

Summary Grids: Building Accurate Multidimensional Histograms

Pedro Furtado and H. Madeira

Depto Eng. Informática Univ. Coimbra
Pinhal Marrocos
3030 Coimbra, Portugal
pnf@dei.uc.pt

Abstract

Data summarization is very important for many data analysis tasks. In this paper we propose a simple but efficient data summarization algorithm, which outputs a histogram for multidimensional data, and make a comparative study of its usage with different distributions and with existing algorithms. The idea is to iteratively grow and modify regions of homogeneous data. This is a different strategy from a commonly used strategy of iteratively fracturing subspaces using straight lines. This work compares both strategies and concludes that the new technique is better and holds good results. We also concluded that discriminate handling of outliers is important to provide good approximates.

1. Introduction

Data reduction and summarization are important bases for such diverse tasks as datacube representations, efficient OLAP [12], data mining, selectivity estimation and in any data analysis task. Among the alternative techniques [11], multidimensional histograms are a good choice as they are simple, easy to construct and maintain, present a small run-time overhead, do not require the data to fit a probability distribution and occupy a reasonably small space. For this reason histograms are frequently used in tasks such as query result size estimation to help query processors optimize the query execution strategy. MOLAP Datacubes can have gigabytes of size on disk

and being able to access reduced or summarized versions which retain most of the distribution form is important to allow efficient analysis or to give preliminary approximated answers to queries, as alternative methods to sampling for online data analysis [2]. We emphasize here that summarizing has a different objective from aggregation as it analyses the data to maintain the form of the distribution whereas aggregation simply computes averages or sums over the ranges given by the user. Both techniques are important and we understand summarization in MOLAP as a tool to be used for regions explicated by the user. In this paper we propose a new data summarization algorithm called Summary Grid (SGRID) which builds a histogram for multidimensional data. This algorithm addresses a shortcoming of a previous technique (MHIST) [10] used to build histograms for data summarization purposes. By comparing SGRID with MHIST we are confronting two alternative histogram construction strategies. MHIST iteratively fractures subspaces using straight lines, while SGRID iteratively coalesces neighbors until clustering constraints are met. End-to-end line fracturing as in MHIST often prematurely cuts homogeneous subregions and this obstructs the best partitioning. The SGRID algorithm avoids this by marrying the ideas of clustering with histogramming. We evaluate the merits of this algorithm for different data distributions and compare it to the MHIST algorithm and its possible variants. MHIST was compared favorably with other alternative techniques such as singular value decomposition (SVD) in terms of error. We will show that our technique

presents better results than MHIST. For simplicity we restrict our experiments to the two dimensional space. Still, a major concern with SGRID and indeed with any clustering technique is the computational overhead and the related scalability issue. We show that this overhead is bounded and easily adjustable. We also compare SGRID and MHIST with a technique similar to the Quasi-Cubes in [1]. This compares another algorithm based on regression and a fixed grid partitioning of the datacube (REGR) with the adaptive SGRID and MHIST strategies and therefore is a very interesting experiment.

Discriminate handling of outliers is important to avoid large errors in the approximation. A few of the most distant values from the average are not summarized and instead histogram buckets must include the complete characterization for those points. This is important for handling non-uniform data as shown by the results. The paper is organized as follows: Section 2 discusses related work. Section 3 briefly reviews key concepts related to histograms and our variations of the MHIST technique. Section 4 presents the SGRID technique. Section 5 studies the complexity of the algorithms and Section 6 briefly presents the REGR technique. Section 7 presents the results from evaluation of MHIST, SGRID and REGR techniques and Section 8 concludes the paper.

2.Related Work

Several techniques have been described for data reduction in [11]. Parametric techniques such as singular value decomposition or the discrete wavelet transform assume a model for the data. Non-parametric techniques that do not assume a model for data include histograms, cluster-based reduction of data and index trees. Sampling was also mentioned as providing a way to obtain quick but approximate answers.

Examples of efficient clustering techniques are BIRCH [14], which uses the concept of a clustering features tree (CF-tree) to dynamically build a cluster tree and CLARANS [7] which progressively refine clusters based on some heuristics. These techniques do not produce histograms.

The statistical information grid STING [13] is another technique that effectively summarizes the data distribution for spatial data mining. The space is divided into a set of hierarchical grids and the data must be characterized as belonging to some well-defined distribution with parameters such as the mean and standard deviation. This technique relies on statistical information associated with the individual cells and is hierarchical in the sense that a number of grids with increasing resolution is used to accommodate different degrees of precision and progressive refinement of regions satisfying a given query. The main drawback of

this technique is the need to identify and quantify a distribution and this is often not possible. SGRID adapts the grid to patterns in raw data and doesn't need user intervention.

Quasi-Cubes were presented in [1] as an effective way to support approximate datacubes using regression. This technique is used in our experiments for comparison.

We have mentioned in section 1 some of the advantages of histograms. Simple histograms are frequently used in commercial systems for selectivity estimation in query processors. One-dimensional histograms were comprehensively studied in [9], focusing on database estimation problems: for a query involving a single attribute of a relation, the result size depends on its data distribution. In [8] an extensive framework is presented for histogram characterization and the results for different combinations of partitioning-related parameters are compared to identify the histogram partitioning strategy of choice which would be close to the best in both construction time and data approximation error.

Multidimensional histograms were also studied in [9]. The result size of a query involving two or more attributes of the same relation depends on the joint data distribution of those attributes, that is, the frequencies of all combinations of attribute values in the database. To approximate the joint data distributions most commercial systems adopt the attribute value independence assumption under which the results size is derived by computing from one-dimensional histograms of individual attributes, regardless of the actual data dependencies. [10] proposes the use of multidimensional histograms to drop the attribute value independence assumption, as it was noticed that real-life data rarely satisfies that assumption and therefore the estimate is very inaccurate. The construction of a multidimensional histogram poses some difficulties. It was perceived that a multidimensional sort parameter is harder to handle, as it requires finding arbitrary rectangular non-overlapping regions. There are a very large number of choices for these rectangular regions. The approach adopted in [9] was taken from [6] and imposes a priority on the dimensions of the sort parameter, so that regions are first created along the highest priority dimension, each such region is then broken into subregions along the second highest priority dimension, etc. An equivalent approach called PHASED was also used in [10] and compared with other techniques, namely, the attribute value independence assumption (AVI), singular value decomposition (SVD), Hilbert Numbering and a new technique named MHIST which presented the most accurate estimation results of them all. The strict order in attribute partitioning of the PHASED approach results in

a low quality approximation. On the other hand, MHIST exhibits more flexibility by picking a dimension in each step based on its criticality to the partitioning constraint.

3. The MHIST Technique

The purpose of histograms is to approximate a particular data distribution using precomputed tabular information. Histograms have been extensively studied (for example in [3,4,5,8, 9]) and we only review here the most important concepts.

A given attribute A takes a set of values V_i with frequency f_i , denoting the attribute value frequency. One-dimensional histograms on an attribute X partition the attributes' data distribution into β subsets called buckets. This partitioning process uses a *partitioning rule* and approximate the values and frequencies in each bucket using some other rules.

Multidimensional histograms approximate the joint data distribution of the attributes they represent. Figure 1 shows an example of a two-dimensional data distribution with frequencies for some combinations of values and a possible division into buckets for that data distribution. We adopted a representation similar to that in [10].

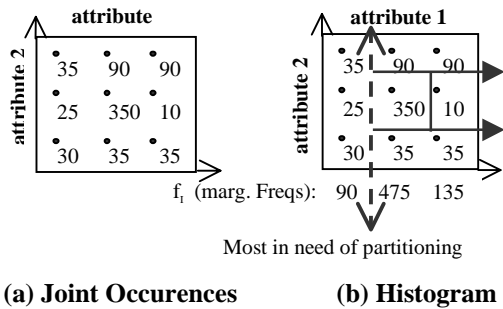


Figure 1: Joint Data Distribution

The MHIST technique in [10] partitions the space by finding the attribute whose marginal distribution is the most in need of partitioning. The marginal distribution is the individual data distribution of each attribute (number of occurrences of each possible value (f_i) as depicted in figure 1(b)). MHIST achieves the best results by using a partitioning constraint that was labeled $Maxdiff(V,A)$, which determines that there is a boundary in each of the $\beta-1$ largest differences between source values, where the source value is the area (spread x frequency). Splitting is done along straight end-to-end fracture lines.

We introduced some variations in the calculation of the differences to obtain the splitting point:

0. Marginal difference (MHIST-0) – absolute value of the difference between marginal frequencies: $|f_{i+1} - f_i|$ (ex. figure 1: $|475 - 90|$).
1. Normalized marginal difference (MHIST-1) –

divide marginal difference by the number of pairs of values considered: $|f_{i+1} - f_i|/N$ (ex: $|475 - 90|/3$).

2. Sum of absolute differences (MHIST-2) - Instead of using the marginal distributions, compute individual differences and then sum their absolute values: $\sum(|f_{i+1,j} - f_{i,j}|)$ (ex. figure 1: $|90 - 35| + |350 - 25| + |35 - 30|$).
3. Normalized sum of absolute differences (MHIST-3) - divide (MHIST-2) by the number of pairs of values considered: $\sum(|f_{i+1,j} - f_{i,j}|/N)$ (for 2 dimensions).
4. Maximum differences approach (MHIST-4) – consider only the greatest difference between two neighboring points in the multidimensional space: $\max |f_{i+1,j} - f_{i,j}|, j=1,n$ (ex. figure 1: $|350 - 25|$).

Our motivation for considering these different variations of the same basic algorithm comes from the fact that alternative strategies give more weight to different aspects of the multi-dimensional problem of finding a good partitioning strategy. This way we are able to more exhaustively study and evaluate the different alternatives. Approaches MHIST-0 and MHIST-2 use absolute sums and therefore favor consecutive fracturing lines parallel to the longest sub-dimension. In figure 1 this means that the subspace in the left of the dashed fracture line is consecutively favored, creating fracturing lines parallel to it. MHIST-2 sums individual differences as opposed to the cumulative differences in MHIST-0. This gives more weight to extremes. Approaches 1 and 3 are simple normalizations of MHIST-0 and MHIST-2 obtained by dividing the result by the number of pairs considered. This avoids the strategy of longest dimensions favoring which is typical of MHIST-0 and MHIST-2, but also favors smaller dimensions, which may result in over-fragmentation of those small regions. The computational burden is equivalent in all approaches because even marginal distributions must be recomputed for subspaces. We defer the analysis of these approaches to the evaluation section.

4. Summary Grids (SGRID)

The MHIST technique is easy to implement but induces some estimation error by splitting along an end-to-end line. Although this fracture line is the most critical considering the cumulative marginal source value differences, **it often prematurely separates subsets of uniform valued neighbors**. This problem is more relevant when the space volume is big. Reflecting on those issues we created an alternative SGRID approach which drops the line fracturing strategy and instead

groups homogeneous data in a way that tries to minimize the resulting error in estimates.

The SGRID technique uses an incremental strategy (SGRID-incr) which works through maximizing the area of regions within given objective constraints and search restrictions. Those parameters are pre-calculated by the (SGRID-heur) module. As with most histograms this technique then assumes uniform spread and frequencies within buckets. The strategy SGRID-incr tries alternative expansion directions for each region according to the heuristics. Figure 2 shows the process diagrammatically. In our implementation it starts coalescing points in one corner up to the point when the objective constraints cannot be met. Several directions are tried as in the figure to attempt to maximize the region area. The degree of exhaustiveness in region search is totally configurable to balance execution time overhead with optimality. After one subregion is set the remaining space goes through the same process to determine the other regions.

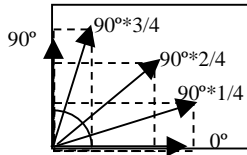


Figure 2: SGRID-incr Module Probing a Region

The algorithm is,

Next_StartPoint = Origin

while Next_StartPoint != space end point

for each direction

while error metric limits not reached

 grow region (delta-large growth)

if maxe passed **then** extract up to x% outliers

end if

end while

 recall last region within error limits

 // optional smaller step growth

while error metric limits not reached

 grow region (delta-small growth)

if maxe passed **then** extract up to x% outliers

end if

end while

 save last region within error limits

end for

 choose the region with largest area from saved regions

 assign the point immediately after the boundaries to

 Next_StartPoint (follow a predefined order for dimensions).

end while

Regions have some parameters used as heuristic constraints to find region boundaries. The radius and diameter values tend to smooth variations because they are normalized. For this reason we added the accumulated or maximum error parameter, corresponding to the sum of the absolute values of the differences to the

bucket average for every point represented in it. The last heuristic constraint is the number of outliers allowed in each bucket. This way the parameters which are tuned by the SGRID-heur module are:

Radius – this is equivalent to the standard deviation and measures the normalized deviation from the average.

$$r = \left(\frac{\sum_{i=1}^n (v_i - \mu)^2}{n} \right)^{1/2}$$

Diameter – measures the mean difference between points in the region

$$d = \left(\frac{\sum_{i=1}^n \sum_{j=1}^n (v_i - v_j)^2}{n(n-1)} \right)^{1/2}$$

Maximum Error – measures the cumulative error. Our implementation actually uses the absolute deviation or cumulative square error,

$$cse = \left(\sum_{i=1}^n (v_i - \mu)^2 \right)^{1/2}$$

Maximum Number of Outliers – outliers are points that are not approximated. Each histogram bucket must represent them separately. The points most distant from the mean are chosen for outliers. The result is a better approximation because outliers will not contribute to the error. Let O_i be the candidate set of outliers in bucket region β_i and (p,v) be a pair (point,value), where point are attribute coordinates and value is the corresponding joint occurrence frequency. Let mo be the maximum number of outliers in each bucket.

$$O_i = \{ (p,v) \in \beta_i : |v - \mu| \text{ is one of the } mo \text{ maximum values for the expression } |v_j - \mu|, \forall (q,v_j) \in \beta_i \}$$

The actual number of outliers in a bucket is determined by the algorithm.

Directional Search – Another important parameter determines which directions to try in the greedy growth algorithm. Figure 2 shows this for the two dimensional case, where k trial generating vectors are determined by dividing the 90° into sectors with $90^\circ/(k-1)$ aperture.

The value of the k parameter is crucial for the balance of performance and precision in SGRID. The first direction is tried until the error measures are overpassed. Then the algorithm determines a region with at least the area just computed for the next direction as defined by k . This way the algorithm tries to find the maximal area in all trial directions such that the error measures are not overpassed. The corresponding region is chosen and excluded from the search space. For more dimensions k can be replaced by k_i for each dimension i allowing a greater selectivity in preferred dimensions. Another improvement is to optionally pre-define end-to-end borders in selected attribute values for hierarchical categorical information in datacubes.

Support for One-pass over Regions – Region growing should not require complete recomputation of

measures, otherwise the overhead would be too big. We prevent this by the incremental nature of the algorithm which is well supported by the set (N,LS,SS) of clustering features (CF) from BIRCH [14] (N - number of elements, LS – linear sum of values, SS – square sum of values) and the corresponding additivity theorem, which in our case allows the technique to try with boundary modification without recalculating the whole region features. The additivity theorem states that when coalescing two regions with CF values (N1,LS1,SS1) and (N2,LS2,SS2) respectively, the resulting CF values are (N1+N2,LS1+LS2,SS1+SS2). All the other parameters except the outliers are easily calculated from the CF vector. But the incremental property is also applied to outliers by simply keeping two additional arrays with “maximum number of outliers” upper and lower extremes from each bucket. This way the upper and lower extremes from the incremented region are just the “maximum number of outliers” number of extremes from the correspondent upper and lower extremes of subregions.

Heuristics precalculation – The heuristics module determines the maximum admissible values for the parameters. The user can specify either the maximum admissible average error to obtain a high-precision histogram no matter how much space is used, or a fixed number of buckets (nbuckets) to obtain a compact histogram such as the ones used in selectivity estimation. Bucket sampling is used to retrieve NSAMPLING buckets and this way derive error parameters limits.

5. Complexity Estimation

– It is our major concern to study the scalability properties of the SGRID algorithm. Both SGRID, MHIST and any fixed grid scheme exhibit $O(n)$ complexity, n being the size (number of data values) of the original space. Still, SGRID and MHIST exhibit a larger overhead because they do some data analysis to determine the partitions, while a fixed partitioning scheme would require access to exactly n values. We first demonstrate this statement by reminding that SGRID makes k directional trials and from these trials an area a_i is chosen and excluded from the search space. A majorant to the number of data values accessed in this process is $k \times a_i$ (as a_i is the maximal area discovered in the process) and the total number of data values accessed is less than

$$k \times a_1 + \dots + k \times a_p = k \times \sum_{i=1}^p a_i = k \times n \rightarrow O(n) \quad (1)$$

(p is the number of buckets determined)

Even though the SGRID algorithm is $O(n)$, a careful study is needed because its performance strongly depends on the number k and this number must be chosen carefully depending on the number of dimensions

(usual dimensions would range from 1 to 10). For this reason we do a brief comparative study next. In the 2 dimensional experiments of section 7 we have fixed $k=5$ and obtained good results. Let k_i be the value of k per dimension and equal for any dimension for simplicity. The following equations determine a majorant for the number of data values visited in the SGRID and MHIST algorithms and their derivation is simple from the previous rationale. We have determined a possible equation for the number of total trials in the $ndim$ dimensions $k(k_i, ndim)$ for the SGRID algorithm:

$$k(k_i, 0) = 1, \quad k(k_i, 1) = 1 \quad (2)$$

$$k(k_{i+1}, ndim) = \sum_{i=0}^{k_i} k(i, ndim-1) \quad (3)$$

and the total number of data values accessed is less than $k(k_{i+1}, ndim) \times n$ (4)

Furthermore, we introduce a correcting factor in equation (1) by considering that the average area taking into account both chosen and rejected areas in each trial set is $f \times a_i$ for the trial i ($0 < f < 1$). This way equations (1) and (4) become

$$k \times f \times n \quad (5)$$

$$k(k_{i+1}, ndim) \times f \times n \quad (6)$$

The number of data values visited for the MHIST algorithm is approximated by considering the fact that initially $ndim \times n$ values were accessed to determine the marginal values ($ndim$ is the number of dimensions) and using the simplifying approximation of quad-tree like partitioning each successive line fracture would require the recomputation of a number of marginal values requiring successive access to:

line fracture 1 – n values
line fracture 2-3 – $n/2$ values
line fracture 4-7 – $n/4$ values
line fracture 8-15 – $n/8$ values

(as stated before each fracture requires the recalculation of the marginal values in the directions orthogonal to the fracture line (plane in 3D, hyperplane in n dimensions)). The resulting formula is,

$$ndim \times n + \sum_{i=0}^{\log_2 nb-1} n = (\log_2 nb + ndim) * n \quad (7)$$

The actual value is smaller than this because usually the space is not partitioned as in a quad-tree, some subspaces are much more partitioned than others.

We have used these equations to obtain the values in figures 3 and 4 for multiplying factors in SGRID and MHIST. For instance, if 1024 buckets are to be determined in MHIST for 5 dimensions this would imply an overhead of at most $(5+10) \times n = 15 \times n$ accesses to

data values. In the case of SGRID the factor f should also be considered. For instance, to determine a histogram for 5 dimension data with 35 trials covering a wide range of directions in all five dimensions the SGRID algorithm requires $35 \times f \times n$ accesses to data values. Clearly the k_i parameter must be chosen in accordance to the number of dimensions to avoid a large overhead for higher dimensional datasets. We highlighted some acceptable choices in figure 3, where f is the ratio (average area on trials)/(maximum area on trials) for the SGRID technique.

ndim\k _i	5	4	3	2	1
1	1f	1f	1f	1f	1f
2	5f	4f	3f	2f	1f
3	15f	10f	6f	3f	1f
4	35f	20f	10f	4f	1f
5	70f	35f	15f	5f	1f
6	126f	56f	21f	6f	1f
7	210f	84f	28f	7f	1f
8	330f	120f	36f	8f	1f

Figure 3: Multiplying Factors (SGRID) $f:0-1$

ndim\nb	32	64	128	256	512	1024
1	6	7	8	9	10	11
2	7	8	9	10	11	12
3	8	9	10	11	12	13
4	9	10	11	12	13	14
7	12	13	14	15	16	17
8	13	14	15	16	17	18

Figure 4: Multiplying Factors in MHIST nb Buckets

6.The Regression Technique

The regression technique is important in the context of our experiments because it provides both a good algorithm and one of the simplest forms of partitioning, that is, fixed grid partitioning. This effectively allows us to compare two types of adaptive partitioning strategies and a fixed grid strategy. The technique is similar to the one used in the Quasi-Cubes experiment [1] and divides the whole space into equal sized regions. Approximation is achieved by keeping the coefficients of a regression line (regression plane in 3D, regression hyperplane in n-D) that best approximates the bucket data. As in Quasi-Cubes we used rows and columns marginal sums as coordinates in the regression and reconstruct the points from those coordinates and regression coefficients. Outlier handling is also supported.

7.Evaluation

In this section we evaluate the merits of the SGRID algorithm for different data distributions and compare it

to the MHIST algorithm [10] and its possible variations.

Error Measurement: The total error of the histogram approximation is given by,

$$Error = \sum_{i=1}^n |v_i - \mu_{\beta_j}|, \beta_j \text{ is the bucket } v_i \text{ belongs to.}$$

We use $Avgerror = Error/avgval$ to give an estimate of the average error for any query, where $avgval$ is the average value in each point of the original space. This error measure is expressed as a percentage and this way we are able to compare results for different datasets. On the other hand it means that in extreme cases a bad approximation can produce an $Avgerror$ value beyond 100% (for instance, if the average data value is 0.5 an average error of 100% means that the average value would be in the interval [0,1]). The error magnitude is adjustable by simply varying the number of buckets. We have chosen a fixed number of buckets in each experiment. **Data distributions:** Our approach to the experimental evaluation of the different techniques was to model very different distributions and study the behavior for each one, following two generating approaches. One approach is based on populating the space with a randomly distributed user defined number of clusters, each following a normal distribution with number of clusters and standard deviation specified by the user. The other approach generates the data according to the Zipf distribution [15], which is said to model well typical joint data distributions in databases because when attributes have dependencies between them, there will often be a few combinations of attribute values that occur much more frequently than others. The level of dependency is modeled by the z parameter in the distribution. High values of z imply strong dependence between the attributes; small values imply uniform data.

Data Sets: For simplicity our experiments focused on a two-dimensional space. The maximum number of outliers considered was 10 or none. We present a representative subset of our results. Figures 5 and 6 identify classes for cluster and zipf generated datasets that we present.

Label	n° clusters	stdev	edge szs	n° elems
A,B,F	5,5,25	0.3	50x50	50k
C	10	0.02	200x200	500k
D,E	10	0.12, 0.3	200x200	500k
G,H,J	30,100x2	0.02	50,30,50	20k

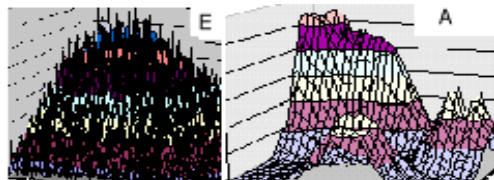
Figure 5: Legend for the Clustered Dataset

Label	z	edges	n°elems	Label	z	edges	n°elems
O	0.7	50	10k	S	0.7	200	100k
P	0.7	50	500k	T	0.7	200	500k

Figure 6: Legend for the Zipf Dataset

We have organized the results according to dataset

characteristics that we observed after generation. In the figures SGRID refers to the new technique, REGR refers to the fixed grid regression technique described in section 6 and MHIST is evaluated for the 5 different variants of marginal calculation described in section 3. The results labeled E are the point estimation error and those labeled E-outliers refer to reduction with outliers. Figure 7 shows the shape of typical distributions of the first two types studied here.



(a) Quasi-normal (b) Horse Sell

Figure 7: Distribution for Quasi-normal and Horse Sell Data Sets

• **Dense Space , Quasi-Normal Distribution (D, E)**

These datasets (figure 7(a)) are the result of many data values and reasonably large standard deviation. Figure 8 shows the estimation error results for these data sets.

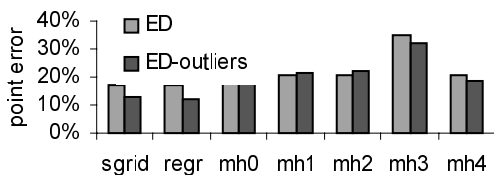


Figure 8: Estimation Errors in Data Sets D, E

SGRID and REGR obtain similar results in the presence of these distributions, while MHIST results are worse. These distributions are relatively easy for REGR, as the data shows a smooth monotone incline well modeled by lines. MHIST has difficulty deciding the appropriate splitting lines in these smooth inclines.

• **Half Dense Space, Horse Sell Distrs (B, A, F)**

These datasets have the format of figure 7(b) and the results are shown in figure 9.

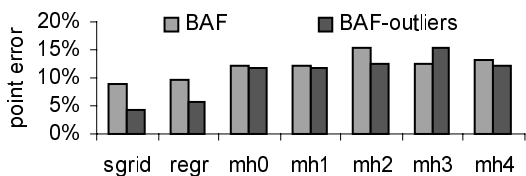
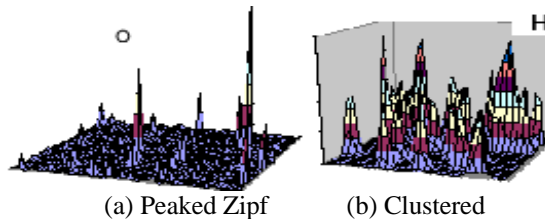


Figure 9: Estimation Errors Data Sets B, A, F

Again SGRID and REGR present quite similar results and both are much better than MHIST. Although

different, the smooth monotone inclines are present as well in these data sets.

Figure 10 shows the next two typical peaked distributions corresponding to clustered and zipf data sets.



(a) Peaked Zipf (b) Clustered

Figure 10: Distribution for Peaked Zipf and Clustered Data Sets

• **Sparse Space, Clustered Distributions (C,G, J, H)**

These datasets contain many cluster peaks because they have a reasonable large set of clusters and small standard deviation (figure 10(b)).

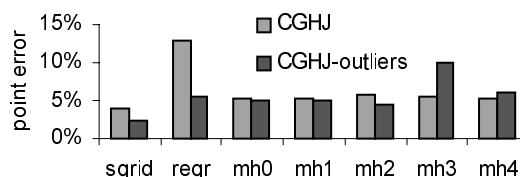


Figure 11: Estimation Errors in Data Sets C,G,H,J

These data sets are irregular. For this reason data analyzing algorithms (SGRID, MHIST) build better histograms than fixed grid algorithms (REGR), although the use of outliers can help these to adapt to the distribution. SGRID had the best results overall.

• **Sparse Space, Peaked Distributions (O, P, S,T)**

These zipf generated datasets (figure 10(a)) have high single data value peaks, while most of the remaining data is sparse and small valued. Again a careful data analysis is important and outlier support is crucial.

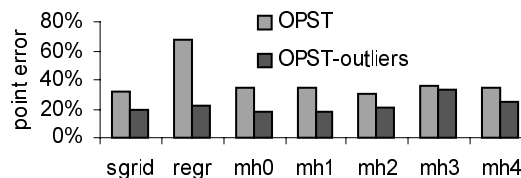


Figure 12: Estimation Errors in Data Sets (O,P,S,T)

SGRID achieves the best results again, though MHIST is able to adapt as well. Thin peaks can be put as outliers, reducing the error significantly.

- **Execution Times:** the average execution times are shown in figure 13. SGRID exhibits a larger overhead than MHIST and REGR is the fastest algorithm as it is not adaptive. The algorithms were only partially optimized, so these results can only give a fuzzy idea of the actual execution time overhead.

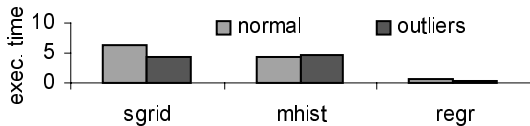


Figure 153: Mean Execution Times

- **Experiment Conclusions:** these experiments have allowed us to determine what important features algorithms should have to handle various datasets. The REGR algorithm is the fastest as it partitions the space in a fixed grid manner, but it has some problems with zipfian and heavily clustered data. The MHIST algorithm often cuts homogeneous regions and therefore produces a not so good partitioning for some data sets. The SGRID algorithm is the best overall. The directional trial parameter of SGRID must be carefully chosen, otherwise the algorithm will not be scalable to higher dimensions. Discriminate outlier handling is also important to eliminate large deviations. MH0 and MH1 are the most reliable variants of MHIST (they have similar results because the computation is equivalent).

8. Conclusion

In this paper we have presented a new technique for histogram construction for multidimensional data by region coalescing (SGRID) and used it to compare alternative strategies for histogram construction. The other techniques are the MHIST line fracturing algorithm and the REGR fixed grid regression algorithm. Several datasets were studied to understand the behavior of these algorithms. It was shown that irregular distributions require data analysing algorithms (SGRID, MHIST), although the use of outliers can help fixed grid techniques adapting to those irregularities. SGRID consistently presents lower error than MHIST in data approximation and for some important data distributions it is also better than the REGR algorithm. We have also shown that adaptivity pays an execution overhead and scalability price. Additionally, we have shown that outlier handling is an important addition to reduce the error in any algorithm.

10. References

- [1] D. Barbara and M. Sullivan, "Quasi-Cubes: A space-efficient way to support approximate multidimensional databases," Technical Report, ISE Dept., September 1997.
- [2] Joseph M. Hellerstein, Peter J. Haas, Helen J. Wang. Online Aggregation..Proceedings of the ACM SIGMOD international conference on Management of data, May 13-15, 1997, Tucson, AZ.
- [3] Yannis E. Ioannidis: Universality of Serial Histograms. 19th International Conference on Very Large Data Bases, August 24-27, 1993, Dublin, Ireland, Proceedings. Morgan Kaufmann, pp.256-267.
- [4] Yannis E. Ioannidis, Stavros Christodoulakis: Optimal Histograms for Limiting Worst-Case Error Propagation in the Size of Join Results. Transactions on Database Systems TODS 18(4): 709-748(1993)
- [5] Yannis E. Ioannidis, Viswanath Poosala: Balancing Histogram Optimality and Practicality for Query Result Size Estimation. Proceedings of the 1995 ACM SIGMOD Intl. Conf. on Management of Data, San Jose, California, May 22-25, 1995, pp. 233-244.
- [6] M. Muralikrishna, David J. DeWitt: Equi-Depth Histograms For Estimating Selectivity Factors For Multi-Dimensional Queries. Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data, Chicago, Illinois, June 1-3, 1988.
- [7] Raymond T. Ng. And Jiawei Han, Efficient and Effective Clustering Methods for Spatial Data Mining, Proceedings of the 20th International Conference on Very Large Data Bases, (VLDB'94), September 12-15, 1994, Santiago de Chile, Chile.
- [8] Viswanath Poosala, Yannis E. Ioannidis, Peter J. Haas, Eugene J. Shekita: Improved Histograms for Selectivity Estimation of Range Predicates. Proceedings of the 1996 ACM SIGMOD Intl. Conf. on Management of Data, Montreal, Canada, June 4-6, 1996., pp 294-305.
- [9] V. Poosala, Histogram-Based Estimation Techniques in Database Systems, PhD Thesis, University of Wisconsin-Madison, 1997.
- [10] V. Poosala, Y. Ioannidis. Selectivity Estimation Without the Attribute Value Independence Assumption. Proceedings of the 23rd VLDB Conference, Athens, Greece, 1997.
- [11] Special issue on data reduction techniques of the bulletin of the Technical Committee on Data Engineering of the IEEE Computer Society, December 1997, Vol. 20, n 4.
- [12] Special issue on Supporting Online Analytical Processing of the bulletin of the Technical Committee on Data Engineering of the IEEE Computer Society, March 1997, Vol. 20, n 1.
- [13] W. Wang, J. Yang and R. Muntz, STING: A statistical Information Grid Approach to Spatial Data Mining, in Proceedings of the 23rd VLDB Conference, Athens, Greece, 1997.
- [14] T. Zhang, R. Ramakrishnan, M. Livy. BIRCH: An Efficient Data Clustering Method for Very Large Databases, in SIGMOD'96, Montreal, Canada 1996.
- [15] George Kingsley Zipf: Human Behaviour and the Principle of Least Effort: an Introduction to Human Ecology. Addison-Wesley 1949.