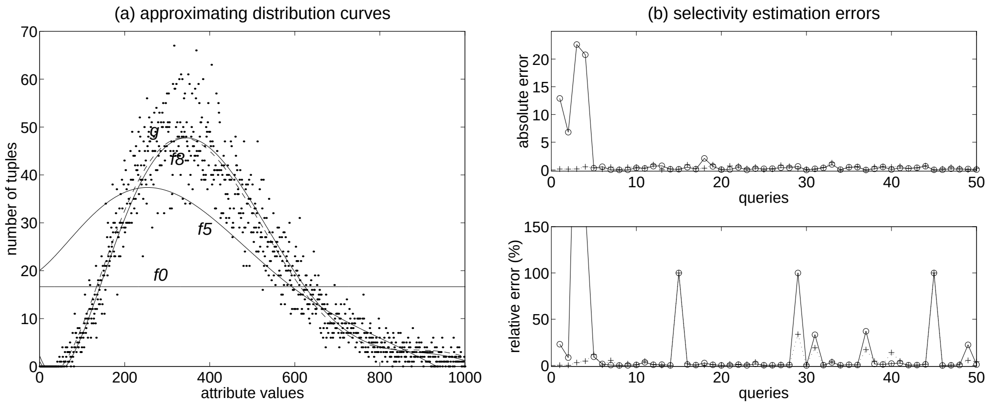

Figure 3: Chi-Square Distribution

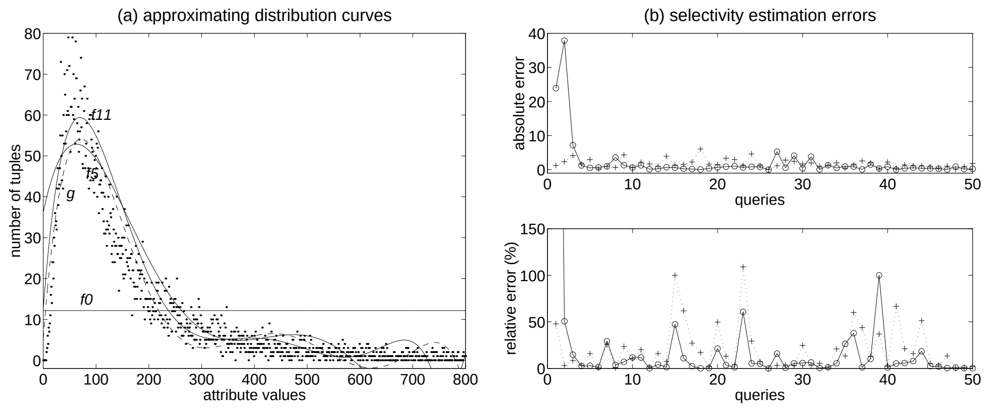

Figure 4: the F Distribution

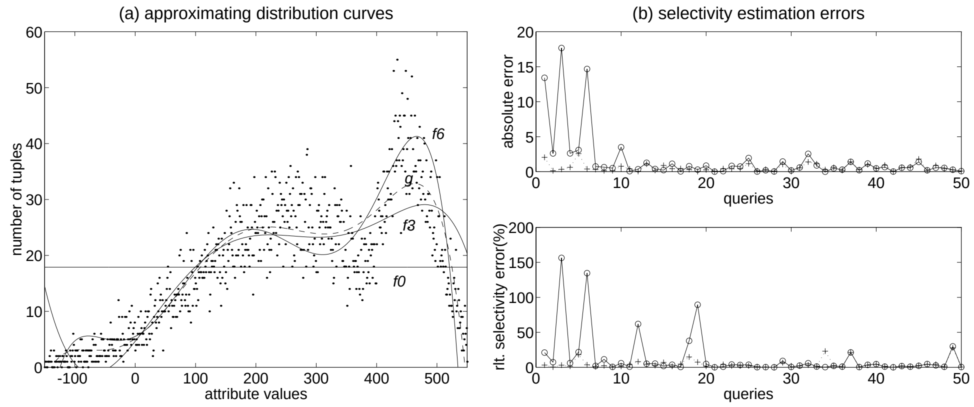

Figure 5: a bi-modal Distribution

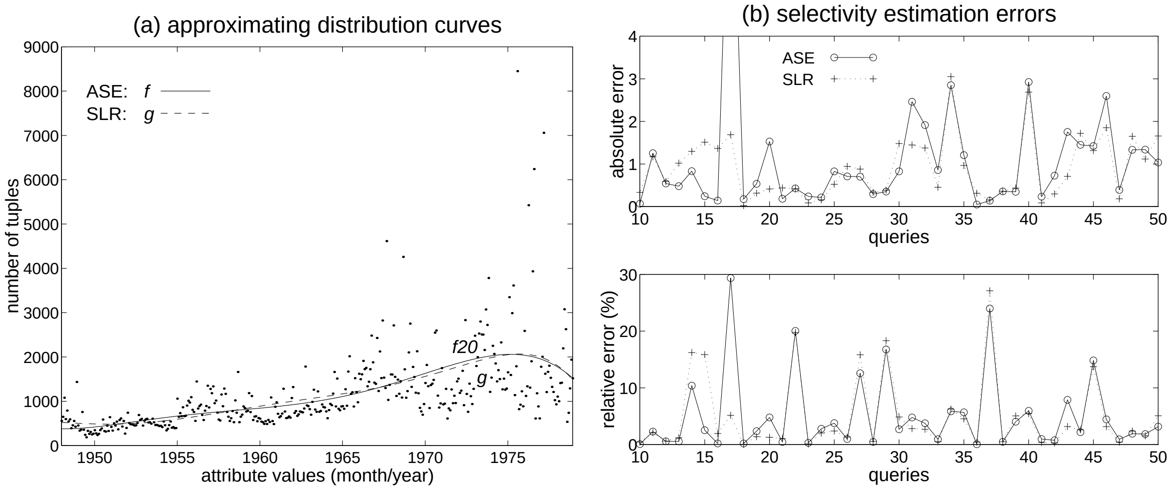

Figure 6: Adaptation in LowQL (Low Query Locality)

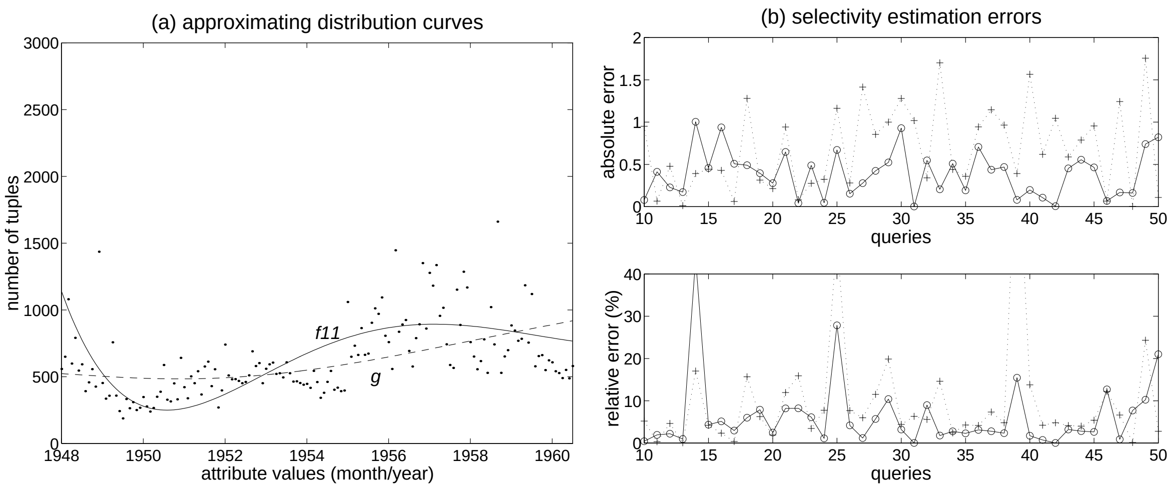

Figure 7: Adaptation in MedQL (Medial Query Locality)

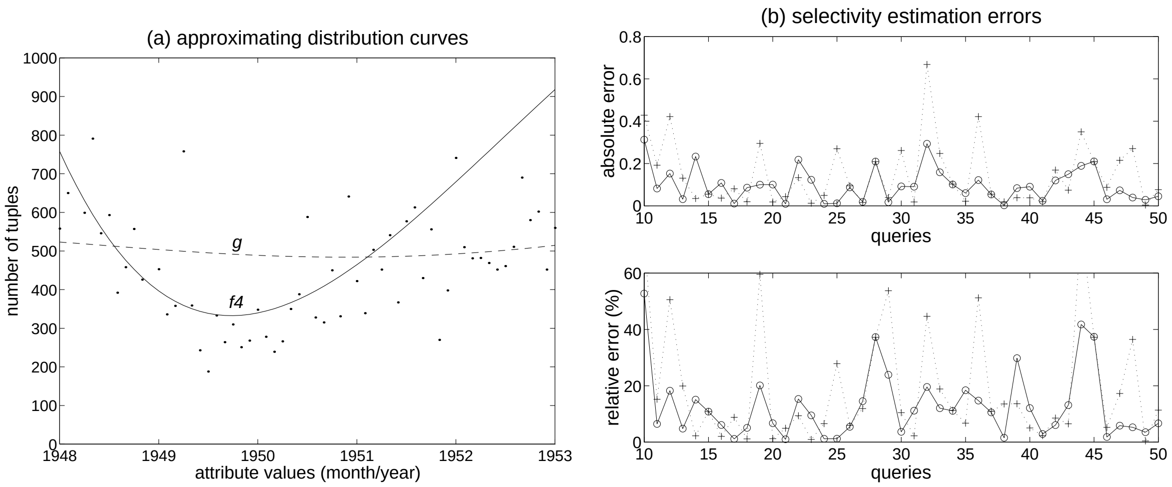

Figure 8: Adaptation in HighQL (High Query Locality)

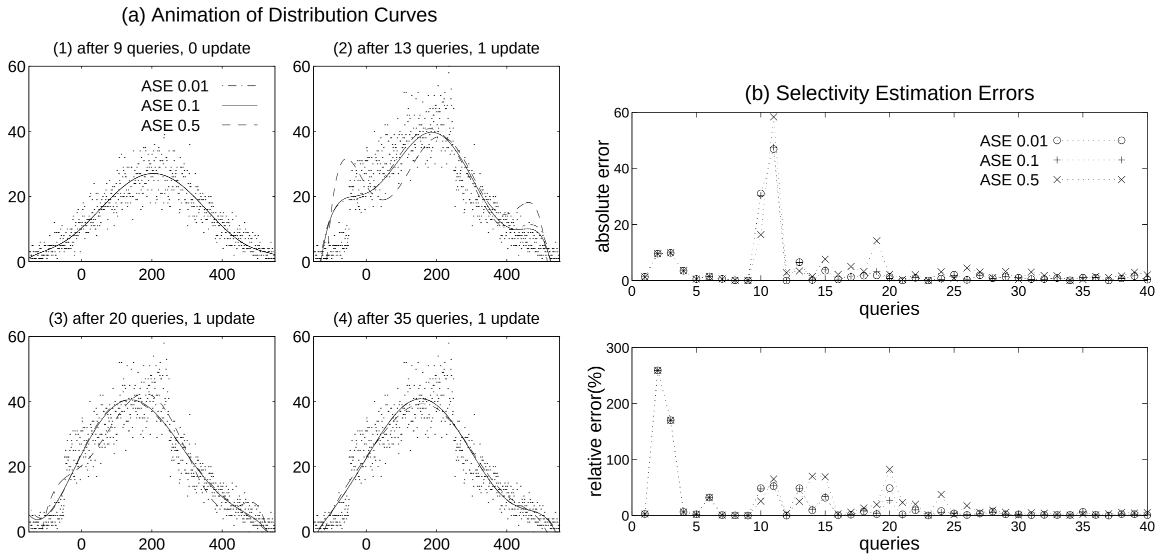

Figure 9: Adaptation in Update LOAD1

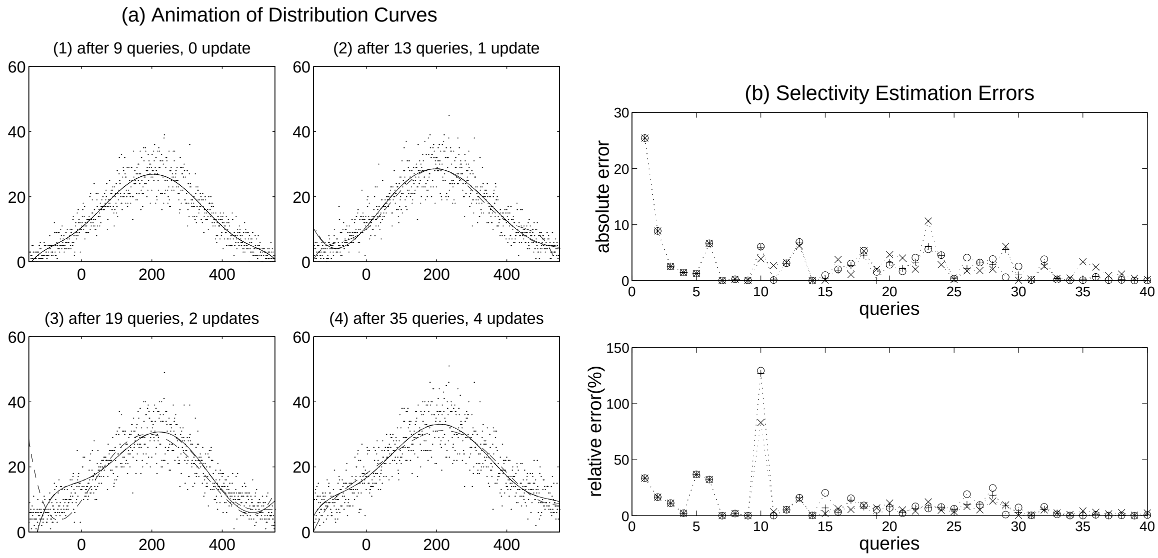

Figure 10: Adaptation in Update LOAD2

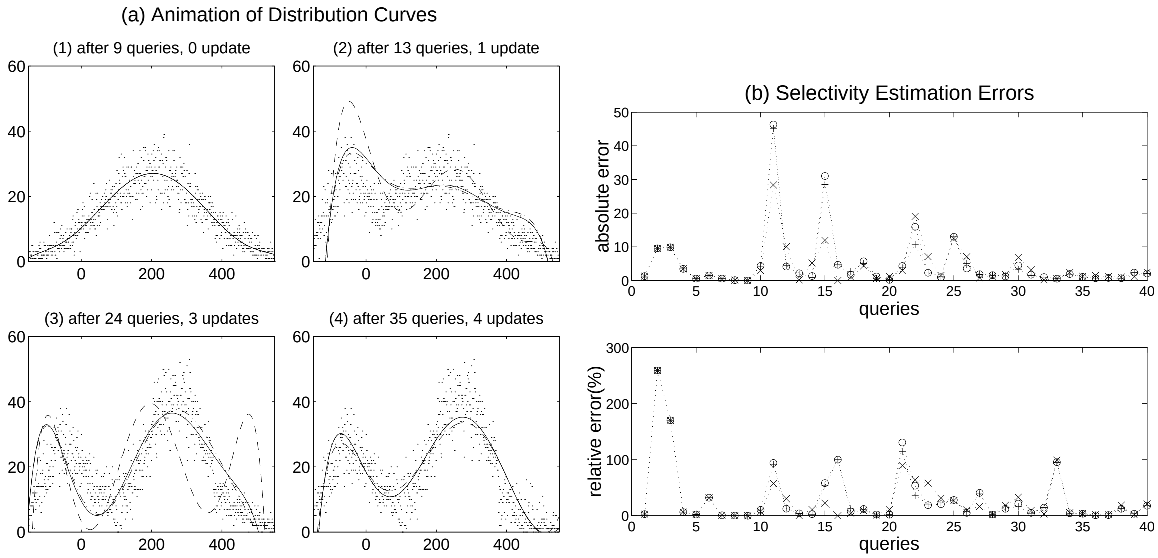

Figure 11: Adaptation in Update LOAD3