Computer Engineering Department
Faculty of Engineering
Deanery of Higher Studies
The Islamic University-Gaza
Palestine

An Improvement for DBSCAN Algorithm for Best Results in Varied Densities

Mohammad N. T. Elbatta

Supervisor

Dr. Wesam M. Ashour

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering

Gaza, Palestine

(September, 2012) 1433 H

Acknowledgements

Apart from the efforts of myself, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project. I would like to thank my parents for providing me with the opportunity to be where I am. Without them, none of this would be even possible to do. You have always been around supporting and encouraging me and I appreciate that. I would also like to thank my brothers and sisters for their encouragement, input and constructive criticism which are really priceless. Also, special thanks goes to Dr. Homam Feqawi who did not spare any effort to review and audit my thesis linguistically.

My heartiest gratitude to my wonderful wife, Nehal, for her patience and forbearance through my studying and preparing this study.

I would like to express my sincere gratitude to my advisor Dr. Wesam Ashour for the continuous support of my master study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my master study.

Also, I would like to thank all my teachers in master degree, for their perceptiveness, understanding, and their skillful teaching. And many thanks to all my friends for their moral support.

And above all, many thanks to Allah, who helps me get all required resources for completing this research.

Contents

List of Abbreviations

Core	The core object for each cluster is the object that has the minimum density function value according.
DBCLASD	A Distribution-Based CLustering Algorithm for mining in large Spatial Databases.
DBSCAN	Density-Based Spatial Clustering of Applications with Noise.
DENCLUE	Clustering Based on Density Distribution Functions "DENsity-based CLUstEring".
DMDBSCAN	Dynamic Method DBSCAN.
DVBSCAN	A Density based Algorithm for Discovering Density Varied Clusters in Large Spatial Databases.
E	Total Density Function represents the difference among the data points, which is based on the core.
E-neighborhood	The neighborhood within a radius Eps of a given object is called the E-neighborhood of the object.
Eps	The radius of a number of objects.
MDBSCAN	A Multi-Density DBSCAN.
MinPts	The Minimum number of Points (objects) each point of a cluster the neighborhood of a given radius (Eps) has to contain.
p	Some Point in a data set.
ST-DBSCAN	An algorithm for clustering Spatial–Temporal data.
VDDBSCAN	Vibration and Dynamic DBSCAN.
VMDBSCAN	Vibration Method DBSCAN.

List of Figures

List of Tables

تعديل على خوارزمية DBSCAN للحصول علي أفضل نتائج عند الكثافة المتغيرة

محمد نديد تكريم البطة

ملخص

DBSCAN هو خوارزمية تستخدم لتجميع البيانات بالاعتماد على الكثافة. بواسطته يمكن معرفة مجموعات من مختلف الأشكال والأحجام من كمية كبيرة من البيانات، والتي تحتوي على بيانات فيها ضوضاء وقيم غير مرتبطة بالبيانات الاصلية المطلوب تصنيفها. ومع ذلك، فإنه يفشل في التعامل مع وجود اختلاف في الكثافة المحلية الموجودة ضمن البيانات. وبالتالي، نحتاج الي طريقة للتعامل مع اختلاف كثافة البيانات في المجموعة.

في هذه الأطروحة تم عرض اقتراح لتعزيز الخوارزمية DBSCAN، الذي يكشف عن مجموعات من مختلف الاشكال والاحجام التي تختلف في الكثافة. و نقدم ثلاث خوارزميات جديدة. الخوارزمية الاولي المقترحة VMDBSCAN، حيث تم استخدام فكرة الاهتزاز لحل مشكلة فصل مجموعة البيانات المرتبطة عن بعضها عند استخدام خوارزمية DBSCAN القديمة. الخوارزمية الثانية المقترحة هي DMDBSCAN حيث تتغلب هذه الخوارزمية الجديدة علي مشكلة استخدام قيمة واحدة لنصف القطر EPS المستخدم لقياس المسافات بين النقاط. أي نقاط يتم ايجاد المجموعة التي تنتمي اليها يتم عزلها وتطبيق الخوارزمية علي باقي النقاط. الخوارزمية الاخيرة تجمع الخوارزمية الاولي مع الثانية لإيجاد افضل تجميع للبيانات، وبالتالي نحصل علي افضل النتائج.

التجارب اجريناها علي بيانات صناعية وثلاث مجموعات من البيانات الحقيقية من UCI. هذه البيانات تحتوي علي نقاط لها كثافة متغيرة لاختبار الخوارزميات التي اقترحناها. النتائج النهائية أظهرت كفاءة ودقة الخوارزميات المقترحة، حيث حصلنا علي نتائج ممتازة عند مقارنة الخوارزميات المقترحة مع خوارزمية DBSCAN الأساسية وخوارزمية DVBSCAN. في البيانات الحقيقية حصلنا علي اقل نسبة خطأ عند دمج الخوارزمية VMDBSCAN مع DMDBSCAN وصلت الي 9.76 % لبيانات IRIS، بينما DVBSCAN وصلت نسبة الخطأ الي 17.22 %. لبيانات Haberman وصلت نسبة الخطأ الي 12.54 %، بينما DVBSCAN وصل الي 32.65 %. لبيانات Glass وصلت نسبة الخطأ الي 33.43 %، بينما في DVBSCAN وصلت نسبة الخطأ الي 41.23 %.

An Improvement for DBSCAN Algorithm for Best Results in Varied Densities

Mohammad N. T. Elbatta

Abstract

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a base algorithm for density based clustering. It can find out the clusters of different shapes and sizes from a large amount of data, which is containing noise and outliers. However, it fails to handle the local density variations that exist within the cluster. In this thesis, an enhancement of DBSCAN algorithm is proposed, which detects the clusters of different shapes, sizes that differ in local density. We introduce three new algorithms. Our first proposed algorithm Vibration Method DBSCAN (VMDBSCAN) first finds out the “core” of each cluster – clusters generated after applying DBSCAN -. Then it “vibrates" points toward cluster that has the maximum influence on these points.

The second proposed algorithm is Dynamic Method DBSCAN (DMDBSCAN). It selects several values of the radius of a number of objects (Eps) for different densities according to a k-dist plot. For each value of Eps, DBSCAN algorithm is adopted in order to make sure that all the clusters with respect to corresponding density are clustered. And for the next process, the points that have been clustered are ignored, which avoids marking both denser areas and sparser ones as one cluster.

The last algorithm Vibration and Dynamic DBSCAN (VDDBSCAN) combines the first and second algorithms to produce best clustering results. It begins by searching for each level of density its corresponding Eps, then it will use DBSCAN to find all clusters, finally, it will use vibration method of VMDBSCAN to solve the problem of splitting clusters.

Experimental results are obtained from artificial data sets and three real data sets from UCI. These data sets are of varied densities to match our goal for testing the proposed algorithms. The final results show that our algorithms get a good results with respect to the original DBSCAN algorithm and DVBSCAN algorithm. We obtain the correct number of clusters of artificial data sets. In the real data sets, the error rate is decreased when merging VMDBSCAN with DMDBSCAN and reach 9.76 % for IRIS data set, while when using DVBSCAN it was 17.22 %. For Haberman data set it reach 12.54 %, while when using DVBSCAN it was 32.65 %. For Glass data set it reach 33.43 %, while when using DVBSCAN it was 41.23 %.

Keywords: Cluster, Vibrating, Core, Density Different Cluster, Variance Density, DBSCAN, Total Density Function, K-dist.

Chapter 1
Introduction

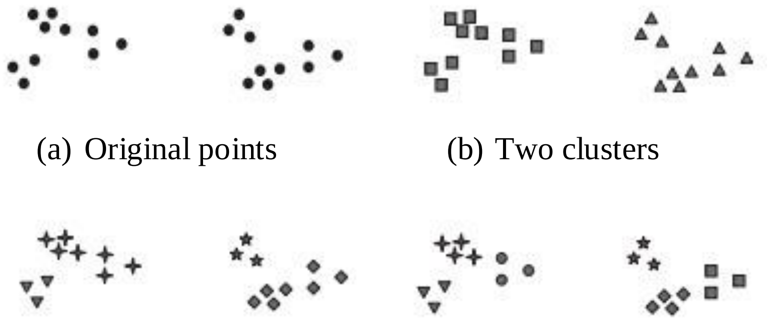
1.1 What Is Clustering

The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering. A cluster is a collection of data points that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. A cluster of data points can be treated collectively as one group and so may be considered as a form of data compression [1].

Clustering is also called data segmentation in some applications because clustering partitions large data sets into groups according to their similarity. Clustering can also be used for outlier detection, where outliers (values that are "far away" from any cluster) may be more interesting than common cases. Applications of outlier detection include the detection of credit card fraud and the monitoring of criminal activities in electronic commerce [2].

Data mining has attracted a great deal of attention in the information industry and in society as a whole in recent years, due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge which can be used for applications ranging from market analysis, fraud detection, and customer retention, to production control and science exploration. Data mining can be viewed as a result of the natural evolution of information technology in a lot of functionalities such as data collection and database creation, data and advanced data analysis (involving data warehousing and data mining). Clustering, which divides the data to disparate clusters, is a crucial part of data mining. The objects within a cluster are "similar," whereas the objects of different clusters are "dissimilar" [3].

In many applications, the notion of a cluster in not well defined. To better understand the difficulty of deciding what constitutes a cluster, look at Figure 1.1, which shows twenty points and three different ways of dividing them into clusters. The shapes of markers indicate cluster membership. Figures 1.1(a) and 1.1(d) divide the data into two and six parts, respectively. However, the apparent division of each of the two larger clusters into three sub clusters may simply be an artifact of human visual system. Also, it may not be unreasonable to say that the points form four clusters, as shown in Figure 1.1(c). This figure illustrates that the definition of a cluster is imprecise and that the best definition depends on the nature of data and the desired results [2].



(a) Original points

(b) Two clusters

(c) Four clusters

(d) Six clusters

Figure 1.1 Different ways of dividing points into clusters [2]

1.2 Types of Clustering

Clustering methods can be categorized into two main types: fuzzy clustering and hard clustering. In fuzzy clustering, data points can belong to more than one cluster with probabilities. In hard clustering, data points are divided into distinct clusters, where each data point can belong to one and only one cluster. These data points can be grouped with many different techniques, such as partitioning, hierarchical, density based, grid based, and model based Figure 1.2 [4, 5, 6].

Partitioning algorithms minimize a given clustering criterion by iteratively relocating data points between clusters until a (locally) optimal partition is attained. The most popular partition-based clustering algorithms are the k-means [7] and the k-mediod [8]. The advantage of the partition-based algorithms is the use an iterative way to create the clusters, but the limitation is that the number of clusters has to be determined by user and only spherical shapes can be determined as clusters. This limitation because clusters are separable in a way so that the mean value converges towards the cluster center.

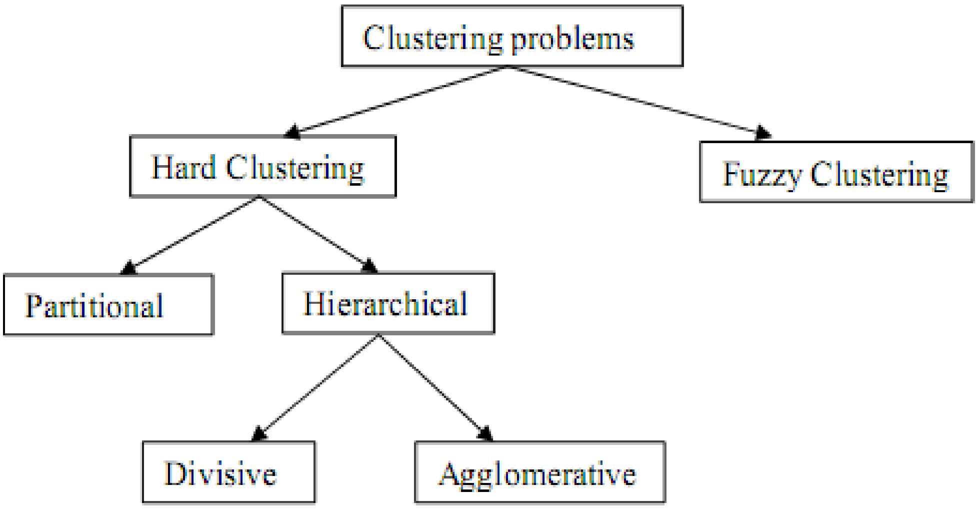


Figure 1.2 Diagram of clustering algorithms [4]

Hierarchical algorithms provide a hierarchical grouping of the objects. These algorithms can be divided into two approaches, the bottom-up or agglomerative and the top-down or divisive approach. In case of agglomerative approach, at the start of the algorithm, each object represents a different cluster and at the end, all objects belong to the same cluster. In divisive method at the start of the algorithm all objects belong to the same cluster, which is split, until each object constitute a different cluster. Hierarchal algorithms create nested relation-ships of clusters, which can be represented as a tree structure called dendrogram [9]. The resulting clusters are determined by cutting the dendrogram by a certain level. Hierarchal algorithms use distance measurements between the objects and between the clusters. Many definitions can be used to measure distance between the objects, for example Euclidean, City-block (Manhattan), Minkowski etc.

Between the clusters one can determine the distance as the distance of the two nearest objects in the two clusters (single linkage clustering) [10], or as the two furthest (complete linkage clustering) [11], or as the distance between the mediods of the clusters. The disadvantage of the hierarchical algorithm is that after an object is assigned to a given cluster it cannot be modified later. Also only spherical clusters can be obtained. The advantage of the hierarchical algorithms is that the validation indices (correlation, inconsistency measure), which can be defined on the clusters, can be used for determining the number of the clusters. The popular hierarchical clustering methods are CHAMELEON [9], BIRCH [7] and CURE [8].

Density-based algorithms like DBSCAN [10] and OPTICS [11] find the core objects at first and they grow the clusters based on these cores and search for objects that are in a neighborhood within a radius of a given object. The advantage of these types of algorithms is that they can detect arbitrary forms of clusters and they can filter out the noise.

Grid-based algorithms quantize the object space into a finite number of cells (hyper-rectangles) and then perform the required operations on the quantized space. The advantage of this approach is the fast processing time that is in general independent of the number of data points. The popular grid-based algorithms are STING [12], WaveCluster [13], and CLIQUE [14].

Model-based algorithms find good approximations of model parameters that best fit the data. They can be either partitional or hierarchical, depending on the structure or model they hypothesize about the data set and the way they refine this model to identify partitioning. They are closer to density-based algorithms, in that they grow particular clusters so that the preconceived model is improved. However, they sometimes start with a fixed number of clusters and they do not use the same concept of density. The most popular model-based clustering methods is EM [15].

Fuzzy algorithms suppose that no hard clusters exist on the set of objects, but one object can be assigned to more than one cluster. The best known fuzzy clustering algorithm is FCM (Fuzzy C-MEANS) [16].

So we can summarize the clustering algorithms as follows [12]:

1. Hierarchical Methods
    - Agglomerative Algorithms
    - Divisive Algorithms
2. Partitioning Methods

- Relocation Algorithms
- Probabilistic Clustering
- k-medoids Methods
- k-means Methods
- Density-Based Algorithms

3. Density-Based Connectivity Clustering
4. Density Functions Clustering
5. Grid-Based Methods
6. Methods Based on Co-Occurrence of Categorical Data
7. Constraint-Based Clustering
8. Clustering Algorithms Used in Machine Learning
    - Gradient Descent and Artificial Neural Networks
    - Evolutionary Methods
9. Scalable Clustering Algorithms
10. Algorithms For High Dimensional Data
    - Subspace Clustering
    - Projection Techniques
    - Co-Clustering Techniques

1.3 Types of Clusters

There are many algorithms that deal with the problem of clustering large number of objects. The different algorithms can be classified regarding different aspects. These methods can be categorized into partitioning methods [4, 5, 6], hierarchical methods [4, 7, 8], density based methods [10, 11, 17], grid based methods [12, 13, 14], and model based methods [18, 19]. Figure 1.3 depicts different types of clusters.

Well-Separated: In this type of cluster, each object is closer (or more similar) to every other object in the cluster than to any object not in the cluster Figure 1.3(a).

Prototype-Based: It is a cluster in which each object is closer(more similar) to the prototype that defines the cluster than the prototype of any other cluster. We commonly refer to prototype-based clusters as center-based clusters Figure 1.3(b).

Graph-Based: A cluster can be defined as a connected component; i.e., a group of objects that are connected to one another, but that have no connection to objects outside the group. An important example of graph-based clusters are contiguity-based clusters, where two objects are connected only if they are within a specified distance of each other. This implies that each object in a contiguity-based cluster is closer to some other object in the cluster than to any point in a different cluster Figure 1.3(c).

Density-Based: A cluster is a dense region of objects that is surrounded by a region of low density. Objects in these sparse areas - that are required to separate clusters - are usually considered to be noise and border points Figure 1.3(d).

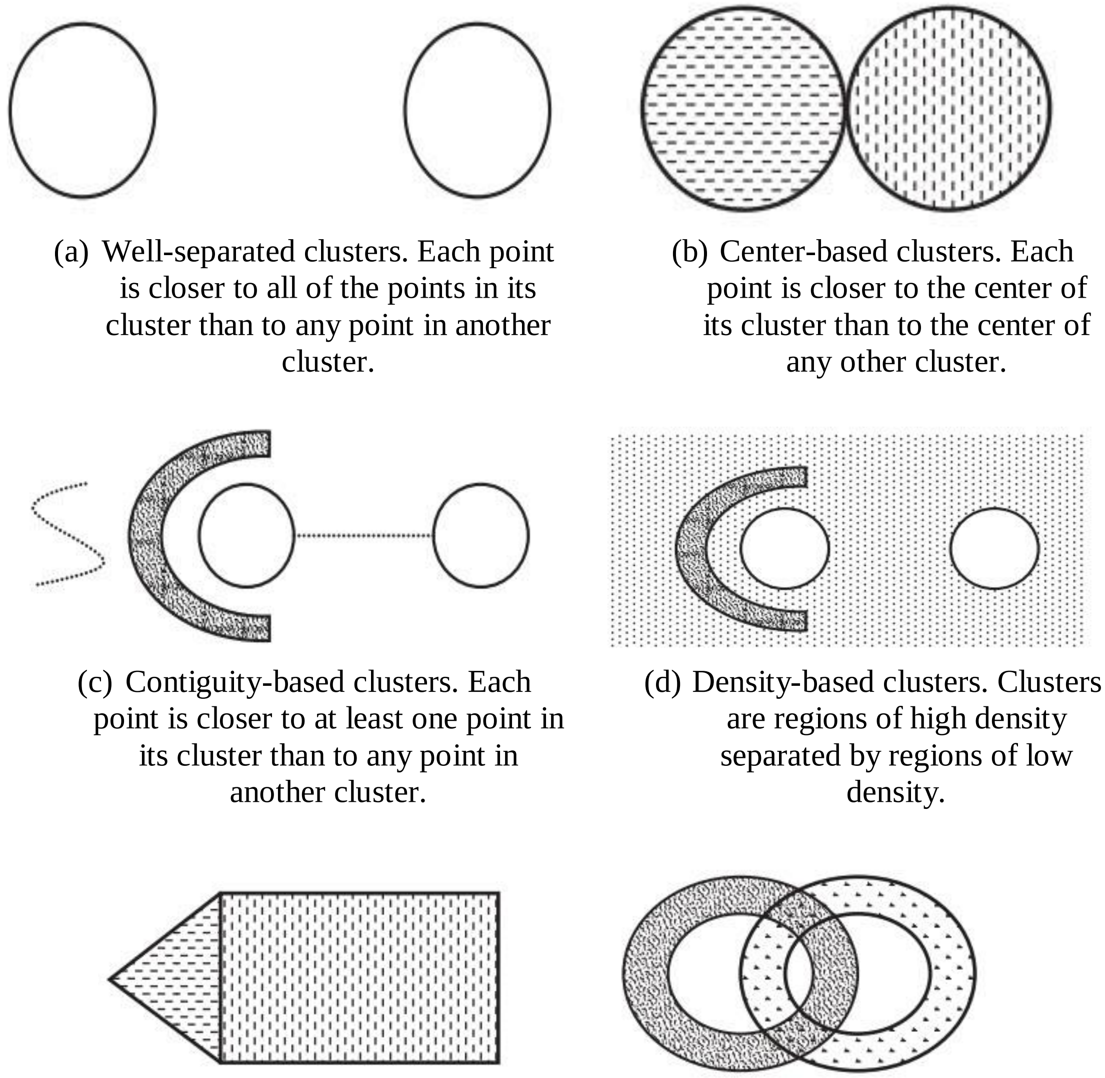
The famous density based clustering method is DBSCAN algorithm. In contrast to many newer methods, it features a well-defined cluster model called "density-reachability". It is based on connecting points within certain distance thresholds. However, it only connects points that satisfy a density criterion, in the original variant defined as a minimum number of other objects within this radius. A cluster consists of all density-connected objects plus all objects that are within these objects range. So, it will define any arbitrary shape of original data. Also, DBSCAN complexity is fairly low - it requires a linear number of range queries on the database - and that it will discover essentially the same results in each run, therefore there is no need to run it multiple times.

Shared-Property (Conceptual Clusters): A cluster of objects that share some property. It is distinguished from ordinary data clustering by generating a concept description for each generated class. Most conceptual clustering methods are capable of generating hierarchical category structures Figure 1.3(e).

1.4 Requirements of Clustering Algorithms

Clustering is a challenging field of research in which its potential applications pose their own special requirements. The following are typical requirements of clustering algorithms in data mining:

Ability to deal with different types of attributes [1]: Many algorithms are designed



(a) Well-separated clusters. Each point is closer to all of the points in its cluster than to any point in another cluster.

(b) Center-based clusters. Each point is closer to the center of its cluster than to the center of any other cluster.

(c) Contiguity-based clusters. Each point is closer to at least one point in its cluster than to any point in another cluster.

(d) Density-based clusters. Clusters are regions of high density separated by regions of low density.

(e) Conceptual clusters. Points in a cluster share some general property that derives from the entire set of points. (Points in the intersection of the circles belong to both.)

Figure 1.3 Different types of clusters [2]

to cluster numeric (interval-based) data. However, applications may require clustering other data types, such as binary, nominal (categorical), and ordinal data, or mixtures of these data types. Recently, more and more applications need clustering techniques for complex data types such as graphs, sequences, images, and documents.

Scalability [4]: Many clustering algorithms work well on small data sets containing fewer than several hundred data points; however, a large database may contain millions or even billions of objects, particularly in Web search scenarios. Clustering on only a sample of a given large data set may lead to biased results. Therefore, highly scalable clustering algorithms are needed.

Discovery of clusters with arbitrary shape [3]: Many clustering algorithms determine clusters based on Euclidean or Manhattan distance measures. Algorithms based on such distance measures tend to find spherical clusters with similar size and density. However, a cluster could be of any shape. Consider sensors, for example, which are often deployed for environment surveillance. Cluster analysis on sensor readings can detect interesting phenomena. We may want to use clustering to find the frontier of a running forest fire, which is often not spherical. It is important to develop algorithms that can detect clusters of arbitrary shape.

Able to deal with noise and outliers [3]: Most real-world data sets contain outliers and/or missing, unknown, or erroneous data. Sensor readings, for example, are often noisy, some readings may be inaccurate due to the sensing mechanisms, and some readings may be erroneous due to interferences from surrounding transient objects. Clustering algorithms can be sensitive to such noise and may produce poor-quality clusters. Therefore, we need clustering methods that are robust to noise.

Requirements for domain knowledge to determine input parameters [2]: Many clustering algorithms require users to provide domain knowledge in the form of input parameters such as the desired number of clusters. Consequently, the clustering results may be sensitive to such parameters. Parameters are often hard to determine, especially for high-dimensionality data sets and where users have yet to grasp a deep understanding of their data. Requiring the specification of domain knowledge not only burdens users, but also makes the quality of clustering difficult to control.

Time complexity [20, 21]: The time complexity of a clustering algorithm quantifies the amount of time taken by an clustering algorithm to run as a function of the size of the input to the problem. Real world problems need the time of any algorithm to be fast to do the processing and then output the results.

High dimensionality [22, 23]: A database or a data warehouse can contain several dimensions or attributes. Many clustering algorithms are good at handling low-dimensional data, involving only two to three dimensions. Human eyes are good at judging the quality of clustering for up to three dimensions. Finding clusters of data points in high-dimensional space is challenging, especially considering that such data can be sparse and highly skewed.

Incremental clustering and insensitivity to input order [24]: In many applications, incremental updates (representing newer data) may arrive at any time. Some clustering algorithms cannot incorporate incremental updates into existing clustering structures and, instead, have to re-compute a new clustering from scratch. Clustering algorithms may also be sensitive to the input data order. That is, given a set of data points, clustering algorithms may return dramatically different clustering depending on the order in which the objects are presented. Incremental clustering algorithms and algorithms that are insensitive to the input order are needed.

Labeling or assignment: hard or strict vs. soft or fuzzy [25, 26, 27].

Interpretability and usability of results [1]: Users want clustering results to be interpretable, comprehensible, and usable. That is, clustering may need to be tied in with specific semantic interpretations and applications. It is important to study how an application goal may influence the selection of clustering features and clustering methods.

Insensitive to the initial conditions[3]:

However, clustering is a difficult combinatorial problem, and differences in assumptions and contexts in different communities have made the transfer of useful generic concepts and methodologies slow to occur [1].

1.5 Our Contribution

The contributions of this research is that we developed a new three clustering algorithms named VMDBSCAN, DMDBSCAN, and VDDBSCAN. All these algorithms are Density-based Multi-density algorithms. The first algorithm VMDBSCAN uses the idea of vibration to find the good clustering results in varied densities data sets. The second algorithm will solve the problem of using one global value of Eps for all data set, by find local value of Eps for each density in data set. The last algorithm will merge the first algorithm and second algorithm to find the best clustering results by find local values of Eps and apply vibration after that.

Main advantages of our proposed algorithms are highlighted hereunder:

- Enhancing the quality of clustering by solving the problem of varied densities.
- Because for large spatial databases it is very difficult to identify the initial parameters like number of clusters, shape and density in advance. Our proposed algorithms select several values of input parameter Eps for different densities, so, no need from the user to input these values.
- The clusters which are formed based on the proposed algorithms are easy to understand and it does not limit itself to the shapes of clusters.

Experimental results are shown in this thesis to demonstrate the effectiveness of the proposed algorithms. We compared our proposed algorithm results with other famous related algorithms results. And we present that our new proposed algorithm is the best one.

1.6 Thesis Structure

The rest of the thesis is organized as follows: Chapter 2 talks about related work which discusses the clustering problem. Chapter 3 summarizes the methodologies of the new proposed clustering algorithms and a number of concepts related to the techniques used in my proposed algorithms. Chapter 4 discusses and explains our proposed algorithms VMDBSCAN, DMDBSCAN and VDDBSCAN, and shows our contribution for improving efficiency of our proposed algorithms to cluster data sets. Chapter 5 shows the experimental results which compare our new proposed algorithms with other density-based algorithms; and finally, chapter 6 concludes the thesis and presents suggestions for future work.

Chapter2
Related Works

Data mining is the process of identifying hidden and interesting patterns from large data set, which can further be used in decision making and future prediction [28]. Clustering is an important technique of class identification in spatial databases. Objective of the clustering is to maximize the intra cluster similarity and minimizing the inter cluster similarity. Clustering is used to find useful patterns in unlabeled data.

Clustering has been extensively studied for over 40 years and across many disciplines due to its broad applications. Most books on pattern classification and machine learning contain chapters on cluster analysis or unsupervised learning [29].

In this chapter we survey the techniques proposed in the literature to overcome the limitations of DBSCAN. DBSCAN will be survey in more details on chapter 3.

2.1 Background

Several textbooks are dedicated to the methods of cluster analysis, including Hartigan [30], Jain and Dubes [31], Kaufman and Rousseeuw [32], and Arabie, Hubert, and De Sorte [33]. There are also many survey articles on different aspects of clustering methods. Recent ones include Jain, Murty, and Flynn [34], Parsons, Haque, and Liu [35], and Jain [36].

For partitioning methods, the k -means algorithm was first introduced by Lloyd [37], and then by MacQueen [38]. Arthur and Vassilvitskii [39] presented the k-means++ algorithm. A filtering algorithm, which uses a spatial hierarchical data index to speed up the computation of cluster means, is given in Kanungo, Mount, Netanyahu, Piatko, Silverman, and Wu [40].

The k-medoids algorithms of PAM and CLARA were proposed by Kaufman and Rousseeuw [32]. The k-modes (for clustering nominal data) and prototypes (for clustering hybrid data) algorithms were proposed by Huang [41]. The k-modes clustering algorithm was also proposed independently by Chaturvedi, Green, and

Carroll [42, 43]. The CLARANS algorithm was proposed by Ng and Han [44]. Ester, Kriegel, and Xu [45] proposed techniques for further improvement of the performance of CLARANS using efficient spatial access methods, such as R*-tree and focusing techniques. A k-means-based scalable clustering algorithm was proposed by Bradley, Fayyad, and Reina [46].

Partitioning techniques like k-means and PAM clustering algorithms assume clusters are globular and are of similar sizes. Both fail in large variation in cluster sizes and when cluster shapes are convex as in Figure 2.1 below. The data set in Figure 2.1 below contains two convex clusters. K-means and PAM clustering algorithms fail to find the correct clusters, so that the right cluster take points from the left one and the vice versa, and it is wrong result.

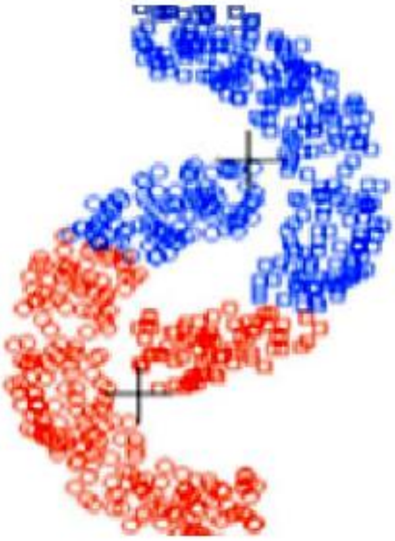


Figure 2.1 Clustering with k-means and PAM algorithms

Hierarchical Techniques like CURE [47] and ROCK [48] clustering algorithms use static models to determine the most similar cluster to merge in the hierarchical clustering. CURE measures the similarity of two clusters based on the similarity of the closest pair of the representative points belonging to different clusters, without considering the internal closeness (i.e., density or homogeneity) of the two clusters involved. It fails to take into account special characteristics and shapes as in Figure 2.2 below, we get a wrong clustering result. ROCK measures the similarity of two clusters by comparing the aggregate inter-connectivity of two clusters against a user-specified static inter-connectivity model, and thus it ignores the potential variations in the inter-connectivity of different clusters within the same data set.

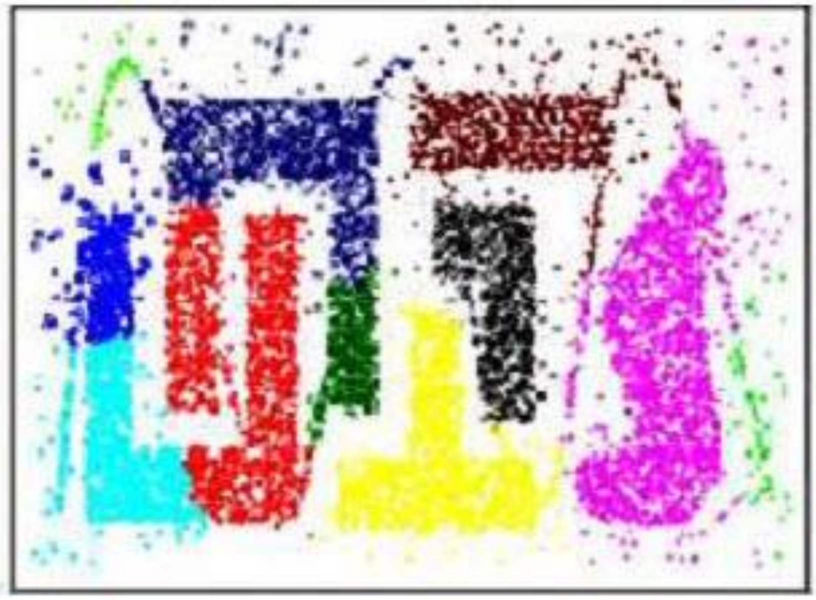


Figure 2.2 Clustering an artificial data set with CURE algorithm

DBSCAN is an important and widely used technique for class identification in spatial databases [3]. Many variations to the DBSCAN exist. DDBSCAN was first proposed in [14], for clustering over large data sets. Two major drawbacks are seen with DBSCAN:

1. The time complexity which reaches to O(n^2) in worst case.
2. The accuracy of clustering over the varied densities [15].

Ankerst, Breunig, Kriegel, and Sander [49] developed OPTICS, a cluster ordering method that facilitates density-based clustering without worrying about parameter specification.

OPTICS [11] algorithm is an improvement of DBSCAN to deal with variance density clusters. OPTICS does not assign cluster memberships but this algorithm computes an ordering of the objects based on their reachability distance for representing the intrinsic hierarchical clustering structure. Pei et al [50] proposed a nearest-neighbor cluster method, in which the threshold of density (equivalent to Eps of DBSCAN) is computed via the Expectation-Maximization (EM) [15] algorithm and the optimum value of k (equivalent to minimum points MinPts of DBSCAN) can be decided by the lifetime individual k. As a result, the clustered points and noise were separated according to the threshold of density and the optimum value of k.

In order to adapt DBSCAN to data consisting of multiple processes, an improvement should be made to find the difference in the mth nearest distances of processes. Roy and Bhattacharyya [51] developed new DBSCAN algorithm, which may help to find different density clusters that overlap. However, the parameters in this method are still defined by users. Lin and Chang [52] introduced new approach called GADAC, which may produce more precise classification results than DBSCAN does. Nevertheless, in GADAC, the estimation of the radius is dependent upon the density threshold δ, which can only be determined in an interactive way.

Pascual et al [53] developed density-based cluster method to deal with clusters of different sizes, shapes, and densities. However, the parameters neighborhood radius R, which is used to estimate the density of each point, have to be defined using prior knowledge and finding Gaussian-shaped clusters and is not always suit for clusters with arbitrary shapes.

EDBSCAN (An Enhanced Density Based Spatial Clustering of Application with Noise) [54] algorithm is another extension of DBSCAN; it keeps tracks of density variation which exists within the cluster. It calculates the density variance of a core object with respect to its E-neighborhood. If density variance of a core object is less than or equal to a threshold value and also satisfying the homogeneity index with respect to its neighborhood then it will allow the core object for expansion. But it calculates the density variance and homogeneity index locally in the E-neighborhood of a core object.

DD_DBSCAN [55] algorithm is another enhancement of DBSCAN, which finds the clusters of different shapes, sizes which differ in local density. But, the algorithm is unable to handle the density variation within the cluster. DDSC [56] (A Density Differentiated Spatial Clustering Technique) is proposed, which is again an extension of the DBSCAN algorithm. It detects clusters, which are having non-overlapped spatial regions with reasonable homogeneous density variations within them.

CHAMELEON [9] finds the clusters in a data set by two-phase algorithm. In first phase, it generates a k-nearest neighbor graph. In the second phase, it uses an agglomerative hierarchical clustering algorithm to find the cluster by combining the sub clusters.

In the following sections, we will study in more details the DBSCAN and some of algorithms variations to solve shortcomings of DBSCAN.

2.2 DENCLUE

DENCLUE (DENsity-based CLUstEring) [1, 17] is a clustering method based on a set of density distribution functions. The method is built on the following ideas:

1. The influence of each data point can be formally modeled using a mathematical function, called an influence function, which describes the impact of a data point within its neighborhood.
2. The overall density of the data space can be modeled analytically as the sum of the influence function applied to all data points.
3. Clusters can then be determined mathematically by identifying density attractors, where density attractors are local maxima of the overall density function.

Let x and y be objects or points in F^d, a d-dimensional input space. The influence function of data point y on x is a function, $f_B^y : F^d \to R_0^+$, which is defined in terms of a basic influence function f_B:

$$f_B^y(x) = f_B \ (x, y) \qquad (2.1)$$

This reflects the impact of y on x. In principle, the influence function can be an arbitrary function that can be determined by the distance between two objects in a neighborhood.

Problems of Denclue:

The method requires careful selection of the density parameter σ and noise threshold ξ – any point has density function value less than ξ is considered as outlier -, as the selection of such parameters may significantly influence the quality of the clustering results [29].

2.3 DBCLASD

This new clustering Algorithm DBCLASD [57] detects clusters with arbitrary shape and it does not require any input parameters. The efficiency of DBCLASD on large spatial databases is also very attractive.

DBCLASD algorithm works as follow:

1. DBCLASD is an incremental algorithm that is the assignment of a point to a cluster is based only on the points processed so far without considering the whole database.
2. It incrementally augments an initial cluster by its neighboring points as long as the nearest neighbor distance of the resulting cluster fits the expected distance distribution.
3. A set of candidates of a cluster is constructed using region queries which is supported by Spatial Access Methods (SAM).
    - The incremental approach implies an inherent dependency of the discovering clusters from the order of generating and testing candidates. The order of testing the candidates is crucial. Candidates which are not accepted by the test for the first time are called unsuccessful candidates.

DBCLASD Problems:

DBCLASD [57] Algorithm is based on the assumption that the points inside a cluster are uniformly distributed. The application of DBCLASD to earthquake catalogues shows that it also works effectively on real databases where the data is not exactly uniformly distributed. It is very efficient for large spatial databases. This algorithm fulfills all the requirements needed for designing a good clustering algorithm for spatial databases. So, it suffers when there are non-uniform points [29].

2.4 ST-DBSCAN

ST-DBSCAN [58] algorithm is constructed by modifying DBSCAN [10] algorithm. In contrast to existing density-based clustering algorithm, ST-DBSCAN [12] algorithm has the ability of discovering clusters with respect to non-spatial, spatial and temporal values of the objects.

The three modifications done in DBSCAN algorithm are as follows:

1. ST-DBSCAN algorithm can cluster spatial-temporal data according to non-spatial, spatial and temporal attributes.
2. DBSCAN does not detect noise points when it is of varied density but this algorithm overcomes this problem by assigning density factor to each cluster.
3. In order to solve the conflicts in border objects it compare the average value of a cluster with new coming value.

The algorithm starts with the first point p in database D:

1. This point *p* is processed according to DBSCAN algorithm and next point is taken.
2. Retrieve_Neighbors (object, Eps1, Eps2) function retrieves all objects density-reachable from the selected object with respect to Eps1, Eps2 and Minpts. If the returned points in E-neighborhood are smaller than Minpts input, the object is assigned as noise.
3. The points marked as noise can be changed later that is the points are not directly density-reachable but they will be density reachable.
4. If the selected point is a core object, then a new cluster is constructed. Then all the directly-density reachable neighbors of this core objects is also included.
5. Then the algorithm iteratively collects density-reachable objects from the core object using stack.

6. If the object is not marked as noise or it is not in a cluster and the difference between the average value of the cluster and new value is smaller than ΔE , it is placed into the current cluster.
7. If two clusters C1 and C2 are very close to each other, a point p may belong to both C1 and C2. Then point p is assigned to cluster which discovered first.

ST-DBSCAN Problems:

One of the main problems of ST-DBSCAN algorithm it needs three input parameter from the user [29].

2.5 DVBSCAN

In contrast to DBSCAN [10], DVBSCAN [59] algorithm handles local density variation within the cluster. The input parameters used in this algorithm are minimum objects (μ), radius, threshold values (α, λ). It calculates the growing cluster density mean and then the cluster density variance for any core object, which is supposed to be expanded further by considering density of its E-neighborhood with respect to cluster density mean. If cluster density variance for a core object is less than or equal to a threshold value and is also satisfying the cluster similarity index, then it will allow the core object for expansion.

It outperforms the DBSCAN, especially in case of local density as shown in Figure 2.3 and Figure 2.4.

The DVBSCAN algorithm has the following steps:

1. A cluster is formed by selecting core object.
2. Then it computes cluster density mean (CDM) is calculated for the growing cluster before allowing the expansion of an unprocessed core object.
3. Computation of the cluster Density variance (CDV) includes the E-neighborhood of the unprocessed core object with respect to CDM.
4. If CDV of growing cluster with respect to CDM is less than a specified threshold value α and the difference between the minimum and maximum object lying in the E-neighborhood of the object is less than a specified

threshold value λ then only an unprocessed core object is allowed for expansion.

5. Otherwise the object is simply added into the cluster.

DVBSCAN Problems:

The parameters α and λ are used to limit the amount of allowed local density variations within the cluster. These input parameters must be determined by the user. Other problem of DVBSCAN is the time complexity, which is high compared with other density based algorithms [29].

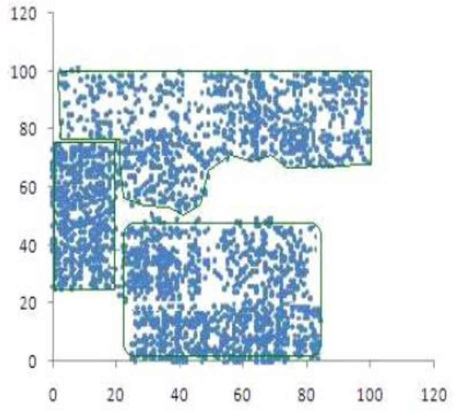


Figure 2.3 Clusters generated by DBSCAN algorithm [29]

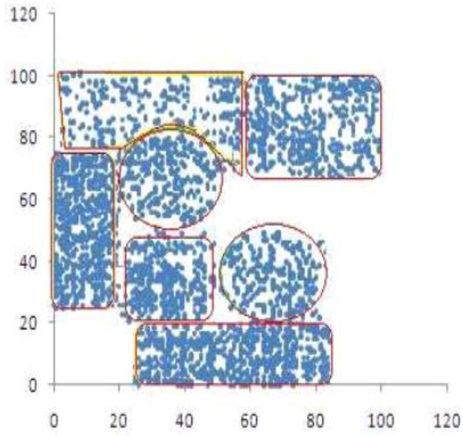


Figure 2.4 Clusters generated by DVBSCAN algorithm [29]

Chapter 3
Background

Our thesis presents new ideas to solve the problem of variant densities which most DBSCAN based algorithms suffer from. This chapter will explain some background and techniques used in clustering, that we will use it when applying our proposed algorithms. Also, we will review the DBSCAN, its advantages and its shortcomings.

3.1 Data Types in Clustering Analysis

The type of data is directly associated with data clustering, and it is a major factor to consider in choosing an appropriate clustering algorithm. The attributes can be Binary, Categorical, Ordinal, Interval-scaled or Ratio-scaled [1].

Binary: Have only two states: 0 or 1, where 0 means that the variable is absent and 1 means that it is present.

Categorical: also referred to as nominal, are simply used as names, such as the brands of cars and names of bank branches. That is, a categorical attribute is a generalization of the binary variable; it can take on more than two states.

Ordinal: resembles a categorical variable, except that the M states of the ordinal value are ordered in a meaningful sequence. For example, professional ranks are often enumerated in a sequential order.

Interval-scaled: are continuous measurements of a linear scale such as weight, height and weather temperature.

Ratio-scaled: make a positive measurement on a nonlinear scale. For example an exponential scale and the volume of scales over time are ratio-scaled attributes.

There are many other data types, such as image data, though we believe that once readers get familiar with these basic types of data, they should be able to adjust the algorithms accordingly.

3.2 Similarity and Dissimilarity

Similarity and Dissimilarity (Distances) play an important role in cluster analysis. Similarity measures, similarity coefficients, dissimilarity measures, or distances are used to describe quantitatively the similarity or dissimilarity of two data points or two clusters, that how similar two data points are or how similar two clusters are: the greater the similarity coefficient, the more similar are the two data points. Dissimilarity measure and distance are the other way around: the greater the dissimilarity measure or distance, the more dissimilar are the two data points or the two clusters. Consider the two data points x and y example. The Euclidean distance between x and y is calculated as:

$$f\text{Euclidean}(x, y) = \sqrt{(x_{i1} - y_{j1})^2 + (x_{i2} - y_{j2})^2 + \cdots + (x_{in} - y_{jn})^2} \qquad (3.1)$$

where i =(x_{i1}, x_{i2}, … , x_{in}) and j =(y_{j1}, y_{j2}, … , y_{jn}) are two *n*-dimensional data points.

The lower the distance between x and y, the more probability that x and y fall in the same cluster.

Every clustering algorithm is based on the index of similarity or dissimilarity between data points [7].

3.3 Scale Conversion

In many applications, the variables describing the objects to be clustered will not be measured in the same scales. They may often be variables of completely different types, some interval, others categorical. Scale conversion is concerned with the transformation between different types of variables. There are three approaches to cluster objects described by variables of different types.

One is to use a similarity coefficient, which can incorporate information from different types of variable. The second is to carry out separate analyses of the same set of objects, each analysis involving variables of a single type only, and then to synthesize the results from different analyses. The third is to convert the variables of different types to variables of the same type, such as converting all variables describing the objects to be clustered into categorical variables [6]. Any scale can be converted to any other scale. Several cases of scale conversion are described by Anderberg (1973) [6], including interval to ordinal, interval to nominal, ordinal to nominal, nominal to ordinal, ordinal to interval, nominal to interval, dichotomization (binarization) and so on.

3.4 Data Standardization and Transformation

In many applications of cluster analysis, the raw data, or actual measurements, are not used directly unless a probabilistic model for pattern generation is available. Preparing data for cluster analysis requires some sort of transformation, such as standardization or normalization [1].

Data standardization makes data dimensionless. It is useful for defining standard indices. After standardization, all knowledge of the location and scale of the original data may be lost. It is necessary to standardize variables in cases where the dissimilarity measure, such as the Euclidean distance, is sensitive to the differences in the magnitudes or scales of the input variables. The approaches of standardization of variables are essentially of two types: global standardization and within-cluster standardization[2].

Global standardization standardizes the variables across all elements in the data set. Within-cluster standardization refers to the standardization that occurs within clusters on each variable. Some forms of standardization can be used in global standardization and within-cluster standardization as well, but some forms of standardization can be used in global standardization only.

It is impossible to directly standardize variables within clusters in cluster analysis, since the clusters are not known before standardization [60].

For convenience, let $D^* = \{\boldsymbol{X}_1^*, \boldsymbol{X}_2^*, \dots, \boldsymbol{X}_n^*\}$ denotes the d-dimensional raw data set. Then the data matrix is an $n \times d$ matrix given by:

$$(\boldsymbol{X}_1^*, \boldsymbol{X}_2^*, \dots {}' \boldsymbol{X}_n^*)^{\mathrm{T}} = \begin{pmatrix} x_{11}^* & \cdots & x_{1d}^* \\ \vdots & \ddots & \vdots \\ x_{n1}^* & \cdots & x_{nd}^* \end{pmatrix} \qquad (3.2)$$

To standardize the raw data given in equation (3.1), we can subtract a location measure and divide a scale measure for each variable. That is,

$$x_{ij} = \frac{x_{ij}^*}{M_j} - \frac{L_j}{M_j} \qquad (3.3)$$

where x_{ij} denotes the standardized value, L_j is the location measure, and $M_j$is the scale measure.

We can obtain various standardization methods by choosing different L_j and M_j in equation (3.3) [1]. Some well-known standardization methods are mean, median, standard deviation, range, Huber's estimate, Tukey's biweight estimate, and Andrew's wave estimate [3]. Table 3.1 gives some forms of standardization, where $\overline{x}_j^*$, R_j^*, and σ_j^* are the mean, range, and standard deviation of the jth variable, respectively, i.e.,

$$\overline{x}_j^* = \frac{1}{n} \sum_{i=1}^{n} x_{ij}^* \qquad (3.4a)$$

$$R_j^* = \max_{1 \le i \le n} x_{ij}^* - \min_{1 \le i \le n} x_{ij}^* \qquad (3.4b)$$

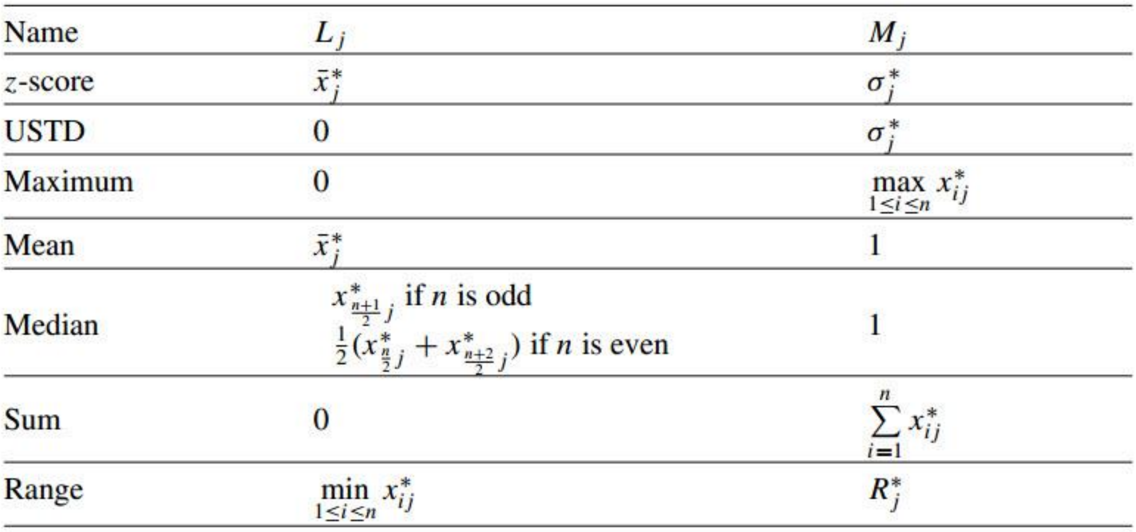
$$\sigma_j^* = \left[\frac{1}{n-1} \sum_{i=1}^{n} (x_{ij}^* - \overline{x}_{ij}^*)^{\,2}\right]^{\frac{1}{2}} \qquad (3.4c)$$

3.5 Overview of DBSCAN Algorithm

The DBSCAN [10] is density fundamental cluster formation. Its advantage is that it can discover clusters with arbitrary shapes and size. The algorithm typically regards clusters as dense regions of objects in the data space that are separated by regions of low-density objects. The algorithm has two input parameters, radius Eps and MinPts.

There are some basic concepts related to DBSCAN algorithm, we will present these concepts:

Table 3.1 Some data standardization methods, where $\overline{x}_j^*$, R_j^*, and σ_j^* are defined in equation (3.4)



Name	L_j	M_j
z-score	$\bar{x}_j^*$	σ_j^*
USTD	0	σ_j^*
Maximum	0	$\max\limits_{1 \le i \le n} x_{ij}^*$
Mean	$\bar{x}_j^*$	1
Median	$x^*_{\frac{n+1}{2}j}$ if n is odd $\frac{1}{2}(x^*_{\frac{n}{2}j} + x^*_{\frac{n+2}{2}j})$ if n is even	1
Sum	0	$\sum\limits_{i=1}^{n} x_{ij}^*$
Range	$\min\limits_{1 \le i \le n} x_{ij}^*$	R_j^*

1. The neighborhood within a radius Eps of a given object is called the E-neighborhood of the object.
2. If the E-neighborhood of an object contains at least a minimum number, MinPts, of objects, then the object is called a core object.
3. Given a set of objects, *D*, we say that an object *p* is directly density-reachable from object *q* if p is within the E-neighborhood of *q*, and *q* is a core object Figure 3.1.
4. An object *p* is density-reachable from object *q* with respect to Eps and MinPts in a set of objects, *D*, if there is a chain of objects $p_1, \ldots , p_n$, where $p_1 = q$ and $p_n = p$ such that p_{i+1} is directly density-reachable from *pi* with respect to Eps and MinPts, for $1 \le i \le n$, $p_i \in D$ Figure 3.1.
5. An object *p* is density-connected to object *q* with respect to Eps and MinPts in a set of objects, *D*, if there is an object $o \in D$ such that both *p* and *q* are density-reachable from *o* with respect to Eps and MinPts Figure 3.1.
6. Density reachability is the transitive closure of direct density reachability, and this relationship is asymmetric. Only core objects are mutually density reachable. Density connectivity, however, is a symmetric relation.

According to the above definitions, it only needs to find out all the maximal density-connected spaces to cluster the data points in an attribute space. And these density-connected spaces are the clusters. Every object not contained in any cluster is considered noise and can be ignored.

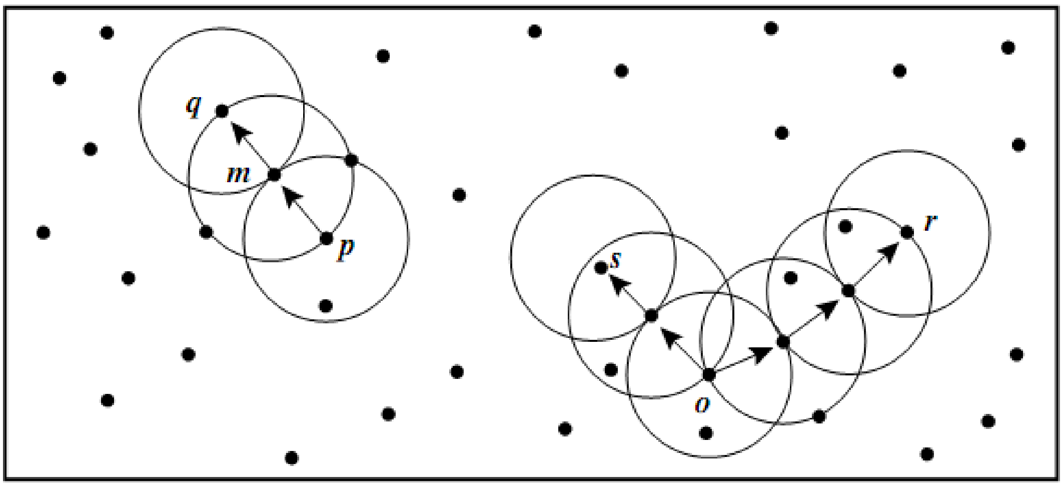


Figure 3.1 Density reachability and density connectivity in density-based clustering [2]

Explanation of DBSCAN Steps:

1. DBSCAN requires two parameters: Eps and MinPts. It starts with an arbitrary starting point that has not been visited. It then finds all the neighbor points within distance Eps of the starting point.
2. If the number of neighbors is greater than or equal to MinPts, a cluster is formed. The starting point and its neighbors are added to this cluster and the starting point is marked as visited. The algorithm then repeats the evaluation process for all the neighbors' recursively.
3. If the number of neighbors is less than MinPts, the point is marked as noise.
4. If a cluster is fully expanded (all points within reach are visited) then the algorithm proceeds to iterate through the remaining unvisited points in the data set.

DBSCAN Problems [29]:

1. When applying the DBSCAN algorithm, we must input two parameters:
   - Eps
   - MinPts

2. Another issues in Eps value that it is determined globally, so due to a single global parameter Eps, it is impossible to detect some clusters using one global-MinPts.
3. DBSCAN performance is poor on multi-density data sets. In the multi-density data set, DBSCAN may merge different clusters and may also neglect other clusters that assign them as noise.
4. Also the runtime complexity of constructing R*-tree and implementation of DBSCAN are not linear.

Chapter 4
Proposed Algorithms

In this chapter we will introduce a three new improved algorithms in the following, for the purpose of effective clustering analysis of data sets with varied densities. First algorithm will solve the problem of splitting clusters resulted from varied densities, by using the idea of vibration. Second algorithm will solve the problem of using one global Eps for all data set in varied densities, so it will use the idea of K-dist to find suitable Epsi for each density level. The last algorithm combines the first and second algorithms to produce best clustering results.

4.1 First Proposed Algorithm VMDBSCAN

In this section, we will present new method to solve the problem of splitting clusters when there are densities variations in data sets.

DBSCAN is a base algorithm for density based clustering. It can find out the clusters of different shapes and sizes from a large amount of data, which is containing noise and outliers. However, it fails to handle the local density variations that exists within the cluster. Thus, a good clustering method should allow a significant density variation within the cluster because, if we go for homogeneous clustering, a large number of smaller unimportant clusters may be generated. In this section, an enhancement of DBSCAN algorithm is proposed, which detects the clusters of different shapes, sizes that differ in local density. Our proposed method VMDBSCAN first finds out the “core” of each cluster – clusters generated after applying DBSCAN -. Then it “vibrates" points toward cluster that has the maximum influence on these points.

As shown in Figure 4.1, if we use DBSCAN directly to find the clusters of data set, we will obtain three clusters C1, C2, and C3. This is because there are some varied density between C2 and C3, which result in splitting one correct cluster into 2-clusters. So, when using Vibration method, we will have the correct number of clusters, which are C1, and merging C2 with C3 into one cluster.

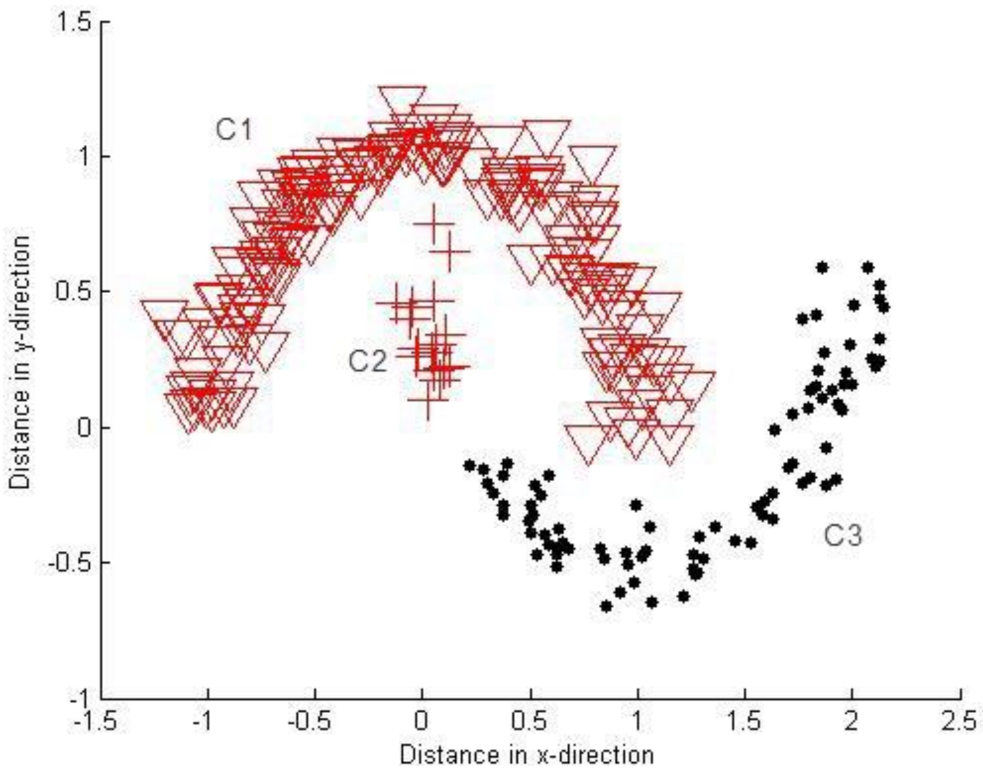


Figure 4.1 Problem of splitting clusters when using DBSCAN for two-moons data set

4.1.1 Description of Proposed Algorithm to Find Correct Number of Clusters:

One of the problems with DBSCAN is that it is has wide density variation within a cluster. To overcome this problem, new method based on DBSCAN algorithm is proposed in this section. It first clusters the data points using DBSCAN. Then, it finds the density functions for all data points within each cluster. The data point that has the minimum density function value will be the core for that cluster, since this point will be local maximum of the density function. After that, it computes the density variation of the data point with respect to the density of core object of its cluster against all densities of other core's clusters. According to the density variance, we do the movement for data points toward the new core. New core is one of other core's clusters, which has the maximum influence on the tested data point.

We intuitively present some definitions:

Definition 1. Suppose x and y are be two data points in a d-Dimensional feature space, D. The influence function of data point y on x is a function $f_B^y: D \to R_0^+$, where R_0^+ is real positive numbers, and can be defined as basic influence function f_B :

$$f_B^y(x) = f_B\ (x, y) \qquad (4.1)$$

In principle, the influence function can be an arbitrary function that can be determined by the distance between two objects in a neighborhood. The distance function, d(x, y), should be reflexive and symmetric, such as the Euclidean distance function.

$$f\text{Euclidean}(x, y) = \sqrt{(x_{i1} - y_{j1})^2 + (x_{i2} - y_{j2})^2 + \cdots + (x_{in} - y_{jn})^2} \qquad (4.2)$$

where i =(x_{i1}, x_{i2}, … , x_{in}) and j =(y_{j1}, y_{j2}, … , y_{jn}) are two n-dimensional data points.

Definition 2. Given a d-Dimensional feature space, D.The density function at a data point $x \in D$ is defined as the sum of all the influence to x from the rest of data points in D.

$$f_B^D(x) = \sum_{i=1}^{n} f_B^{xi}\ (x), 1 \leq i \leq n \qquad (4.3)$$

According to Definition 1 and Definition 2, we can calculate the density function for each data point in the space by applying equation (4.2) to calculate the Euclidian distance for every data point with respect to every data point in the data set, then applying equation (4.3) to sum all distances from that data point to every data point in the data set.

Definition 3. Core, the core object for each cluster is the object that has the minimum density function value according to Definition 2, since this point will be local maximum of the density function. That is, we can calculate the density function for each object in the cluster, which given initially by DBSCAN, and the object which has the minimum connection to all other objects will be the core for that cluster.

Definition 4. Total Density Function (*E)* represents the difference among the data points, which is based on the core. That is, the E for data point x_i is the difference between the data point x_i and the core of its cluster.

$$Ei = d(x_i, Cj) \qquad (4.4)$$

where $1 \leq i \leq n$, $1 \leq j \leq k$, n is number of points, k is number of cores.

In addition, according to our initial clusters which is given by the density-based clustering methods, we can take over the influence function (Definition 1) and density function (Definition 2) to calculate the E of the data points by subtracting the value of their density function to the value of the core's:

$$Ei = |f_B^D(xi) - f_B^D(C)| \qquad (4.5)$$

4.1.2 Vibration Process:

The main idea is to vibrate data points according to the density of the data point with respect to core (Definition 3), the core that represents each cluster, and measure the E of each data point as in (4.5). Then, if its Ei with respect to its core is greater than Ei for some other cores, vibrate all points in that cluster toward the core object which has the maximum influence on that object point, according to [62]:

$$x(i+1) = x(i) + \eta * \big(x(c) - x(i)\big) * (e^{-\frac{1}{2\sigma^2}}) \qquad (4.6)$$

where:

$\sigma = e^{-i/T}$

$x(i)$: is the current tested point

$x(c)$: is the current tested core

η: is the learning rate, determined by user.

T: is the control of reduction in sigma, it take the value of length of the data set.

Uses of η in the vibration equation to control the winner of the current cluster, and we can adapt it to get the best clustering results. T is used in formula to control the reduction in sigma, that is, as the time increased, the movement (vibrate) of the point toward the new core is reduced.

VMDBSCAN first clusters of the given data points using DBSCAN in order to find out the core of the clusters. And then, it vibrates the data points such that the difference between data points are lowered, and the similar clusters can be merged. We will describe the process of the VMDBSCAN by an example as below.

Suppose that: there are two-moons data set with 256 data points. Each data point is a 2-dimensional data point, and each point has two different attributes (x, y). Eps is 0.2, and MinPts are 5.

Step 1: Initial Clustering

The purpose of initial clustering is to get an initial understanding of the data points and find out the core of the clusters.

Firstly, we adopt DBSCAN algorithm to find initial clusters. Secondly, we compute the value of each data object's density function by using equation (4.3). Based on the data points' density function, we divide the data points into three clusters. Among each cluster, we choose the data point, which have the maximal density function value, as the core of the cluster. The result is shown in Figure 4.2.

Step 2: Vibrating

In this step, we calculate the E_i of the data points at first. In this example, as described in Definition 4, the E_i of each data point is the difference between the value of its density function and the value of the density function of the core as in equation (4.5). Next, if its E_i with respect to its core's density function is greater than Ei for some other core's density functions, vibrate all points in that cluster toward the core which has the maximum influence on that point, according to equation (4.6).

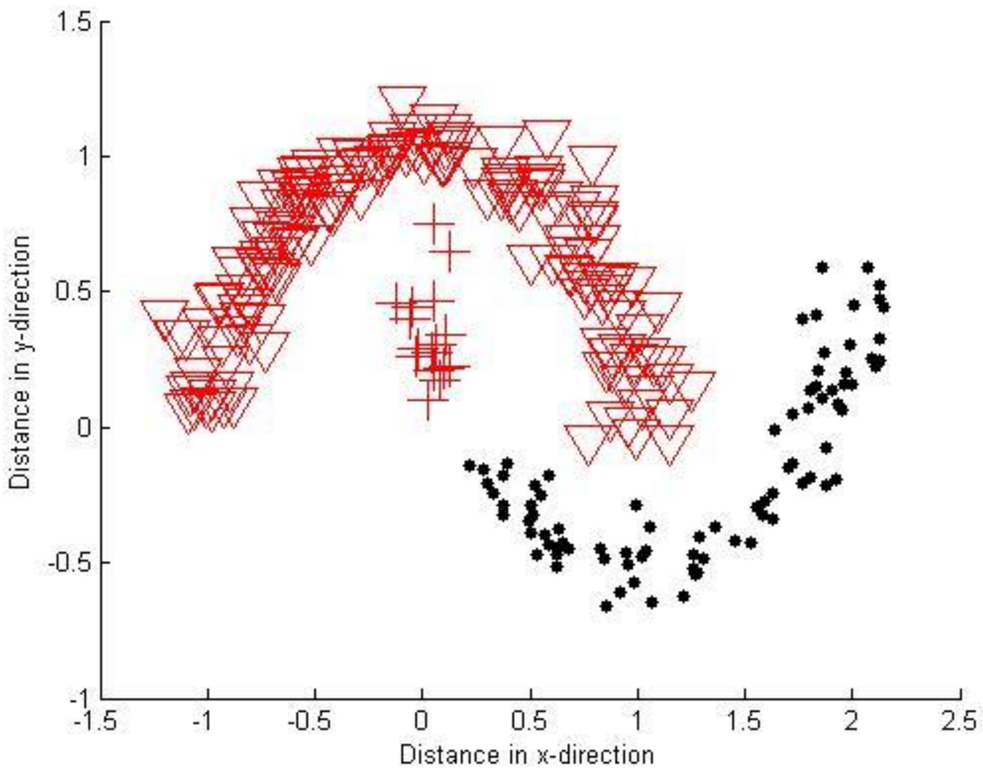


Figure 4.2 The distribution of the data points after applying DBSCAN algorithm

It shows in Figure 4.3 that the clusters with clear boundaries have somewhat merged. From the comparison of Figure 4.2 to Figure 4.3, we can figure out that those clusters, which are close to each other but are not density connected, have somewhat mixed together.

Formally, the proposed algorithm can be described as follow:

1. Data sets input and Data standardization.
2. Calculate the Density Function for all the data points.
3. Do Clustering for the data points using traditional DBSCAN algorithm.
4. Find out the core of each cluster.
5. Calculate the Density Function for all the data points within each cluster generated by traditional DBSCAN.
6. For each data point, if its E with respect to its core's density function is greater than with respect to other core's density function, then vibrate the data points in that cluster toward the core which has the maximum influence on that point.

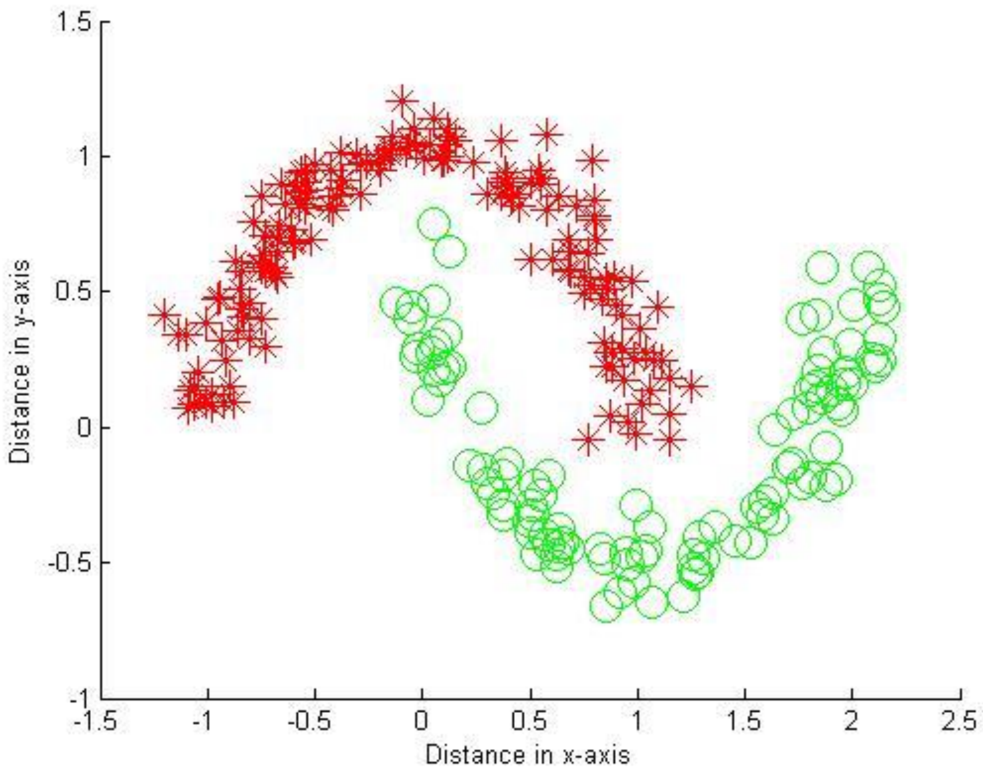


Figure 4.3 The Optimized distribution of the data points after applying VMDBSCAN

4.1.3 VMDBSCAN Algorithm Pseudo-Code:

The proposed algorithm to find the correct number of clusters in varied densities is shown as pseudo code in Algorithm 4.1.

The first step initializes the value of learning rate η, it can takes small values from [0,1] ; n is the number of data points in the data set. For each data point in the data set, algorithm computes the density function of this data point according to equation (4.3), and then store results in an array list of Point Density (d). Lines 8 and 9 of the algorithm call the DBSCAN algorithm to make initial clustering. From lines 9-12, algorithm search the core object for each cluster resulted from DBSCAN. Line 14 calculates the E for each point x_i with respect to its core object. Line 16 calculates the E for that point x_i with respect to all other core objects. From line 16 to Line 19 algorithm checks the effect of core objects on the data point x_i if the effect of its core object is less than other core objects c_j then vibrate the whole points which data point belongs to it toward the core c_j.

Algorithm 4.1: The pseudo code of the proposed technique VMDBSCAN to find correct number of clusters in varied densities	
Purpose:	*Merge splitting clusters if found*
Input:	*Data set of size n*
Output:	*Correct number of clusters*

Procedure:

```
Begin initialize $\eta$
For $i = 1$ to $n$
    find density$(x_i)$
end_For
 Initial Clusters with number of cores $c \leftarrow$
 DBSCAN(  )
For $j = 1$ to $c$
    find core$(c_j)$
end_For
For $i = 1$ to $n$
    $E_i = |density(x_i) - density(c_i)|$
    For $j = 1$ to $c$
        $E_{ic} = |density(x_i) - density(c_j)|$
        if $E_{ic} < E_i$
            vibrate the point by move it toword
            cluster which has maximum influence
            on the point
        else no vibrate
        end_If
    end_For
end_For
```

4.2 Second Proposed Algorithm DMDBSCAN:

In this section, we will present new method to solve the problem of using one global value of parameter Eps for all densities in data set, instead DMDBSCAN will use dynamic method to find suitable value of Eps for each density level of data set.

One of data mining primary method is clustering analysis. Clustering analysis has many methods such as density clustering. This method has advantages as:

1. Its clusters are easy to understand.
2. It does not limit itself to shapes of clusters.

But existing density-based algorithms have trouble in finding out all the meaningful clusters for data sets with varied densities. In this section we will introduce a new algorithm called DMDBSCAN for the purpose of varied-density data sets analysis. The basic idea of DMDBSCAN is that we need some methods to find the suitable values of parameters Eps for different densities according to k-dist plot, then we can use traditional DBSCAN algorithm to find clusters. For each value of Eps, DBSCAN algorithm is adopted to find all the clusters with respect to corresponding level of density. Then in the next step of algorithm, all points which clustered ignored. The final result will avoids marking both denser areas and sparser ones as one cluster.

To determine the parameters Eps and MinPts we need to look at the behavior of the distance from point to its k^{th} nearest neighbor, which is called k-dist. This k-dists are computed for all data points for some (k), then the plot sorted values in ascending order, after that, we expect to see the sharp change in the plotted graph. This sharp change at the value of k-dist corresponds with a suitable value of Eps for each density level of data set. In the K-dist plot some little changes show up for the changing density level of the examining dataset. But finally after a certain time a sharp change shows up and according to the DMDBSCAN algorithm the suitable value of (k) corresponds to this sharp changed level are 3. For example the Line (A) in Figure 4.4 shows a simple k-dist line for the value of k=3. We notice the value of Eps determined in this way depends on (k), but doesn't change dramatically as (k) changes.

DBSCAN can find many clusters which could not be found using some other clustering algorithms, like k-means, because DBSCAN uses a density-based definition of a cluster, which result in less relatively resistant to noise and can handle clusters of different shapes and sizes. However, the main weakness of DBSCAN is that it has trouble when the clusters have greatly varied densities.

For more description of new algorithm DMDBSCAN, 2-dimension data is chosen. Figure 4.5 shows the data. Obviously, there are two regions with respect to different densities in the data and data points of each region are uniformly distributed. The data set provides a clustering standard to estimate the accuracy of the result, for it has strong regularity and obvious clusters. In addition, as it has been already acknowledged that density-based clustering algorithms can find out clusters with any shape.

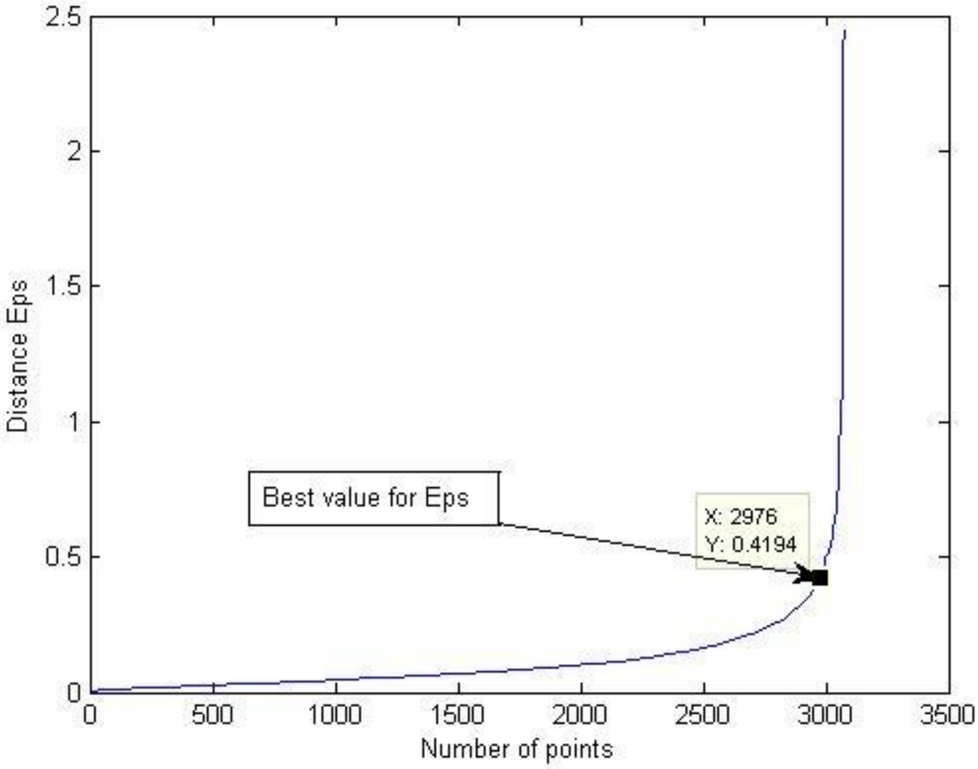


Figure 4.4 Points sorted in ascending order distance to the 3rd nearest neighbor to find the best value of Eps in varied densities data set

Suppose that the noise around the denser cluster C1 has the same density as the other cluster C2. If the Eps threshold is low enough that DBSCAN finds C2 as cluster, then C1 and the points surrounding it will become a single cluster. If the Eps threshold is high enough that DBSCAN finds C1 as a separate cluster, and the points surrounding are marked as noise, then C2 and the points surrounding it will also be marked as noise. DBSCAN also has trouble with high-dimensional data because density is more difficult to define for such data.

4.2.1 Description of Finding suitable Epsi For Each Density Level:

Formally, algorithm can describe our proposed to find suitable Epsi for each density level of data set as follow:

1. Data sets input and Data standardization.
2. Calculates and stores k-dist for each point and partition k-dist plots.
3. The number of densities is given intuitively by k-dist plot.

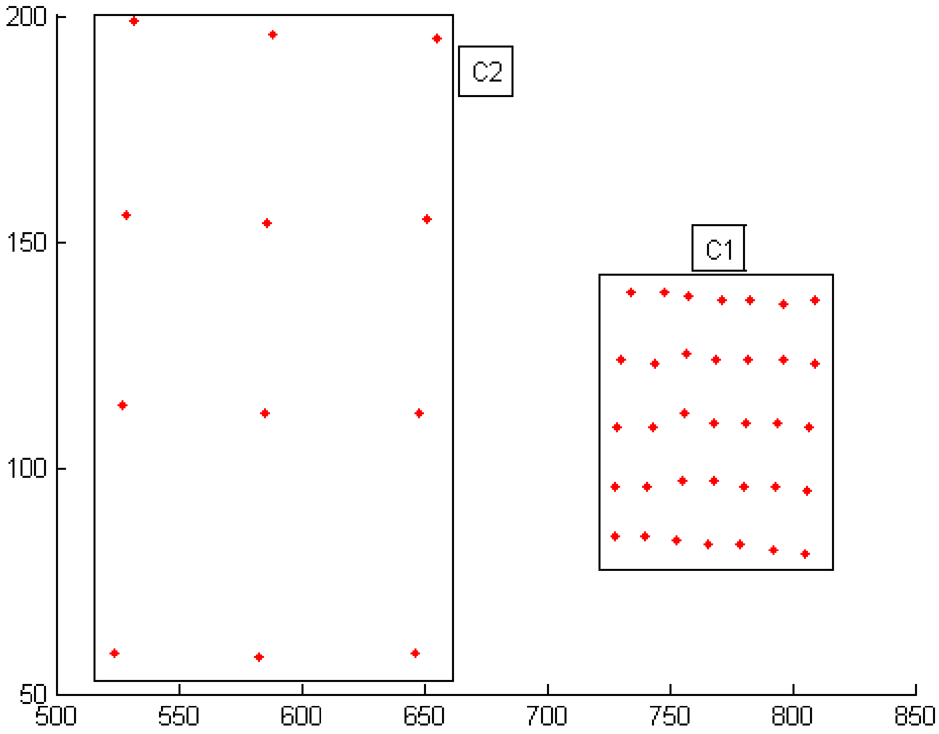


Figure 4.5 Problem of varied densities when there are two regions with respect to two different densities

4. Choose parameters Epsi automatically for each density.

In the first step, K-dist plot is drawn for not only selection of parameters Eps, but also analysis of density levels of the data set. If we have data sets with widely varied density, we notice that there will be some variation, depends on the density of the cluster and the random distribution of points, but the points of the same density level, the range of the variation will not be huge while a sharp changes that expected to see between two density levels. Thus there will be several smooth curves connected by greatly variation ones. For a data set of single-density, if its density does not vary widely, there is only one smooth curve in its k-dist plot.

Figure 4.2 shows a simple k-dist plot. Line A shows a simple k-dist line of a single-density data set. Figure 4.6 shows a simple line of a three varied-densities data set. We notice that there are sharp change in the curves which correspond to noise points connecting two smooth curves which stand for two density levels, as Line b and d which can be called level-turning lines. Line b connects line a and c, and line d

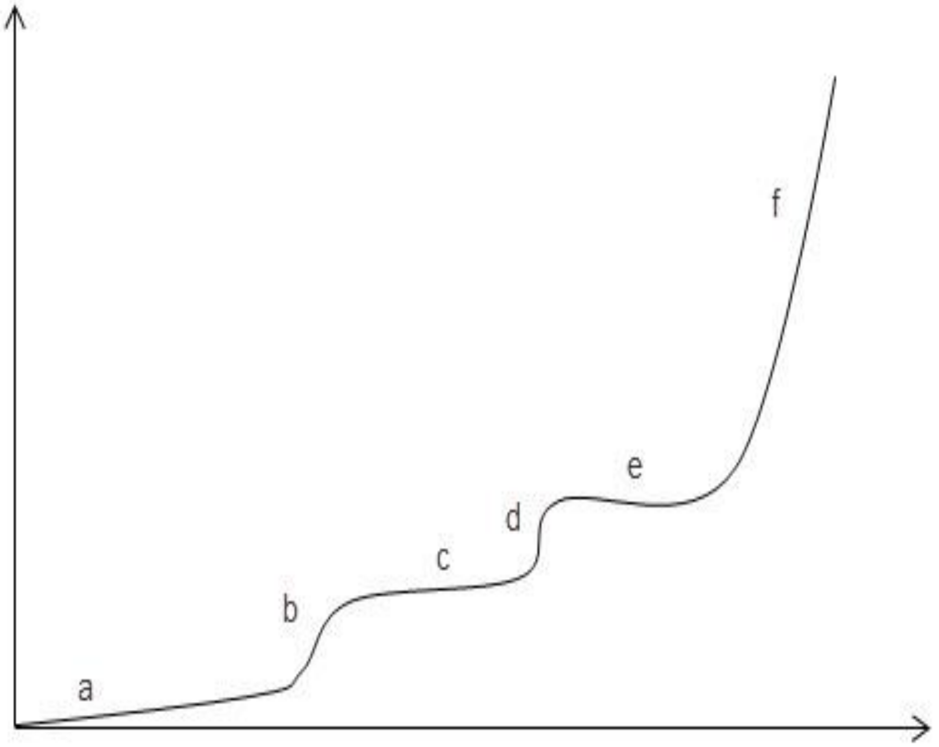


Figure 4.6 Data set with three levels of density

connects c and e, while a, c and e stand for different density levels. The outliers are shown with line f are not a level-turning line for it does not connect two smooth lines.

In Figure 4.4 we have three density levels, the result of that three suitable Eps. Combine line a and b as a sub-k-dist plot to select Eps1, and then take line c and d as a sub-k-dist plot for Eps2, e and f for Eps3.

4.2.2 DMDBSCAN Algorithm Pseudo-Code:

The proposed method of the algorithm to find suitable Epsi for each level of density is shown as pseudo code in Algorithm 4.2.

Algorithm 4.2: The pseudo code of the proposed technique DMDBSCAN to find suitable Epsi for each level of density.	
Purpose:	1. *To find suitable values of Eps*
Input:	2. *Data set of size n*
Output:	3. *Eps for each varied density*

Procedure:

```
$For\ i = 1\ to\ n$
    $For\ j = 1\ to\ n$
        $d(i,j) \leftarrow find\ distance(x_i, x_i)$
        $find\ minimum\ values\ of\ distances\ to\ nearest\ 3$
    $end\ for$
$end\ for$
$sort\ distances\ ascending\ and\ plot\ to\ find\ each\ value$
$Eps\ corresponds\ to\ critical\ change\ in\ curves$
```

4.3 Third Proposed Algorithm VDDBSCAN:

In this section, we will use the idea of algorithm VMDBSCAN with algorithm DMDBSCAN to introduce new algorithm VDDBSCAN. This algorithm will results in best clustering. It will solve the problem of using one global value of parameter Eps for all densities in data set, and overcomes the problem of splitting clusters when there are densities variations in data sets.

Our new algorithm can be divided and organized into three steps:

1. Data sets input and data standardization.
2. Overcoming on problems of varied densities.
3. Noise Elimination.

4.3.1 Data sets Input and Data Standardization:

This is the first step, it is necessary to standardize variables in cases where the dissimilarity measure, such as the Euclidean distance, is sensitive to the differences in the magnitudes or scales of the input variables. Section 3.4 describes in more details how data point attributes are standardized.

The z-score is a well-known form of standardization used for transforming normal variants to standard score form. We will use the z-score standardization to standardize the data point attributes. So, the first step in our proposed algorithm is to input the data set which is in a multi-density format. And then make the data standardization step.

4.3.2 Overcoming on Problems of Varied Densities:

At this step we need to find the different values of Epsi for each level of density. Then we will use this values of Epsi to implement the idea of vibration method to find the correct number of clusters.

Firstly, to determine the parameters Eps and MinPts we need to look at the behavior of the distance from point to its kth nearest neighbor, which is called k-dist. This k-dists are computed for all data points for some (k), then the plot sorted values in ascending order, after that, we expect to see the sharp change in the plotted graph. This sharp change at the value of k-dist corresponds with a suitable value of Eps for each density level of data set.

K-dist plot is drawn for not only selection of parameters Eps, but also analysis of density levels of the data set. If we have data sets with widely varied density, we notice that there will be some variation, depends on the density of the cluster and the random distribution of points, but the points of the same density level, the range of the variation will not be huge while a sharp changes that expected to see between two density levels. Thus there will be several smooth curves connected by greatly variation ones.

Secondly, it clusters the data points using traditional DBSCAN algorithm. Then, it finds the density functions for all data points within each cluster. The data point that has the minimum density function value will be the core for that cluster. After that, it computes the density variation of every data point with respect to the density of core object of its cluster against all densities of other core's clusters. According to the density variance, we do the movement for data points toward the new core. New core is one of other core's clusters, which has the maximum influence on the tested data point.

Formally, algorithm can describe our proposed to find suitable Epsi for each density level of data set and then use Vibration method to overcome on the problem of splitting clusters as follow:

1. Data sets input and Data standardization.

2. Calculates and stores k-dist for each point and partition k-dist plots.
3. The number of densities is given intuitively by k-dist plot.
4. Choose parameters Epsi automatically for each density.
5. Calculate the Density Function for all the data points.
6. Do Clustering for the data points using traditional DBSCAN algorithm,
7. Calculate the Density Function for all the data points again, and then find out the core of each generated cluster.
8. For each data point, if its *E* with respect to its core is greater than with respect to other cores, then vibrate the data points in that cluster.

4.3.3 Noise Elimination:

Any data set almost always contains outliers. These do not belong to any of the clusters. That is, the neighborhoods of outliers are generally sparse compared to points in clusters, and the distance of an outlier to the nearest cluster is comparatively higher than the distances among points of the points in clusters themselves. Each clustering method needs mechanism to eliminate outliers. In our proposed algorithm, outliers due to their larger distances from other points, tend to merge with other points less and typically grow at a much slower rate than actual clusters. Thus the clusters which are growing very slowly are identified and eliminated as outliers. Also, since the number of points in a collection of outliers is typically much less than the number in a cluster and that outliers form very small clusters, we can easily identify such small groups and eliminate them. Consequently, the final step, the outlier elimination, is necessary step for good clustering.

4.3.4 Proposed Algorithm VDDBSCAN Pseudo-Code:

Our proposed technique to do best clustering is shown as pseudo code in Algorithm 4.3.

Algorithm 4.3: The pseudo code of the proposed technique VDDBSCAN	
Purpose:	1. *Find Clusters From Data set*
Input:	2. *Data set of size n*
Output:	3. *Number of clusters*

Procedure:

```
$For\ i = 1\ to\ n$
    $For\ j = 1\ to\ n$
        $d(i,j) \leftarrow find\ distance(x_i, x_j)$
        $find\ minimum\ values\ of\ distances\ to\ nearest$
    $end\ for$
$end\ for$
$sort\ distances\ ascending\ and\ plot\ to\ find\ each\ value$
$initialize\ \eta$
$For\ i = 1\ to\ n$
    $find\ density(x_i)$
$end\_For$
$Initial\ Clusters\ c\ \leftarrow\ DBSCAN()$
$For\ j = 1\ to\ c$
    $find\ core(c_j)$
$end\_For$
$For\ i = 1\ to\ n$
    $E_i = |density(x_i) - density(c_i)|$
    $For\ j = 1\ to\ c$
        $E_{ic} = |density(x_i) - density(c_j)|$
        $if\ E_{ic} < E_i$
            $vibrate\ the\ point$
        $else\ no\ vibrate$
        $end\_If$
    $end\_For$
$end\_For$
```

4.4 Proposed Algorithms Properties:

- The proposed algorithms will solve the problem of splitting clusters resulted from varied densities. Also, they will solve the problem of using one global Eps for all data set in varied densities, by using local value for each data set density.
- The proposed algorithms can deal with varied data types of data sets.
- DBSCAN requires two input parameters Eps and MinPts, but the proposed algorithms does need these input parameters.
- The proposed algorithms discovers clusters of arbitrary shape. It holds good for large spatial databases.

Chapter 5
Experimental Results

In this chapter we will show a sufficient number of results with various types of data sets and with various numbers of points, we will compare between five algorithms namely: DBSCAN, DVBSCAN, and our proposed algorithms VMDBSCAN, DMDBSCAN, and VDDBSCAN.

5.1 Implementation Environment

The experiments are implemented on windows 7 operating system using MATLAB R2012a. Our hardware are hp laptop of Intel Core 2 Due processing power of 2.4 GHz CPU with 2GB RAM.

5.2 Artificial Data sets

In this section we will show all results when implement the above mentioned algorithms on artificial data sets.

We use three artificial two-dimensional data sets, since the results are easily visualized. The first data set is ball data set which is shown in Figure 5.1(a). It consists of 226 data points, each data point is 2-dimensional. The ball data set is one cluster and has different densities. The second data set is two-moons data set which is shown in Figure 5.1(b). It consists of 256 data points , each data point is 2-dimensional. The two-moons dataset is two nested moons and it is two-clusters with different densities. The third data set is shown in Figure 5.1(c) which consists of 385 data points with five clusters and some points of noise. Each data point is 2-dimensional.

5.2.1 DBSCAN

Figure 5.1(a) shows the original data set plotting. In Figure 5.2, after applying the DBSCAN algorithm, with MinPts = 5, Eps = 11.8, we get 2-clusters. And we notice

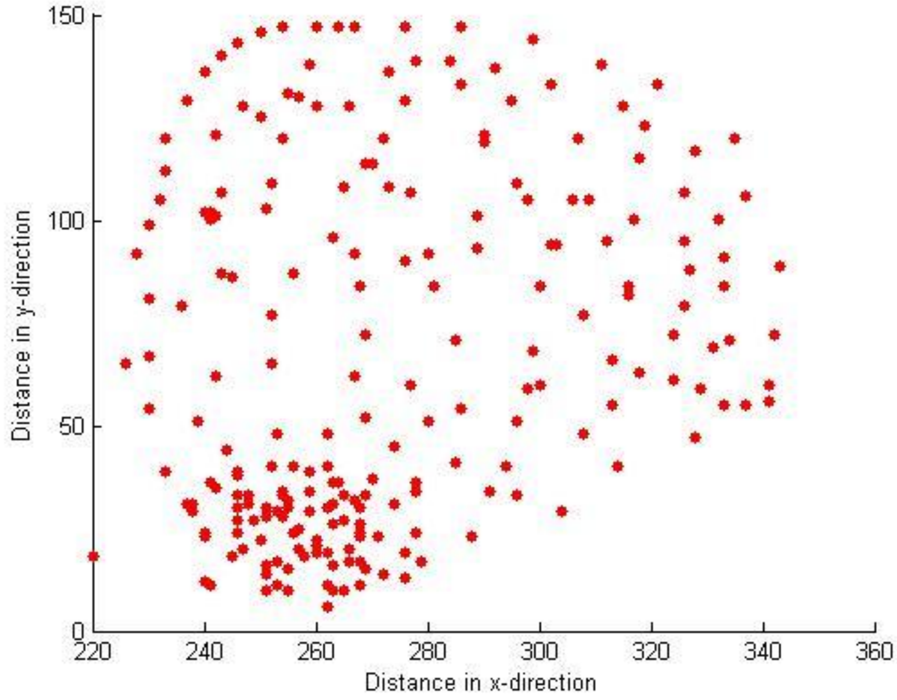


Figure 5.1(a) Ball data set with 226 data points with one cluster

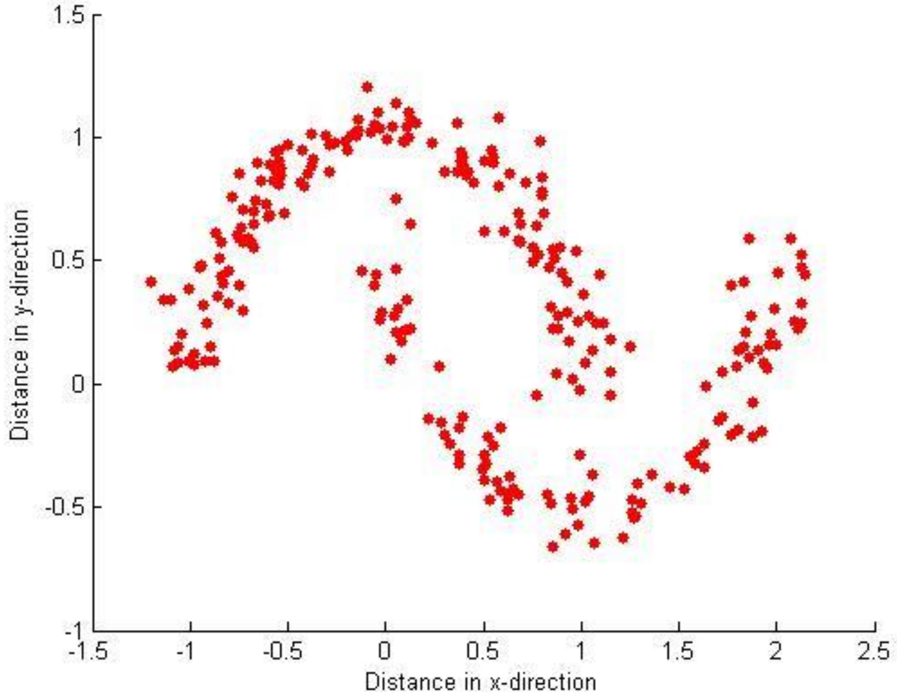


Figure 5.1(b) Two-moons data set with 256 data points with two clusters

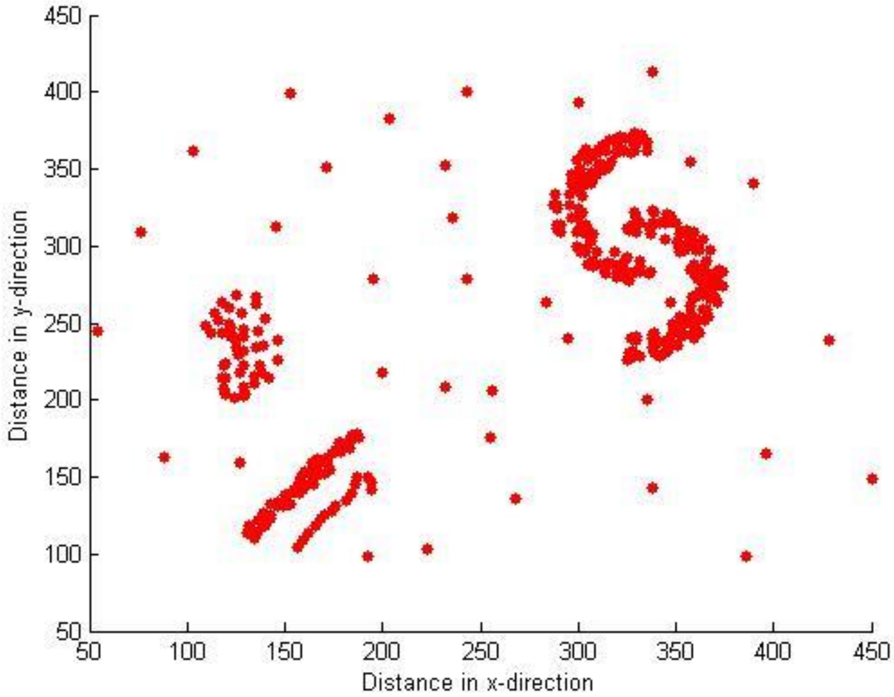


Figure 5.1(c) Moon data set with many points of noises and with 385 data points with five clusters

that there some points deleted by DBSCAN, as DBSCAN considered it as noise points, which resulted from varied density area.

Figure 5.1(b) shows the original data set plotting. Figure 5.3 shows the result of applying DBSCAN on the second data set, with MinPts = 5, and Eps = 0.2. The resulted clusters are 3-clusters.

Figure 5.1(c) shows the original data set plotting. In Figure 5.4, after applying the DBSCAN algorithm, with MinPts = 5, Eps = 8, we get 4-clusters. In this data set, DBSCAN treats some points as noise and remove them.

5.2.2 DVBSCAN

Figure 5.1(a) shows the original data set plotting. In Figure 5.5, after applying the DVBSCAN algorithm, with α = 100, λ = 50, μ = 20, Eps = 12, we get 1-cluster. A parameters α and λ are used to limit the amount of allowed local density variations within the cluster. The selection of parameter is very crucial. If the value of α is small, then it generates a large number of small unimportant clusters, on the other hand if we have large value of α then it merges a number of good quality clusters into single cluster.

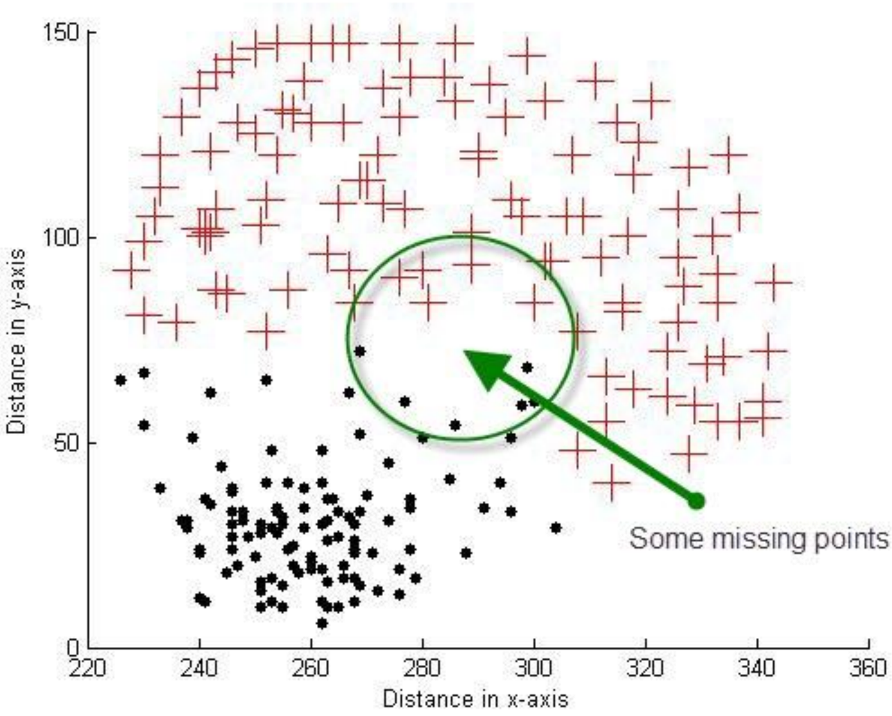


Figure 5.2 Two clusters generated by applying DBSCAN algorithm on ball data set for the values Eps = 11.8, and MinPts = 5

So if we select suitable value of α then it not only separates the sparse region but also separate the region which does not have the significant density variation.

After many tries, the previous parameters are the best values for ball data set. We notice that there some points deleted by DVBSCAN, as DVBSCAN considered it as noise points, which resulted from varied density area.

Figure 5.1(b) shows the original data set plotting. Figure 5.6 shows the result of applying DVBSCAN on the second data set, with α = 100, λ = 50, μ = 20, Eps = 0.25. The resulted clusters are 4-clusters.

Figure 5.1(c) shows the original data set plotting. In Figure 5.7, after applying the DVBSCAN algorithm, with α = 100, λ = 50, μ = 20, Eps = 8.5, we get 4-clusters.

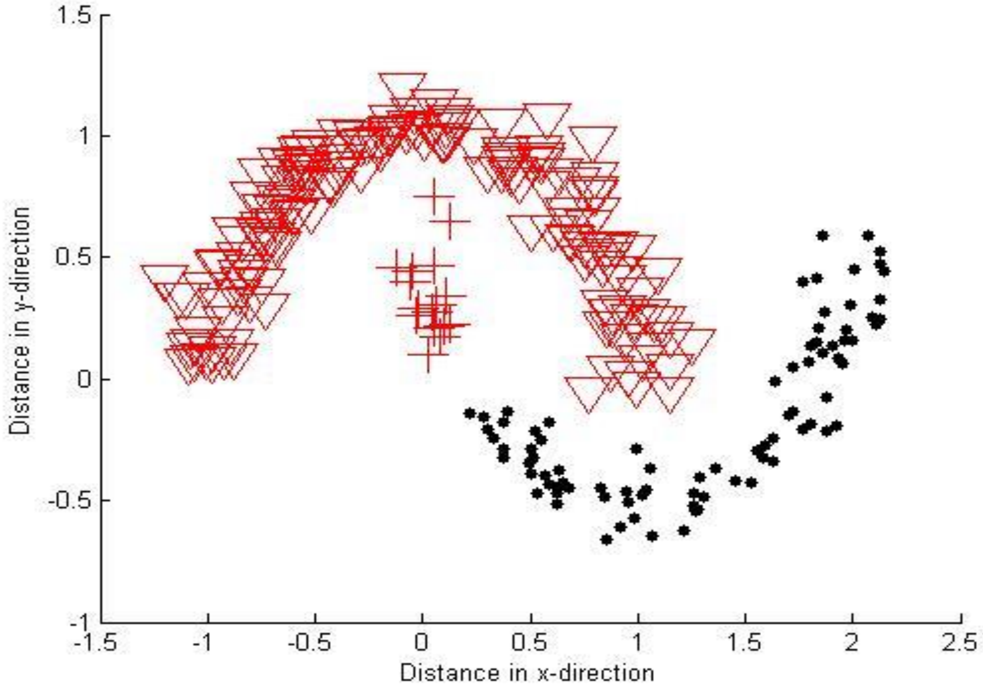


Figure 5.3 Three clusters generated by applying DBSCAN algorithm on two-moons data set for the values Eps = 0.2, and MinPts = 5

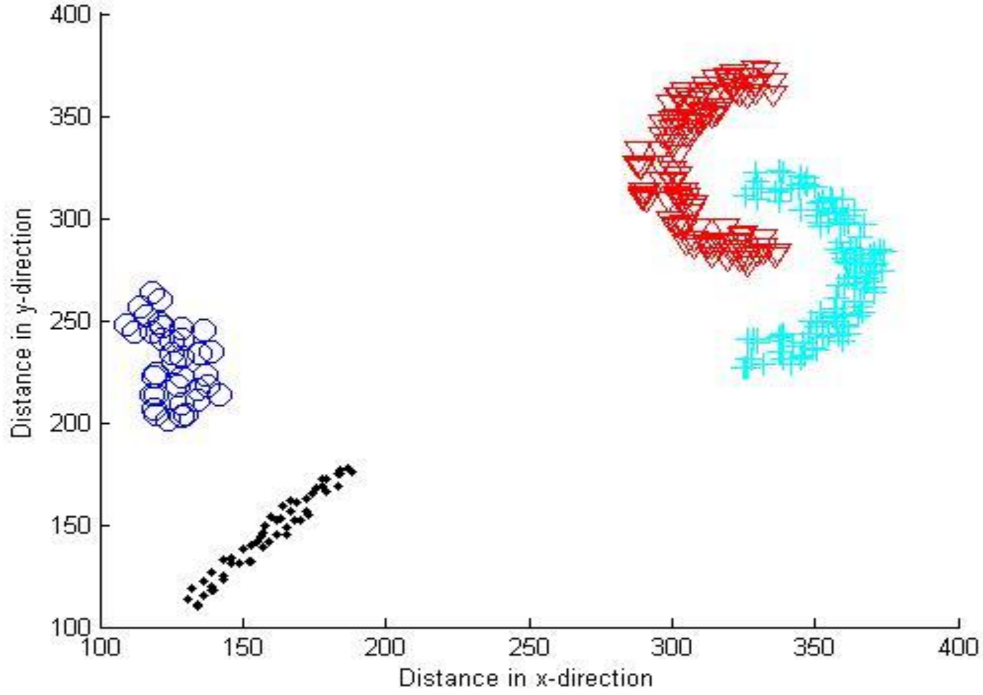


Figure 5.4 Four clusters generated by applying DBSCAN algorithm on moon data set for the values Eps = 8, and MinPts = 5

5.2.3 VMDBSCAN

Figure 5.1(a) shows the original data set plotting. In Figure 5.8, after applying our proposed algorithm with η = 0.0005 of equation (4.6), and Eps = 11.8, we get the correct number of clusters, that is, we have only 1-cluster. The used value of η which equal 0.0005 in the vibration equation controlled the winner of the current cluster and

we adapted it to get the best clustering results. Its best value in this experiment was found by trying many values and then selecting the best one that get correct clusters. We notice that the points which deleted by DBSCAN, as DBSCAN considered it as noise points, now they are appeared after applying our proposed algorithm.

Figure 5.1(b) shows the original data set plotting. Figure 5.9 shows the result of applying VMDBSCAN on the second data set with η = 0.005 of equation (4.6), its best value in this experiment was found by trying many values and then selecting the best one that get correct clusters. The Eps value = 0.26. We get the correct number of clusters, which are 2-clusters.

Figure 5.1(c) shows the original data set plotting. In Figure 5.10, after applying our proposed algorithm with η = 0.0005 of equation (4.6), its best value in this experiment was found by trying many values and then selecting the best one that get correct clusters. We get the correct number of clusters, that is, we have only 5-clusters.

5.2.4 DMDBSCAN

In this section, we will apply the proposed algorithm DMDBSCAN to solve the problem of using one global value of parameter Eps for all densities in data set, instead of DMDBSCAN will use dynamic method to find suitable values of Eps for each density level of data set. Figures 5.11, 5.12 and 5.13 show the plotting of k-dist with k=1,2 and 3 respectively (the nearest 3 distance) for ball data set, and sorting them in ascending order to find the best values of Eps which will adopt to DMDBSCAN to find the correct number of clusters.

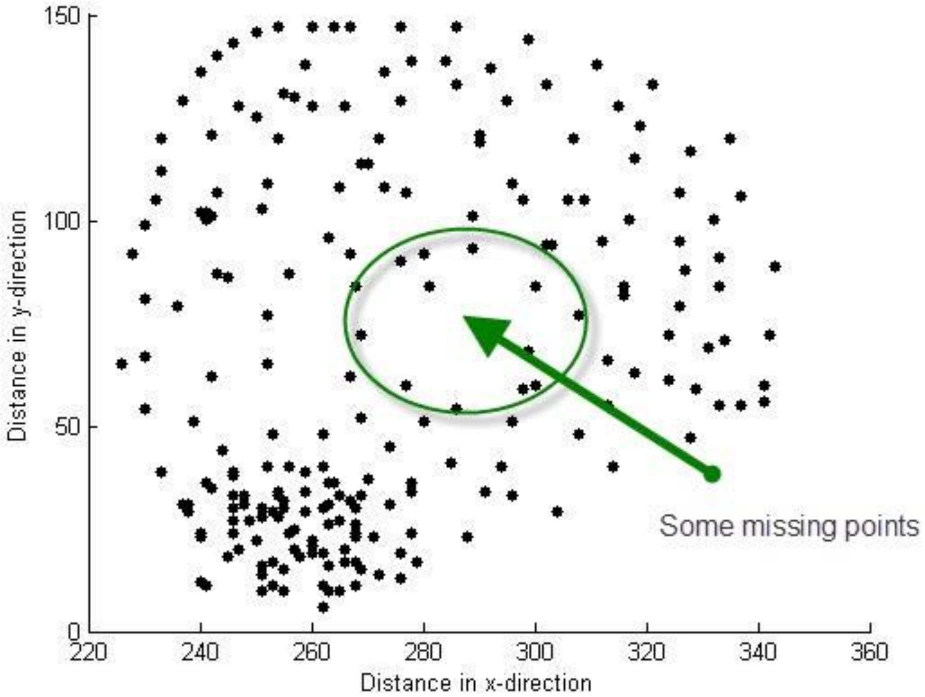


Figure 5.5 One cluster generated by applying DVBSCAN algorithm on ball data set for the values α = 100, λ = 50, μ = 20, and Eps = 12

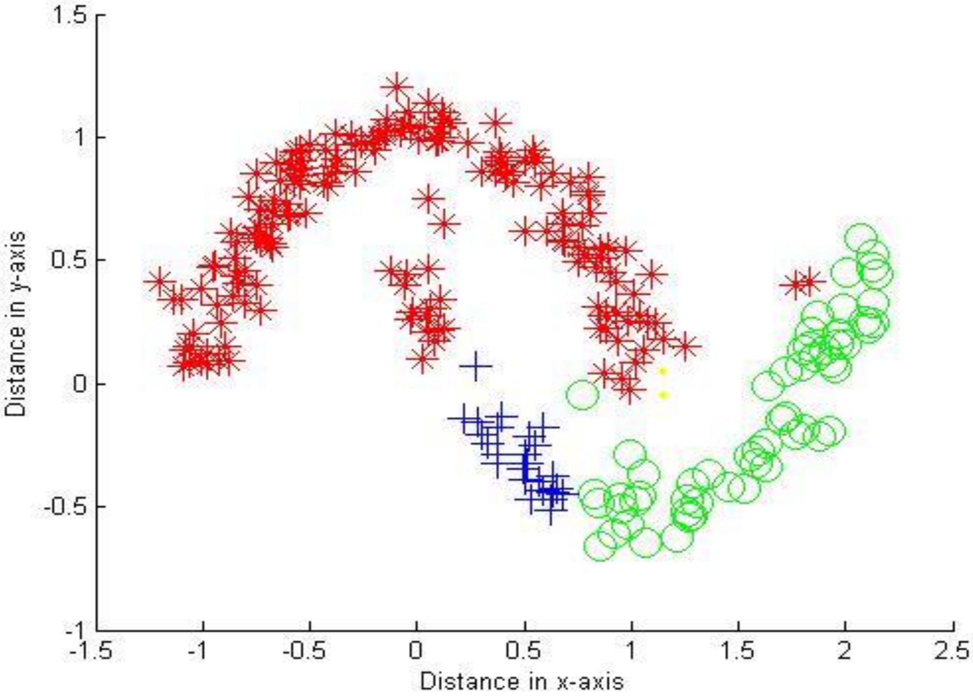


Figure 5.6 Four clusters generated by applying DVBSCAN algorithm on two-moons data set for the values, α = 100, λ = 50, μ = 20, and Eps = 0.25

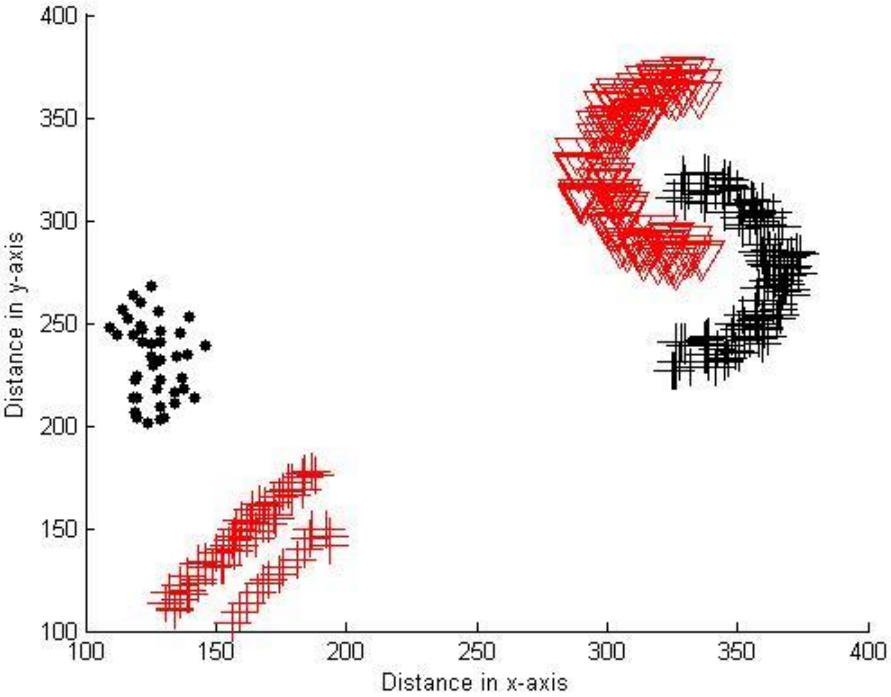


Figure 5.7 Four clusters generated by applying DVBSCAN algorithm on moon data set for the values, α = 100, λ = 50, μ = 20, and Eps = 8.5

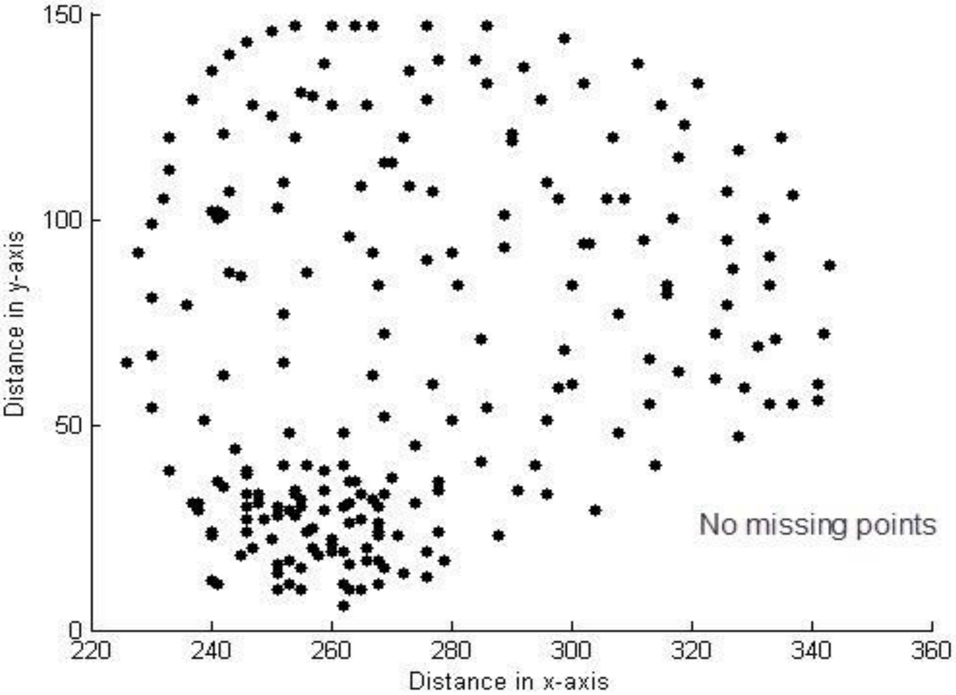


Figure 5.8 One Cluster generated by applying VMDBSCAN algorithm on ball data set for the values η = 0.0005 and Eps = 11.8

We split the plotting in 3 figures for simplifying and visualizing the critical point (Eps) for each density level. When applying DMDBSCAN, firstly we find the nearest 3 distance for each point in data set. We notice that there is sharp change (critical) in Figures 5.11, 5.12 and 5.13 at best values of Eps equal 10.77, 12.17 and 18.25 respectively. Figure 5.14 is the result of applying DMDBSCAN and we got the correct number of cluster which is 1-cluster.

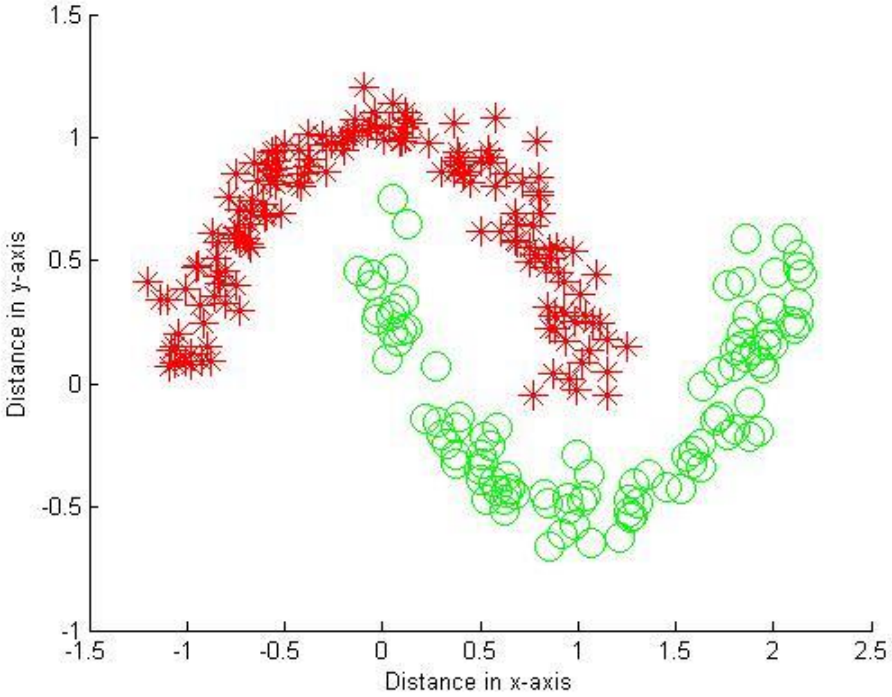


Figure 5.9 Two Clusters generated by applying VMDBSCAN algorithm on two-moons data set for the values η = 0.005, Eps = 0.26, and MinPts = 5

Figures 5.15, 5.16 and 5.17 show the plotting of k-dist with k=1, 2 and 3 respectively (the nearest 3 distance) for two-moons data set, and sorting them in ascending order to find the best values of Eps which will adopt to DMDBSCAN to find the correct number of clusters. We split the plotting in 3 figures for simplifying and visualizing the critical point (Eps) for each density level. When applying DMDBSCAN, firstly we find the nearest 3 distance for each point in data set. We notice that there is sharp change (critical) in Figures 5.15, 5.16 and 5.17 at best values of Eps equal 0.1007, 0.1208 and 0.171 respectively. Figure 5.18 is the result of applying DMDBSCAN and we got the correct number of cluster which equal 2-clusters.

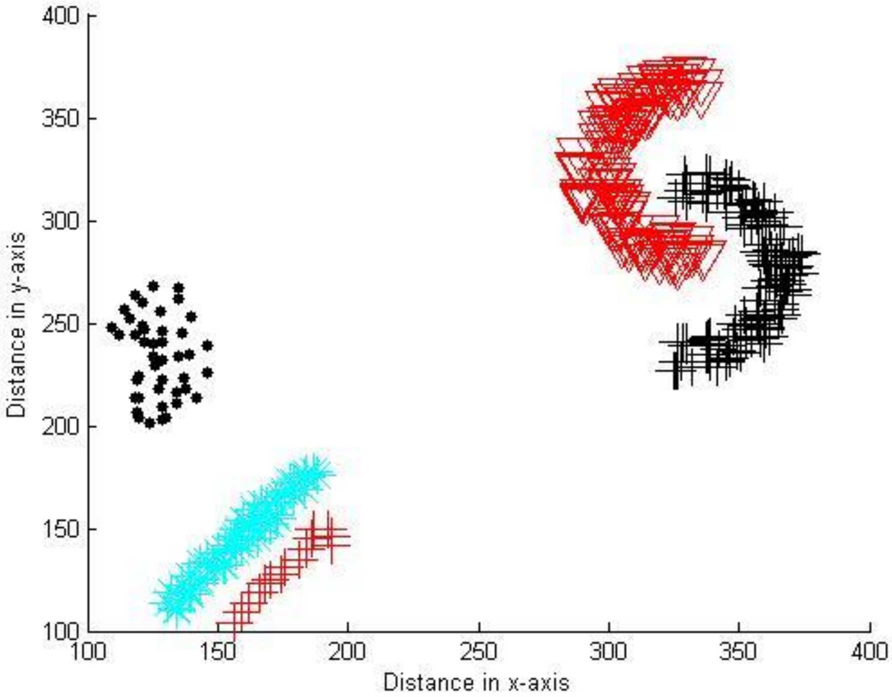


Figure 5.10 Five clusters generated by applying VMDBSCAN algorithm on moon data set for the values η = 0.0005, Eps = 8.8 , and MinPts = 5

Figures 5.19, 5.20 and 5.21 show the plotting of k-dist with k=1,2 and 3 respectively (the nearest 3 distance) for moon data set, and sorting them in ascending order to find the best values of Eps which will adopt to DMDBSCAN to find the correct number of clusters. We split the plotting in 3 figures for simplifying and visualizing the critical point (Eps) for each density level. When applying DMDBSCAN, firstly we find the nearest 3 distance for each point in data set. We notice that there are sharp changes (critical) in Figures 5.19, 5.20 and 5.21 which represent the best values of Eps equal 8.062, 13.15 and 18.03 respectively. Figure 5.22 is the result of applying DMDBSCAN and we got the correct number of cluster which equal 5-clusters.

5.2.5 VDDBSCAN

In this section, we will apply the proposed algorithm VDDBSCAN to solve the problem of using one global value of parameter Eps for all densities in data set, instead VDDBSCAN will use dynamic method to find suitable values of Eps for each density level of data set. Also, it will solve the problem of splitting one cluster into two or more clusters by using vibration method. Figures 5.11, 5.12 and 5.13 show the plotting of k-dist with k=1,2 and 3 respectively (the nearest 3 distance) for ball data set, and sorting them in ascending order to find the best values of Eps which will adopt to VDDBSCAN to find the correct number of clusters.

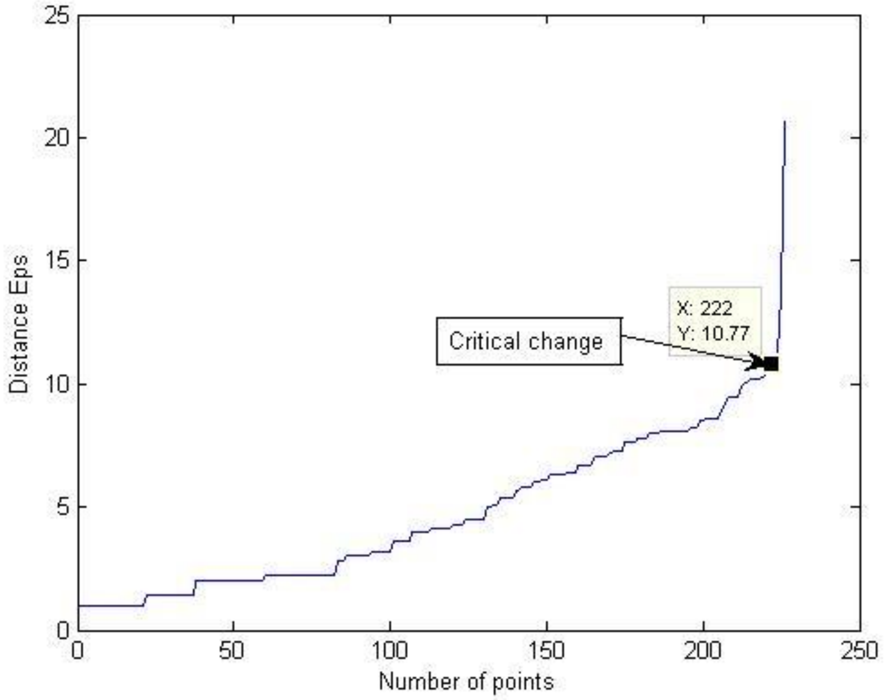


Figure 5.11 Nearest 1-dist sorted points in ascending order to find critical change

We split the plotting in 3 figures for simplifying and visualizing the critical point (Eps) for each density level. When applying VDDBSCAN, firstly we find the nearest 3 distance for each point in data set and with η = 0.0005 of equation (4.6). We notice that there are sharp changes (critical) in Figures 5.11, 5.12 and 5.13 which represent the best values of Eps and equal 10.77, 12.17 and 18.25 respectively. Figure 5.23 is the result of applying VDDBSCAN and we got the correct number of cluster which equal one. Figures 5.15, 5.16 and 5.17 show the plotting of k-dist with k=1, 2 and 3 respectively (the nearest 3 distance) for two-moons data set, and sorting them in ascending order to find the best values of Eps which will adopt to VDDBSCAN to find the correct number of clusters.

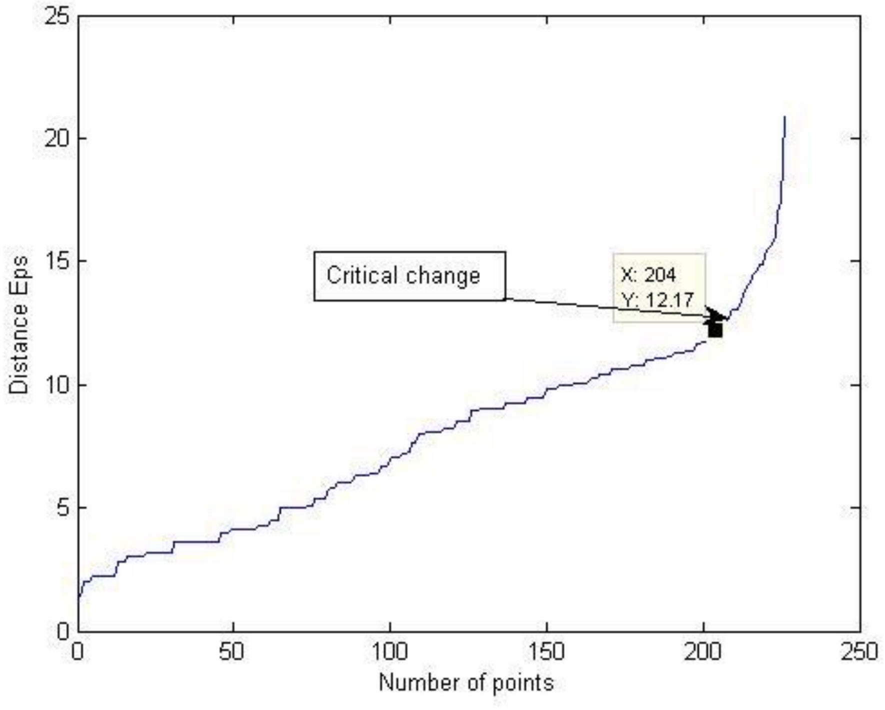


Figure 5.12 Nearest 2-dist sorted points in ascending order to find critical change

We split the plotting in 3 figures for simplifying and visualizing the critical point (Eps) for each density level. When applying VDDBSCAN, firstly we find the nearest 3 distance for each point in data set and with η = 0.005. We notice that there are sharp change (critical) in Figures 5.15, 5.16 and 5.17 at best values of Eps and equal 0.1007, 0.1208 and 0.171 respectively. Figure 5.24 is the result of applying VDDBSCAN and we got the correct number of cluster which equal 2-clusters.

Figures 5.19, 5.20 and 5.21 show the plotting of k-dist with k = 1, 2 and 3 respectively (the nearest 3 distance) for moon data set, and sorting them in ascending order to find the best values of Eps which will adopt to VDDBSCAN to find the correct number of clusters. We split the plotting in 3 figures for simplifying and visualizing the critical point (Eps) for each density level. When applying VDDBSCAN, firstly we find the nearest 3 distance for each point in data set and with η = 0.0005. We notice that there are sharp changes (critical) in Figures 5.19, 5.20 and 5.21 which represent the best values of Eps and equal 8.062, 13.15 and 18.03 respectively. Figure 5.25 is the result of applying VDDBSCAN and we got the correct number of cluster which equal 5-clusters.

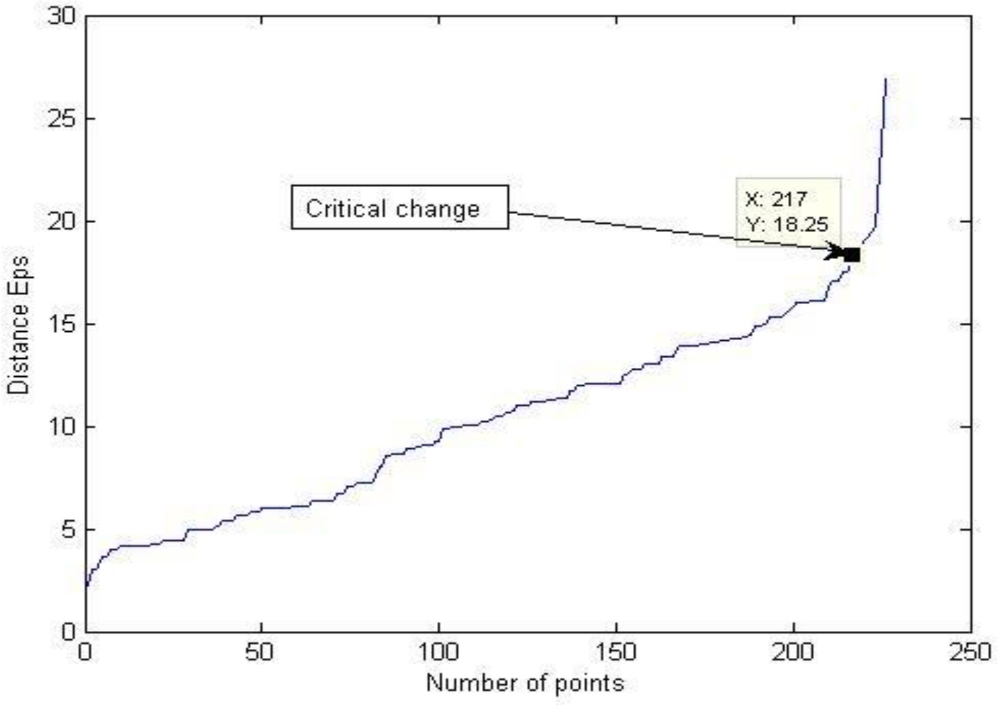


Figure 5.13 Nearest 3-dist sorted points in ascending order to find critical change

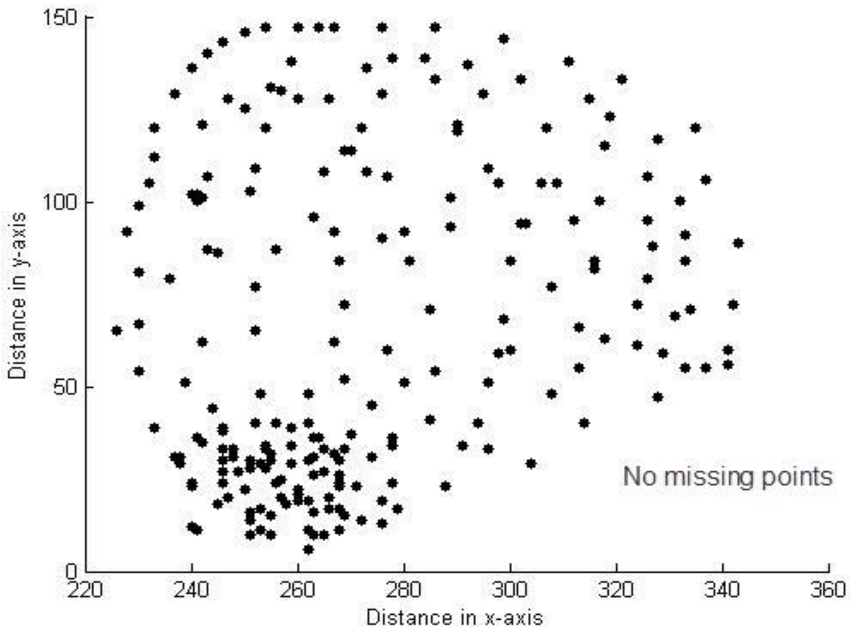


Figure 5.14 One cluster generated by applying DMDBSCAN algorithm on ball data set with three values of Eps equal 10.77, 12.17 and 18.25 respectively

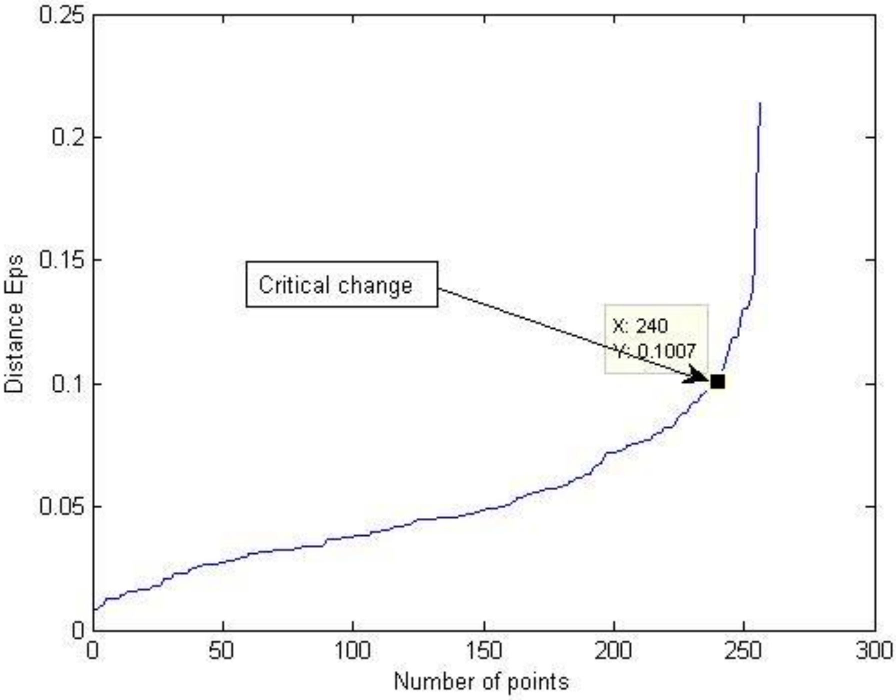


Figure 5.15 Nearest 1-dist sorted points in ascending order to find critical change

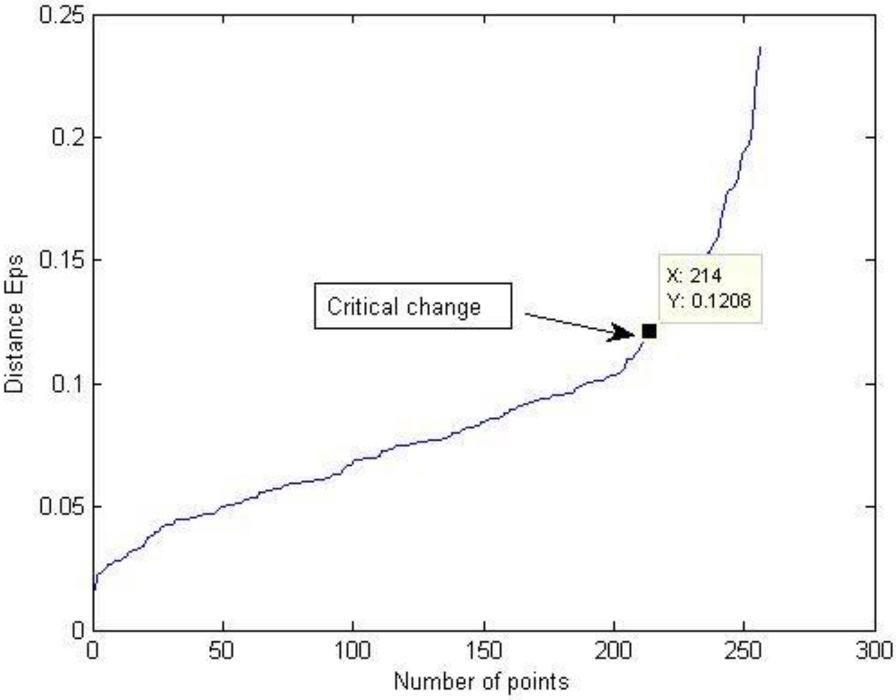


Figure 5.16 Nearest 2-dist sorted points in ascending order to find critical change

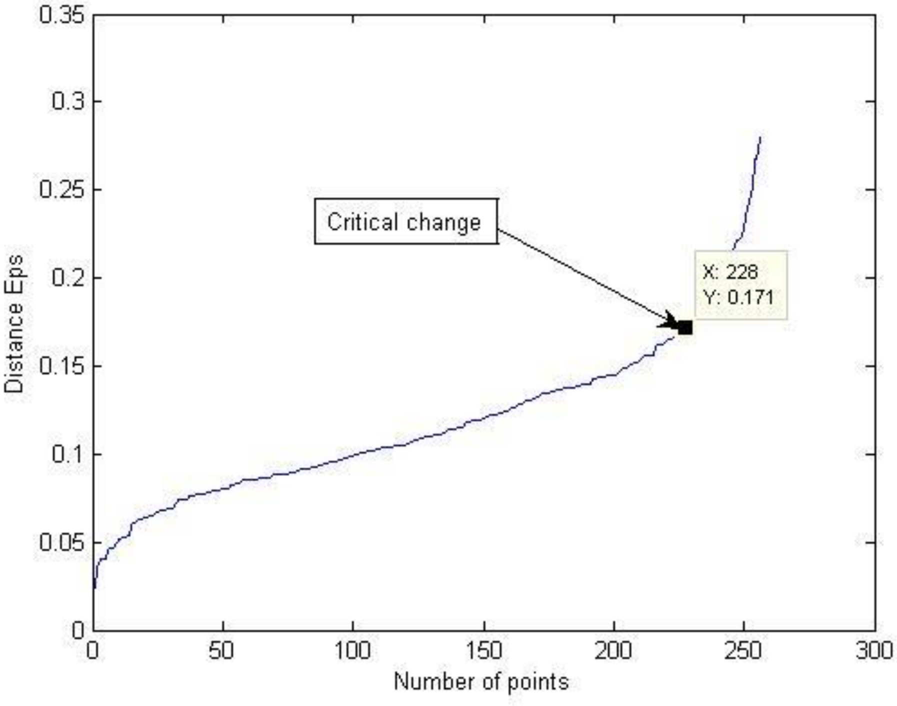


Figure 5.17 Nearest 3-dist sorted points in ascending order to find critical change

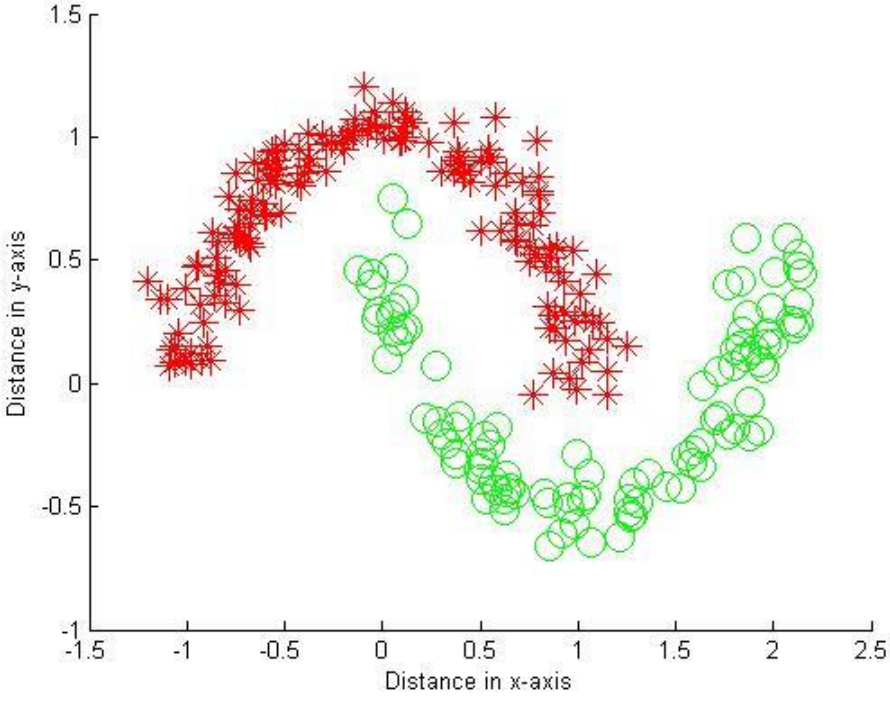


Figure 5.18 Two Clusters generated by applying DMDBSCAN algorithm on two-moons data set with three values of Eps equal 0.1007, 0.1208 and 0.171 respectively

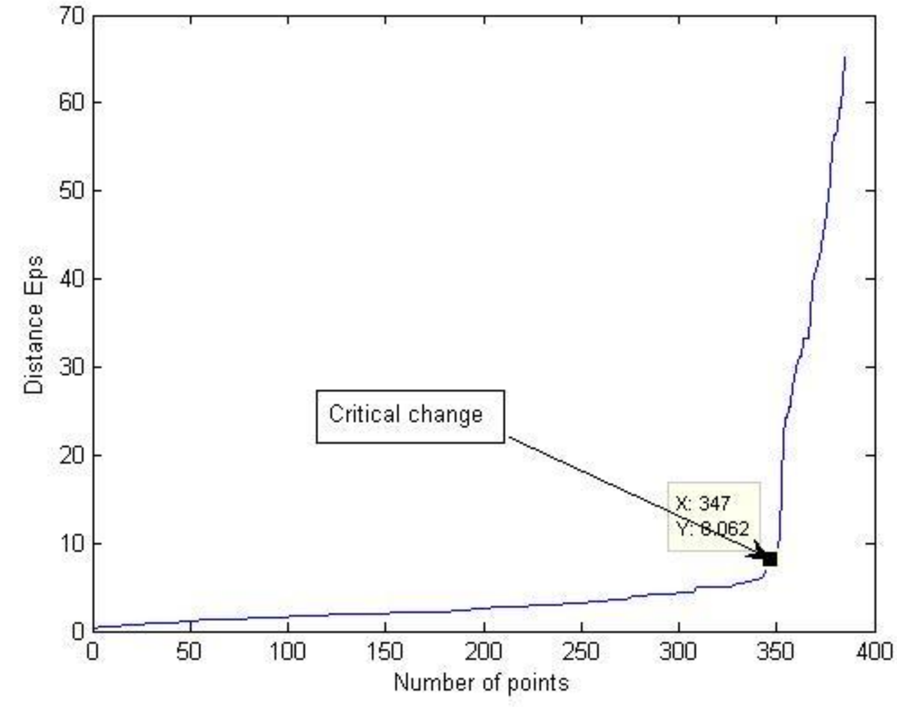


Figure 5.19 Nearest 1-dist sorted points in ascending order to find critical change

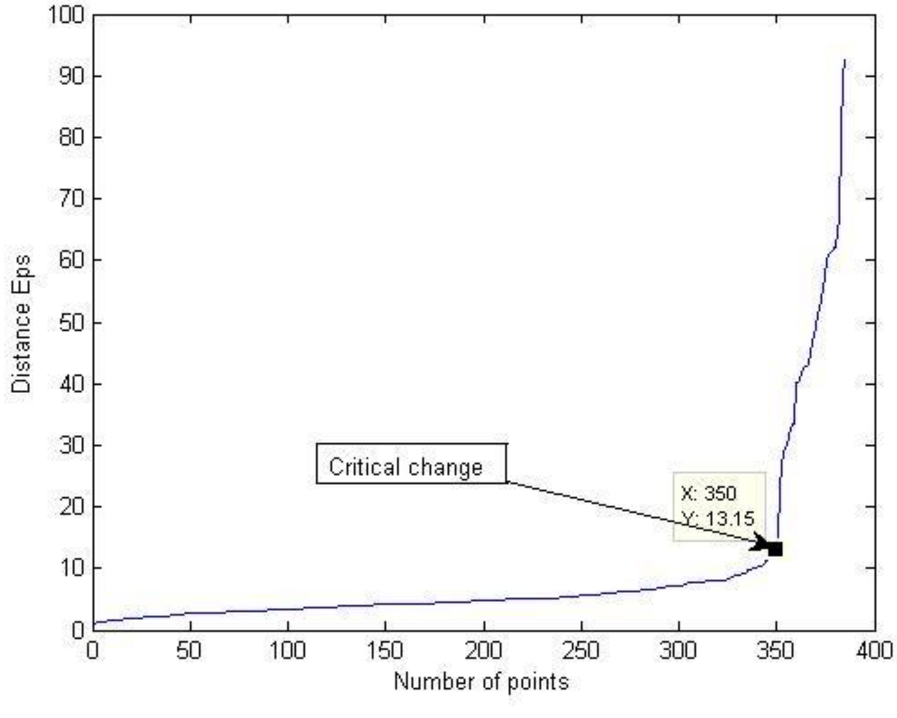


Figure 5.20 Nearest 2-dist sorted points in ascending order to find critical change

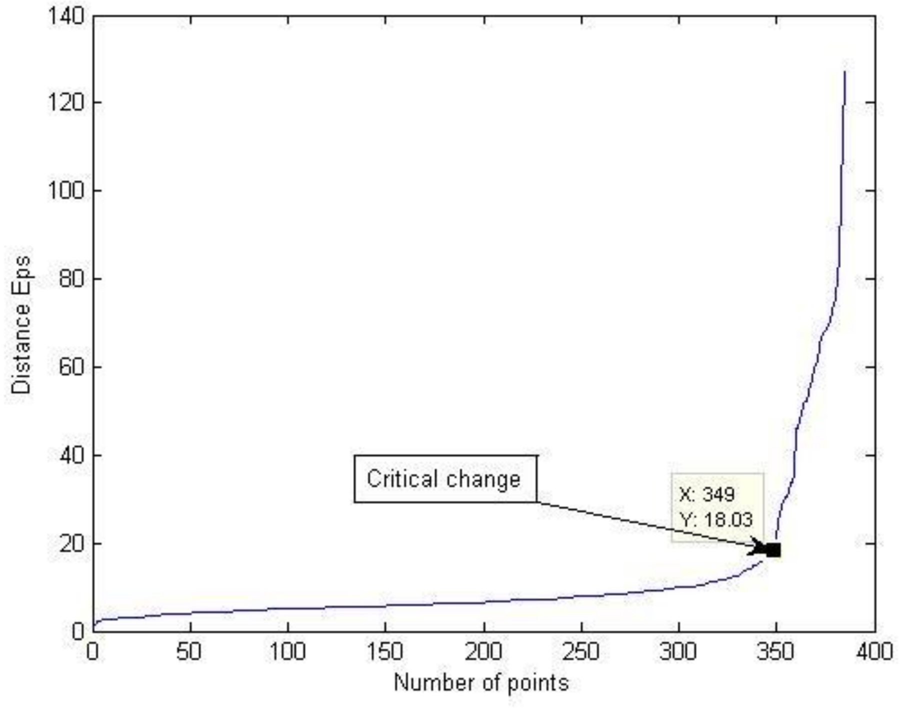


Figure 5.21 Nearest 3-dist sorted points in ascending order to find critical change

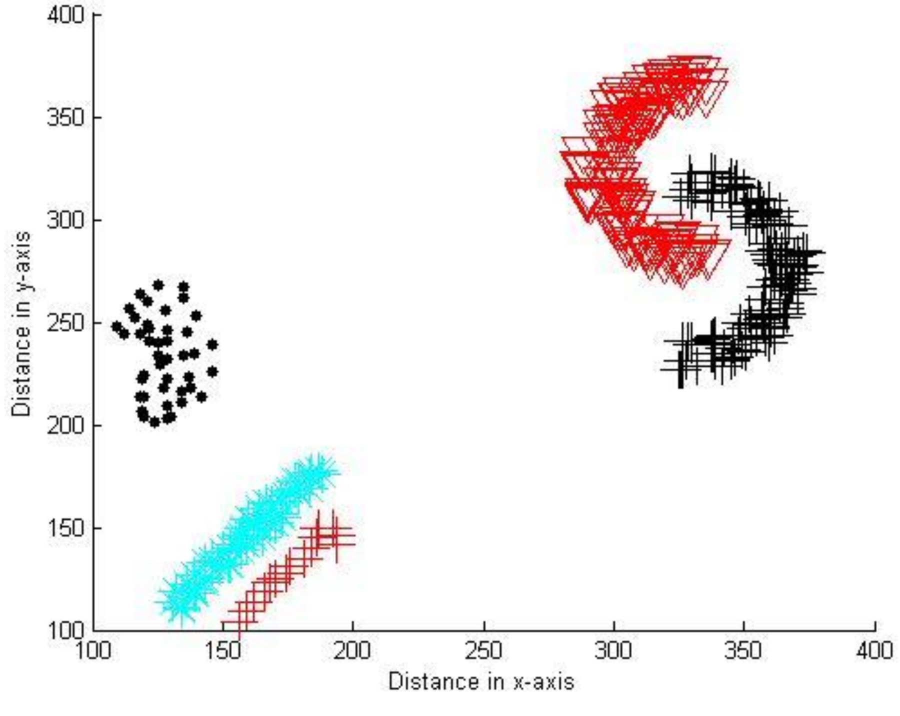


Figure 5.22 Five clusters generated by applying DMDBSCAN algorithm on moon data set with three values of Eps equal 8.062, 13.15 and 18.03 respectively

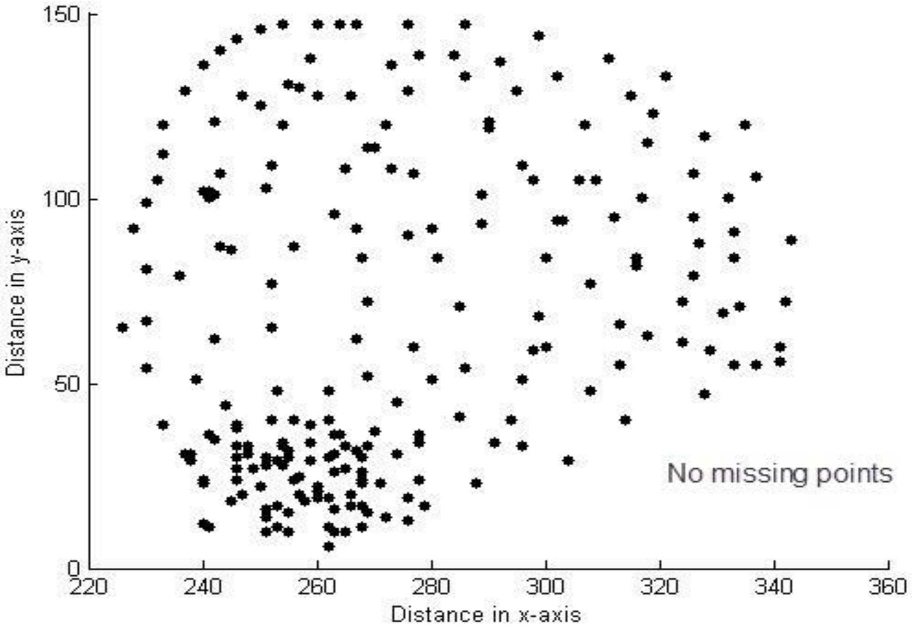


Figure 5.23 One cluster generated by applying VDDBSCAN algorithm on ball data set with three values of Eps and equal 10.77, 12.17 and 18.25 respectively, η = 0.0005, and MinPts = 5

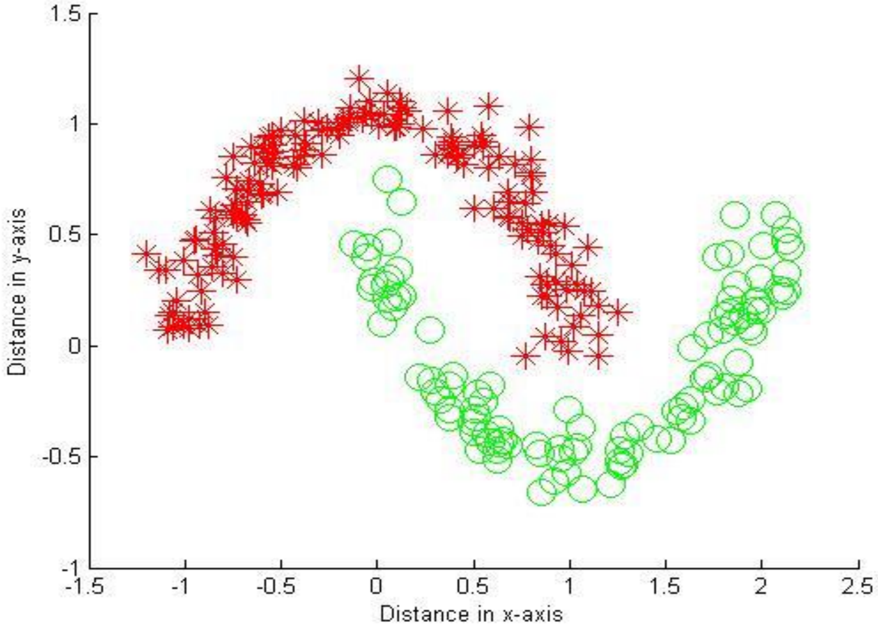


Figure 5.24 Two clusters generated by applying VDDBSCAN algorithm on two-moons data set with three values of Eps equal 0.1007, 0.1208 and 0.171 respectively, η = 0.005, and MinPts = 5

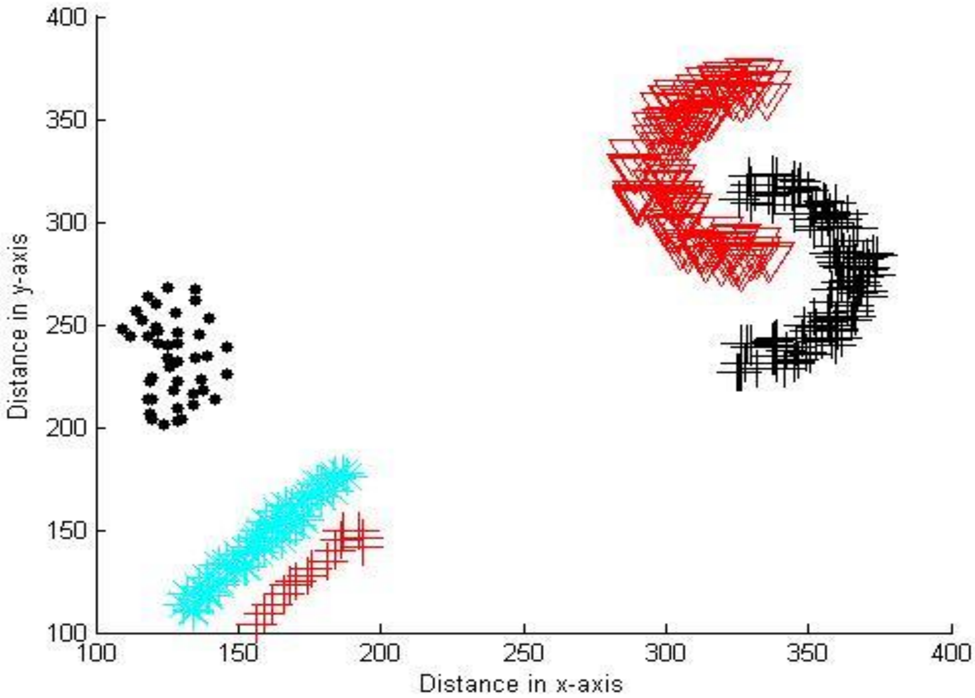


Figure 5.25 Five clusters generated by applying VDDBSCAN algorithm on moon data set with three values Eps equal 8.062, 13.15 and 18.03 respectively, η = 0.0005, and MinPts = 5

5.3 Real Data sets

In this section we will show all results when implement the proposed algorithms on real data sets.

Data sets which will be used found at UCI [61], and we use three real data sets from it. Table 5.1 shows the characteristics of these data sets. The IRIS data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other. The second data set is Haberman, which contains 306 instances, and 3 attributes. The first attribute is the age of patient, the second attribute is year of operation, the third attribute is number of positive axillary nodes detected. Each instance has one of 2 possible classes. The last data set is Glass, which contains 214 instances, and 10 attributes. Each instance has one of 6 possible classes. For measuring the accuracy of our proposed algorithms, we use an average error index in which we count the misclassified samples and divide it by the total number of samples.

Table 5.1 Real data set characteristics

Data set Name	Data set Characteristics	Number of Instances	Number of Attributes	Attribute Characteristics	True Clusters
IRIS	Multivariate	150	4	Real	3
Haberman	Multivariate	306	3	Integer	2
Glass	Multivariate	214	10	Real	6

5.3.1 DBSCAN

We apply the DBSCAN algorithm on IRIS data set with Eps = 0.35 and MinPts = 5, and obtain an average error index of 45.33%. Number of determined clusters = 2 while number of true clusters = 3. While applying DBSCAN on Haberman data set, we get an average error index of 33.33% with Eps = 4.3 and MinPts = 5. Number of determined clusters = 1 while number of true clusters = 2. Another real data set is Glass data set and when we apply DBSCAN on it, we get an average error index of 68.22% with Eps = 0.85 and MinPts = 5**.** Number of determined clusters = 3 while number of true clusters = 6.

Table 5.2 shows all results when apply DBSCAN on these real data sets.

Table 5.2 Average error index of applying DBSCAN on real data sets

Data set	True Clusters	Determined Clusters DBSCAN	Time (ms)	DBSCAN Error %
IRIS	3	2	35	45.33
Haberman	2	1	51	33.33
Glass	6	3	42	68.22

5.3.2 DVBSCAN

Here in this section, we will offer the results of the second comparison algorithm. We apply the DVBSCAN algorithm on IRIS data set with α = 100, λ = 50, μ = 20, Eps = 0.4, and obtain an average error index of 17.22%. Number of determined clusters = 3 and number of true clusters = 3. While applying DVBSCAN on Haberman data set, we get an average error index of 32.65% with α = 100, λ = 50, μ = 20, Eps = 4.5. Number of determined clusters = 2 and number of true clusters = 2. Another real data set is Glass data set and when we apply DVBSCAN on it, we get an average error index of 41.23% with α = 100, λ = 50, μ = 20, Eps = 0.9. Number of determined clusters = 5 while number of true clusters = 6.

From the previous results, DVBSCAN gives results better than DBSCAN in real data sets.

Table 5.3 shows all results when apply DVBSCAN on these real data sets.

Table 5.3 Average error index of applying DVBSCAN on real data sets

Data set	True Clusters	Determined Clusters DVBSCAN	Time (ms)	DVBSCAN Error %
IRIS	3	3	24	17.22
Haberman	2	2	33	32.65
Glass	6	5	29	41.23

5.3.3 VMDBSCAN

Here we will introduce the results of apply our first proposed algorithm VMDBSCAN.

We apply the VMDBSCAN algorithm on IRIS data set with $\eta = 0.00005$, we have an average error index of 20.00%. Number of determined clusters = 3 and number of of true clusters = 3. While applying VMDBSCAN on Haberman data set with $\eta = 0.0005$, we have an average error index of 27.78%. Number of determined clusters = 2 and number of true clusters = 2. Another real data set is Glass data set and when we apply VMDBSCAN on it with $\eta = 0.0005$, we have an average error index of 62.15%. Number of determined clusters = 4 while number of true clusters = 6

Table 5.4 shows all results when apply VMDBSCAN on these real data sets.

Table 5.4 Average error index of applying VMDBSCAN on real data sets

Data set	True Clusters	Determined Clusters VMDBSCAN	Time (ms)	VMDBSCAN Error %
IRIS	3	3	39	20.00
Haberman	2	2	55	27.78
Glass	6	4	47	62.15

5.3.4 DMDBSCAN

Here we will introduce the results of apply our second proposed algorithm DMDBSCAN.

We apply the DMDBSCAN algorithm on IRIS data set, and applying k-dist for 3-nearest points, we have 2 values of Eps which are 0.37 and 0.42. The average error index is 15.00%. Number of determined clusters = 3 and number of true clusters = 3. While applying DMDBSCAN on Haberman data set, and applying k-dist for 3-nearest points, we have 3 values of Eps which are 4.5, 4.8, and 5. The average error index is 20.33%. Number of determined clusters = 2 and number of true clusters = 2.

Another real data set is Glass data set and when we apply DMDBSCAN on it, and applying k-dist for 3-nearest points, we have 3 values of Eps which are 0.85, 9.1, and 9.4. The average error index is 50.34%. Number of determined clusters = 5 while number of true clusters = 6.

Table 5.5 shows all results when apply DMDBSCAN on these real data sets.

Table 5.5 Average error index of applying DMDBSCAN on real data sets

Data set	True Clusters	Determined Clusters DMDBSCAN	Time (ms)	VMDBSCAN Error %
IRIS	3	3	37	15.00
Haberman	2	2	53	20.33
Glass	6	5	46	50.34

5.3.5 VDDBSCAN

Here we will introduce the results of apply our first proposed algorithm VDDBSCAN.

We apply the VDDBSCAN algorithm on IRIS data set with $\eta = 0.00005$, and applying k-dist for 3-nearest points, we have 2 values of Eps which are 0.37 and 0.42. The average error index is 9.76%. Number of determined clusters = 3 and number of true clusters = 3. While applying VDDBSCAN on Haberman data set with $\eta = 0.0005$, and applying k-dist for 3-nearest points, we have 3 values of Eps which are 4.5, 4.8, and 5. The average error index is 12.54%. Number of determined clusters = 2 and number of true clusters = 2.

Another real data set is Glass data set and when we apply VDDBSCAN on it with $\eta = 0.0005$, and applying k-dist for 3-nearest points, we have 3 values of Eps which are 0.85, 9.1, and 9.4. The average error index is 33.43%. Number of determined clusters = 6 and number of true clusters = 6.

Table 5.6 shows all results when apply VDDBSCAN on these real data sets.

Table 5.6 Average error index of applying VDDBSCAN on real data sets

Data set	**True Clusters**	**Determined Clusters VDDBSCAN**	**Time (ms)**	**VDDBSCAN Error %**
IRIS	3	3	41	9.76
Haberman	2	2	56	12.54
Glass	6	6	50	33.43

5.4 The Results Summary

5.4.1 Artificial Data sets

In this section we will discuss the previous results from using proposed algorithms to three artificial data sets. In the first artificial data set, ball data set, when we apply traditional DBSCAN algorithm, we got 2-clusters. This is due the fact that DBSCAN algorithm uses one global Eps for the whole data set, and there are two levels of density. But when we used our proposed algorithms, we get the correct number of clusters, which is 1-cluster. This is due the fact that we use vibration in VMDBSCAN algorithm, which will merge the clusters if DBSCAN spitted them.

Also, when using DMDBSCAN, we got 1-cluster. That is because we have two levels of density. But when we use VDDBSCAN algorithm, the correct number of clusters are generated which is 1-cluster. By using VDDBSCAN, we first get 3-Eps values, and then we use vibration, so we got 1-clusetr.

In the second data set, two-moons data set, because there are varied density, the DBSCAN result in 3-clusters. But applying our proposed algorithm VMDBSCAN, the correct number of clusters are generated which is 2-clusters. That is, when there are varied density in the second moon of data set, the vibration will merge the 2-clusters generated by using DBSCAN into one cluster.

In the last data set, moon data set, Using DBSCAN resulted in 3-clusters, and removed one of cluster. But using our proposed algorithm VMDBSCAN will generate the correct number of clusters, which is 5-clusters. Also, the same results were in DMDBSCAN and VDDBSCAN.

5.4.2 Real Data sets

In this section we will compare the efficiency and time complexity of our algorithms compared to DBSCAN and DVBSCAN.

From our experiments, and as Tables 5.7, 5,8 and 5.9 show, by using DBSCAN algorithm for multi-densities data sets, we get bad quality results with long times. DBSCAN algorithm is a time consuming algorithm when dealing with large multi-densities datasets. This is due to Eps parameter value which is very important for DBSCAN algorithm, but it's calculation is a time-consuming. In other words, clustering algorithms is in need to discover a better version of DBSCAN algorithm to deal with these special multi-densities datasets.

DVBSCAN algorithm gives efficiency - in terms of error rate - less than DBSCAN algorithm. Also, it takes less time to get the clustering results.

VMDBSCAN gives better efficiency results than DBSCAN, but it takes more time compared with DBSCAN. This is due that algorithm need to call DBSCAN algorithm to make initial clustering, then it needs to find cores of each returned clusters from DBSCAN.

DMDBSCAN gives better efficiency results than DBSCAN or DVBSCAN or VMDBSCAN clustering algorithms, but takes more time compared with DBSCAN and DVBSCAN. This is due that algorithm needs to call DBSCAN algorithm for each value of Eps.

VDDBSCAN gives the best efficiency results compared with all algorithms which we compare with them here, but it is the worst in time compared with them. This is due that algorithm need to find Eps for each level of data density, then to call DBSCAN algorithm to find initial clusters, and applying Vibration to find the correct number of clusters.

Table 5.7 Time complexity comparison between varies algorithms applied on IRIS data set

Algorithm	True C's	Determined C's	Time (ms)	Error %
DBSCAN	3	2	35	45.33
DVBSCAN	3	3	24	17.22
VMDBSCAN	3	3	39	20.00
DMDBSCAN	3	3	37	15.00
VDDBSCAN	3	3	41	9.76

We notice in Glass real data set that the error rate resulted by using DBSCAN or our proposed algorithms is large. This is due to the fact that as the number of dimensions increase, the clustering algorithms fail to find the correct number of clusters. But for IRIS or Haberman real data sets, the error rate is less than in Glass, that is because the dimension in these real data sets are smaller. Using DBSCAN will result in the largest error rate, but when using our proposed algorithms, the error rates are small compared with DBSCAN and DVBSCAN algorithms.

Table 5.8 Time complexity comparison between varies algorithms applied on Haberman data set

Algorithm	True C's	Determined C's	Time (ms)	Error %
DBSCAN	2	1	51	33.33
DVBSCAN	2	2	33	32.65
VMDBSCAN	2	2	55	27.78
DMDBSCAN	2	2	53	20.33
VDDBSCAN	2	2	56	12.54

Table 5.9 Time complexity comparison between varies algorithms applied on Glass data set

Algorithm	True C's	Determined C's	Time (ms)	Error %
DBSCAN	6	3	42	68.22
DVBSCAN	6	5	29	41.23
VMDBSCAN	6	4	47	62.15
DMDBSCAN	6	5	46	50.34
VDDBSCAN	6	6	50	33.43

Chapter 6
Conclusion

6.1 Summary and Conclusion Remarks

DBSCAN is technique used widely for clustering in spatial databases. DBSCAN needs less knowledge of input parameters. Major advantages of DBSCAN are to identify arbitrary shape objects and removal of noise during the clustering process. Beside its familiarity, DBSCAN has problems with handling large databases and in worst case its complexity reaches to $O(n^2)$. Also, a major limitation of DBSCAN is that it requires to know the parameters Eps and MinPts, and it uses global parameter of Eps, so it cannot handle data containing clusters of differing densities, since its density based definition of core points cannot identify the core points of varying density clusters. New algorithms were proposed to work with efficient way for these problems. In this thesis we proposed new algorithms in which original DBSCAN is modified or enhanced with improvement in results on varied densities data sets to find the correct number of clusters over many different types of data of different shapes and sizes. We compare these variations one with another and other algorithm DVBSCAN to show the efficiency of our proposed algorithms.

In this thesis, we introduce some enhancement to DBSCAN algorithm by estimating its parameter based on the rank of points distances. In which we use the k-dist plotting to find Eps for each level of density in data set. Also, we eliminate the wrong splitting of two clusters when using traditional DBSCAN algorithm by testing density variation of the data point with respect to the density of core object of its cluster against all densities of other core's clusters. According to the density variance, we do the movement for data points toward the new core. Then merging old cluster with new cluster.

We have proposed an enhancement algorithm based on DBSCAN to cope the problems of one of the most used clustering algorithm. Our proposed algorithm VMDBSCAN gives far more stable estimates of the number of clusters than existing DBSCAN over many different types of data of different shapes and sizes. The second new algorithm DMDBSCAN overcomes on the problem of using one local value of Eps, by using local value of Eps for each level of density in a data set. After merge algorithm VMDBSCAN with DMDBSCAN, we get the best results to find the correct number of clusters over many different types of data of different shapes and sizes.

Experimental results demonstrate that our algorithm is effective and efficient and outperform DBSCAN in detecting clusters of different densities and in eliminating noises.

The experiments show the efficiency of the new algorithms, and get best results with minimum errors.

6.2 Future Work

Future work will focus on determining the best value of the parameter η which used in VMDBSCAN and VDDBSCAN algorithms, and improve the results for high dimensions data sets.

Several opportunities for future research, how to select all the parameters automatically is one of the interesting challenges as parameter k has to be chosen subjectively in DMDBSCAN and VDDBSCAN algorithms.

The future work can be focused on to reduce the time complexity of algorithms. Future research will have to consider cases when the points inside of a cluster are non-uniformly distributed. Other unknown distributions of the points should be investigated.

Bibliography

[1] J. Han and M. Kamber, "Data Mining: Concepts and Techniques," Morgan Kaufmann Publishers, 2006.

[2] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar, "Introduction to Data Mining," Pearson Addison Wesley, 2006.

[3] L. Kaufman and P. Rousseeuw, "Finding groups in Data: an Introduction to cluster," John Wiley & Sons, 1990.

[4] Gan, Guojun, Chaoqun Ma, and Jianhong Wu, "Data Clustering: Theory, Algorithms, and Applications," ASA-SIAM Series on Statistics and Applied Probability, SIAM, Philadelphia, ASA, Alexandria, VA, 2007.

[5] J. Han, M. Kamber, and A. K. H. Tung, "Spatial Clustering Methods in data mining," A Survey, Geographic Data Mining and Knowledge Discovery, 2001.

[6] M. Anderberg (1973), "Cluster analysis for applications," New York: Academic Press.

[7] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," In Proc. 1996 ACMSIGMOD Int. Conf. Management of data (SIGMOD’96), 1996.

[8] S. Guha, R. Rastogi, and K. Shim, "Cure: An efficient clustering algorithm for large databases," In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD’98), 1998.

[9] G. Karypis, E. H. Han, and V. Kumar, "CHAMELEON: A hierarchical clustering algorithm using dynamic modeling," Computer, vol. 32, no. 8, pp. 68–75, 1999.

[10] Ester M., Kriegel H.-P., Sander J., and Xu X. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD’96), Portland: Oregon, pp. 226-231,1996.

[11] M. Ankerst, M. Breunig, H.P. kriegel, and J. Sander, "OPTICS: Ordering points to identify the clustering structure," In Proc. 1999 ACM-SIGMOD Int. Conf. Management of data (SIGMOD'96), 1999.

[12] W. Wang, J. Yang, and R. Muntz, "STING: A statistical information grid approach to spatial data mining," In Proc. 1997 Int. Conf. Very Large Data Bases (VLDB'97), 1997.

[13] G. Sheikholeslami, S. Chatterjee, and A. Zhang, "WaveCluster: A multi-resolution clustering approach for very large spatial databases," in Proc. 24th Int. Conf. Very Large Data Bases, VLDB, pp. 428– 439, 24–27, 1998.

[14] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," pp. 94– 105, 1998.

[15] R.M. Neal and G. E. Hinton. "A new view of the EM algorithm that justifies incremental, sparse and other variants," In M. I. Jordan, editor, Learning in Graphical Models", Kluwer Academic Publishers, pp. 355–3681, 998.

[16] J. C. Bezdeck, R. Ehrlich, and W. Full, "Fcm: Fuzzy c-means algorithm," Computers and Geoscience, vol. 10, no. 2-3, pp. 191-203, 1984.

[17] A. Hinneburg and D. A. Keim, "An efficient approach to clustering in large multimedia databases with noise," In Proc. 1998 Int. Conf. Knowledge discovery and Data mining (KDD'98), 1998.

[18] J. W. Shavlik and T.G. Dietterich, "Reading in machine learning," Morgan Kaufmann Publishers, 1990.

[19] T. Kohonen, "Self organized formation of topologically correct feature maps," Biological Cybernetics, 1982.

[20] C. H. Chou, M. C. Su, and E. Lai, "A new cluster validity measure and its application to image compression," Pattern Analysis and Applications, vol. 7, no. 2, pp. 205–220, July 2004.

[21] G. Hammerly, and C. Elkan, "Alternatives to the k-means algorithm that find better clusterings," Proc. ACM on Information and Knowledge Management, pp. 600– 607, November 2002.

[22] P.S. Bradley, and U.M. Fayyad, "Refining Initial Points for k-means Clustering," ICML 1998, pp. 91–99, January 1998.

[23] P. Berkhin, "Survey of Clustering Data Mining Techniques," Accrue Software, Technical Report, nnnn 2002.

[24] J. Kogan, "Introduction to Clustering Large and High-Dimensional Data," Cambridge University Press, New York, 2007.

[25] J. Valente de Oliveira, and W. Pedrycz, "Advances in Fuzzy Clustering and its Applications," John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, vvv England, 2007.

[26] M. Sato-Ilic, and L. C. Jain, "Innovations in Fuzzy Clustering," Springer-Verlag Berlin Heidelberg, New York, 2006.

[27] W. Pedrycz, "KNOWLEDGE-BASED CLUSTERING From Data to Information Granules," John Wiley & Sons, Inc., Hoboken, New Jersey, 2005.

[28] Usama Fayyad and Ramasamy Uthurusamy, "Data mining and knowledge discovery in databases: Introduction to the special issue," Communications of the ACM, 39(11), November 1999.

[29] M.Parimala, Daphne Lopez, N.C. Senthilkumar, "A Survey on Density Based Clustering Algorithms for Mining Large Spatial Databases," International Journal of Advanced Science and Technology, Vol. 31, June, 2011.

[30] J. A. Hartigan, "Clustering Algorithms," John Wiley & Sons, 1975.

[31] A. Jain and R. Dubes, "Algorithms for Clustering Data," Englewood Cliffs, NJ: Prentice–Hall, 1988.

[32] L. Kaufman and P. J. Rousseeuw, "Finding groups in data: an Introduction to cluster analysis," John Wiley & Sons Inc., New York, USA, p. 112, 1990.

[33] P. Arabie, L. J. Hubert, and G. De Soete, "Clustering And Classification," World Scientific Publ., River Edge, NG, 1996.

[34] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A survey," ACM Computing Surveys, vol.31, no.3, 31:264–323, September 1999.

[35] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: A review," ACM SIGKDD Explorations Newsletter, ACM New York, NY, USA, 6, 90-105 6:90–105, 2004.

[36] A. K. Jain, "Data clustering: 50 years beyond k-means," 19th European Conference on Artificial Intelligence Proceedings, 2010.

[37] S. P. Lloyd. "Least squares quantization in PCM," 28:128–137, 1982 (original version: Technical Report, Bell Labs, 1957).

[38] J. Mac Queen, "Some methods for classification and analysis of multivariate observations," 1:281–297, 1967.

[39] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," In, pages 1027–1035, Tokyo, Japan, 2007.

[40] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. "An efficient k-means clustering algorithm: Analysis and implementation," IEEE Transactions on pattern analysis and machine intelligence, vol.24, no.7, July 2002.

[41] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," 2:283–304, 1998., 24:881–892, 2002.

[42] A. Chaturvedi, P. Green, and J. Carroll, "k-means, k-medians and k-modes: Special cases of partitioning multi-way data," In, Houston, TX, 1994.

[43] A. Chaturvedi, P. Green, and J. Carroll, "K-modes clustering," 18:35–55, 2001.

[44] R. Ng and J. Han, "Efficient and effective clustering method for spatial data mining," In, pages 144–155, Santiago, Chile, Sept. 1994.

[45] M. Ester, H.-P. Kriegel, and X. Xu, "Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification," In, pages 67–82, Portland, ME, Aug. 1995.

[46] P. Bradley, U. Fayyad, and C. Reina, "Scaling clustering algorithms to large databases," In, pages 9–15, New York, NY, Aug. 1998.

[47] S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," In, pages 73–84, Seattle, WA, June 1998.

[48] S. Guha, R. Rastogi, and K. Shim, "ROCK: A robust clustering algorithm for categorical attributes," In, pages 512–521, Sydney, Australia, Mar. 1999.

[49] M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: Ordering points to identify the clustering structure," In, pages 49–60, Philadelphia, PA, June 1999.

[50] Pei T, Zhu AX, Zhou CH, Li BL, Qin CZ, "A new approach to the nearest-neighbor method to discover cluster features in overlaid spatial point processes," Int J Geogr Inf Scipp 153–168, 2006.

[51] Roy S, Bhattacharyya DK, "An approach to find embedded clusters using density based techniques," Lecture Notes Computer vol. 3816, pp. 523–535, 2005.

[52] Lin CY, Chang CC, "A new density-based scheme for clustering based on genetic algorithm," Fundam Inform vol. 68, pp. 315–331, 2005.

[53] Pascual D, Pla F, Sanchez JS, "Non parametric local density-based clustering for multimode overlapping distributions," In: Proceedings of intelligent data engineering and automated learning," (IDEAL2006), Spain, Burgos, pp. 671–678, 2006.

[54] Ram, A.; Sharma, A.; Jalal, A.S.; Agrawal, A.; Singh, R., "An Enhanced Density Based Spatial Clustering of Applications with Noise," Advance Computing Conference, 2009. IACC 2009. IEEE International, vol., no., pp.1475-1478, 6-7 March 2009.

[55] Borach, B., Bhattacharya, D.K., "A Clustering Technique using Density Difference," In proceedings of International Conference on Signal Processing, Communications and Networking. pp. 585–588, 2007.

[56] B. Borah, D.K. Bhattacharyya, "DDSC: A Density Differentiated Spatial Clustering Technique", Journal Of Computers, Vol. 3, No. 2, February 2008.

[57] Xiaowei Xu, Martin Ester, Hans-Peter Kriegel, Jörg Sander, "DBCLASD: A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases," In Proceedings of the Fourteenth International Conference on Data Engineering, IEEE Computer Society, 1998.

[58] Derya Birant, Alp Kut, "ST-DBSCAN: An algorithm for clustering spatial–temporal data," In Data Knowl. Eng., vol. 60, nr. 1 , p. 208-221, January 2007.

[59] Anant Ram, Sunita Jalal , Anand S. Jalal, Manoj Kumar, "DVBSCAN: A Density based Algorithm for Discovering Density Varied Clusters in Large Spatial Databases," International Journal of Computer Applications (0975 – 8887) Volume 3 – No.6, June 2010.

[60] G. Milligan and M. Cooper, "A study of standardization of variables in cluster analysis," Journal of Classification, 5:181–204, 1988.

[61] UCL Machine Learning Repository, Available at "http://archive.ics.uci.edu/ml/data sets," last visit 1 Aug 2012.

[62] Kohonen, T., "The self-organizing map", Proceedings of the IEEE, Volume 78, Issue Pages: 1464 – 1480, Sep 1990.