Optimizing Dimensionality Reduction with Convergence-Based Cluster Detection

Anil Rajput

Professor Computer Science & Mathematics

Govt.C.S.A.P.G.College, Sehore, MadhyaPradesh

e-mail: dranilrajput@hotmail.com

Kshmasheel Mishra

Former Reader Computer Science

Vikram University,Ujjain,Madhya Pradesh

e-mail:mishra_ks@hotmail.com

Abstract:

Businesses worldwide are experiencing unprecedented growth in data volume and a continuous demand for enhanced database performance. The need to mine, cleanse, and integrate increasing amounts of data across enterprises is intensifying, alongside the urgency for timely, meaningful insights. Dimensionality reduction is a critical technique used in clustering, classification, and various machine learning and data mining applications. This process involves data convergence, which moves data points along the density gradient, creating condensed and widely-separated clusters. Clusters are then detected by identifying the connected components of dense cells and evaluating their compactness. These steps are applied across a sequence of grids with varying cell sizes. Reducing high-dimensional data to lower dimensions is essential for better visualization and projection into 2D or 3D spaces. The ultimate goal is to produce meaningful and manageable data clusters that facilitate informed decision-making for business leaders.

Keywords: *Big Data Analytics, Data Partitioning, Data Warehousing, Business Intelligence (BI), Data Integration (DI), Advanced Analytics, Database Performance, Dimensionality Reduction, Data Convergence, Clustering Algorithms, Multi-Dimensional Data Analysis (MDDA) and Knowledge Discovery in Databases (KDD)*

INTRODUCTION

Businesses globally are experiencing an exponential growth in data, necessitating continuous improvements in database performance. A comprehensive analysis of current trends in data partitioning and Big Data analytics forms the basis of this research. Despite the widespread adoption of data warehouses and business intelligence solutions, organizations still struggle to deliver reliable

information promptly and accurately. The ever-increasing volume of data to mine, cleanse, and integrate, combined with the rising complexity and urgency of obtaining meaningful insights, intensifies these challenges.

Before information can be effectively displayed in dashboards or reports, several steps are essential: collecting raw data, integrating data from various sources, transforming data formats, creating data cubes, and loading updates into the data warehouse. Performance remains a critical success factor in business intelligence (BI), data warehousing (DW), data integration (DI), and advanced analytics. Modern BI, DW, DI, and analytics demand high performance due to the increasing data volumes, complex analytical workloads, expanding user bases, and real-time operational requirements. Enhancing performance in multidimensional databases involves achieving speed and scale while managing complexity and concurrency. These four dimensions—speed, scale, complexity, and concurrency—are interrelated and crucial for quality information production in data warehouse design.

Dimensionality reduction, the process of converting high-dimensional data into lower-dimensional forms, is crucial for better visualization, efficient storage and retrieval, and noise reduction. The goal is to create meaningful and manageable data cubes for business decision-makers [2].

This paper presents findings on clustering multi-dimensional data and makes the following contributions:

- Introduction of a novel data preprocessing technique that optimizes data distribution characteristics.
- Proposal of a data-convergence process as a key component of this preprocessing technique, resulting in condensed and well-separated clusters.
- Demonstration of how the data-convergence process can enhance the performance of existing clustering algorithms.
- Detection of clusters based on cell density following the data-convergence process.
- Development of a noise-insensitive algorithm capable of detecting clusters of any shape.
- Implementation of a multi-scale gridding scheme to detect clusters at various scales, addressing the challenge of selecting an appropriate cell size and managing data sets with clusters of different densities.

MULTI DIMENSIONAL DATA ANALYSIS (MDDA)

Modern businesses routinely capture data on millions of observations across various subjects, brand SKUs, time periods, predictor variables, and store locations, resulting in massive high-dimensional datasets. Multi-dimensional Data Analysis (MDDA) involves summarizing data across multiple levels (dimensions) and presenting the results in a multi-dimensional grid format. This process, also known as

data mining or Knowledge Discovery in Databases (KDD)[6], aims to uncover interesting patterns from large datasets stored in databases, data warehouses, or other information repositories. Data mining integrates techniques from various disciplines, including database technology, statistics, machine learning, high-performance computing, pattern recognition, neural networks, data visualization, and information retrieval [10].

Cluster analysis identifies homogeneous and well-separated groups of objects within databases, addressing the recognized need to cluster large quantities of multidimensional data. This classical problem is significant in databases, artificial intelligence, and theoretical literature, playing a crucial role in many business and scientific fields. However, real-world data often contains irrelevant features that can reduce the accuracy of clustering algorithms. High-dimensional datasets pose fundamental challenges to these algorithms. One effective technique for enhancing data analysis performance is dimension reduction, which transforms data into a lower-dimensional space while preserving its essential information. This simplification facilitates further processing without compromising the quality of the final results and is commonly used in clustering, classification, and various machine learning and data mining applications [7].

The data convergence process advances data points along the density gradient, creating condensed, widely-separated clusters. After data convergence, clusters are identified by finding the connected components of dense cells and evaluating their compactness. These steps are performed on a sequence of grids with varying cell sizes.

PROPOSED APPROACH FOR DATA PRE PROCESSING

We introduce a novel data preprocessing technique called "convergence," which optimizes the inherent distribution characteristics of data. In real datasets, natural data groups, if they exist, may be very sparse. By moving data points toward the centroids of their respective groups during preprocessing, these sparse groups become denser and more easily detectable, while noise can be further isolated. Essentially, a dense area "attracts" objects from surrounding sparse areas, increasing its density. This process assumes that a data point is attracted to neighboring points, moving in the direction where the attraction is strongest, determined by the distribution of nearby data points. Our data convergence preprocessing simulates this movement, reflecting the "attraction" each data point feels toward its neighbors. Analogous to infiltration mechanisms where materials like water move from dense to sparse areas, our method causes data points to migrate toward denser regions. Points far from a given data point have minimal influence and can be ignored. This data reorganization concept is applicable in various fields, such as pattern recognition, data clustering, and signal processing, to facilitate extensive data analysis [2].

We propose a convergence-based approach for multi-dimensional data analysis to address the limitations of current clustering algorithms in handling multi-dimensional data. This method combines a cluster-wise evaluation measurement to select the best clusters detected at different scales [8]. The algorithm comprises three steps: data convergence, cluster detection, and cluster evaluation and selection. During data convergence, points move along the density gradient, forming condensed and widely separated clusters. Clusters are then identified by finding the connected components of dense cells. Both data convergence and cluster detection are grid-based, employing a sequence of grids with varying cell sizes instead of a fixed cell size. To prevent issues caused by points clustered near grid vertices and separated into different cells, the process uses two interleaved grids for each cell size. Finally, in the cluster evaluation and selection step, clusters detected at different scales are evaluated using a cluster-wise measurement, and the best clusters are selected as the final result.

This paper introduces two main innovations:

- A grid-based convergence and evaluation approach using a sequence of increasing grid sizes to capture cluster structures at different scales. At each scale, two grids are employed, with the second grid shifted diagonally from the first.
- Integration of a compactness-based cluster evaluation into the framework. This evaluation measures both inter-cluster and intra-cluster distances, assessing a cluster's compactness by the ratio of intra-cluster to inter-cluster distances[11].

Grid-based clustering methods depend heavily on the proper selection of grid-cell size. Without prior knowledge of the structure of an input data set, proper grid-cell size selection is problematical. We propose a multi scale gridding technique to address this problem. Instead of choosing a grid with a fixed cell size, we use a sequence of grids of different cell sizes. Data convergence and cluster detection are conducted on these grids, the detected clusters are compared, and those clusters with the best quality are selected as the final result. Throughout this paper, we assume that the input data set X is

X = f~X1 ; ~X2 ; ::: ; ~Xng;

which is normalized to be within the hypercube [0;1)d ½ Rd.

We apply a simple histogram-based approach to get reasonable grid scales for the data convergence process. We scan the input d-dimensional data set X once and get the set of histograms, one for each dimension:

H = fh1; h2; ::: ; hdg:

Each bin of a histogram denotes the number of data points in a certain segment on this histogram. We set up a number n as a quantity threshold. It is used in the following algorithm to help generate density spans.

A density span is a combination of consecutive bins' segments on a certain dimension in which the amount of data points exceeds n. A size of a density span is the sum of the sizes of the bins it includes. For each histogram hi, i=1,...,d, we sort its bins based on the number of data points they contain in descending order.

Then we start from the first bin of the ordered bin set and merge it with its neighboring bins until the total amount of data points in these bins exceeds. At each step, we check the number of points in the bin on the left side and the one on the right side of the currently span, and choose the bin with more points in it to merge with.

Thus a density span is generated as the combination of the segments of these bins. If a current span has less than n data points, but its left and right neighbors have both been assigned to a precious span already, we stop the operation on the current span and call it as an incomplete span which will not be considered in the following procedure of generating multiple grid scales. The operation is continued until all the non-empty bins of this histogram is in some density spans or some incomplete spans. Each histogram has a set of density spans.

Here we just demonstrate two density spans on this histogram. Bin 21 is the one with largest amount of data points.

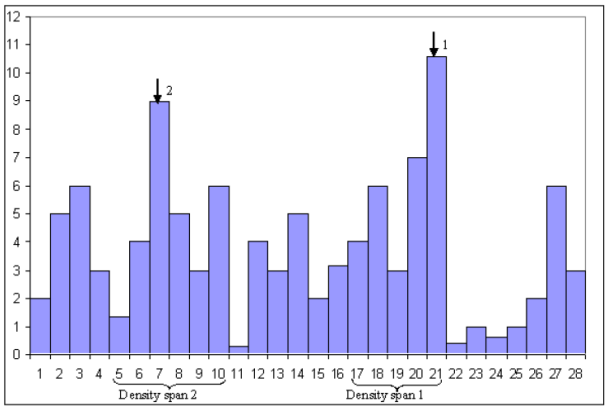

Figure 1 : An example of density span acquirement

We start from Bin 21, check its neighbouring bins 20 and 22, and choose bin 20 to merge with because it has more points than bin 22. Next we check the neighbouring bins of the combination of the segments of bin 20 and 21, which are bin 19 and bin 22, and choose bin 19 because it has more points

than bin 22. In this way the neighbouring bins are merged until the amount of data points included exceeds n. Thus density span 1 is generated. Bin 7 has the second largest amount of data points. Density span 2 is generated starting from bin 7, first including bin 8, then including bin 6, 9, 10 and 5, step by step until the amount of data points included exceeds[7].

After the density span generation operation mentioned above, we have a set T of density spans with different sizes. We cluster the density spans by their sizes.

The value Xn depends on the size of the input data set X. Normally it can be set as a certain percentage of the number of data points in X. There is a balance in choosing a value for Ks: a smaller Ks can increase the precision of cluster detection, while a larger Ks can save time. The time complexity for this method is determined by the dimensionality d of X and the amount of bins Bn in each histogram. The time required to perform Algorithm 1 is O(Bn logBn).

Algorithm 1 (Density span generation)
Input: histogram hi
Output: Density span set of hi
1) Sort the bins of hi in the descending order;
2) Beginning from the first bin of the ordered bin set, merge it with its neighbours until the total amount of data points included exceeds Xn;
3) Repeat step 2 until all non-empty bins are included in some density spans or some incomplete spans;
4) Output the density span set.
The multi scale gridding scheme proposed above not only facilitates the determination of a proper cell size but also offers advantages for handling data sets with clusters of various densities.

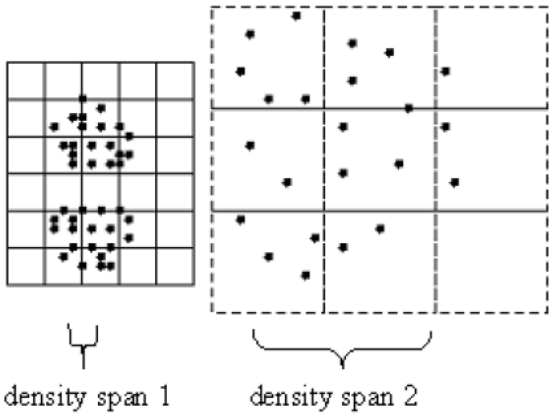

Figure 2 : A data set with three clusters

DATA CONVERGENCE METHODOLOGY

We start with the first step of the proposed method: data convergence. In this step, each data point moves along the direction of the density gradient and the data set shrinks toward the inside of the clusters. Points are "attracted" by their neighbours and move to create denser clusters. This process is repeated until the data are stabilized or the number of iterations exceeds a threshold[5].

The neighbouring relationship of the points in the data set is grid-based. The space is first subdivided into grid cells. Points in sparse cells are considered to be noise or outliers and will be ignored in the data-convergence process. Assume a dense cell C with neighbouring cells surrounding C. Data convergence proceeds iteratively; in each iteration, points in the dense cells move toward the data centroid of the neighbouring cells. The iterations terminate if the average movement of all points is less than a threshold or if the number of iterations exceeds a threshold. The major motivation for ignoring sparse cells is computation time. If the grid cells are small, the number of non-empty cells can be O(n), where n is the number of data points.

The computation of data movement for all non-empty cells takes a length of time quadratic to the number of non-empty cells, which is O(n2). By ignoring sparse cells in the data movement, dramatic time savings can be realized.

Ideally, for a data set covering a manifold with a boundary, the convergence process pushes the boundary points inward until the manifold is reduced to its skeleton. If the skeleton is also a manifold with a boundary, it is skeletonized again. This process is repeated until a manifold with no boundary is produced, as shown in Figure 3. However, most data sets from real-world applications do not have well-defined shapes in high-dimensional spaces. The data sets resulting from the convergence process may also not have well-defined shapes. In general, the convergence process produces individual clusters which are condensed and therefore widely separated, facilitating cluster detection[4].

The second step of our method is cluster detection. Since the data-convergence process generates individual clusters which are condensed and widely separated, it can be used as a pre processing with any cluster-detection algorithm. In this paper, we use a simple grid based cluster-detection method to test the data-convergence process. For a given cell-side length 1 Xn, after the data-convergence process is performed on the input data set, we find the dense cells. Neighboring dense cells are connected and a neighborhood graph of the dense cells is constructed. Each connected component of the neighborhoods graph is a cluster. The cluster-detection method is conducted on two interleaved grids. This avoids the problem caused by points clustered near a vertex of a grid and separated in different cells. After the

cluster detection step, we evaluate the clustering results[6, 7]. There are several ways to define what is a good clustering. Most conventional clustering validity measurements evaluate clustering algorithms by measuring the overall quality of the clusters. A cluster in a data set is a subset in which the included points have a closer relationship to each other than to points outside the cluster. In the literature the intra-cluster relationship is measured by compactness and the inter-cluster relationship is measured by separation. Compactness is a relative term; an object is compact in comparison to a looser surrounding environment. We use the term compactness to measure the quality of a cluster on the basis of intra-cluster and inter-cluster relationships[3].

After the cluster detection step, we evaluate the clustering results. There are several ways to define what is a good clustering. Most conventional clustering validity measurements evaluate clustering algorithms by measuring the overall quality of the clusters. In this section, we introduce a cluster-wise measurement which provides an evaluation method for individual clusters. Compactness is a relative term; an object is compact in comparison to a looser surrounding environment. We use the term compactness to measure the quality of a cluster on the basis of intra-cluster and inter-cluster relationships.

Given an input data set and a defined scale, we first find the dense cells of two interleaved grids at this scale. Compactness is then defined on the complete graph of the dense cells. Because the sparse cells are ignored, running time is reduced and the result is not noise-sensitive. In evaluating a given data set, we run the data-convergence and cluster-detection processes using a sequence of grids of selected cell sizes. We compute the compactness-before-convergence of the clusters detected at all scales[9]. Those clusters with compactness exceeding a specified threshold will be output as the final result. Within the clustering result, a cluster can be a subset of another cluster.

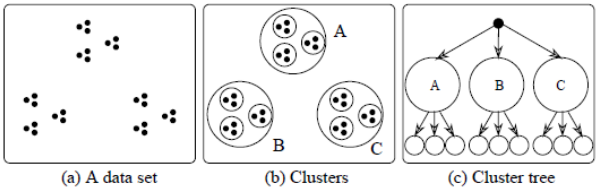

Figure3 : Cluter Tree

To demonstrate the functioning of the convergence process, we will first discuss experiments conducted using a 2D data set. Next we give our experimental results on high-dimensional data sets to show the scalability of our approach. Trials using data sets from real-world applications comparing to other algorithms such as CURE and OPTICS are offered as a demonstration of the accuracy of the proposed approach.

In our experiments, we empirically found that after at most 10 iterations, the movements of most of the data sets which converge eventually will be stabilized, so we set Tit as 10.

To test the scalability of our algorithm over dimensionality and data size, we designed a synthetic data generator to produce data sets with clusters. The sizes of the data sets vary from 5,000, 10,000, ... to 60,000, and the dimensions of the data sets vary from 10, 20, ... to 60. Each data set contains five clusters, with the points in each cluster generated in normal distributions. We set different values like 200, 350, 500, 650, and 800 to the centers of different clusters on each dimension (we can regard the center of a synthetic clusters as the mean of the data distribution in this cluster), and set the standard deviation Xn for different clusters on each dimension as 10, 5, 7, 9 and 4, respectively. An additional 5% of data points are added randomly to each data set as noise. Clusters were effectively detected by our algorithm in all tests performed on these high-dimensional data sets.

In the experiments, Xn was set according to the size of the testing data set. Since for different synthetic data sets, the condition of convergence of the algorithm and the iteration numbers would be different, to make the comparison between data sets of different sizes and dimensionality fair, we fixed the number of iterations ranged as 5. . Figure 3 shows the running time of 12 groups of data sets with dimensions increasing from 10 to 60. Each group has a fixed data size.

EXPERIMENTS ON DATASETS WITH 2 DIMENSIONS

We first conducted experiments on 2-dimensional data sets as intuitive demonstrations for data convergence pre processing procedure. Due to the space limitation, here we just present the convergence result on one data set DS1 which has 2682 points including noisy data. There are two clusters in the data with one is half-embraced by the other. The convergence process generates two well-separated clusters of arbitrary shape and filters outliers, thus facilitating cluster detection. The experiment set was initiated with zero intervention stage and no partitioning .

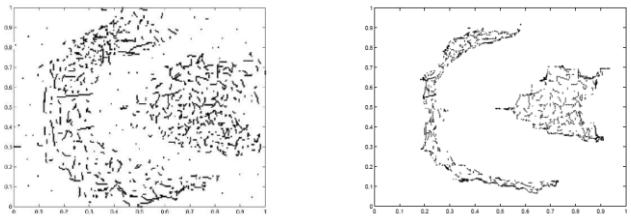

Figure 4 : Convergence process on the data set DS1

(2- dimensional data set DS1, and the data set after the convergence process).

EXPERIMENTS ON HIGH-DIMENSIONAL DATASETS

To test the scalability of our algorithm over dimensionality and data size, we designed a synthetic data generator to produce data sets with clusters. The sizes of the data sets vary from 5,000, 10,000, ... to

60,000, and the dimensions of the data sets vary from 10, 20, ... to 60. Each data set contains five clusters, with the points in each cluster generated in normal distributions. We set different values like 200, 350, 500, 650, and 800 to the centers of different clusters on each dimension, and set the standard deviation for different clusters on each dimension as 10, 5, 7, 9 and 4, respectively. Clusters were effectively detected by our algorithm in all tests performed on these high-dimensional data sets. Figure 5 shows the running time of 6 groups of data sets with sizes increasing from 5,000 to 60,000. Each of these 6 groups has fixed dimensions.

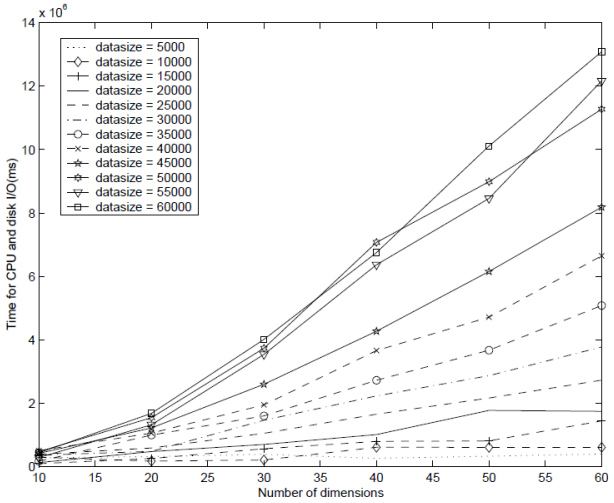

Figure 5 : Running time with fixed dimensions

FINDINGS OF CONVERGENCE-BASED DIMENSION REDUCTION PROCESS.

The alteration of the histogram variances through the data-convergence process on each dimension reflects the characteristic aspects of the data distribution on the dimension better than the histogram variance itself. Figure 6 shows an example for this case. From figure 6 we can see the dimension 3 and dimension 4 have much larger histogram variances than other dimensions both without and after convergence-based dimension reduction process.

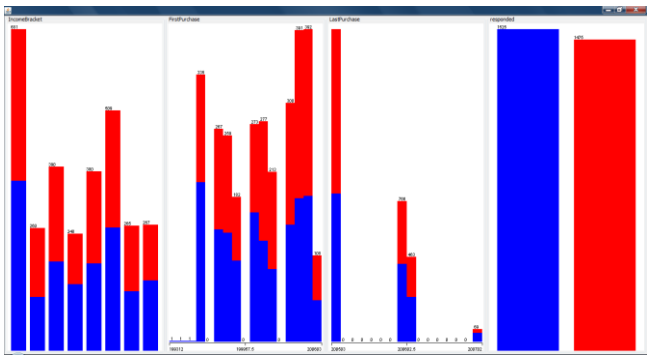

Figure 6 : Histograms variance diagram for convergence-based dimension reduction

CONCLUSION

In this paper, we explored the optimal selection of a subset of existing dimensions to facilitate easier interpretation. Our key idea is that dimensions significantly contributing to effective data analysis, particularly clustering, show notable changes in histogram variance through the data-convergence process. We identify suitable dimension candidates for clustering by examining the differences in histogram states of each dimension following convergence-based dimension reduction.

As companies generate millions of data points about their users, they seek ways to convert this information into increased revenue. Our approach can be particularly valuable in fields like data fusion, where data evolves dynamically, and traditional methods often struggle to maintain an up-to-date system. This technique enables the dynamic supervision of data status, efficient elimination of obsolete data, and reorganization of the data structure, ensuring the system always contains the most relevant information.

REFERENCES

[1] A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Prentice Hall, 1988.

[2] Bellatreche, L., Boukhalfa, K., Richard, P.,Woameno, K.Y.: Referential Horizontal Partitioning Selection Problem in DataWarehouses: Hardness Study and Selection Algorithms. In IJDWM. 5(4), 1–23 (2009)

[3] Ben-Dor, A., Shamir, R. and Yakhini, Z. Clustering gene expression patterns. Journalof Computational Biology, 6(3/4):281–297, 1999.

[4] Beyer, K., Goldstein, J., Ramakrishnan, R. and Shaft, U. When is nearest neighbours meaningful. In Proceedings of the International Conference of Database Theories, pages 217–235, 1999.

[5] Chun Hung Cheng and Ada Wai-Chee Fu and Yi Zhang. Entropy-based Subspace Clustering for Mining Numerical Data. In Knowledge Discovery and Data Mining, pages 84–93, 1999.

[6] D. Yu, G. Sheikholeslami, and A. Zhang. Findout: Finding outliers in very large datasets. The Knowledge and Information Systems (KAIS), (4), October 2000.

[7] E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. In Proceedings of the 24th VLDB conference, pages 392–403, New York, August 1998.

[8] J. H. Wang and J. D. Rau. Vq-agglomeration: A novel approach to clustering. In IEE Proceeding-Vision, Image and Signal Processing, 148(1):36-44, 2001.

[9] M. T. Dickerson and D. Eppstein, Algorithms for proximity problems in higher dimensions, Computational Geometry: Theory and Applications 5, p.277-291 (1996).

[10] P. Ciaccia, M. Patella and P. Zezula, M-tree: An efficient access method for similarity search in metric spaces, Very Large Data Bases (VDLP) Conference 1997, p.426-435 (1997).

[11] T. Gonzalez. Clustering to minimize the maximum intercluster distance. Theoretical Computer Science, 38:311–322, 1985.