The VLDB Journal (2008) 17:683–702
DOI 10.1007/s00778-006-0032-z

REGULAR PAPER

DAWN: an efficient framework of DCT for data with error estimation

Ming-Jyh Hsieh · Wei-Guang Teng · Ming-Syan Chen · Philip S. Yu

Received: 20 October 2004 / Accepted: 10 November 2005 / Published online: 29 September 2006

Abstract On-line analytical processing (OLAP) has become an important component in most data warehouse systems and decision support systems in recent years. In order to deal with the huge amount of data, highly complex queries and increasingly strict response time requirements, approximate query processing has been deemed a viable solution. Most works in this area, however, focus on the space efficiency and are unable to provide quality-guaranteed answers to queries. To remedy this, in this paper, we propose an efficient framework of DCT for dAta With error estimatioN, called DAWN, which focuses on answering range-sum queries from compressed OP-cubes transformed by DCT. Specifically, utilizing the techniques of Geometric series and Euler's formula, we devise a robust summation function, called the GE function, to answer range queries in constant time, regardless of the number of data cells involved. Note that the GE function can estimate the summation of cosine functions precisely; thus the quality of the answers is superior to that of previous works. Furthermore, an estimator of errors based on the Brown noise assumption (BNA) is devised to provide tight bounds for answering range-sum queries. Our experiment results show that the DAWN framework is scalable to the selectivity of queries and the available storage space. With GE functions and the BNA method, the DAWN framework not only delivers high quality answers for range-sum queries, but also leads to shorter query response time due to its effectiveness in error estimation.

M.-J. Hsieh · W.-G. Teng · M.-S. Chen (✉)
Electrical Engineering Department,
National Taiwan University,
Taipei, Taiwan, ROC
e-mail: mschen@cc.ee.ntu.edu.tw

W.-G. Teng
Department of Engineering Science,
National Cheng Kung University,
Tainen city 701, Taiwan, ROC

P. S. Yu
IBM Thomas J. Watson Research Centre, P.O.Box 704,
Yorktown, NY 10598, USA
e-mail: psyu@us.ibm.com

1 Introduction

Approximate query processing has recently emerged as a viable solution for dealing with the huge amount of data, highly complex queries and increasingly strict response time requirements that characterize today's decision-support-system (DSS) applications. In such systems, users usually pose very complex queries to the on-line analytical processing (OLAP) system, which requires complex operations over gigabytes of data and takes a very long time to produce exact answers. Consequently, the issue of approximating OLAP queries becomes critical. Answering range queries is one of the primary tasks of OLAP applications. However, datasets tend to be very large in real data warehousing systems. Thus, answering aggregate queries can be computationally expensive. To address this issue, providing approximate answers to online queries efficiently is an essential means for cost/performance reasons.

In particular, one kind of data cube, called the OP-cube (operational data cube), is widely applied in information systems. An OP-cube is a data cube that stores information for operational purposes rather than demographic data. Examples of OP-cubes include the sales of bookstores or department stores aggregated by different branches, product categories and time intervals

The major difference between demographic data cubes and the OP-cubes is that there is usually a dependency between adjacent data cells in OP-cubes, e.g., different members in the same dimension are relevant in geographic locations or in time. Hence, OP-cubes can usually be modeled as a Brown noise distribution [19]. In this paper, we focus on OP-cubes and explore their characteristics. It is noted that most of real-world data cubes constructed from transaction databases are time-relevant; consequently, those cubes can be regarded as OP-cubes.

In addition to approximate query processing, error bound estimation of the answers to queries is an important functionality of DSS applications. That is, an approximate answer would be inapplicable if the user had no way of knowing whether it is very accurate, or off by many orders of magnitude. Hence, both an efficient approximate query processing algorithm and deterministic error bounds are required for DSS applications.

1.1 Related works

To construct an informative data cube with limited storage space, compression is required. Usually compression techniques exploit the benefits of energy compaction, i.e., information with numerical values can be centralized to fewer transformed coefficients to significantly reduce the amount of storage required. The approaches explored in this section include sampling, histograms, wavelets and discrete cosine transform (DCT).

Random samples of data collection typically provide good estimates for aggregate queries [8], and the construction procedure is very efficient to run. In addition, random samples yield the running aggregates and their confidence intervals [7]. However, deterministic error bounds are not provided, and the sampling methods must collect enough sample data to achieve the desired probabilistic accuracy, resulting in extra storage requirements. Also, when answering non-aggregate queries involving joins, a subset of the exact answer could be empty, which makes it difficult to apply sampling as an approximation tool for answering queries [2].

Histogram methods have also been applied extensively to query selectivity estimation and are considered useful for obtaining approximate answers to queries [11,17]. According to the method of partitioning data into buckets, histogram methods can be classified as the equi-depth [16], the V-optimal and the MaxDiff [10,22], to name but a few approaches. Among the histogram methods, the work in [18] offers the most promising technique. It is known as MaxDiff partitioning, and has been shown to provide the best trade-off between accuracy and computational efficiency. This approach recursively partitions a space by splitting an appropriately chosen dimension, whose aggregate distribution contains two neighboring elements with the maximum difference between their measured values, into two parts. This step is repeated until the number of partitions obtained equals the number of available buckets. It has been empirically verified that histogram methods provide better approximate answers than random sampling methods. However, the histogram-based approach could be problematic for high dimensional datasets [23]. As the number of dimensions of a dataset increases, both the storage overhead and the construction cost of histograms increase dramatically.

Wavelet-based methods [2,23,24] provide a mathematical tool for hierarchical decompositions. Recent studies have demonstrated that wavelet methods are feasible for selectivity estimation and the approximation of range-sum queries over OLAP data cubes. The work in [24] develops techniques that extend the partial sums technique proposed in [9] to accelerate answering of range-sum queries. This approach uses wavelet techniques to compute an approximate version, called the compact partial-sum cube. For a d-dimensional range-sum query, the complexity can be reduced to $O(n_T \cdot 2^d)$, where n_T is the number of coefficients. Though outperforming single-cell estimation, the procedure for computing partial sums may have to process a dataset that is substantially larger than the original data set [23], which is infeasible in practice. The authors of [23] propose a wavelet-based approach that decomposes/reconstructs data without partial sums. The complexity can be further reduced to $2d \times \min\{n_T, 2\prod_{i=1}^{d} \log |D_i|\}$, where $|D_i|$ is the cardinality of dimension i. Note that if a hash table is available for locating coefficients, the cost to retrieve a wavelet coefficient can be reduced to $O(1)$. However, once this technique is applied, we are not able to know which coefficient in the error tree contributes to the submitted query. Thus, extra effort is required for identifying which coefficients are needed. In terms of CPU time, as pointed out in [23] the aggregate cost of $O(2\prod_{i=1}^{d} \log |D_i|)$ and the extra effort for applying hash tables is mostly larger than n_T in practice. Also, authors of [3] extend the previous works to handle multiple measures. The wavelet represents a function in terms of a coarse overall approximation and a series of detailed coefficients that revise the function from coarse to fine. For modern wavelet theory, the Haar basis is the foundation of wavelets. Specifically, thresholding techniques are adopted to select the most representative coefficients in order to contain compressed information within limited storage space.

Haar wavelet decomposition can be extended to multidimensional data arrays by performing a series

Table 1 An example of an 8×8 dataset

	0	1	2	3	4	5	6	7
0	81	89	59	95	77	30	105	43
1	123	116	57	94	17	57	113	36
2	121	109	114	98	86	9	79	116
3	83	115	19	19	72	21	68	24
4	26	73	33	122	42	28	92	34
5	51	36	17	28	28	34	14	77
6	73	50	104	44	33	45	91	107
7	93	30	18	79	109	105	124	23

of one-dimensional decompositions. For example, in a two-dimensional case, we first apply one-dimensional decomposition to each row of data. Next, the transformed rows are treated as an original data set and decomposition is performed on each column. This kind of decomposition, applied in [23,24], is known as standard decomposition.

However, there are two critical issues when wavelet methods are applied to OLAP applications. The first issue is that it requires a lot of memory to keep related coefficients at each step of decomposition. In the worst case, the space cost is half that of the data cube, since the original data is processed pairwisely during decomposition. The second issue is that the quality of reconstruction depends heavily on the order of dimensions when decomposition is performed, subject to the constraint of limited storage space. This problem can be best understood by the following example.

Example 1 Consider the two-dimensional dataset in Table 1. The size of each dimension is eight. By applying multidimensional wavelet decomposition techniques [23] to this dataset, decompositions along both dimensions are performed iteratively. The reconstruction results are shown in Table 2, where Table 2a is the result decomposed along the x-axis first from Table 1 and Table 2b is that decomposed along the y-axis first. The percentage of coefficients to be retained in each step is set at 25%.

If a range-sum query, $Q_s = \text{Sum}(3{:}3, 1{:}6)$, requesting the summation of the cells with bold numbers is submitted, the answer calculated from Table 2a will be very different from that from Table 2b. The former gives 245 and the latter gives 436, whereas the correct answer is $115 + 19 + 19 + 72 + 21 + 68 = 314$, showing that wavelet decomposition is degraded by the order dependency imposed on the corresponding reconstruction results. □

Authors of [2] propose an alternative to approximate query processing using wavelets. The method, called non-standard decomposition, loads data chunks from a sorted dataset and performs the entire computation required to decompose the chunk. Hence, the problem

Table 2 Results of performing two-dimensional wavelet decomposition/reconstruction

(a)along x-axis first							
55	55	75	75	56	75	65	65
55	55	75	75	56	75	65	65
75	75	95	95	114	56	85	85
76	76	16	16	36	55	46	46
65	65	24	106	73	73	37	78
65	65	69	62	73	73	37	78
65	65	84	47	43	43	83	93
65	65	84	47	73	73	104	12
(b)along y-axis first							
58	88	58	58	65	65	79	51
58	88	58	58	65	65	79	51
58	88	82	34	65	65	56	74
58	88	34	82	65	65	102	28
56	48	55	102	65	65	99	32
56	48	102	55	65	65	60	71
89	37	67	67	31	31	79	51
89	37	67	67	99	99	79	51

of dimension order can be avoided. For each chunk, this approach performs one step of pairwise averaging and differencing for each one-dimensional row of array cells along all dimensions. This process is then repeated recursively only on the quadrant containing averages across all dimensions. Also, the range of contributed cells and the corresponding quadrant sign information is kept in a synopsis in order to answer queries efficiently. This approach works well for chunk-based organizations of relational tables. However, for real world applications with huge amounts of data, the extra-preprocessing step required to organize chunks impacts on the performance of underlying database systems significantly. Moreover, for very large databases with numerous chunks, which require approximate query processing, very few coefficients can be kept for each chunk if a dynamic coefficient-thresholding scheme is applied. That is, this degrades the optimality of global thresholding for wavelets proven in [21].

Discrete cosine transform (DCT) is widely used in image and signal processing for 2-dimensional domains since it takes the advantage of unequal distribution in the frequency of natural signals and therefore provides plausible efficiency in compression capability. A DCT-based histogram method that extends the mechanism in image and signal processing to multidimensional data, is discussed in [13], where compressed information from a large number of small-sized histogram buckets is maintained using DCT. Note that, the DCT for multidimensional data cubes (or histograms) is called DCT for data in this paper. The characteristic that differentiates the DCT for data from the DCT for signals is that the former does not compute all DCT coefficients in order to avoid dealing with a huge number of coeffi-

cients. Instead, zonal sampling of coefficients can be employed, i.e., only the coefficients in specific locations are selected.

The work in [13] provides some fundamentals for storing multidimensional data with low storage overhead and low error rates, and also conducts a preliminary analysis of range-sum queries over compressed information. However, the foundation of errors respective to range–sum queries due to coefficient sampling is not explored. Moreover, the work in [13] tends to suffer from the following problems which might limit its practicality. The first is that it uses integrals to estimate the summation of cosine functions, which could cause miscalculation of the answers, regardless of the number of transformed coefficients. An example of this defect is given in Sect. 3.3. The second problem is that this work does not address the error bound of answers and the method of estimating query errors is not available. The quality of answers is thus not guaranteed. To remedy these problems, we propose a robust method for estimating the summations of cosine functions and provide a theoretical foundation of error bounds for queries. Note that accurate error estimation can also be used to improve query response time. For range-sum queries, the errors due to compression increase while the selectivity of queries decreases. Thus, the query response time can be improved if a minimum number of coefficients are processed to compute the answer for a given acceptable error rate.

To estimate the quality of approximate answers, [12] proposes basic methods that require additional storage for information about the accurate answers of each cell or each bucket. However, the approaches cannot be applied to ad hoc range-sum queries in very large data cubes with numerous cells or buckets. The reason is that keeping extra information to estimate error bounds is not economic, since the coverage of each ad hoc query is unknown. The probabilistic wavelet synopsis proposed in [4–6] is a wavelet-based reduction technique with guarantees on the accuracy of individual approximate answers. The authors of [4] and [5] show that the maximum relative error of estimating a single cell in data reconstruction is $\max_i \left\{ \frac{|\widehat{d}_i - d_i|}{\max\{d_i, S\}} \right\}$, where d_i and $\widehat{d}_i$ are accurate and approximate answers, respectively, and S is a user defined parameter. However, with the same problem in [12], the accurate answer d_i will be never known after the decomposition and thresholding are performed. In practice, the error could be estimated by the constant value S, but the guaranteed error bound will be very loose if the answers (cell values) have a large variance. Moreover, as mentioned in [5], it costs $O(N^2)$ storage space to estimate error bounds of ad hoc range-sum queries, where N is the number of cells in the data cube. That is, the cost to estimate error bounds is even higher than keeping the original information. The authors of [6] extend the works in [4] and provide the answers for D-dimensional data cubes that are guaranteed to be within $(1+\epsilon)$ of the optimal maximum absolute error in time $O(\frac{\log R}{\epsilon} 2^{2^D+3D} \cdot N \log^2 N \cdot B \log B)$, with a total space requirement of $O(\frac{1}{\epsilon} 2^{3D} \cdot N \log^2 N \cdot B)$, where R, N, B and ϵ denote the maximum absolute coefficient value, the number of cells in the data cube, the size of synopsis in the error tree and the desired approximation factor, respectively. However, in addition to the considerable complexity in time and space, the provided error bound in the form of "$(1+\epsilon)$ of the optimal" under a synopsis, B, does not match users' requirements in general. DSS applications with approximate query processing should provide the error bounds to guarantee the absolute relative errors directly.

Basically, both wavelet based and DCT based approaches can deliver good approximate answers for the sparse multidimensional datasets seen in most DSS applications. The are two major advantages in applying DCT-based approaches. The first is that each DCT coefficient can be transformed independently. That is, the DSS applications can transform coefficients by several computing nodes simultaneously. Thus, the time cost for compressing a huge data cube can be reduced significantly. Also, the execution time required to approximate answers can be improved by parallel processing. Though some wavelet-based techniques, such as in [2], can process chunks by different computing nodes, the DCT-based technique can achieve maximum parallelism, where all coefficients are independent. The second advantage is that real world applications suffer from the problem of huge data cubes. Even the nonzero data cells may occupy a large amount of memory. For standard decomposition wavelets, thresholding between each decomposition is required unless numerous I/O between memory and disks are performed. For nonstandard decompositions, thresholding is also required after a chunk is decomposed. In contrast, the DCT based technique can take advantage of zonal sampling when processing OP-cubes. Thus, the required working buffer is not larger than the number of retained coefficients, and the approach can work in a single pass over the input.

1.2 Our contributions

In this paper we propose a framework of DCT for dAta With error estimatioN, called DAWN, which focuses on answering range-sum queries from compressed OP-cubes transformed by DCTs. DAWN possesses several

advantages. Our contributions are as follows:

1. We fully explore the essence of the errors in answering queries due to coefficient sampling, and prove the theoretical foundation for the goodness of the reciprocal zonal sampling technique.
2. We devise a robust summation function, called the GE function, to answer range queries in constant time, regardless of the number of data cells involved. The GE function processes the transformation steps in DCT more precisely than previous work; thus the quality of resulting answers is improved.
3. An estimator of errors based on the Brown noise assumption (BNA) is devised to provide tight bounds for answering range-sum queries and cost constant additional storage space only. Although the error bound estimated by BNA is not deterministic, as will be empirically validated later, the errors estimated by BNA are in essence very close to real errors. Also, BNA is able to minimize the resources required for approximating answers within a given acceptable error rate, in order to improve the query response time.

Extensive experimental studies are conducted to provide many insights into DAWN. A real dataset containing transaction data from a major bookstore and a synthetic dataset from TPCH is utilized to evaluate the merits of DAWN. As shown by our experimental results, the DAWN framework is scalable to the query selectivity and the available storage space. Also, the DAWN is shown to deliver quality answers robustly. In addition, the comparison among estimators shows that the BNA method outperforms other approaches by providing much tighter bounds in orders of magnitude. Moreover, quality analysis of the relationship between actual errors and estimated errors has been conducted to show the good error estimation capability of DAWN. With GE functions and the BNA method, the framework DAWN not only delivers answers of high quality for range-sum queries but also leads to shorter query response time due to its effectiveness in error estimation.

The remainder of this paper is organized as follows. Preliminaries are presented in Sect. 2. The DAWN framework is proposed in Sect. 3. Several error estimators, including the BNA method are devised in Sect. 4. Empirical studies are conducted in Sect. 5. Finally, in Sect. 6, we present our conclusions.

2 Preliminaries

2.1 Multidimensional data cubes

The multidimensional view of data usually consists of categorical attributes, such as products and stores, to form the dimensions; and numeric attributes, such as sales and revenue, to form the measures. In addition to categorical attributes, numeric ones can be incorporated into data cubes as dimensions by dividing the numeric attributes into proper categories. Let $D = \{D_1, D_2, \ldots, D_d\}$ denote the set of dimensions of a data cube $\mathcal{C}$, and $|D_i|$ denote the number of distinct values in dimension D_i. Without loss of generality, we assume that each dimension D_i has an index domain $\{0, 1, \ldots, |D_i| - 1\}$. The number of cells needed to store all information of $\mathcal{C}$ can be obtained by $N_A = \prod_{i=1}^{d} |D_i|$.

The measures are aggregated to various dimension levels of attributes using functions such as sum, average, count and variance. OLAP queries are the aggregation queries that aggregate the measures of subcubes. An important class of aggregation queries is the range-sum query which applies SUM operations over a set of continuous data cells [23]. The problem of computing a range-sum query in a d-dimensional data cube can be formulated as follows:

$$\text{Sum}(l_1 : h_1, \ldots, l_d : h_d) = \sum_{i_1=l_1}^{h_1} \cdots \sum_{i_d=l_d}^{h_d} D[i_1, \ldots i_d],$$

where $D[i_1, \ldots i_d]$ is the value of the cell whose dimension-index is i_j for dimension D_j.

2.2 DCT for multidimensional cubes

Discrete cosine transforms are employed in image and signal processing areas because of their good performance of compressing information.

For a series of data $\overrightarrow{f} = (f(0), f(1), \ldots f(N-1))$, the DCT coefficients, $\overrightarrow{F} = (F(0), F(1), \ldots F(N-1))$, are defined as

$$F(u) = \sqrt{\frac{2}{N}} c_u \sum_{n=0}^{N-1} f(n) \cos\left(\frac{(2n+1)u\pi}{2N}\right),$$

$$\text{where } c_u = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u = 0 \\ 1 & \text{for } u \neq 0 \end{cases}.$$

$\overrightarrow{f}$ is recovered by the inverse DCT defined as

$$f(n) = \sqrt{\frac{2}{N}} \left[\sum_{u=0}^{N-1} c_u F(u) \cos\left(\frac{(2n+1)u\pi}{2N}\right) \right].$$

The process of computing the inverse is referred to as *reconstruction*.

We next generalize the above to a d-dimensional DCT recursively. Let f be $N_1 \times N_2 \times \cdots \times N_d$ d-dimensional

data and F the corresponding transformed coefficient set. The d-dimensional DCT for a coefficient $F_{\overrightarrow{u}}$ located at $(u_1, u_2, \ldots u_d)$ is

$$F_{\overrightarrow{u}} = \sqrt{\frac{2^d}{\prod_{i=1}^{d} N_i}} \sum_{m_1=0}^{N_1-1} \sum_{m_2=0}^{N_2-1} \cdots \sum_{m_d=0}^{N_d-1} \widehat{F},$$

where $\widehat{F} = f_{(m_1,m_2\ldots,m_d)} \cdot \prod_{j=1}^{d} c_{u_j} \cos\left(\frac{(2m_j+1)u_j\pi}{2N_j}\right)$, and the inverse-DCT for the data point $f_{\overrightarrow{m}}$ located at $(m_1, m_2, \ldots m_d)$ is

$$f_{\overrightarrow{m}} = \sqrt{\frac{2^d}{\prod_{i=1}^{d} N_i}} \sum_{u_1=0}^{N_1-1} \sum_{u_2=0}^{N_2-1} \cdots \sum_{u_d=0}^{N_d-1} \widehat{f},$$

$$where\ \widehat{f} = \left[F_{(u_1,u_2\ldots,u_d)} \cdot \prod_{j=1}^{d} c_{m_j} \cos\left(\frac{(2m_j+1)u_j\pi}{2N_j}\right)\right].$$

For interested readers, the transformation of Table 1 to its DCT form is given in Appendix A. The number of DCT coefficients is the same as the number of data cells in the original data cube. However, in order to store the information efficiently, only some of the coefficients are kept for further processing. The set of selected coefficients is called the *transformed coefficient set* (TCS), and the number of coefficients in the set is denoted by Size(TCS). Also, the *unselected coefficient set*, which will not be transformed, is abbreviated as UCS. With respect to a d-dimensional data cube, $\mathcal{C}$, a DCT, $\mathcal{D}$, can be viewed as a transformation of cells in $\mathcal{C}$ to a TCS and a UCS respectively, where Size(TCS)+ Size(UCS)=Size($\mathcal{C}$). Note that items (coefficients) in UCS are not actually computed, i.e., their values are unknown after the transformation. In general, an efficient TCS tends to comprise low frequency coefficients, which means that high frequency coefficients are usually discarded. The theoretical properties of the techniques of coefficient selection will be derived in Sect. 3.

The most important property of the DCT is that it reduces the correlation among transformed coefficients. If adjacent sets of data in the data distribution are highly correlated, then the DCT reduces the correlation between adjacent transformed coefficients. Explicitly, if the frequency spectrum of data distribution is skewed so that the magnitudes of low frequency coefficients are large and those of high frequency coefficients are small, we can discard the latter without affecting the original data distribution significantly [13]. As mentioned in Sect. 1, such scenarios occur frequently in real-world databases. For example, sales of a product in each month of a year, regarded as OP-cubes, vary smoothly. As a result, a DCT for data works very well for OP-cubes.

3 The DAWN framework

The objective of the DAWN framework is to deliver high quality answers for range-sum queries and provide good estimators for errors. We first examine the effect of coefficient selection on the quality of reconstruction. Note that, as shown in [15], when the error rates are obtained by aggregating those of individual cells, DCT-based approaches do not perform well. However, as will be shown later, DAWN remedies this drawback by evaluating the error rates with respect to the range-sum queries. Specifically, we will show that the errors of answers for range-sum queries due to discarding of high frequency coefficients are predictable and relatively insignificant; thus, the query answers are of high quality.

In order to answer OLAP queries within a small amount of memory space, the answers can be estimated approximately by using an appropriate TCS. The fewer the DCT coefficients included in the TCS, the smaller the memory space required. However, the error rate will increase if some coefficients are discarded. The trade-off between space and quality can be balanced by proper coefficient selection. Therefore, for those algorithms based on compression mechanisms, the effectiveness of coefficient selection techniques indeed determines the quality of reconstructed information. In particular, the DCT for data usually performs coefficient selection spatially, since the cost of the decomposition step is expensive and it is infeasible to compute all coefficients for selection. Thus, the coefficient selection in the DCT for data can also be viewed as coefficient sampling. This question of the goodness of a TCS is analogous to that of energy preservation in signal processing. That is, the efficiency in energy preservation is regarded as the ability to reconstruct the original information and the most important task of the DCT for data is to develop an efficient mechanism for coefficient sampling. Consequently, reciprocal zonal sampling, which will be proved to be the most efficient approach for OP-cubes, is employed in this paper.

In Sect. 3.1, the essence of errors caused by coefficient sampling is introduced first. The differences between the errors of individual cells and those of range-sum queries are then examined. An efficient coefficient selection technique and the proof of its optimality are presented in Sect. 3.2. A fast approximation technique for range-sum queries is devised in Section 3.3, and three different error estimation techniques are proposed in Sect. 4

to provide different levels of estimated error rates for answers to OLAP queries.

3.1 Transformation errors

Since the TCS does not include all the DCT coefficients of a data cube, errors occur in the transformation and reconstruction process. An error is often measured as a mean squared error (MSE).

$$\text{MSE} = \frac{1}{N} \sum_{i_1,i_2,\ldots,i_d} (D[i_1, i_2, \ldots, i_d] - f[i_1, i_2, \ldots, i_d])^2,$$

where N is the number of cells and $f[i_1, i_2, \ldots, i_d]$ is computed by applying the inverse DCT with a TCS, i.e., information is reconstructed by truncated DCT coefficients. To estimate the error due to coefficient sampling, the case of an error, $E(u)$, corresponding to $F(u)$ is discussed first. Consider the case of one-dimensional data sequence of size N. The estimated value of a data point can be represented as

$$\begin{aligned} f(m) &= \sqrt{\frac{2}{N}} \sum_{u=0}^{N-1} c_u [F(u) + E(u)] \cos\left(\frac{(2m+1)u\pi}{2N}\right) \\ &= D(m) + \sqrt{\frac{2}{N}} E(u) c_u \cos\left(\frac{(2m+1)u\pi}{2N}\right), \end{aligned} \tag{1}$$

where m and u denote the indices for the original data sequence and transformed coefficients, respectively.

Thus, the maximum error of $D(m) - f(m)$ due to $E(u)$ can be represented as

$$\mathcal{E} = \sqrt{\frac{2}{N}} |E(u)| \underset{m=0\ldots N-1}{Max} |Z(m)|, \tag{2}$$

$$\text{where } Z(m) = \begin{cases} \cos(\frac{1}{4}\pi), & u = 0 \\ \cos\left(\frac{(2m+1)u\pi}{2N}\right), u \neq 0 \end{cases}. \tag{3}$$

Without loss of generality, we assume that the number of dimensions is d and the size of dimension D_i is $|D_i|$. The maximum transformation error due to discarding the coefficient at u_p can be obtained by

$$\mathcal{E} = |E(u_p)| \times \sqrt{\left(\frac{2}{\prod_{j=1}^{d} |D_j|}\right)^d} \prod_{j=1}^{d} \cos\left(\frac{u_{pj}\pi}{2|D_j|}\right), \tag{4}$$

where u_{pj} is the index in j-th dimension of u_p.

The error of range-sum queries due to coefficient sampling is expected to be the sum of the errors of individual cells involved in the range-sum. The simplest way to estimate the error of range-sum queries is to sum up the estimated errors of all cells computed by Eq. (4). However, since not all the cells suffer the worst case, the accuracy of a range-sum query should be estimated by a set of adjacent cells, rather than the sum of the individual cells. From Eq. (1), it can be seen that discarding a coefficient will affect those cells located around $\frac{2t+1}{2}\pi$ in different directions. In general, for a range-sum query, $Q_s = \text{SUM}(i_l, i_h)$, in an N-point data source, the transformation error of Q_s due to discarding of a coefficient, F_k can be estimated as $\sum_{i=i_l}^{i_h} \sqrt{\frac{2}{N}} F_k c_k \cdot \cos\left(\frac{(2i+1)k\pi}{2N}\right)$ by single cell error estimation. If $(i_h - i_l) \cdot k > N$, there must exist some cell pairs whose errors are compensated for by each other. As a result, the overall error for range-sum queries is reduced significantly. In the optimal case, if the cells in Q_s are distributed evenly over different sides of f_i, where $\frac{(2i+1)k}{2N} = \frac{2t+1}{2}$ and t is a positive integer, all errors can be completely eliminated. Also, it can be seen that for a larger k, these conditions for elimination are more likely to hold. Since the parameter, k, is proportional to the frequency of coefficients, it follows that transformation errors of range-sum queries due to discarding high frequency coefficients tend to be smaller than those due to discarding low frequency coefficients. In addition, the errors corresponding to discarding high frequency coefficients are smaller, in general, since those coefficients contain less energy.

3.2 Coefficient selection

The number of DCT coefficients increases exponentially as the dimensionality increases. Computing all coefficients for possible selection is computationally prohibitive; thus, coefficient sampling is needed. In terms of energy preservation, coefficients with larger absolute values are preferred. In practical OLAP systems, the data distribution usually has a correlation among data items, i.e., the frequency spectrum of the distribution contains large values in its low frequency coefficients and small values in its high frequency coefficients [19, 13]. In general, high frequency coefficients are usually of less interest in OLAP, but low frequency coefficients are of interest since they correspond to range-aggregation queries [9].

To select an appropriate TCS, several geometrical zonal sampling techniques, such as triangular, spherical, rectangular and the reciprocal sampling, have been proposed [14]. The work in [13] extends those techniques to multidimensional datasets. Each of the sampling techniques selects a TCS, which is expected to be efficient to retain, and discards all other coefficients. From the view point of signal processing, the loss of energy during the compression process results in the loss of details of original signals. Thus, the efficiency of energy preservation of compressed signals can be deemed as the capability of providing good answers of quality from reconstructed

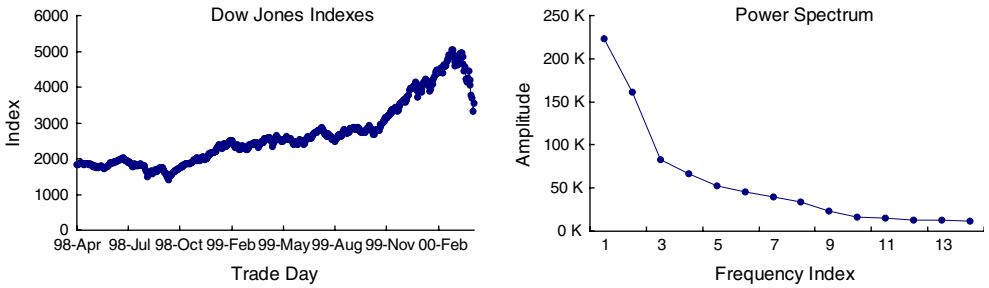

Fig. 1 The *left* figure illustrates the sequence for Dow Jones indices during 500 trading days and the *right* figure shows the corresponding energy spectrum

signals. Since the sum of total errors caused by discarding of all the DCT coefficients of a given data cube is a constant value, the efficiency of a TCS can be evaluated as the function of its coefficients.

Definition 1 *With a TCS* $\mathcal{T} = \{k_{u_{11},\ldots,u_{1d}}, k_{u_{21},\ldots,u_{2d}}, \ldots, k_{u_{n1},\ldots,u_{nd}}\}$ *in a d-dimensional data cube, the efficiency of* $\mathcal{T}$ *is defined as* $H(\mathcal{T},n)=\sum_{i=0}^{n-1}(k_{u_{i1},\ldots,u_{id}})^2$, *where n is the size of* $\mathcal{T}$ *and* u_{ij} *is the index of i-th coefficient for dimension j.*

For ease of presentation, $k_{u_{i1},u_{i2},\ldots,u_{id}}$ in the compressed cube is denoted as k_{p_i}.

Theorem 1 *For two TCSs,* T_1 *and* T_2, *of the same size in a d-dimensional data cube,* T_1 *causes a smaller error due to coefficient sampling if* $H(T_1,n) > H(T_2,n)$.

Proof From the derivation in Sect. 3.1, the MSE due to discarding coefficient k_{p_i} can be estimated as

$$\frac{\sum_{m_1=0}^{N-1}\sum_{m_2=0}^{N-1}\cdots\sum_{m_d=0}^{N-1}\left(k_{p_i}\prod_{j=1}^{d}c_{ij}\cos\left(\frac{(2m_j+1)u_{ij}\pi}{2N}\right)\right)^2}{N^d}$$

$$= \frac{k_{p_i}^2}{N^d}\sum_{m_1=0}^{N-1}\sum_{m_2=0}^{N-1}\cdots\sum_{m_d=0}^{N-1}\left(\prod_{j=1}^{d}c_{ij}\cos\left(\frac{(2m_j+1)u_{ij}\pi}{2N}\right)\right)^2$$

$$= \frac{k_{p_i}^2}{N^d}\prod_{j=1}^{d}\left(\sum_{i=0}^{N-1}c_{ij}\cos^2\left(\frac{(2m_i+1)u_{ij}}{2N}\right)\right)$$

$$= k_{p_i}^2\frac{\left(\frac{N}{2}\right)^d}{N^d} = K(k_{p_i})^2, \text{ where } K=\left(\frac{1}{2}\right)^d.$$

Let MSE due to the discarding of all coefficients be $\text{MSE}_{\text{total}}$. We then have $\text{MSE}_{T_1} = \text{MSE}_{\text{total}} - K\sum_{i=0}^{n-1}(k_{p_i}^1)^2 = \text{MSE}_{\text{total}} - H(T_1,n)$ and $\text{MSE}_{T_2} = \text{MSE}_{\text{total}} - K\sum_{j=0}^{n-1}(k_{p_j}^2)^2 = \text{MSE}_{\text{total}} - H(T_2,n)$, where $k_{p_i}^1 \in T_1$ and $k_{p_j}^2 \in T_2$. It follows that $\text{MSE}_{T1} < \text{MSE}_{T_2}$ if $H(T_1,n) > H(T_2,n)$. □

In order to select an efficient TCS, *reciprocal zonal sampling,* which selects coefficients such that the multiplication of indices is equal to or less than a given value b, is employed. Coefficients we selected by the constraint $\{k_{p_i} | \prod_{j=1}^{d}(u_{ij}+1) \le b\}$, where $u_{ij} = 0,\ldots,|D_j|-1$ and $|D_j|$ is the number of members in the j-th dimension. Note that this technique is able to select coefficients layer by layer, and the set of coefficients, $\{k_{p_i} | \prod_{j=1}^{d}(u_{ij}+1) = b\}$, is called *a layer of degree b*. This feature offers the capabilities of parallel transformation and progressive reconstruction. To our knowledge, the theoretical aspects of the reciprocal zonal sampling have not been explored before. By the definition of reciprocal zonal sampling, low-frequency coefficients in each dimension are likely to be selected. To determine the effectiveness of reciprocal zonal sampling, the distribution of DCT coefficients should be investigated.

Note that real signals have a skewed energy spectrum [19]. For example, stock movements and exchange rates can be successfully modeled as random walks (also known as brown noise) which exhibit an energy spectrum, $O(f^{-2})$. Therefore, the DCT coefficients of such data, referred to as amplitudes of the signals, can be modeled as $O(f^{-1})$[19], i.e., the amplitude is inversely proportional to the frequency. Note that DAWN focuses on OP-cubes and the brown noise assumption is hence appropriate for modeling the distribution of cubes. A scenario of the brown noise assumption for OP-cubes is shown in Fig. 1, where it can be seen that the amplitudes are inversely approximately proportional to the frequencies.

Consequently, we formally prove by Theorem 2 that, based on the assumption of brown noise distribution, reciprocal zonal sampling is in fact the most efficient method for coefficient selection.

Theorem 2 *For a symmetric d-dimensional data cube with brown noise distribution, the most efficient* TCS *is* $T = \{k_{p_i} | \prod_{j=1}^{d}(u_{ij}+1) \le b\}$, *where b is a constant.*

Proof Without loss of generality, we assume that the data cube is two-dimensional. Therefore, the DCT coefficients can be modeled as $k_{(m,n)} = \frac{K}{(m+1)(n+1)}$, where K is a positive constant, and m and n represent

the frequencies of x-axis and y-axis respectively. Given that $k_{(x,y)} \in T=\{k_{(m,n)}|(m+1)(n+1) \leq b\}$, it follows that $(x+1)(y+1) = b$ holds for the smallest coefficient. From Theorem 1, if there exists another TCS, T', which causes $H(T',n) > H(T,n)$, there must exist a coefficient $k_{(x,y')} \in T'$ where $\frac{K}{(x+1)(y'+1)} > \frac{K}{(x+1)(y+1)}$. This implies that $(x+1)(y'+1) < b$, which is a contradiction. Therefore, the theorem follows. □

Actually, reciprocal zonal sampling works well for asymmetric dimensions as well as for symmetric ones under the brown noise assumption of data distribution. In the case of asymmetric data cubes, we can multiply the index for each dimension of a DCT coefficient by a scaling factor in order to make all the indices proportional to the corresponding frequencies. The relationship between the frequency and cell-index can be expressed as $f_{ij} = \frac{u_{ij}+1}{|D_j|}$. Asymmetric reciprocal zonal sampling is defined in Corollary 2.1 below.

Corollary 2.1 *For an asymmetric d-dimensional data cube with brown noise distribution, the most efficient TCS of the data cube is* $\{k_{p_i}|\prod_{j=1}^{d}(u_{ij}+1) \leq b'\}$, *where* b' *is a constant.*

3.3 Answering range-sum queries

To compute the answers of range-sum queries by a TCS, one can intuitively use single-cell estimation, i.e., we reconstruct the values of each cell requested by the query and compute the summation. However, as pointed out earlier, this approach incurs a very high computational overhead. For a range-sum query, which sums up n data cells, it costs $O(n \cdot n_T)$ time to compute the result with a TCS whose size is n_T. Also, the answers to queries provided by DCT-based approaches consist of the summations of cosine functions. Therefore, the accuracy of the estimated summations is the key factor in the quality of the answers. To compute the summations of cosine functions efficiently, the authors in [13] introduced an integral-based approach to approximate the summations. However, this approach suffers from the problem of miscalculation. In the case of small dimensions, as shown by the following example, the non-linearity of the cosine function could cause the results of an integral to be incorrect.

Example 2 Consider the following two functions:

$$f_1(x) = \sum_{x=l}^{h} \cos\left(\frac{(2x+1)u\pi}{2N}\right),$$

$$f_2(x) = \int_{l}^{h} \cos\left(\frac{(2x+1)u\pi}{2N}\right)\mathrm{d}x.$$

In the case of $N = 1,024$, $u = 10$ and $(l{:}h) = (3:10)$, the results of $f_1(x)$ and $f_2(x)$ can be evaluated as 7.7969 and 7.7966 respectively, i.e., $f_2(x)$ approximates $f_1(x)$ well. However, in the case of $N = 15$, the results of $f_1(x)$ and $f_2(x)$ can be evaluated as -0.5 and -0.414. Therefore, the integral approach does not work well when the dimension size is small. □

To compute the sum of a series of cosine functions precisely and efficiently, we have designed a summation function based on the techniques of geometric series and the Euler's formula, referred to as the GE function. The GE function, which offers the fundamentals of query answering and error estimation, is able to produce good estimations for summations in constant time.

Lemma 1 *The value of* $\sum_{x=l}^{h}\cos(\frac{(2x+1)u\pi}{2M})$ *can be computed in constant time by the GE function:*

$$\mathrm{GE}(l,h,u,M) = \begin{cases} \mathrm{Re}\left(\frac{\left(e(l)+\frac{1}{e(h)}\right)\left(e(\frac{2h-2l+1}{2})-1\right)}{2e(\frac{1}{2})-2}\right) & \text{when } u \neq 0 \\ h-l+1 & \text{when } u = 0, \end{cases}$$

where Re*(.) means the real part and* $e(t){=}e^{i(\frac{(2t+1)u\pi}{2M})}$.

As will be validated by our experimental results later, the GE function improves the quality of answers significantly, since the summations are precise. From Lemma 1, the part of the answer for a range-sum query contributed by one coefficient can be computed in $O(d)$. Consider the case of answering the range-sum query $Q = \mathrm{SUM}(l_1 : h_1, l_2 : h_2, \ldots, l_d : h_d)$ by the TCS $T = \{k_{p_1}, k_{p_2}, \ldots k_{p_n}\}$ in a d-dimensional $(|D_1| \times |D_2| \times \ldots \times |D_d|)$ data cube. The answer to Q can be estimated by

$$R_Q = \sqrt{\frac{2^d}{\prod_{i=1}^{d}|D_i|}}\sum_{k_{p_i}\in T}\left(k_{p_i}\prod_{j=1}^{d}c_{u_{ij}}\mathrm{GE}(l_j,h_j,u_{ij},|D_j|)\right).$$

For ease of presentation, this equation is rewritten as

$$R_Q = \mathcal{K}\sum_{k_{p_i}\in T}k_{p_i}\mathcal{E}(p_i), \tag{5}$$

where $\mathcal{K}$ is a constant for the same Q, and $\mathcal{E}(p_i)$, referred to as the *error function*, is the product of GE functions along all dimensions.

By applying the GE function, the time complexity can be reduced to $O(n_T d)$, where n_T is the size of TCS. This derives a significantly better result than those of single-cell estimation and the partial sum approach.

Note that an important characteristic of a DCT for data is that the time required to transform a cube is only of complexity $O(n \cdot n_T)$, where n is the number of tuples in the original cube. For OP-cubes in practical OLAP systems, it is shown that the cube data space is extremely sparse [20], implying that n is much smaller than the cardinality of the data cube. Therefore, it is feasible to transform more coefficients in the working buffer by zonal sampling, and choose those with the largest n_T absolute values as the final selection for the TCS. This hybrid method, implemented in our experiments later, is called the *buffered DCT for data*. In previous DCT-based approaches was called the *unbuffered DCT*.

4 Error estimation of DAWN

Another important advantage of DAWN is the error estimation, which is a very difficult task in other compression based approaches. Estimating the errors of range-sum queries is the same as estimating the answers contributed by the unselected coefficient set (UCS). In other words, the correct value of a data cell equals the sum of the answers computed by applying Eq. (5) to TCS and UCS. The error estimation problem can be regarded as finding a good estimator for the errors of answers due to discarding UCS. Hence, Eq. (5) can be rewritten as follows:

$$E_q = \sum\nolimits_{k_{p_i} \in T} \left(\mathcal{K} k_{p_i} \cdot \mathcal{E}(p_i) \right). \tag{6}$$

From Eq. (6), the calculation requires the understanding of all unselected coefficients in the UCS, which is infeasible in practice since the unselected coefficients are never computed. In order to estimate the error of discarding the UCS, the equation presenting the error can be divided into two parts to simplify the estimation. The first part, referred to as the k-part, is derived from the value of the coefficient and is independent of queries. The second part, referred to as the q-part, is generated by the product of GE functions and varies from query to query. For example, $\mathcal{K} \cdot k_{p_i}$ in Eq. (6) is the k-part and $\mathcal{E}(p_i)$ forms the q-part. For ease of presentation, the set of k-parts of coefficients in UCS is denoted as a vector, $\overrightarrow{k}$, and that of the q-parts is denoted as $\overrightarrow{q}$. Thus, the problem of finding an error estimator for answering queries can be transformed to the one of finding a good estimator for the value of the inner-products of $\overrightarrow{k}$ and $\overrightarrow{q}$ with respect to the UCS.

Example 3 Let D be a two-dimensional (3×3) data cube. D and its DCT coefficients are shown in Table 3 a and Table 3 b respectively.

Table 3 D and its DCT coefficients

10	15	13
14	**20**	**16**
9	13	11

(a)

DCT $\longrightarrow$

40.33	-2.858	-5.421
2.041	-0.5	-0.289
-6.835	-0.289	1.167

(b)

Given a TCS= $\{k_{0,0}, k_{0,1}, k_{1,0}, k_{0,2}, k_{1,1}, k_{1,2}\}$, UCS= $\{k_{2,0}, k_{2,1}, k_{2,2}\}$, and a user-submitted query Q = SUM (1:1, 1:2) (marked in bold font in left table), the k-part of $k_{2,0}$ can be calculated as $\sqrt{\frac{4}{3 \cdot 3}} \cdot (-6.835)$ and the q-part can be estimated as $c_2 c_0 \cdot \mathrm{GE}(1,1,2,3) \cdot \mathrm{GE}(1,2,0,3) = -\sqrt{2}$. Thus, the error of answering Q due to discarding $k_{2,0}$ is -6.444, and the total error due to discarding the UCS can be obtained by

$$\mathrm{Error}_{\mathrm{total}} = \overrightarrow{k} \cdot \overrightarrow{q} = \begin{bmatrix} \overline{k}_{(2,0)} & \overline{k}_{(2,1)} & \overline{k}_{(2,2)} \end{bmatrix} \begin{bmatrix} q_{(2,0)} \\ q_{(2,1)} \\ q_{(2,2)} \end{bmatrix}$$

$= -6.444 + 0.167 - 0.389 = -6.67$, where $\overline{k}_{(i,j)}$ is the magnitude of $k_{i,j}$. □

Note that two major problems make the estimation of the total error difficult. The first is that the details of $\overline{k}_{(i,j)}$ are unknown due to the definition of UCS, i.e., not all unselected coefficients are actually calculated. The second is the inefficacy of calculating $\overrightarrow{q}$ since UCS is very large. In what follows, to provide an estimation of the errors due to discarding the UCS, we present three different approaches, namely, the Basic approach, the LM approach and the BNA method, to estimate $\overrightarrow{k}$ and $\overrightarrow{q}$ under different assumptions on the coefficient distribution.

4.1 The basic approach

To estimate the error bound of answers with basic information, we describe a basic error estimation first. Based on the extension of Eq. (4), the worst case of the q-part can be estimated as

$\mathcal{E}_{\mathrm{Max}}(u_p) = \sqrt{\frac{2^d}{\prod_{i=1}^{d} |D_i|}} \cdot \prod_{j=1}^{d} \left(\cos\left(\frac{u_{pj}\pi}{2|D_j|} \right) \right) \cdot N_c$, where N_c is the number of data cells involved in the query.

Note that $\mathcal{E}_{\mathrm{Max}}$ is a constant for a given query and is independent of the positions of coefficients in UCS. By substituting $\mathcal{E}_{\mathrm{Max}}$ for items in $\overrightarrow{q}$, the problem is reduced to the one of estimating the worst case of $\mathcal{E}_{\mathrm{Max}} \cdot \sum \overline{k}_{(i,j)}$. In order to sketch the distribution of the items in $\overrightarrow{k}$, Parseval's theorem offers an important inspection.

Theorem [Parseval] *Let* $\overrightarrow{F}$ *be the DCT of the sequence* $\overrightarrow{f}$, *we have*

$$\sum_{t=0}^{n-1} |f_t|^2 = \sum_{t=0}^{n-1} |F_t|^2 .$$

That is, the energy in the time domain is the same as the energy in the frequency domain.

By Parseval's Theorem, the total energy of UCS can be estimated from the energy of the original data cells, i.e., $\sum \overline{k}^2_{(i,j)}$ can be calculated after a TCS is generated. However, the distribution of individual energy is still unknown. By applying some algebraic manipulations, $\sum \overline{k}_{(i,j)}$ reaches its maximum when all coefficients are equal and $\sum \overline{k}^2_{(i,j)}$ is a constant, i.e., the energy is distributed evenly over all unselected coefficients. Thus, the Basic approach for error estimation is formulated as

$$\text{Error}_{\text{total}} = \sqrt{\frac{H(\text{All}, n) - H(TCS, n_T)}{\text{Size(UCS)}}} \cdot \text{Size(UCS)} \cdot \mathcal{E}_{\text{Max}},$$

where n and n_T denote the total number of coefficients and size of the TCS respectively.

Example 4 Consider the two-dimensional cube in Example 3. By Parseval's Theorem, we have

$$\sum_{i=0}^{2} k^2_{2,i} = 48.2 \quad \text{and} \quad \mathcal{E}_{\text{Max}} = \sqrt{\frac{2^2}{3 \cdot 3}} \cdot \cos^2\left(\frac{\pi}{6}\right) \cdot 2 = 1.$$

Therefore, $\sum \overline{k}_{(i,j)} = \sqrt{\frac{48.2}{3}} \cdot 3 = 12.025$ and $|\text{Error}_{\text{total}}| = 12.025$ in the worst case. □

This method considers the worst cases for both the k-part and the q-part of the $\text{Error}_{\text{total}}$ and provides guaranteed error bounds of query answers. However, as shown in Example 4, the bound is very loose since the worst cases are far from their true values.

4.2 The Lagrange multiplier approach

Unlike the Basic approach, the Lagrange multiplier (LM) approach can be applied to find the bound of the error estimation problem. In general, the more constraints there are on the variables, the tighter bound the LM approach can achieve. However, there are two major considerations that hinder us in formulating more constraints. The first is the difficulty in solving equations as the number of equations grows. The second is the lack of information about the distribution of items in $\overrightarrow{k}$ to generate constraints. Thus, only four statistics, i.e., (1) those aggregated from the sum of items in $\overrightarrow{k}$ (2) the sum of the squares of items in $\overrightarrow{k}$ and (3) and (4) those two of $\overrightarrow{q}$, are utilized to be constraint equations for LM, i.e., four Lagrange multipliers are applied. For a UCS in which the k-part $\overrightarrow{k} = \{k_i d_i | 1 \le i \le n_u, k_i \ge 0, d_i = \pm 1\}$ and the q-part $\overrightarrow{q} = \{C_i | 1 \le i \le n_u\}$, there are $2n_U + 4$ constraint equations. Let $\text{Error}_{\text{total}} = \sum_{i=1}^{n_U} k_i d_i C_i$ be the last equation. The solutions for $\text{Error}_{\text{total}}$ can thus be estimated as

$$\text{Error}_{\text{total}} = \frac{\text{SK} \cdot \text{SC} \pm \sqrt{\left(-n \cdot \text{SSK} + \text{SK}^2\right)\left(-n \cdot \text{SSC} + \text{SC}^2\right)}}{n_U}, \quad (7)$$

where $\text{SC} = \sum_{i=1}^{n_U} C_i$, $\text{SSC} = \sum_{i=1}^{n_U} C_i^2$, $\text{SK} = \sum_{i=1}^{n_U} k_i$ and $\text{SSK} = \sum_{i=1}^{n_U} k_i^2$.

Example 5 From the UCS and the query in Example 3, we get $\text{SK} = -5.957$, $\text{SSK} = 48.1626$ and $\overrightarrow{q} = \{-1.414, 0.866, 0.5\}$. $\text{Error}_{\text{total}}$ can then be estimated as $\text{Max}\{|-10.343|, |10.535|\} = 10.535$. □

It can be seen from Example 5 that the bound of $\text{Error}_{\text{total}}$ is much better than that of the Basic approach as the assumptions of the k-parts and the q-parts in the LM approach are more optimistic than those in the Basic approach. However, SK for the UCS is unavailable. Also, the cost of aggregating SSC and SC, estimated as $O(n_U)$, is expensive if the value of each C_i is computed individually.

4.2.1 Fast Computing for SC and SSC

In order to compute the SC of the UCS, denoted as SC_U, efficiently, the information about the SC of the TCS, denoted as SC_T, and the SC of all coefficients, denoted as SC_A, can be utilized. Note that it is feasible to estimate SC_T by summing all items, since $n_T \ll n_U$. The fast computing methods for SC and SSC are based on the following lemma:

Lemma 2 *For the SC_A, SC_U, and SC_T of a DCT, $\mathcal{D}$, and a query $Q = SUM(l_1{:}h_1, \ldots, l_d{:}h_d)$ corresponding to a d-dimensional data cube $\mathcal{C}$, the sum of items (SC_U) and the sum of squares of items (SSC_U) in the q-parts can be computed in $O(n_T d)$ time, where n_T is the size of corresponding TCS.*

4.2.2 Fast Computing for SK

The computation of SK and SC differs in tow major respects. The first is that information about items of the k-parts in the UCS is unavailable for a compressed cube. The second is that SK is independent of submitted queries, i.e., this value needs to be computed only once for the same UCS of a DCT corresponding to a data cube. The purpose of computing SK of the UCS efficiently is to collect information about the sum of coefficient values for the TCS during the transformation process. This

approximates the sum of all coefficient values so that the SK of the UCS can be estimated efficiently. For ease of presentation, the subscripts of SK are the same as those of SC.

Lemma 3 *For the SK_A, SK_U, and SK_T of a DCT, $\mathcal{D}$, corresponding to a d-dimensional data cube, $\mathcal{C}$, the cost of estimating SK_U is the same as that of computing a single coefficient during the transformation process.*

From Lemma 2 and Lemma 3 the SK_U, being unchanged for a compressed cube, and SC_U and SSC_U, varying with queries, can be estimated in $O(n_T d)$ time.

4.3 The BNA approach

Although the LM method is able to attain a better error bound than the Basic approach, errors could be overestimated when the sign of an item in the k-part is the same as that of the corresponding item in the q-part. The overestimation may be hundreds of times larger than the real error. Thus, to achieve a better estimator, we have developed a greedy method based on the BNA for error estimation. Based on an assumption about the distribution of k-parts, we apply some heuristics to estimate $\text{Error}_{\text{total}}$. The results of our experiments show that the heuristics improve the quality of the estimator significantly. By extending Eq. (6), the square of the total error can be formulated as

$$(Error_{\text{total}})^2 = \sum_{x=1}^{n_U} k_x^2 C_x^2 + 2 \sum_{i,j=1, i\neq j}^{n_U} k_i d_i C_i k_j d_j C_j. \quad (8)$$

Those two parts of Eq. (8) are handled separately, since the first can be estimated precisely under BNA, but the second part can only be estimated approximately.

4.3.1 Computing the sum of $k_x^2 C_x^2$

As mentioned in Sect. 3.2, the distribution of coefficients for OP-cubes can be modeled as brown noise; thus, the magnitude of coefficients can be modeled as $k_{p_i} = \prod_{i=1}^{d} \frac{C}{|(u_{ij}+1)|}$.

To satisfy both Parseval's Theorem and the BNA, the DC factor, i.e., the estimation of the coefficient whose frequency on all dimensions is zero must be computed. Assuming that the TCS, $\mathcal{T}$ and UCS, $\mathcal{U}$, of a d-dimensional data cube $\mathcal{C}$, are known, we denote the energy of $\mathcal{T}$, $\mathcal{U}$, and the total energy as $H_{\text{total}} = \sum_{f_{(p)} \in \mathcal{C}} |f_{(p)}|^2$, $H_T = \sum_{k_p \in T_1} |k_p|^2$ and $H_U = \sum_{k'_p \in T_2} |k'_p|^2$, respectively, where $p_i = (u_{i1}, u_{i2}, \ldots u_{id})$. From Parseval's Theorem, the DC factor, C_U, can be estimated as:

$$C_U = \sqrt{\frac{H_{\text{total}} - H_T}{\sum_i \frac{1}{\left|\prod_{j=1}^{d}(u_{ij}+1)\right|^2}}}. \quad (9)$$

The subscript U means that this factor is derived from the UCS. Also, the DC factor derived from the TCS, denoted as C_T, and the actual value of the first coefficient, C_0, in TCS must also be considered. The DC factor is important for error estimation because several calculations for statistics of the coefficients depend on it. Therefore, the C_m, estimated as the geometric mean of C_0, C_T and C_U, is considered as the estimator of the DC factor. Hereafter, the values of the k-parts are computed by Eq. (9). Note that the value of $\sum_{x=0}^{n_U-1} k_i^2 C_i^2$ can be computed in $O(n_T)$ time from Lemma 2 by substituting term $(\frac{\sqrt[d]{C_m}}{(u_{ij}+1)} G_{u_{ij}})^2$ for $G_{u_{ij}}$.

4.3.2 Computing the sum of $k_i d_i C_i k_j d_j C_j$

For the second part of Eq. (8), to approach the bound , we propose a heuristic based on an assumption about the distributions of $k_i k_j$ and $C_i C_j$. Let the second part of Eq. (8) be denoted as $\mathcal{S}$, and six subsets be defined as follows:

$$\begin{aligned} K_A &= \{k_i k_j | i, j \in \{1..n_U\}, i \neq j\} \\ C_A &= \{C_i C_j | i, j \in \{1..n_U\}, i \neq j\} \\ K_\pm &= \{k_i k_j | (k_i k_j \in K_A) \& (d_i \cdot d_j = \pm 1)\} \\ C_\pm &= \{C_i C_j | (C_i C_j \in C_A) \& (C_i \cdot C_j \pm 1)\} \end{aligned}, \text{ and } \langle Y_x \rangle$$

be defined as the sum of all terms in set Y_x.

Though the numbers of the positive and negative items in the UCS are not always the same, most transforms produce coefficients with sign bits that have an approximately equal probability of being 1 or 0, and being highly uncorrelated with the sign bits of neighboring coefficients [25]. Therefore, three assumptions can be made based on this property of the DCT: (1) the numbers of positive and negative coefficients will be close to each other and $\text{Size}(C_+)$ will be close to $\text{Size}(C_-)$; (2) $\text{Size}(S_{++}) \simeq \frac{1}{2}\text{Size}(K_+)$, where $S_{++} = \{k_i k_j C_i C_j | k_i k_j \in K_+, C_i C_j \in C_+\}$ and (3) $k_i k_j C_i C_j = k_i k_j \cdot \frac{\sum C_i C_j}{\text{Size}(S_{++})}$, where $k_i k_j C_i C_j \in K_+$. These estimators consider the moderate case in order to avoid both the worst and best case scenarios. Also, the other sets, S_{+-}, S_{-+}, and S_{--} can be defined similarly. From these assumptions, the second part of Eq. (8) can be rewritten as

$$\mathcal{S} = 2 \sum_{x,y \in \{+,-\}} \langle S_{xy} \rangle = \frac{4 \sum_{x,y \in \{+,-\}} \langle K_x \rangle \langle C_y \rangle}{n(n-1)}.$$

It is noted that for the summations of $\langle K_x \rangle$ and $\langle C_y \rangle$, it is not appropriate to derived them from the TCS and the UCS directly. Instead, an efficient method for this purpose is proposed as follows:

$$\langle K_{\pm} \rangle = \frac{\sum k_i k_j \pm \sum |k_i k_j|}{2} = \frac{(\mathrm{SK}_U)^2 + \left(\sum |k_i|\right)^2 - \mathrm{SSK} \mp \mathrm{SSK}}{4}$$

Further, $\sum_{i=1}^{n_U} |k_i|$ can be computed similarly from Lemma 3, and $\left(\sum |k_i|\right)^2$ can then be estimated efficiently. $\langle C_+ \rangle$ and $\langle C_- \rangle$ can be computed in a similar manner.

4.3.3 Computing the sum of two parts

To compute the accurate parts and approximate parts of Eq. (8), the extreme cases must be considered. The term inside the square brackets of the second part of the equation will be close to 0 if the positive and negative terms of $k_i k_j C_i C_j$ are distributed evenly. Extreme values of $(\mathrm{Error}_{\mathrm{total}})^2$ occur when $\mathcal{S} = 4\left[\frac{\langle K_+ \rangle \langle C_+ \rangle + \langle K_- \rangle \langle C_- \rangle}{n(n-1)}\right]$ or $\mathcal{S} = 4\left[\frac{\langle K_+ \rangle \langle C_- \rangle + \langle K_- \rangle \langle C_+ \rangle}{n(n-1)}\right]$, where the former is the maximum value and the latter is the minimum value. To make the estimation more reliable, the worst case, $\mathcal{S}_+$, is considered. Also, a parameter, $\beta = \frac{n_U}{N}$, should be added to adjust the effect of the size of the UCS, i.e., the fewer terms the UCS contains, the smaller the impact of the second part of Eq. (8).

Thus, based on the BNA approach, Eq. (8) can be rewritten as follows.

$$\mathrm{Error}_{\mathrm{total}} = \sqrt{\sum_{i=1}^{n} k_i^2 C_i^2 + 4\beta\left[\frac{\langle K_+ \rangle \langle C_+ \rangle + \langle K_- \rangle \langle C_- \rangle}{n(n-1)}\right]}.$$

Lemma 4 *The computational complexity of the BNA for estimating errors is the same as to that for answering a single query.*

Lemma 4 can be proven by inspecting the time cost for both first and second parts. Note that $\left(\sum |k_i|\right)^2$ can be computed during the transformation process and for further use. Thus, the computational complexity of the BNA is bounded in $O(n_T d)$.

In the BNA method, the estimator of errors in answers is better than those of the Basic and LM methods. Note that, although the error bound estimated by BNA is not deterministic, the errors estimated by BNA will be empirically verified to be very close to real errors, showing the very advantage of BNA.

5 Experimental results

To evaluate the framework of DAWN, three sets of empirical studies on both synthetic and real datasets were conducted. In Sect. 5.1, the scalability of DAWN in both query answering and error estimation is evaluated. In Sect. 5.2, the quality of answers to range-sum queries using DAWN is compared to that of other query approximating techniques. Moreover, the performance of the BNA is verified in Sect. 5.3. The major findings of our studies can summarized as follows:

- Improved quality of answers: By exploiting our GE functions, the quality/accuracy of the approximate answers is better than that of other approximating techniques for both the synthetic and real-world datasets.
- Better error bound estimation: In addition to better quality answers, DAWN provides a tight error bound for a given query, but only generates a constant additional space cost. The results show that the error bounds provided by DAWN are tighter than those of traditional mathematical models by several orders of magnitude.

5.1 Testbed and methodology

5.1.1 Datasets

The synthetic dataset, denoted as $\mathrm{D}_{\mathrm{TPC}}$, the source of which is decision support benchmark, TPC-H [1], consists of eight relations. Since DAWN focuses on OP-cubes, a derived table consisting of six dimensions (Suppliers, Nationkey, Orderstatus, Mkesegment, Mfgr and Brand) was combined with six relations to produce a cube with 169,665 tuples and 2,250,000 cells. Meanwhile, the real world OP-cube, denoted as $\mathrm{D}_{\mathrm{BOOK}}$, was obtained from the sales log of a large chain bookstore. Since the dataset $\mathrm{D}_{\mathrm{BOOK}}$ is not generated synthetically from the brown noise assumption, performance of DAWN on real world applications can hence be evaluated from those experiments based on $\mathrm{D}_{\mathrm{BOOK}}$. Four common dimensions were considered, and the cube was comprised of 89,100 cells and 29,125 tuples.

5.1.2 Techniques

Our experimental method compare several approximate query processing techniques with DAWN. Also, the performance and quality of error bounds estimation were evaluated. For the approximating techniques, sampling, histograms, wavelets, and DCT-based approaches were considered. Range-Sum queries were approximated by

a multidimensional MaxDiff(V,A) histogram. As the work in [11] had already shown that MaxDiff(V, A) histograms provide quality approximations for aggregate queries, we adopted the MaxDiff(V, A) technique. To implement the sampling technique, we random-sampled some cells from non-zero data points in the multidimensional datasets and the measures for cells were scaled appropriately. For example, if c cells were random-sampled from t non-zero cells, then the measure of every cell in the sample was multiplied by $\frac{t}{c}$. For wavelet-based approaches, we adopted the standard decomposition technique [24] and the improved algorithms for answering on-line queries [23]. Recall that the cardinality of data cubes in real world applications is usually huge and the working buffer for decomposition or transformation should be limited to improve the execution efficiency. Hence, a memory buffer $T_B = r_b \cdot N$ is allocated as temporary storage for required coefficients, where r_b is the buffer ratio and N is the total number of cells. The thresholding strategy in [23] is implemented. That is, the thresholding process, which discards insignificant coefficients, is performed after decomposition along each dimension. Note that we do not employ the non-standard decompositions since these methods require either a huge working buffer or considerable processing time for extensive I/O access to sort and organize the original dataset, which are in general prohibitive in our applications. For the DCT-based approach, the work in [13] is implemented to compare the quality of answers provided by the integral functions and GE functions.

5.1.3 Queries and coefficients

To simulate the range-sum queries, the range of each dimension of a query is randomly decided by the criterion that the product of all ranges is close to the query selectivity, d_q. The parameter MaxD is used to model the maximum number of dimensions having where-conditions. Also, the issue of different space costs for approximate query processing techniques mentioned in [2] is considered. Thus, for a given amount of space corresponding to c samples/wavelet(DCT) coefficients, we keep $h \simeq \frac{c}{3}$ histogram buckets to ensure a fair comparison.

Approximation-error metrics. We use the one-norm average relative error [23] in the aggregate value as a measure of the accuracy of the approximate query answer. Let m_i denote the accurate value for i-th query and $\widehat{m_i}$ denote the approximate answer. The metric is then formulated as $\frac{1}{n}\sum_{i=1}^{n}\frac{|m_i-\widehat{m_i}|}{m_i}$.

All experiments conducted in this section were performed on a Pentium 4 2.4 GHz machine running Windows 2000 with 1 GB of main memory.

5.2 Performance and scalability of estimation

To verify the performance scalability of query answering and error estimation, 100 queries ($dq = 0.03$) were generated for $\mathrm{D_{TPC}}$ and $\mathrm{D_{BOOK}}$. The execution times for query answering of DAWN and other methods, and the error estimation for BNA were recorded.

Figures 2 and 3 show the execution times with various Size(TCS) and settings of d_q where the prefixes ANS-,DWT-, SAMP- and HIST denote the execution time for answering queries of the DAWN, wavelets, sampling and histogram based approaches. Also, the prefix EST denotes the time for error estimation of BNA. Note that the execution time of the histogram-based technique of $\mathrm{D_{TPC}}$ is plotted along the right y-axis since the values are relatively high. It can be seen that the time cost of answering queries for the histogram-based technique increases as the query selectivity increases, except for $\mathrm{D_{BOOK}}$ when $d_q = 0.03$. The effort required to check the bucket boundaries and query ranges becomes an issue if the query selectivity is large. In contrast, the execution times for DAWN and wavelets are linear to the number of coefficients kept and the time cost is almost the same for different settings of query selectivity. This result conforms with the theoretical time complexity $O(n_T d)$ and $\mathrm{O}(2d \times \min\{n_T, 2\prod_{i=1}^{d}\log|D_i|\})$ for DAWN and wavelets, respectively. Also, the time for error estimation is close to that of answering queries for DAWN, indicating that the complexity of error estimation is close to answering the query. This meets the scenario where most users' requirements. That is, the time cost depends on the how accurate answers they want and the response time will not grow if a large (selectivity) ad hoc range-sum query is submitted.

It is noted that the execution time for the sampling-based technique is very short due to it's simplicity in calculating the contribution of every sampled cell. However, it will be shown that the sampling-based technique produces low quality answers.

5.3 Quality of answers

To evaluate the quality of answers for range-sum queries, the wavelet (abbreviated as WAVE), the histogram (HIST), the sampling (SAMP) and the DCT with integral based (DINT) approaches were implemented for comparison purposes. There were 100 queries for each dataset, and values of MaxD for $\mathrm{D_{BOOK}}$ and $\mathrm{D_{TPC}}$ were set to three and four, respectively. Various settings of storage constraints, i.e., using different numbers of coefficients C_M, were also considered.

From the results in Fig. 4, where $r_b = 0.1$, it is noted that DAWN outperforms other approaches in most

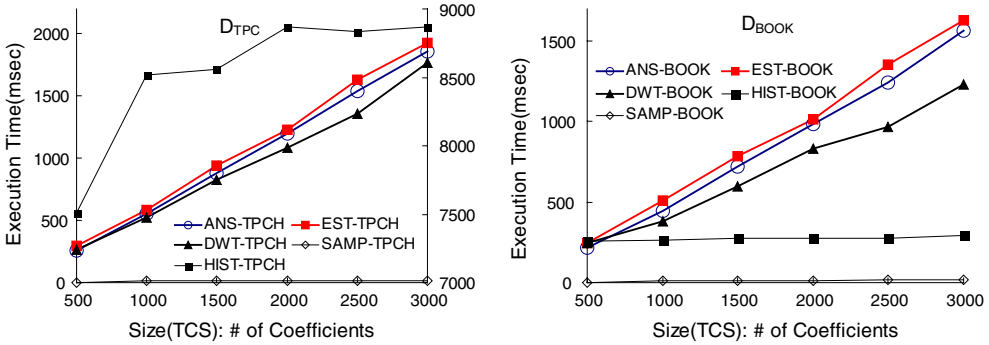

Fig. 2 Execution time with various Size(TCS). ($d_q = 0.03$)

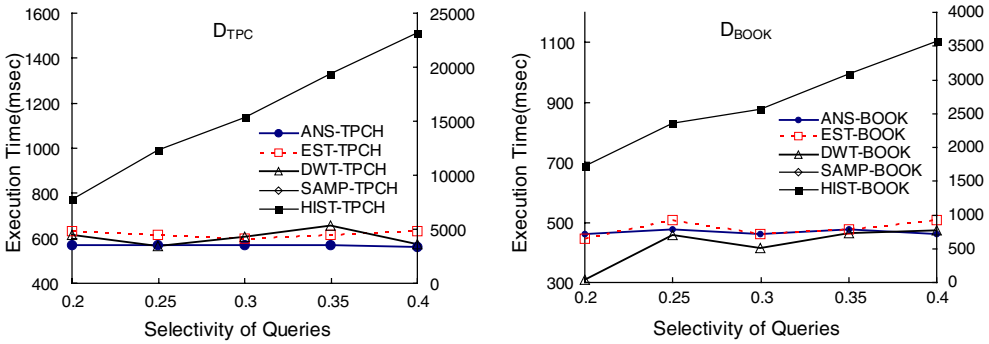

Fig. 3 Execution time with various selectivity of queries. (No. of coefficients = 1000)

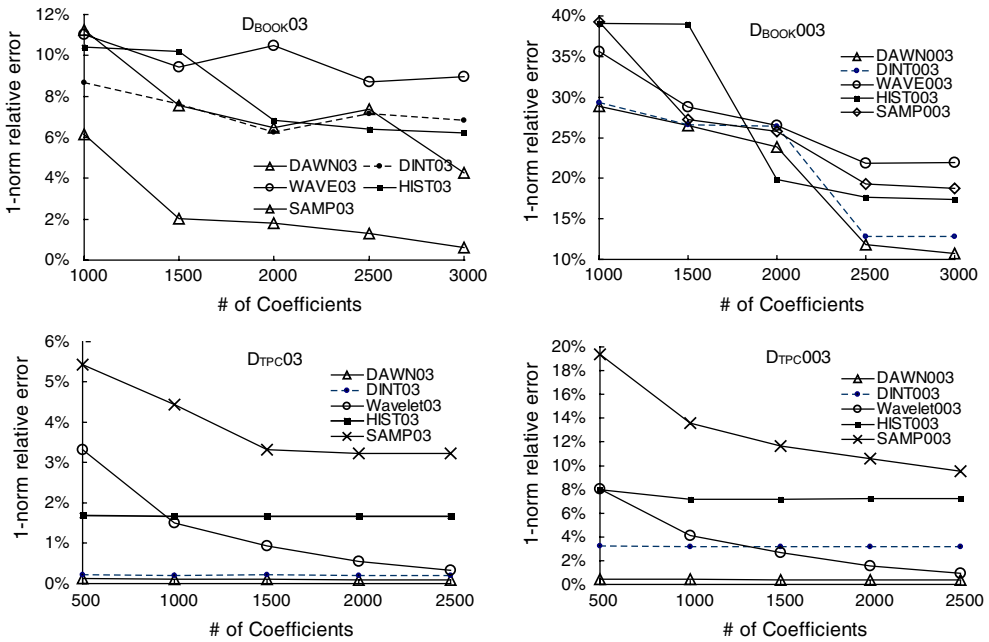

Fig. 4 One-norm average relative error rates of $D_{BOOK}0.3$, $D_{BOOK}0.03$, $D_{TPC}0.3$, and $D_{TPC}0.03$ where the numbers attached are the values of d_q

cases. It can be seen that DAWN, DINT and the wavelet-based approach performs better in sparse datasets than that in dense ones. It is noted that DINT is not stable due to its tendency to miscalculate, which was described in Sect. 3.3. Thus, the error rate of DINT in D_{BOOK} increases, even when the number of coefficients increases only slightly. The defect of integral functions diminishes the efficiency of first 500 buffered

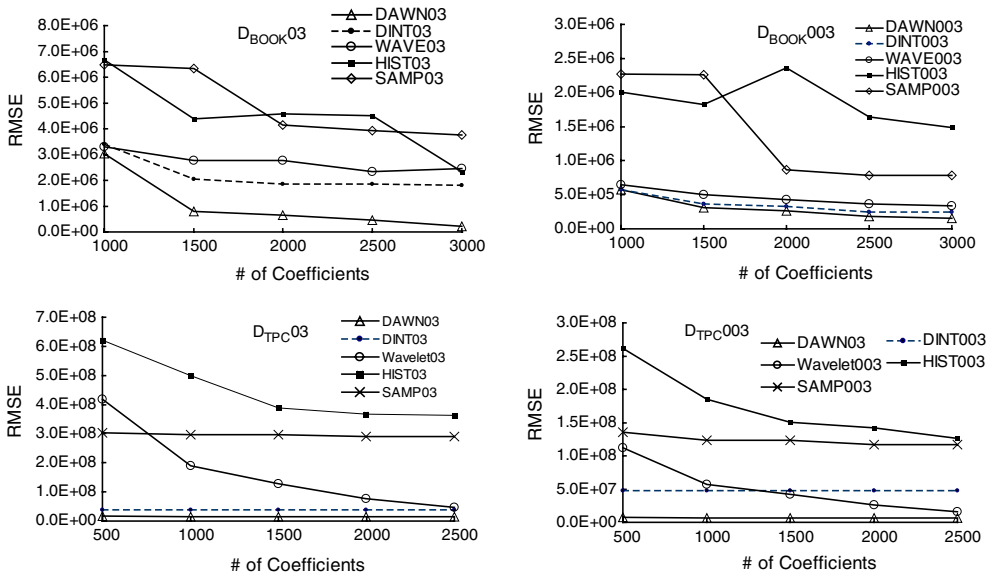

Fig. 5 Root mean square error rates of $D_{BOOK}0.3$, $D_{BOOK}0.03$, $D_{TPC}0.3$ and $D_{TPC}0.03$ where the numbers attached are the values of d_q

DCT coefficients; however, they are actually very representative of D_{TPC}.

It can also be noted that the wavelet-based approach suffers from the problem of limited computation buffers, i.e., not all the coefficients generated during the decomposition of a dimension can be kept in the buffer. Also, the thresholding mechanism may lose important information about the distribution, as shown by the fact that the wavelet coefficients for D_{BOOK} are not very efficient. Using the same number of coefficients (500) is enough for DAWN for the queries of $d_q = 0.3$. In contrast, DAWN performs well in energy conservation since the first 500 coefficients are very efficient for both $d_q = 0.3$ and $d_q = 0.03$. Specifically, DAWN outperforms the wavelet based approach in error rates by a margin ranging from 250 to 2,600% for D_{TPC}, and 140 to 600% for D_{BOOK}.

Note that the performance of DAWN in $D_{TPC}0.03$ is also superior to that of DINT. It outperforms DINT by about 200% on average in $D_{TPC}0.03$. Explicitly, DAWN not only improves the quality of answers but also provides stability.

In addition to one-norm error rates, the root-mean-square-error (RMSE) is used to demonstrate the quality of results. Some approaches, such as the wavelets approach, are only optimal for this error metric. Thus, the experiments with respect to RMSE were conducted to explore the theoretical optimality of those approaches. Fig. 5 shows the RMSE of each method. In all cases, the performance of DAWN on both datasets is superior to other approaches.

5.4 Performance of error estimation

To assess the performance of the BNA method, two sets of queries consisting of 300 queries each are generated for both D_{TPC} and D_{Book}. The query sets are answered by an unbuffered DCT for data under various settings of Size(TCS) to verify the capability of BNA for error estimation. The errors here are also measured by the one-norm average relative error; therefore, the maximum error is not bounded.

Figure 6 shows that the actual errors and those estimated by BNA are close to each other and in the same order of magnitude, since the two curves almost overlap. Note that the y-axis is in a logarithmic scale, suggesting that both the Basic method and the LM method fail to provide a tight bound on errors. Hence, the estimations from these two approaches are indeed too loose to use in practice.

Moreover, estimation results for both datasets are given in Fig. 7 to show the quality of estimations made by BNA. Note that the y-axis indicates the estimated errors of answers while the x-axis indicates the actual errors. A perfect estimation should be located at the diagonal though it is difficult to achieve in the case of small errors. In addition, the estimations can be classified into either overestimated ones, i.e., "△" in Fig. 7, or underestimated ones, i.e., "×" in Fig. 7. It is shown that the errors estimated by BNA in D_{TPC} and D_{BOOK} are in fact close to the actual errors, except that some errors of queries are underestimated. Note that the y-axis in the lower figures is finer than that in the upper ones. Hence,

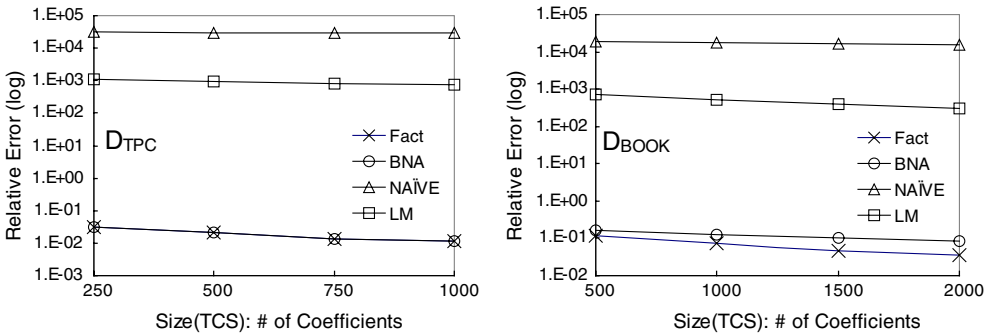

Fig. 6 Error rates under various number of coefficients for D_{TPC} and D_{BOOK}

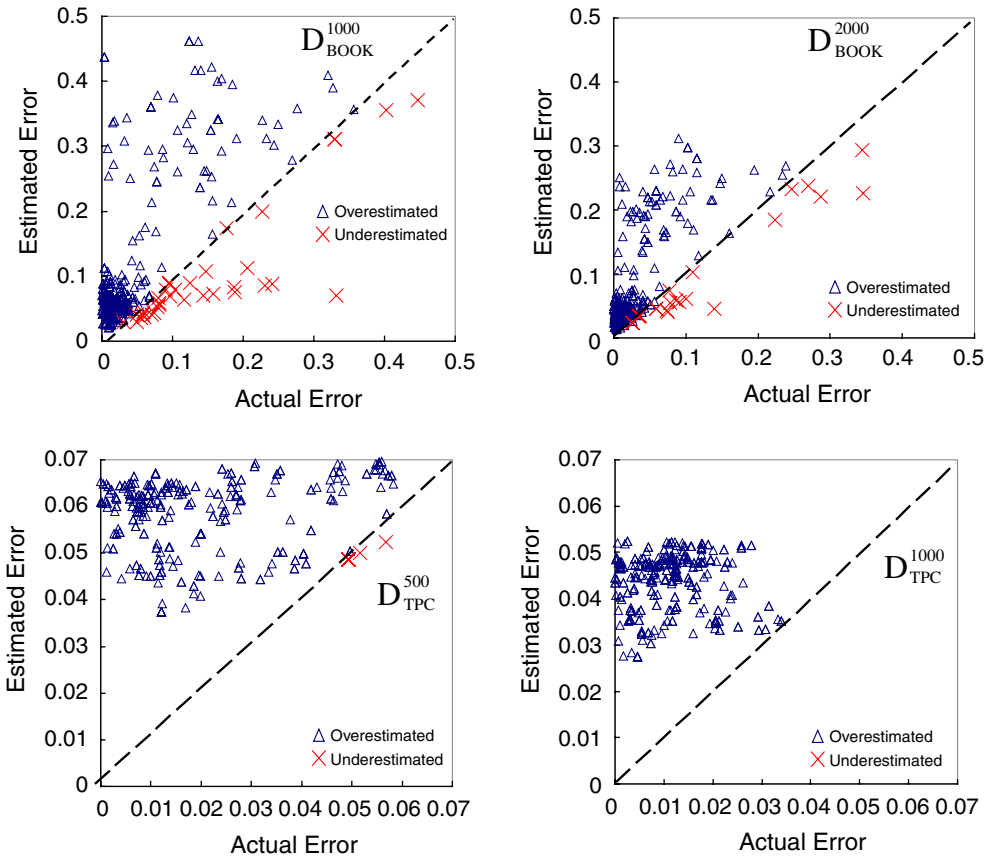

Fig. 7 Quality plots of D^{1000}_{BOOK}, D^{2000}_{BOOK}, D^{500}_{TPC}, and D^{1000}_{TPC} where the superscripts indicate the sizes of TCS

the points there can be deemed as close to the diagonals as these points in upper figures. Indeed, for those very few underestimates, the degree of difference is mostly within 1%, showing the very good quality of the BNA error estimator.

Note that BNA also improves the query response time. To show this capability, five sets of queries with different selectivities are generated for both D_{TPC} and D_{BOOK}. All the queries are estimated by BNA to obtain error bounds. Each query set has 100 queries. The average number of coefficients required maintain error rates below 10% is illustrated in Fig. 8. It can be seen that the number of coefficients needed decreases with increasing selectivity of queries. For online range-sum queries, the system needs to retrieve the minimum number of coefficients from storage systems for further computation for different selectivities, i.e., the number of I/O times can be optimized. Thus, the accurate error estimation performed by BNA is powerful in improving query response time.

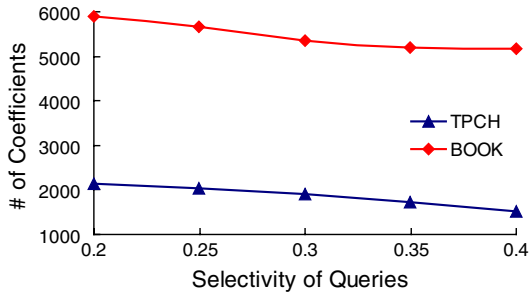

Fig. 8 The number of coefficients needed to keep the error rate below 10% with various query selectivities

6 Conclusions

In this paper, we propose a framework called DAWN that answers range-sum queries from compressed OP-cubes transformed by DCT. The optimality of the reciprocal zonal sampling technique to be utilized in coefficient selection has been theoretically proved, and the most efficient TCS can thus be decided. By utilizing the techniques of Geometric series and Euler's formula, a robust summation function, called the GE function, that answers range queries precisely in constant time has been devised. As shown by our experimental results, DAWN not only delivers answers of high quality for range-sum queries, but also provides a powerful tool of error estimation. The comparison among estimators shows that the BNA method outperforms other approaches by providing much tighter bounds in orders of magnitude.

Appendix A: additional example

Example A Table 4 shows the DCT coefficients of the data cube in Example 1.

Table 4 Table 4 DCT coefficient of the example data cube in Table 1

526.0	36.1	48.6	3.2	12.5	50.6	−85.9	6.8
52.3	73.8	18.8	−29.3	−6.4	5.4	−51.5	−6.3
56.9	−23.5	−12.8	42.2	−9.3	46.2	18.0	14.1
−77.2	−23.2	−8.7	−35.5	−3.1	−1.1	9.6	−30.9
−18.3	−14.9	−64.7	65.3	−9.2	9.7	−67.4	−1.1
−0.5	5.8	34.5	−32.4	3.1	−47.8	23.5	53.5
−3.8	−8.8	−16.2	−0.2	53.4	−31.4	15.5	15.1
64.2	−13.5	−26.3	−76.1	11.4	32.3	−0.8	10.1

Appendix B: proofs of Theorems and Lemmas

Corollary 2.1 *For an asymmetric d-dimensional data cube with brown noise distribution, the most efficient TCS of the data cube is* $\{k_{p_i} | \prod_{j=1}^{d}(u_{ij}+1) \le b'\}$ *where* b' *is a constant.*

Proof From the definition of asymmetric cubes and Theorem 2, the most efficient TCS can be defined as $T = \{k_{p_i} | \prod_{j=1}^{d}(\frac{u_{ij}+1}{|D_j|}) \le b\}$, where $|D_j|$ is the size of dimension D_j. Since $\prod_{j=1}^{d}|D_j|$ is a constant for a given cube, T can be rewritten as $T = \{k_{p_i} | \prod_{j=1}^{d}(u_{ij}+1) \le b \cdot \prod_{j=1}^{d}|D_j|\} = \{k_{p_i} | \prod_{j=1}^{d}(u_{ij}+1) \le b'\}$.[1] □

Lemma 1 *The value of* $\sum_{x=l}^{h}\cos(\frac{(2x+1)u\pi}{2M})$ *can be computed in constant time by the GE function:*

$$GE(l,h,u,M) = \begin{cases} \mathrm{Re}\left(\frac{\left(e(l)+\frac{1}{e(h)}\right)\left(e(\frac{2h-2l+1}{2})-1\right)}{2e(\frac{1}{2})-2}\right) & \text{when } u \neq 0 \\ h-l+1 & \text{when } u = 0 \end{cases}$$

where Re(.) *means real part and* $e(t) = e^{i(\frac{(2t+1)u\pi}{2M})}$.

Proof Consider the summation of a cosine series $S = \sum_{x=l}^{h}\cos(\frac{(2x+1)u\pi}{2M})$. Let $e^{ix} = \cos(\frac{(2x+1)u\pi}{2M}) + i\sin(\frac{(2x+1)u\pi}{2M})$. We have

$$S = \sum_{x=l}^{h}\frac{e^{i\left(\frac{(2x+1)u\pi}{2M}\right)} + e^{-i\left(\frac{(2x+1)u\pi}{2M}\right)}}{2} = \sum_{x=l}^{h}\frac{f_1+f_2}{2}.$$

Since the difference between frequencies of adjacent entries in f_1 and f_2 equals to the constant $\frac{u}{2M}$, S can be treated as the summation of two geometric series. Hence, S can be formulated as

$$S = \frac{1}{2}\frac{\left(e^{i\left(\frac{(2l+1)u\pi}{2M}\right)} + e^{-i\left(\frac{(2h+1)u\pi}{2M}\right)}\right)\left(e^{i(\frac{u\pi}{M})(h-l+1)}-1\right)}{e^{i(\frac{u\pi}{M})}-1}.$$

Next, we define the GE function as

$$GE(l,\ h,\ u,\ M) = \begin{cases} \frac{1}{2}\mathrm{Re}\left(\frac{\left(e(l)+\frac{1}{e(h)}\right)\left(e\left(\frac{2h-2l+1}{2}\right)-1\right)}{e(\frac{1}{2})-1}\right) & \text{when } u \neq 0 \\ h-l+1 & \text{when } u = 0 \end{cases}$$

where $e(t) = e^{i\left(\frac{(2t+1)u\pi}{2M}\right)}$.
Therefore, the summation $S = \sum_{x=l}^{h}\cos(\frac{(2x+1)u\pi}{2M})$ can be rewritten as the real part of a GE function which needs only constant time to estimate. This lemma follows. □

[1] Note that the frequencies of different dimensions are regarded as independent ones. Therefore the frequency of multidimensional data is not the same as that in [19] which assumes that $\widehat{f} = \sqrt{\sum_i f_i^2}$.

Lemma 2 *For the SC_A, SC_U and SC_T of a DCT D and a query $Q = SUM(l_1 : h_1, \ldots, l_d : h_d)$ corresponding to a d-dimensional data cube C, the sum of items (SC_U) and the sum of squares of items (SSC_U) in q-parts can be computed in $O(n_T d)$ where n_T is the size of corresponding TCS.*

Proof From the definition of q-parts, we have $\mathrm{SC}_A = \mathrm{SC}_T + \mathrm{SC}_U$ and

$$\mathrm{SC}_A = \sum_{x=0}^{N-1} C_x = \prod_{j=1}^{d} \sum_{i=0}^{|D_j|-1} (G_{u_{ij}})$$

where $G_{u_{ij}} = \prod_{j=1}^{d} c_{u_{ij}} GE(l_j, h_j, u_{ij}, |D_j|)$.

From Lemma 1, $G_{u_{ij}}$ can be estimated in constant time and thus the time complexity of estimating SC_A can be reduced to $\mathrm{O}(|D_1|+|D_2|+\cdots|D_d|)$. Hence, the time cost to estimate SC_U is $O(n_T+n_T d+|D_1|+|D_2|+\cdots|D_d|)\sim O(n_T d)$. Also, the *SSC* can be computed similarly by replacing $G_{u_{ij}}$ with $\left(G_{u_{ij}}\right)^2$. □

Lemma 3 *For the SK_A, SK_U and SK_T of a DCT D corresponding to a d-dimensional data cube C, the cost for estimating SK_U is the same as that of computing single coefficient during transformation process.*

Proof Let $S^d = \sum_{n_1=0}^{|D_1|-1} \sum_{n_2=0}^{|D_2|-1} \cdots \sum_{n_d=0}^{|D_i|-1}$, $\mathcal{K}=\sqrt{\frac{2^d}{\prod_{i=1}^{d}|D_i|}}$, W is the set of all transformed coefficients, and f_p be the measure value of data cell located at $(m_1, m_2 \ldots, m_d)$, we have

$$\begin{aligned}\mathrm{SK}_A &= \sum_{x=0}^{N-1} K_x \\ &= \mathcal{K} \sum_{u_q \in W} S^d \left(f_p \prod_{k=1}^{d} c_{(n_k)} \cos\left(\frac{(2m_k+1)u_q\pi}{2\,|D_k|}\right)\right)\end{aligned}$$

where $c_{(m_k)} = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } (m_k) = 0 \\ 1 & \text{for } (m_k) \neq 0 \end{cases}$.

Since the iteration of u_q is independent of m_i and f_p, we move the summation on u_q to the inner iteration. It follows that

$$\mathrm{SK}_A = \mathcal{K} S^d \left[f_p \sum_{u_q \in W} \prod_{k=1}^{d} c_{(m_k)} \cos\left(\frac{(2m_k+1)u_q\pi}{2\,|D_k|}\right)\right].$$

Also, the aggregation of $\sum_{u=p_i \in W} \prod_{k=1}^{d}$ includes all combinations of the product of cosine terms, and thus the result will not change after the summation is moved to inner iteration, i.e.,

$$\mathrm{SK}_A = \mathcal{K} S^d f_p \prod_{k=1}^{d} \left[\sum_{u=0}^{D_k-1} c_{(m_k)} \cos\left(\frac{(2m_k+1)u\pi}{2D_k}\right)\right]. \tag{10}$$

The summation inside the square brackets can be estimated by applying the same technique of GE function and then the computation cost of the summation is reduced to constant time. The time costs for the iterations outside and inside the square brackets are $O(n')$ and $O(d)$ respectively where n' is the number of non-zero data points in $\mathcal{C}$. By incorporating Eq. (10) into transformation process of DCT, the cost of computing SK_A is the same as that of computing a single coefficient. Also, the SK_T can be accumulated simultaneously. Since $\mathrm{SK}_U = \mathrm{SK}_A - \mathrm{SK}_T$, it follows that the cost for SK_U is the same as computing a single coefficient during transformation process. □

References

1. Transactional Processing Performance Council. TPC Benchmark. http://www.tpc.org
2. Chakrabarti, K., Garofalakis, M.N., Rastogi, R., Shim, K.: Approximate query processing using wavelets. In: Proceedings of VLDB Conference, pp. 111–122 (2000)
3. Deligiannakis, A., Roussopoulos, N.: Extended wavelets for multiple measures. In: Proceedings of SIGMOD Conference (2003)
4. Garofalakis, M., Gibbons, P.B.: Wavelet synopses with error guarantees. In: Proceedings of SIGMOD Conference (2002)
5. Garofalakis, M., Gibbons, P.B.: Probabilistic wavelet synopses. ACM Trans. Database Syst. **29**(1), (2004)
6. Garofalakis, M., Kumar, A.: Deterministic wavelet thresholding for maximum-error metrics. In: Proceedings of PODS Conference (2004)
7. Haas, P.J.: Large-sample and deterministic confidence intervals for online aggregation. In: Proceedings of SSDBM Conference (1997)
8. Haas, P.J., Naughton, J.F., Seshadri, S., Stokes, L.: Sampling-based estimation of the number of distinct values of an attribute. In: Proceedings of VLDB Conference, pp. 311–322 (1995)
9. Ho, C.T., Agrawal, R., Megiddo, N., Srikant, R.: Range queries in OLAP data cubes. In: Proceedings of SIGMOD Conference, pp. 73–88 (1997)
10. Ioannidis, Y., Poosala, V.: Balancing histogram optimality and practicality for query result size estimation. In: Proceedings of SIGMOD Conference, pp. 223–244 (1995)
11. Ioannidis, Y.E., Poosala, V.: Histogram-based approximation of set-valued query-answers. In: Proceedings of VLDB Conference (1999)
12. Jagadish, H.V., Koudas, N., Muthukrishnan, S., Poosala, V., Sevcik, K.C., Suel, T.: Optimal histograms with quality guarantees. In: Proceedings of VLDB Conference, pp. 275–286 (1998)

13. Lee, J.H., Kim, D.H., Chung, C.W.: Multi-dimensional selectivity estimation using compressed histogram information. In: Proceedings of SIGMOD Conference, pp. 205–214 (1999)
14. Lim, J.: Two Dimensional Signal and Image Processing. Prentice Hall (1990)
15. Matias, Y., Vitter, J.S., Wang, M.: Dynamic maintenance of wavelet-based histograms. VLDB J., 101–110 (2000). http://www.citeseer.nj.nec.com/matias00dynamic.html
16. Piatetsky-Shapiro, G., Connell, C.: Accurate estimation of the number of tuples satisfying a condition. In: Proceedings of SIGMOD Conference, pp. 256–275 (1984)
17. Poosala, V., Ganti, V.: Fast approximate answers to aggregate queries on a data cube. In: Proceedings of SSDBM Conference (1999)
18. Poosala, V., Ioannidis, Y.E.: Selectivity estimation without the attribute value independence assumption. VLDB J., 486–495 (1997)
19. R. Agrawal, C.F., Swami, A.N.: Efficient similarity search in sequence databases. In: Proceedings of FODO Conference, pp. 69–84 (1993)
20. Sarawagi, S.: Indexing OLAP data. Data Eng. Bull. **20**(1), 36–43 (1997)
21. Stollnitz, E.J., DeRose, A.D., Salesin, D.H.: Wavelets for Computer Graphics – Theory and Applications. Morgan Kaufmann, San Francisco (1996)
22. V. Poosala Peter J. Haas, Y.E.I.: Improved histograms for selectivity estimation of range predicates. In: Proceedings of SIGMOD (1996)
23. Vitter, J.S., Wang, M.: Approximate computation of multidimensional aggregates of sparse data using wavelets. In: Proceedings of SIGMOD Conference, pp. 193–204 (1999)
24. Vitter, J.S., Wang, M., Iyer, B.: Data cube approximation and histograms via wavelets. In: Proceedings of CIKM Conference, pp. 96–104 (1998)
25. Zeng, W., Lei, S.: Efficient frequency domain selective scrambling of digital video. IEEE Trans Multimedia **5**, 118 (2002)