"Design of Orthogonal Grey Wolf Optimization for clustering

A thesis submitted
by

MOHIT SHARMA
2017PEC5507

Under the guidance of
Dr. Satyasai Jagannath Nanda

In partial fulfillment for the award of the degree of
MASTER OF TECHNOLOGY
to the
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Electronics & Communication Engineering Department
Malaviya National Institute of Technology, Jaipur
JUNE 2019

CERTIFICATE

This is to certify that the dissertation report entitled **"Design of Orthogonal Grey Wolf Optimization for clustering"** has been successfully completed and presented by **Mohit Sharma** of Second year, IV semester in partial fulfillment of degree of **Master of Technology** in **Electronics and Communication** during the academic year **2018-2019** to the best of my knowledge and belief that this work has not been submitted elsewhere for the award of any other degree.

The work has been found satisfactorily carried out by him under my guidance and supervision in the department and is approved for submission.

Dr. Satyasai Jagannath Nanda
Assistant Professor
Department of Electronics and
Communication Engineering
Malaviya National Institute of Technology,
Jaipur
India 302017.

Date:
Place: Jaipur

DECLARATION

I, **Mohit Sharma**, declare that this Dissertation titled as **"Design of Orthogonal Grey Wolf Optimization Algorithm for Clustering"** and the work presented in it is my own and that, to the best of my knowledge and belief.

I confirm that major portion of the report except the refereed works, contains no material previously published nor present a material which to be substantial extent has been accepted or the award of any other degree by university or other institute of higher learning. Wherever I used data (Theories, results) from other sources, credit has been made to that source by citing them (to the best of my knowledge). Due care has been taken in writing this thesis, errors and omissions are regretted.

Date:
Place: Jaipur

Mohit Sharma
M.Tech
Department of Electronics and
Communication Engineering
Malaviya National Institute of Technology,
Jaipur, India 302017.

Acknowledgments

I would like to thank all people who have helped and inspired me in the research contributing to this thesis. I take this opportunity to express my deep sense of gratitude and respect towards my Supervisor of the dissertation, Dr. Satyasai Jagannath Nanda, Associate Professor, Department of Electronics and Communication Engineering, Malviya National Institute of Technology. I am very much indebted to him for the generosity, expertise, and guidance; I have received from him while working on this project and through- out my studies. Without his support and timely guidance, the completion of my project would have seemed a far-fetched dream. In this respect, I find myself lucky to have him as my Project Supervisor. He has guided me not only with the subject matter but also taught me the proper style and techniques of working.

I express my sincere gratitude to Dr.K.K.Sharma, Professor, Dr. Ravi Kumar Maddila, Associate Professor,and Dinesh Kumar, Senior Research Fellow, Department of Electronics and Communication Engineering, Malviya National Institute of Technology, for all the knowledge they shared with me, without which our research would have been difficult to conclude.

At last but not the least, I am also and always be grateful to my parents and express my deep respect towards them, who always support me and encourage me at each and every step of my career and life besides this project.

Mohit Sharma

Contents

Certificate i

Declaration ii

Acknowledgments iii

Contents v

List of Figures vii

List of Tables ix

List of Abbreviations xi

Abstract xii

1 Introduction 1
- 1.1 Objectives . . . 2
- 1.2 Thesis Organisation . . . 2

2 Literature review 3
- 2.1 K-means clustering algorithm . . . 3
- 2.2 Trimmed K-means clustering algorithm . . . 4
- 2.3 BFR algorithm . . . 6
- 2.4 CURE algorithm . . . 8
- 2.5 Density-based clustering algorithms . . . 9
  - 2.5.1 DBSCAN . . . 9
  - 2.5.2 Peak density-based clustering . . . 10
- 2.6 Orthogonal experimental design based Optimization algorithms . . . 12
  - 2.6.1 Orthogonal array generation . . . 12
  - 2.6.2 Orthogonal genetic algorithm with quantization . . . 15
  - 2.6.3 Orthogonal Particle swarm optimization . . . 17
- 2.7 Grey wolf optimization . . . 20

3 Robust clustering 33
- 3.1 Implementation of Trimmed K-means algorithm . . . 33
- 3.2 Outliers removal using BFR algorithm . . . 34
- 3.3 Outlier removal using density based clustering algorithms . . . 35
- 3.4 Important discussions . . . 35

4 Orthogonal Grey Wolf Optimization **41**
4.1 Proposed OGWO algorithm 41
4.2 Simulation on benchmark mathematical test function 43
4.3 Simulation on benchmark clustering dataset 44

5 Conclusion and future aspects **53**

References **55**

List of Publications **59**

List of Figures

2.1 Figure showing K vs average distance plot by which correct value of K can be selected . . . 6
2.2 Example of K-means clustering: (a) Original unclustered dataset (b) Well clustered dataset. . . . 24
2.3 Figure showing normally distributed clusters, appropriate for BFR algorithm . . . 25
2.4 Figure showing the procedure to assign an incoming point to different sets in BFR algorithm . . . 26
2.5 Shapes of clusters that cannot be identified using K-means and BFR algorithm . . . 26
2.6 Figure showing the representation of a cluster by multiple represebtative points. . . . 27
2.7 Figure showing (a) initialization and (b) clustering of initially loaded datapoints in CURE algorithm . . . 28
2.8 Figure showing (a) representation of mini cluster and (b) assigning incoming points to cluster in CURE algorithm . . . 29
2.9 Complex shapes that can be clustered using density-based clustering algorithms . . . 30
2.10 figure showing high density core points and outlier points at low density 30
2.11 figure showing decision graph and poits with high δ values but low ρ value are rejected as outliers and points with high ρ and δ are selected as centroids 31
2.12 figure showing (a) unclustered dataset and (b) decision graph for the dataset 32

3.1 Effect of outliers on the performance of K-means clustering: (a)Result on Iris dataset without outliers (b) Result on Iris dataset after adding outliers 34
3.2 figure showing deviation of centroids from original position when outliers are present and correct centroid estimation by trimmed K-means algorithm 34
3.3 Figure showing: (a) original artificial dataset without outliers (b) dataset with outliers (c) result of BFR algorithm . . . 36
3.4 figure showing two differnt non clustered datasets . . . 37
3.5 figure showing results of dbscan algorithm . . . 38
3.6 figure showing (a) unclustered dataset and (b) decision graph for the dataset (c)the result of peak density based clustering . . . 39

4.1 Figure showing benchmark functions used (a) Unimodal (b) Multimodal and (c) Fixed dimensional multimodal benchmark functions[1] . . . 46
4.2 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F1 and (b) F2 unimodal benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms . . . 47

4.3 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F3 and (b) F4 unimodal benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms 47
4.4 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F5 and (b) F6 unimodal benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms 47
4.5 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F8 and (b) F9 multimodal 30-dimensional benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms 48
4.6 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F10 and (b) F11 multimodal 30-dimensional benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms 48
4.7 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F12 and (b) F13 multimodal 30-dimensional benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms 48
4.8 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F14 and (b) F15 fixed dimensional multimodal benchmark functions, with 50 population size and 50 iterations for both the algorithms 49
4.9 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F16 and (b) F17 fixed dimensional multimodal benchmark functions, with 50 population size and 50 iterations for both the algorithms 49
4.10 Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F18 and (b) F19 fixed dimensional multimodal benchmark functions, with 50 population size and 50 iterations for both the algorithms 49
4.11 Figure (a) showing clustering results on diabetes dataset using OGWO and (b) showing convergence comparison of OGWO with OGA, PSO, OPSO, and GWO 50
4.12 Figure (a) showing clustering results on gaussian G260 dataset using OGWO and (b) showing convergence comparison of OGWO with OGA, PSO, OPSO, and GWO 50
4.13 Figure (a) showing clustering results on highdimensional highdim-1 dataset using OGWO and (b) showing convergence comparison of OGWO with OGA, PSO, OPSO, and GWO 50

List of Tables

2.1 Table showing(a)An experimental design problem with three factors and three levels and (b) Orthogonal combinations generated from this information [2] 13
2.2 Table showing:(a) Showing different levels of three features, (b) Orthoganal array for N=3 Q=4 (c) The generation of orthogonal combination with the help of orthogonal array 16
2.3 Table showing factors and levels for an arbitrary experiment 18
2.4 Table showing orthoganal array for N=3 Q=3 25
2.5 deciding the best combination level of given experiment using OED method [3] 27

4.1 Table showing orthoganal array for N=3 Q=3 [**?**] 42
4.2 Results of OGWO algorithm on benchmark functions 45
4.3 table showing details of datasets used 51
4.4 Table showing percentage misclassification on different dataset when subjected to differnt clustering algorithms including the proposed algorithm 51

List of Abbreviations

GWO	Grey Wolf Optimization
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
OPSO	Orthogonal Particle Swarm Optimization
OGA	Orthogonal Genetic Algorithm
SI	Swarm Intelligence
BFR algorithm	Bradley-Fayyad-Reina algorithm
RS	Retained Set
CS	Compression Set
DS	Discard Set
SumSQ	Sum Squared
CURE	Clustering Using REpresentatives
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
OED	Orthogonal Experimental Design
OA	Orthogonal Array
IMM	Intelligent Move Mechanism

Abstract

Clustering is a complex problem of segrigating data points with similar characteristics into groups within any available complex dataset which may contain various, numerous data points with various properties. Many algorithms have been developed in the past to tackle this problem using various techniques. The problem of clustering becomes more complicated when outliers are introduced in the dataset. Outliers are the data-points which have the most distinct characteristics compared to the other data-points in the dataset. Hence these contaminated datasets cannot be clustered using orthodox clustering algorithms and therefore robust clustering techniques were developed by researchers. Some of these robust techniques are modified version of conventional algorithms to make them more adaptive to contamination while others are based on completely new concepts. Researchers have also applied optimization algorithms to solve clustering problems. The optimization is the method of picking the most optimal component from an available set of components with regard to some benchmark. Basically an optimization problem is maximizing or minimizing a mathematical function by meticulously picking input values from within an available domain and calculating the value of function. Optimization can be applied on variety of functions and variety of domains. There exist a variety of methods to perform optimization. To test the general performance and other characteristics of these methods, like precision and robustness, researchers have developed some benchmark mathematical test functions. A bencmark function can be highdimensional unimodal(one minima/maxima) or multimodal(more than one minima/maxima) mathematical landscape. These test functions are general for all types of optimization methods. The benefit of these test function is that the result obtained from testing different optimization metods can be compared to each other to find the most competent algorithms. In the past few decades tackling the problem of clustering using nature-inspired optimization algorithms have become popular by considering it as an optimization problem. Researchers have developed various nature-inspired algorithms to solve the optimization problem. A new metaheuristic named as Grey Wolf Optimizer(GWO) was proposed in the year 2014.This algorithm is inspired by the behavior of grey wolves (Canis lupus) in their natural habitat. This algorithm was tested on benchmark tested on benchmark functions and the results obtained were very accurate and had low deviation. However these results can be made more precise md the deviation can be lowered further. For this a new varient of the algorithm is proposed and tested on the benchmark functions to obtain even more robust results and compare them with previous one.

Chapter 1

Introduction

Clustering is unsupervised learning by which a dataset can be categorized into different smaller sets of similar data-points. Clustering is a complex problem because the target value is unknown for all the data-points that are to be clustered. Clustering has found its application in marketing where it can be used to classify and identify customers for marketing purposes. It has also been used by scientists to categorize different species of plants and animals. Researchers have also used the concept to learn about earthquake and flood-affected areas and to classify them into dangerous zones.[4], [5], [6]

Many algorithms have been developed to solve clustering problems, some of them use the concept separation between the centroids whereas some of them use density to identify the clusters. The result of clustering depends on the algorithm used and the dataset on which it is applied. It is not necessary that a technique is best suited for all the datasets as every dataset has a different orientation of data-points. The performance of an algorithm also gets affected by the presence of outliers therefore either the algorithm should be robust to take care of the outliers all by itself or the data should be preprocessed before applying an algorithm. In this project robust clustering algorithms [7], [8]

In the past few decades, solving clustering problems using nature-inspired optimization algorithm, by considering clustering as an optimization problem has also become popular. The general idea is to formulate a fitness function which can either be the inter-cluster distance which is maximized or it can be the intra-cluster distance which is to be minimized. Apart from that, the users can also formulate their own fitness function depending on the problem. In 2014 authors Seyedali Mirjalili, Seyed Mohammad Mirjalili,

and Andrew Lewis propose a new metaheuristic named as Grey Wolf Optimizer(GWO). [1] This algorithm is inspired by the behavior of grey wolves (Canis lupus) in their natural habitat. This algorithm imitates their hunting techniques and the hierarchy of social status. [9], [10]

In this project, a different method is proposed to improve the performance of GWO which employs an orthogonal movement mechanism using an orthogonal experimental design to adjust the movement of every single particle.

1.1 Objectives

The purpose of this project is to study different methods of clustering including conventional and robust clustering techniques to cluster datasets contaminated with outliers. In addition to that other nature-inspired optimization algorithms were studied including Grey Wolf Optimization(GWO) which was introduced in 2014 [] and a new variant of GWO is proposed in order to make it more robust and accurate which is based on orthogonal experimental design. Another objective is to test the algorithm on benchmark test functions for comparative study and to solve clustering problems.

1.2 Thesis Organisation

The thesis is organized in five chapters including this introduction. Chapter-2 gives a basic idea about clustering algorithms including K-means, Trimmed K-means, BFR, and CURE algorithm. Density-based clustering techniques are discussed in the same chapter. Later in chapter-2 optimization algorithms based on orthogonal experimental design along with orthogonal array generation are discussed. Lastly, in the chapter, GWO is discussed. In chapter-3 results of robust clustering algorithms(Trimmed K-means, BFR, and density-based clustering algorithms) are shown when data is corrupted by outliers. In chapter-4 the proposed algorithm is discussed in detail and comparative results of the performance of the algorithm on benchmark test functions and clustering datasets are included. In the last chapter the whole project is concluded and the future work that can be done after this work is explained.

Chapter 2

Literature review

2.1 K-means clustering algorithm

K-means clustering algorithm is a very popular and strong clustering algorithm. It is very simple to implement and also converges quickly, no data preprocessing is required and Euclidean distance is used to determine the similarity between two points. In K-means clustering algorithm, data is to be clustered into 'K' different clusters and each cluster is represented by the centroid point. K-means algorithm is initialized by selecting 'k' points from the dataset. There are several methods for selecting points from dataset. The most common method is random initialization where 'k' points can be chosen in a completely random way, another method suggests that initial points should be chosen in such a way that they are as dispersed as possible which means that they should not be close to each other.

Once the initialization is done the distance of every single point is calculated from these points and if i^{th} point from the dataset, is found to be closest to the j^{th} centroid point then this point is assigned to that centroid. In this way, whole data is scanned and when scanning is done all the points are going to be assigned to a centroid in such a way that not a single remains unassigned and all the centroid points have at least one point under them. The centroids are now replaced with the mean of the points which are assigned to them. The process is repeated several times and final results are obtained. It quite evident from above that k-means is very simple to implement and it is also computationally faster as compared to other algorithms. But if sufficient information

about data is not available then it is difficult to choose the value of 'K' correctly. Apart from choosing the correct value of the number of centroids, initialization also affects the results of the algorithm, different initialization may result in different final results. There is a method to pick 'k' correctly if the dataset is large, then some samples from the dataset are loaded and a smaller value of 'k' is chosen initially and the algorithm is applied. The average distance of all the points from their centroid is calculated and the value of 'K' is increased , this process is repeated and 'K' vs 'average distance to centroid' is plotted from the plot it can be observed that the average drops quickly until a value of 'K' and then either the rate of drop becomes decreases significantly or it becomes almost zero. The value of 'K' for which rate of drops by a significant amount and no longer drops on increasing the value of 'K' is chosen. This whole scenario is explained in figure-2.1. Results of K-means algorithm are shown in figure-2.2. K-means clustering algorithm can easily detect normally distributed clusters but cannot detect complex shaped clusters. Apart from that, the presence of outliers affects the algorithm and there is no method by which the algorithm can deal with the outliers, in fact, the presence of outliers deviate the position of centroids from their original position which results in misclassification of data points.[4], [11], [12].

2.2 Trimmed K-means clustering algorithm

The modified version of K-means can be used to deal with the presence of outliers and the problem of misclassification can be prevented. The modified algorithm is known as Trimmed K-means algorithm where the data is trimmed before the algorithm is applied, this algorithm is also iterative. The trimming can either be random or it can be selective, in random trimming 30-percent of data points are randomly loaded in memory whereas in selective trimming first, 'K' points are chosen randomly and then only those data points are considered which are in close proximity of these 'K' points.[13], [14] After trimming is done, the k-means algorithm is applied and final centroids are obtained and this process is repeated again and again to a certain number of iterations. In the end, the final centroid can be obtained by taking an average of centroids obtained at every iteration. Trimmed K-means algorithm can also be modified by using Mahalanobis Distance **Mahalanobis**

distance instead of using Euclidean distance.

Most of the time Euclidian distance is used to determine the proximity of a point to a cluster, however, there are some issues with Euclidian distance. Generally, datasets are multidimensional and each dimension has different range and different units it is possible that a dimension has low range but is crucial for clustering point of view, therefore, it is not going to contribute much to Euclidian distance however it might happen that the information associated with this dimension is sufficient to partition the whole data-set into clusters correctly but this information would be lost while using Euclidian distance. This problem can be taken care of either by normalizing the data or by using some other distances to calculate the proximity of a point such as Bhattacharyya distance, Mahalanobis distance or earth movers distance in some cases. Mahalanobis distance is very popular and is discussed here. Mahalanobis distance calculates the distance between a point 'p' and the mean 'c' of a distribution. [8], [15].

In the case of clusters, it can be used to calculate the distance of a point from a cluster. Mahalanobis distance quantifies the likelihood of a point belonging to a centroid and it depends critically on the assumption of data-points being normally distributed along the centroid. Let a cluster has centroids$C_1C_2....C_d$ and standard deviation $\sigma_1\sigma_2.....\sigma_d$. let a point 'P' be represented by coordinates $X_1X_2...Xd$. then the normalized distance along i^{th} dimension (Y_i) from centroid 'C' is given by equation-2.1 and Mahalanobis distance is given by equation-2.2. The benefit of using Mahalanobis distance over Euclidean distance is that now the different units and range of dimensions no longer affects the distance. Even if the range for a particular dimension is small but its standard deviation remains the same thus this distance can be used without taking care of units and the range of dimensions and in this all the dimensions are going to contribute by the same magnitude regardless of their smaller values.[8]

$$Y_i = (X_i - C_i)/\sigma_i \tag{2.1}$$

$$M.D = \sqrt{\sum_{i=1}^{d} Y_i^2} \tag{2.2}$$

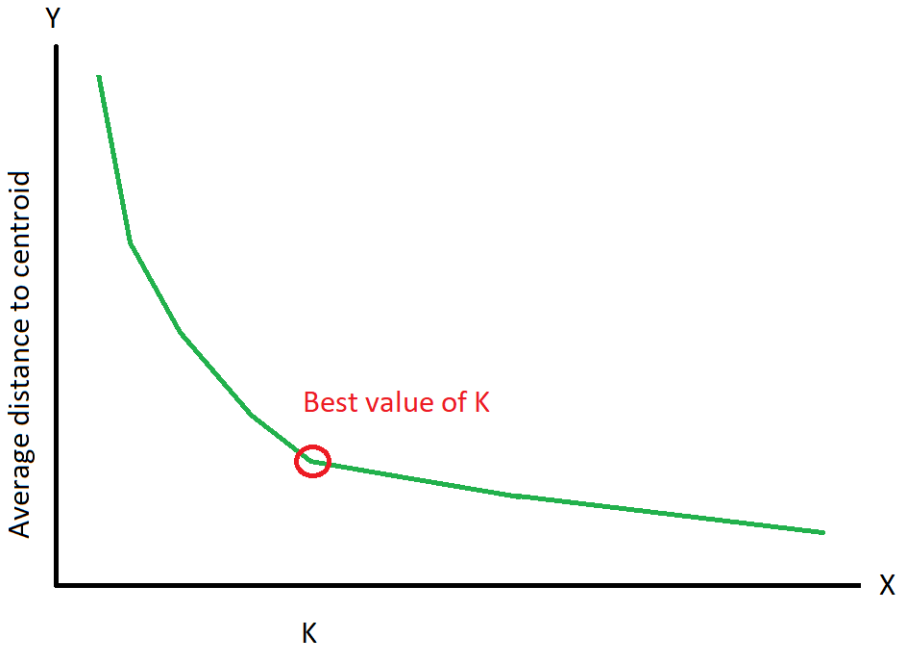


Figure 2.1: Figure showing K vs average distance plot by which correct value of K can be selected

2.3 BFR algorithm

K-means clustering and Trimmed K-means clustering algorithms can produce good results but they have some drawbacks of their own such as the deviation in the position of centroid points in the presence of outliers and computational complexity when data-set is very large. To deal with the problem of outliers and large data-sets a variant of k-means also known as **Bradley Fayyad Reina algorithm(BFR algorithm)** can be used. It can be used when the data-set is so large that it cannot be loaded into the main memory at one time. This algorithm is optimized for really large datasets. In order to perform this task, BFR-Algorithm makes a very strong assumption that each cluster is normally distributed about its centroid in Euclidean space which means that most of the points (68 percent) lie within one standard deviation of cluster whereas 95 percent of total points are located within two standard deviations from the centroid point and this assumption is made for all the dimensions. It is also assumed that all the dimensions are independent of each other and each dimension can have a different mean and standard deviation. Thus the likelihood of finding any point at a certain distance from the centroid of a cluster can be quantified. The normally distributed clusters are shown in the figure 2.3 where it can be seen that cluster-1 is an ellipse with more spread along x-axis and less spread

along y-axis which shows that the standard deviation is more for x-dimension and less for y-axis whereas for cluster-2 the deviation is more along y-axis and comparatively less along x-axis. For cluster-3 the deviation is almost the same for both the dimension which is evident from its circular shape.[7], [8]

The whole dataset is divided into smaller chunks or subsets of data-sets which can be loaded into the memory since complete data-set cannot be loaded into the memory because of its large size. One such chunk of data is then loaded in the memory at a time. The size of these chunks should be kept such that they can easily be loaded and processed in main memory because apart from these data-chunks, the information about cluster metadata has to be kept and is updated after every data read at the same time. In the metadata, the information about indices of the points is stored which occupy very less memory as compared to the whole data-set. For initializing first data-subset is loaded into primary memory from the disk and 'k' points are chosen within this dataset. These points can either be chosen randomly or by some other different methods such as chaotic initialization, orthogonal initialization or they can also be chosen from as far apart from each other as possible to get better results. Once K-points are initialized, the remaining points are to be categorized into three different categories or sets. These sets are Discard-set (DS), Compression-set (CS), and Retained-set (RS). Any point which lies close enough to the chosen centroid point belongs to DS. If the point is not close enough to the centroid point then it can belong either to CS or to RS. After this, the metadata in DS is updated and loaded points are discarded and another subset can now be loaded. If a point is not assigned to DS then it is first loaded into RS then from there it can be moved either to CS or it can remain to RS. In RS points are classified using any of the clustering algorithms, either k-means or hierarchical and then the points which are close to each other but not sufficiently close to any centroid are then transferred to CS whereas remaining points which are not close to any centroid point and are isolated, are retained in this set. The decision that whether a point is 'sufficiently close' to centroid point or not is taken with the help of Mahalanobis distance. Generally, 5 standard deviations can be taken as the threshold but it can vary depending on the problem. The whole procedure to categorize a point is explained with the help of figure-2.4. The discard

set(DS) is summarized in such a way that it can hold more information and occupy less space in terms of memory. The discard set(DS) is maintained for each centroid point in which the number of points belonging to that cluster(N) is to be maintained and updated. Another parameter is vector SUM which is nothing but the sum of all these data points within that cluster and vector SUMSQ which is nothing but the sum of the square of all elements along i_{th} dimension. These two vectors and the value 'N' helps in calculating the Mahalanobis distance. Once all the data chunks are loaded and whole data is scanned, the mini clusters in RS are assigned to the closest centroid and the points in RS can be considered as outliers or noise. [7]

BFR algorithm is a powerful technique to handle massive datasets but it has some shortcomings which are its inability to detect clusters with complex shapes which means that the clusters which are not normally distributed around their centroid cannot be identified correctly and even if the cluster is normally distributed but the axes along which it is inclined are not true 'X' and true 'Y' but to some other inclined axes, in that case, BFR algorithm fails to identify the cluster. These issues arise because of the assumption that BFR algorithm makes about the normal distribution of cluster around its centroid and about fixed axes.[8]

2.4 CURE algorithm

To deal with complex shaped clusters and massive datasets another algorithm can be used. This algorithm is known as CURE algorithm and not only can it handle massive datasets but it can also detect outliers. CURE is an acronym that stands for **Clustering Using Representatives**. In CURE algorithm, Euclidean distance is used and therefore it is recommended to normalize the data in the dataset before applying this algorithm. Normalization helps by scaling all the features and therefore all the features contribute to the distance by the same amount. CURE algorithm does not make any assumption regarding the shape of the cluster thus it is applicable to all the datasets and the results of the algorithm are unaffected as far as the shape of the cluster is concerned.[16]

Unlike in the K-means clustering algorithm and the BFR algorithm, the cluster is not merely represented by its centroid, rather multiple representative points are used to repre-

sent a cluster in Euclidean space. Similar to BFR algorithm the initialization for CURE algorithm is quite similar. Instead of loading complete dataset, a fraction of dataset is loaded into main memory and then the loaded points are clustered. The clustering algorithm used here can either be K-means, Hierarchical algorithm or density-based clustering algorithm. Once the data points are clustered, another step is the representation of cluster in Euclidean space and as mentioned before multiple points are used to represent a cluster. These points are chosen in such a way that they are as separated or dispersed from one another as possible. The purpose of choosing representative points dispersed is to cover as much cluster volume as possible. The whole dataset is then rescanned and all the points are then assigned to the closest cluster. The decision about the closeness of a point with a particular cluster can be taken either by calculating its average distance from all the representative point of all the clusters one by one and comparing or by simply calculating its distance from all the representative points and assigning this point to the cluster for which this distance is minimum. The whole process is explained with the help of figures 2.7 and 2.8.[8]

2.5 Density-based clustering algorithms

Density-based algorithms are very popular and are widely used to carry out clustering. They can be used to easily declutter noise or outliers from the main and useful data which is also their main advantage. Density-based algorithms have proven their importance in clustering of complex shaped datasets. In this section, two such algorithms are discussed which are based on the concept of clustering using density. In these algorithms, high-density data is classified into different clusters whereas low-density points are considered as noise or outliers.

2.5.1 DBSCAN

DBSCAN is one such algorithm where DBSCAN is an acronym to **Density-Based Spatial Clustering of Application with Noise**. Low and high-density data points are shown in the figure-2.10 where it can be seen that high-density points are part of the

cluster while low-density points are outliers. These outliers can be easily detected using this algorithm and can be removed from the data. Two parameters are required to be defined before proceeding with DBSCAN. These parameters are (ϵ) and min-points. The parameter 'ϵ' is nothing but a radial distance, calculated from a point which is the measure of how close a point should be from another point to be considered as a core point. Before the completion of DBSCAN, when a cluster is not completely formed, a cluster is represented by its core points and a core point is a point which is surrounded by other neighboring points such that the number of neighboring points exceeds min point and all those neighboring points should be within the proximity of 'ϵ'. This process of selecting a point as the core point is explained in the figure. The parameter ϵ is user-defined and has to be chosen carefully, it represents the distance and if a point has a distance less than this from currently visited point then this point is considered as a neighboring point and this condition has to be checked in all the possible direction.[17], [18],

The algorithm starts from the with an arbitrary point which is not yet visited and its distance is calculated from all other points, any point having a distance less than ϵ is considered as its neighboring point and in this way, all the neighboring points are extracted. If the number of neighboring points is higher than 'min-point' then this point is marked as core-point and also marked as visited whereas if the condition is unsatisfied then this point is marked as noise but, later on, this point can also become part of the cluster. If a point is a core-point then its neighboring points are also the part of the cluster and the above-described process is repeated for all the neighboring points and this process is repeated for all the remaining unvisited points and they will either become the neighboring points or the core points. After that, all the core points sharing the same neighborhood are joined and the cluster is formed and the remaining points are marked as noise.[19], [20]

2.5.2 Peak density-based clustering

Another popular density-based algorithm is the peak density-based clustering algorithm where regions of high density are calculated and based on that, the decision is made. In this algorithm, a cluster is characterized with a relatively high density region as compared

to its neighbors and using this idea which is proposed in this algorithm, outliers can be easily spotted and removed. As mentioned above, clusters with arbitrary shapes can be easily identified and categorized using this algorithm. This algorithm makes an assumption that the cluster centers are at higher density than to their immediate neighbors and the immediate neighbors of cluster centroids are at relatively higher density than their neighbors.[21], [22], In this algorithm, for every single point in the dataset, two parameters, local density of $i_{th}point$ (ρ_i) and distance of i_{th} point from a point which is closest to it but is at relatively higher density(δ_i) are calculated. To calculate the local density can be calculated from equation no. 2.3

$$\rho_i = \sum_j X(d_{ij} - d_c) \tag{2.3}$$

Here d_c is cutoff distance, any point which is at a higher distance from that from i_{th} point, contributes zero while a point with a distance less than or equal to the threshold contributes one to the density. Another parameter δ which is a measure of the minimum possible distance between i_{th} point and a point with relatively higher density. The mathematical expression to evaluate δ_i is given by equation-2.4.

$$\delta_i = min(d_{ij}); \rho_j > \rho_i \tag{2.4}$$

It is not possible to assign δ value to a point with the highest density. For a point with the highest density, maximum distance is used in equation-3 instead of the minimum. From the values of δ and ρ, a decision graph between ρ_i vs δ_i is plotted, and points with relatively high ρ and δ are chosen as centroid points whereas points with higher δ but relatively lower ρ can be considered as outliers. This process is illustrated in the figure. The next step is to assign points to the cluster centroids, the remaining points are assigned to the same cluster as the closest neighbor of higher density. After this, the border region of the cluster is found which can be defined as the set of points whose distance is less than or equal to d_c from other points which are assigned to another cluster. In the border region, a threshold density is found and the density of all the points lying in the border region is compared with the threshold. Any point with a density less than

the threshold is dropped and rest of the points are kept.[23], [24]

2.6 Orthogonal experimental design based Optimization algorithms

Experimental design refers to how a 'model' whose performance is to be tested, performs when subjected to different conditions. These conditions can either take discrete values or they can be continuous or some of them can be discrete and some with continuous values. There can be at least one such condition or there can be many conditions affecting the performance of the model. For example, pressure affects the performance of wristwatch, where user can be a diver who dives deep underwater where pressure is high comparative to another user who is a hill climber where pressure is low. Therefore before launching a product like that the designer has to make sure that the product is going to perform equally well under all such circumstances. The process sounds simple for a few conditions but becomes extremely tedious when the conditions, under which the behavior and the performance of an object are to be tested, become more. For example, let there be 15 conditions affecting the performance of a model and each variable can take 10 different values then there will be 15^{10} possible combinations which are more than five hundred billion. Thus it is impossible for any designer to test the product at every possible combination of variables. Therefore it is required to sample these combinations and to get a small but representative set of combinations in order to test the model.In order to sample the combinations, the orthogonal design was developed. With the help of orthogonal design a series of orthogonal arrays are generated which gives the indices or the information about how to sample the whole combinations in an efficient way. Despite the fact that only a fraction or a subset of the total combinations is considered but this subset contains a diverse set of conditions(conditions).[25], [26]

2.6.1 Orthogonal array generation

The cocept of orthogonal design is explained with the help of an example.The production of a certain vegetable is affected by three factors: 1) Temperature, 2) the amount of

Table 2.1: Table showing(a)An experimental design problem with three factors and three levels and (b) Orthogonal combinations generated from this information [2]

(a)

	Factors		
Levels	Temperature	Amount of fertilizers	ph value
L1	20^0C	100g/m^3	6
L2	25^0C	150g/m^3	7
L3	30^0C	200g/m^3	8

(b)

Combinations	Factors		
	Temperature	Amount of fertilizers	ph value
1	20^0C	100g/m^2	6
2	20^0C	150g/m^2	7
3	20^0C	200g/m^2	8
4	25^0C	100g/m^2	7
5	25^0C	150g/m^3	8
6	25^0C	200g/m^3	6
7	30^0C	100g/m^3	8
8	30^0C	150g/m^3	6
9	30^0C	200g/m^3	7

fertilizer used and 3) the pH of the soil. In this experiment, there are three 'factors' that are responsible for the yield of vegetable. Let each factor can take three different values, the temperature can either be 20 or 25 or 30 degrees and three different fertilizers can be used in the experiment and are marked as 'A', 'B' and 'C'. while the pH value of soil can be 6, 7 or 8. It can be seen clearly that there are three features which can take three different values, producing 27 different combinations but there is a way by which few samples can be selected from these 27 combinations in such a way that they are evenly scattered over the search space yielding better representatives for all the possible combinations. These representatives are mentioned in table 2.1(b) and later in the section, the method to design these samples is described.[2]

The orthogonal array produces an array of indices of different combinations. Let, for an experiment, there be 'N' factors or conditions or dimensions and each of these factors can take 'Q' values therefore there can a total of Q^N possible combinations and out of

these, few scattered combinations are to be selected. Another variable 'M' is defined as $M = Q^J$. Here 'J' is another variable that can be calculated from the equation-2.5

$$N = \frac{Q^J - 1}{Q - 1} \tag{2.5}$$

. Now it can be observed from equation-3.1 that 'J' depends on 'N' and therefore it is not necessary that there exist an integral value of 'J' for every 'N' in that case the condition to find 'J' can be bypassed by inequality-2.6.

$$N' = \frac{Q^J - 1}{Q - 1} \tag{2.6}$$

The smallest possible value of 'J' should be chosen that satisfies inequality. Once J is chosen another variable N' is defined as given by equation-2.7.

$$N \leq \frac{Q^J - 1}{Q - 1} \tag{2.7}$$

The next step is to calculate the basic columns in the orthogonal array. To calculate the basic columns, first the value of 'J' is calculated from equation-2.5, if 'J' turned out to be an integer otherwise 'J' is calculated from inequality-2.6. Now another variable 'k' is defined and 'k' varies from 1 to J. For each 'k' ,'j' is calculated from equation-2.8 and basic elements of orthogonal array can be calculated from equation-2.9 where 'i' varies from1 to Q^J

$$j = \frac{Q^{K^{-1}} - 1}{Q - 1} + 1; \tag{2.8}$$

$$a_{i,j} = \lfloor \frac{i - 1}{Q^{J-k}} \rfloor mod(Q) \tag{2.9}$$

To construct non-basic column 'k' is varied from 2 to J and 'j' is calculated from equation 2.8. For another variables 's' varying from 's' $= 1$ to $j - 1$ and 't' varying from 1 to $Q - 1$ non-basic elements are calculated from equation-2.10

$$a_{j+(s-1)(Q-1)+t} = (a_s \times t + a_j) mod(Q) \tag{2.10}$$

$a_{i,j}$ is Incremented by 1 for all $1 \leq i \leq M$ and $1 \leq j \leq N$ and the last $N' - N$ columns

are eliminated to get final arrray.

The generation of orthogonal combination is explained with the help of an example.Consider a vector with factors, $N = 3$ and levels, $Q = 4$ and each factor is cosidered to be varying from [0 5] linearly.The values that this vector's each feature can take is shown in the table-2.3. From this table it can be observed that total number of possible combinations are 4^3 (64) but using orthogoanal array a variety of different combinations can be tested. Orthogonal array can be created for N=3 and Q=4 which are shown in table-2.4. With the help of table-2.3 and using the indices obtained from table-2.4 orthogonal combinations can be calculated as shown in table-2.5. It can be observed that the number of orthogonal combinations obtained are 16. Thus a wide variety of different combination can be tested.[27], [28]

2.6.2 Orthogonal genetic algorithm with quantization

In Orthogonal genetic algorithm with quantization, initialization is done with the help of orthogonal array which helps in exploring the search space in an efficient way. For a better initialization it is less probable that algorithm will struck at local minima.Consider an example to minimize a function f(.) depending on five variables,($x1$, $x2$, $x3$, $x4$ and $x5$) and each variable has a specified range between ($a1$, $a2$, $a3$, $a4$, $a5$) and ($b1$, $b2$,$b3$, $b4$, $b5$) with 'a_i' as lower and 'b_i' as the upper bound. It can be speculated that for this function the search space is defined as [a_i, b_i]. The next step is to quantize the search space. For quantization, a quantization variable '$Q1$' is chosen and from this variable the search space can be divided into '$Q1$' levels which are linearly spaced. For example if $Q1$ is taken as 10 then every a_i and b_i is divided into 10 linearly spaced values including both a_i and b_i. After quantization initialization has to be done and as discussed above. An orthogonal array is created for 'N' and'$Q1$' and representative combinations can be obtained. These combinations can be used as initial population.

The next step is selection. for selection, the fitness of the initialized population is calculated and the best 80-90 percent population is selected for the crossover. The crossover is also orthogonal, and a variable '$Q2$' is defined and the features of both the parents who are participating for crossover, are sorted and arranged in increasing

Table 2.2: Table showing:(a) Showing different levels of three features, (b) Orthoganal array for N=3 Q=4 (c) The generation of orthogonal combination with the help of orthogonal array

(a)

factor1	factor2	factor3
0	0	0
1.67	1.67	1.67
3.33	3.33	3.33
5	5	5

(b)

1	1	1
2	2	3
2	3	4
2	4	1
2	1	2
3	2	4
3	3	1
3	4	2
3	1	3
4	2	1
4	3	2
4	4	3
4	1	4
1	2	2
1	3	3
1	4	4

(c)

0	0	0
1.67	1.67	3.33
1.67	3.3	5
1.67	5	0
1.67	0	1.67
3.33	1.67	5
3.33	3.33	0
3.33	5	1.67
3.33	0	3.33
5	1.67	0
5	3.33	1.67
5	5	3.33
5	0	5
0	1.67	1.67
0	3.33	3.33
0	5	5

order before proceeding towards crossover. The orthogonal crossover is explained with an example. Consider $P1$ = [1.6, 1.6, 3.3] and $P2$ = [3.3, 0, 1.6] be the pair selected for crossover, then after sorting them a new space is created with lower bound = [1.6, 0, 1.6] and upper bound as = [3.3, 1.6, 3.3]. After defining the lower and upper bound, this space is again divided in $Q2$ linearly spaced levels as done before and another orthogonal array is created with N factors and $Q2$ levels from above mentioned procedure and from that array create another set offspring. Mutation is the very next step. To perform mutation with probability P_m on a chromosome, an interger is randomly generated between [1 N] followed by a real number ranges between lower and upper bound for that factor

and then the original value of that factor is replaced by the randomly generated number. Best chromosomes can be selected out of the original parents, offsprings and from mutated population.[2]

2.6.3 Orthogonal Particle swarm optimization

Particle swarm optimization is an optimization technique which is based on social behaviour of bird flocking. Since it is population based search technique, it has an advantage of exploring the search space with very low probability of getting struck into local minima. First the standard particle swarm optimization is briefly explained.

For an N-dimensional search space, 'K' particles each with 'N' dimensions are initialized in the search space randomly, followed by the velocity initialization. Velocity is also a vector with same dimension as of the swarm and it is also initialized randomly. After that the fitness of each particle is calculated from the fitness function which depends on the function to be optimized. Fitness evaluation gives the best local and global performance of every individual particle and the swarm respectively. Now the velocty of each particle is updated from their own individual local best and global best of the swarm. Velocity update is followed by the position update. The updation equations 2.11 and 2.12 are the update equations [3]

$$V_{in}(t+1) = WV_{in}(t) + c1 \cdot rand() \cdot (P_{in}(t) - X_{in}(t)) + c2 \cdot rand() \cdot (P_{gn}(t) - X_{in}(t)) \quad (2.11)$$

$$X_{in} = X_{in}(t) + V_{in}(t+1) \quad (2.12)$$

where i=1,2..K and denotes particles index, n=1,2..N denotes the n^{th} dimension of i^{th} particle and t denotes the no. of iteraton.

In OPSO the movement of a particle is controlled by IMM(intelligent move mechanism) which works in a different way as compared to the orthodox movement mechanism as explained above. IMM controls the flow of particle with the help of orthogonal experimental design(OED) and factor analysis. Orthogonal experimental design works on Orthogonal Array. First an orthogonal array is created as was explained before in OGA

algorithm for N factors and Q levels. Once the array is created the population can be initialized with the help of OA and the fitness of the population is evaluated. After evaluating the fitness of the population factor analysis is applied on it. The purpose of factor analysis is to discover the best combination of the levels. In factor analysis main effect is evaluvated on the population for all the individual. Mathematically main effect S_{jk} is given by the equation as,

$$S_{jk} = \frac{\sum_{z=1}^{M} F_z.f_z}{\sum_{z=1}^{M} F_z} \tag{2.13}$$

where $z = 1...M$, f_z denotes the fitness function over a z^{th} combination, M is the total no. of combination, k=1...Q and $F_z = 1$ if the level of the factor j of combination is k otherwise $F_z = 0$. The whole procedure is explained with the help of an example. let there be three factors (A, B, C) and each factor has three levels as shown in the table 2.3. Now orthogonal array is created for this setup with $N = 3$ and $Q = 3$.the ortho gonal array is shown in table 2.4

After creating orthogonal array, the next step is to initialize the population, and calculating the fitness of each individual.which is done with the help of array and applying the main effect is shown in table 2.5. Since the experiment is not real therefore the fitness results are assumed and written based on assumption only. Thus, the effect of every level on every single factor is calculated and considering the above experiment as a maximization problem, the best best combination can be selected from analysing the best or greatest value(in this case) for S_{nq}. It is clear from table-2.5 that the best results are the combination of factor A with 3^{rd} level, factor B with 2^{nd} level and factor C with 2^{nd} level. It is also to be noticed that this combination was not present in the previous nine combinations.

The OPSO is a modified version of standard version of PSO, modified with the help

Table 2.3: Table showing factors and levels for an arbitrary experiment

Levels	A	B	C
1	A1	B1	C1
2	A2	B2	C2
3	A3	B3	C3

of OED and factor analysis. In OPSO, two temporary moves are generated H and R. H represents the cognitive learning part while H represents the social learning. H is calculated from individual best position and present position while R is calculated from global best position and present position of particle as shown in equation (8) and (9). Depending on the problem the N dimensional H and R can be further divided into D dimensional non overlapping vectors such that D ¡ N.The efficiency of IMM and the performance of algorithm depends on the value of D. Larger the value of D the more efficient the algorithm will be.

Step1: cognitive learning component H and social learning component R are created from given equation (2.14) and (2.15).

$$H = X_{in} + WV_{in}(t) + c1 \cdot rand() \cdot (P_{in}(t) - X_{in}(t)). \tag{2.14}$$

$$R = X_{in} + WV_{in}(t) + c1 \cdot rand() \cdot (P_{gn}(t) - X_{in}(t)). \tag{2.15}$$

Step 2: H and R are divided into D dimensional space and every dimension is treated as a factor.

Step3: OA is created for factors $N = D$ and levels Q =2.

Step4: from OA find the M different combinatins where M is number of rows in OA.

Step5: the fitness of every single combination is evaluated with the help of fitness function or objective function.

Step6: Main effect S_{jk} is calculated for j = 1...D and k=1,2.

Step6: the best factor is picked up.

Step7:the next move of particle is based on this factor.

Step8: it is to be varified that the new position $X_i(t+1)$ is better then the previous position $X_i(t)$. If the condition is not satisfied then $X_i(t+1)$ is replaced by the best among $X_i(t)$.

2.7 Grey wolf optimization

Grey wolf optimization which is a metaheuristic optimization technique inspired by the behavior of grey wolf scientifically known as canis lupus. The behaviour of grey wolf during hunting has been mathematically modelled.GWO follows the leadership and hunting operation of the grey wolf in the wild [2]. Grey wolf can be categorized into four different categories in a pack, the alpha wolf, the beta wolves, the delta wolves, and the omega wolves. Grey wolves are apex predators that is, they are at the top of the food chain. Like any other wolf, they also like to live in a pack and prefer to hunt together in a pack with mutual efforts. It has also been observed that the size of the pack is usually 5 to 12 on average. The leader of the pack can be male or female and known as alpha. The alpha being the leader of the pack is responsible for making all the important decisions about sleeping time, the time to wake up and time to hunt. Alpha wolf dominates the pack because he is the strongest among them all however it is also been observed that apart from giving orders alpha also follows orders from other comparatively less strong and inferior but experienced wolves. It is only the alpha wolf in the pack who is allowed to mate in the pack. It is also very interesting to note that it may not be necessary that alpha wolf is the strongest wolf although alpha is among the strongest wolves but he/she is also supposed to be the smartest wolf among all, therefore, alpha has to be best in decision making, keeping the pack in discipline because a well-organized pack is stronger than an unorganized one. Another level of hierarchy is of beta wolves they are generally more than one. They can be considered as second-in-command of the pack Their duty is to help alpha wolf in decision making and organizing the pack. They are the best candidates to become or replace alpha wolf in case alpha becomes very old or gets wounded or dies. They are responsible for implementing the decision taken by alpha wolf in the pack. The third category according to the hierarchy is delta wolves. They perform multiple task and they hold the third most important position in the pack. Some of them guard the boundaries of the dominated land of the pack and inform their superior wolves about any such threat to their territory while some of them take part in the hunt along with the other beta wolves and alpha wolf. Some of the delta wolves also take care of wounded wolves. The last of the wolves among the pack are omega wolves. They are the

least important candidates among the pack but it has been seen that pack faces some internal problems when it comes to loosing their omega wolves. They play the role of space goat. Since they are in lowest in hierarchy, they have to submit to all other wolves i.e. the alpha, beta wolves and delta wolves. Their share is lowest in the prey and at the time of hunt they simply follow all other wolves. It has also seen that the do babysitting for the pack by taking care of the young ones of the wolves. The hunting done by the pack is also interesting. The hunting is proceeded in three main steps which includes tracking the prey and then chasing and informing other members of the pack and then quietly approaching the prey. The next step is to pursuing, encircling and harassing the prey until it stops moving. Final step involves attacking the already freezed prey. For an optimization problem the whole scenario is mathematically modelled [/cite grey wolf optimization]. The search space can be considered as the territory of the grey wolf and they cannot hunt beyond their territory meaning,[**?**] solution has to be found inside the prespecified search space. Since the orthogonal grey wolf optimization is a population based meta heuristic optimization algorithm therefore multiple points have to be initialized in the search space which can be considered as grey wolf pack.Apart from population based meta heuristic, grey wolf optimization also follows the concept of swarm intelligence (SI) and so is the orthogonal grey wolf optimization, which preserves the information about the solutions over every iteration. The best solution obtained so far is preserved with the help of memory, which help these algorithms to obtain a better result.The next step is to set the hierarchy as discussed above since there are four main categories of grey wolves, therefore the whole initialized set is to be categorized as the alpha-solution which is the best solution among all, the beta-solution which the second best among the initialized set and the delta-solution which is the third best solution in the search space from the set, rest of the solutions are considered as omega-solutions.This is done by calculating the fitness of every point from objective function which is to be optimized.As discussed above the hunting is carried out in three different steps these steps are mathematically modelled. The encircling prey is given by the equation-2.16 and equation-2.17 [1], [29] [30]

$$\vec{D} = |\vec{C}.\vec{X_p}(t) - \vec{X}(t)| \tag{2.16}$$

$$\vec{X}(t+1) = \vec{X_p}(t) - \vec{A}.\vec{D} \tag{2.17}$$

Where 't' is the current iteration, $\vec{A}$ and $\vec{C}$ are coefficient vectors, X_p is the position vector of the prey and $\vec{X}$ represents the position of grey wolf. Coefficient vector A and C are updated according to equation-2.18 and equation-2.19

$$\vec{A} = 2\vec{a}.\vec{r} - \vec{a} \tag{2.18}$$

$$\vec{C} = 2\vec{r2} \tag{2.19}$$

Where the components of $\vec{a}$ are to be decreased linearly from 2 to 0 over the course of iterations and r1 and r2 are random numbers between 0 and 1.It has been observed that generally in the hunting, the hunting is guided by alpha-wolf and the beta and delta-wolves take part in the hunting by following the lead of alpha the rest of the pack i.e. the omega-wolves follows either the alpha or the beta or the delta-wolves.The position of the prey can be understood as the location of optimal solution but it is not known, not before initializing neither any thing can be said about it after the initialization. It is assumed that the alpha, beta and the delta wolves have the best knowledge about the position of the prey therefore the first three best position are saved and the new variables D_α, D_β, D_δ are calculated from equation-2.20, equation-2.21, and equation-2.22.

$$\vec{D_\alpha} = |\vec{C_1}.\vec{X_\alpha} - \vec{X}| \tag{2.20}$$

$$\vec{D_\beta} = |\vec{C_2}.\vec{X_\beta} - \vec{X}| \tag{2.21}$$

$$\vec{D_\delta} = |\vec{C_3}.\vec{X_\delta} - \vec{X}| \tag{2.22}$$

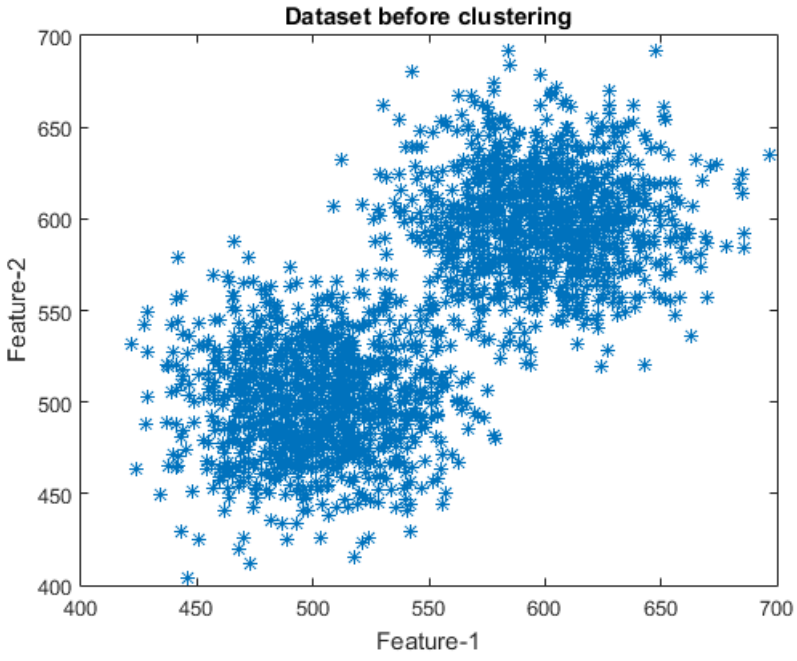
$$\vec{X_1} = \vec{X_\alpha} - \vec{A_1}.(\vec{D_\alpha}) \tag{2.23}$$

$$\vec{X_2} = \vec{X_\beta} - \vec{A_2}.(\vec{D_\beta}) \tag{2.24}$$

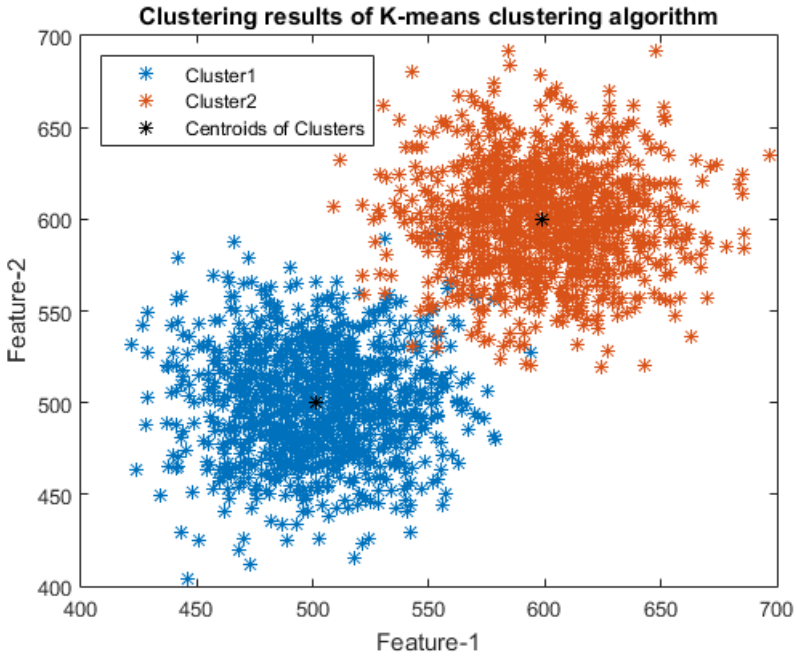
$$\vec{X_3} = \vec{X_\delta} - \vec{A_3}.(\vec{D_\delta}) \tag{2.25}$$

$\vec{X}1$, $\vec{X}2$ and $\vec{X}3$ represents the movement towards the global best, second best and third best solution in the search space respectively and these values has to be clculated for every member in the pack where as $\vec{X_\alpha}$, $\vec{X_\beta}$ and $\vec{X_\delta}$ are the respective positions of global best, second best and third best solution in the search space. The final position of each member is then updated from equation-2.26 in grey wolf optimization.

$$X(\vec{t+1}) = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \tag{2.26}$$



(a)



(b)

Figure 2.2: Example of K-means clustering: (a) Original unclustered dataset (b) Well clustered dataset.

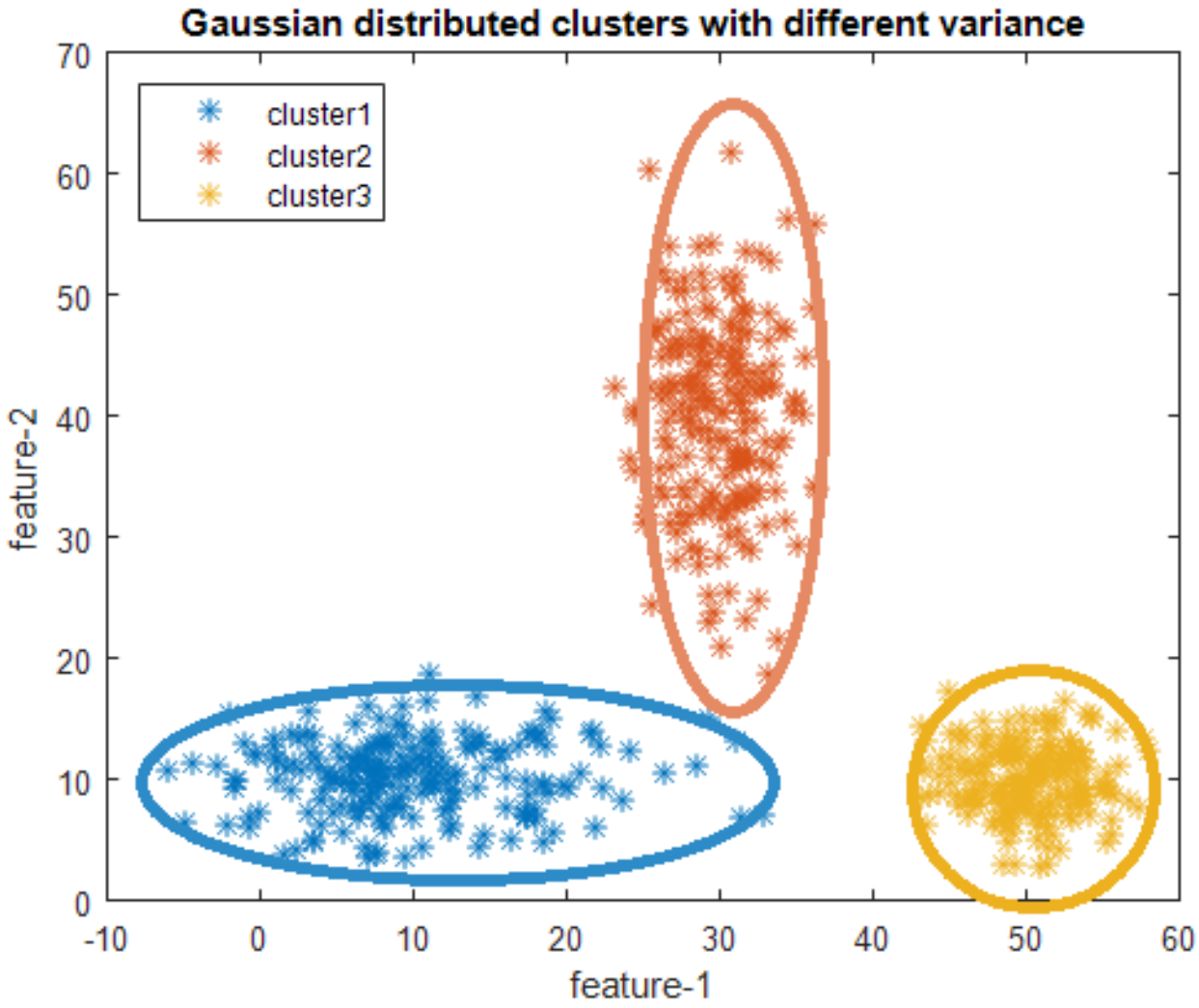


Figure 2.3: Figure showing normally distributed clusters, appropriate for BFR algorithm

Table 2.4: Table showing orthoganal array for N=3 Q=3

1	1	1
1	2	2
1	3	3
2	1	2
2	2	3
2	3	1
3	1	3
3	2	1
3	3	2

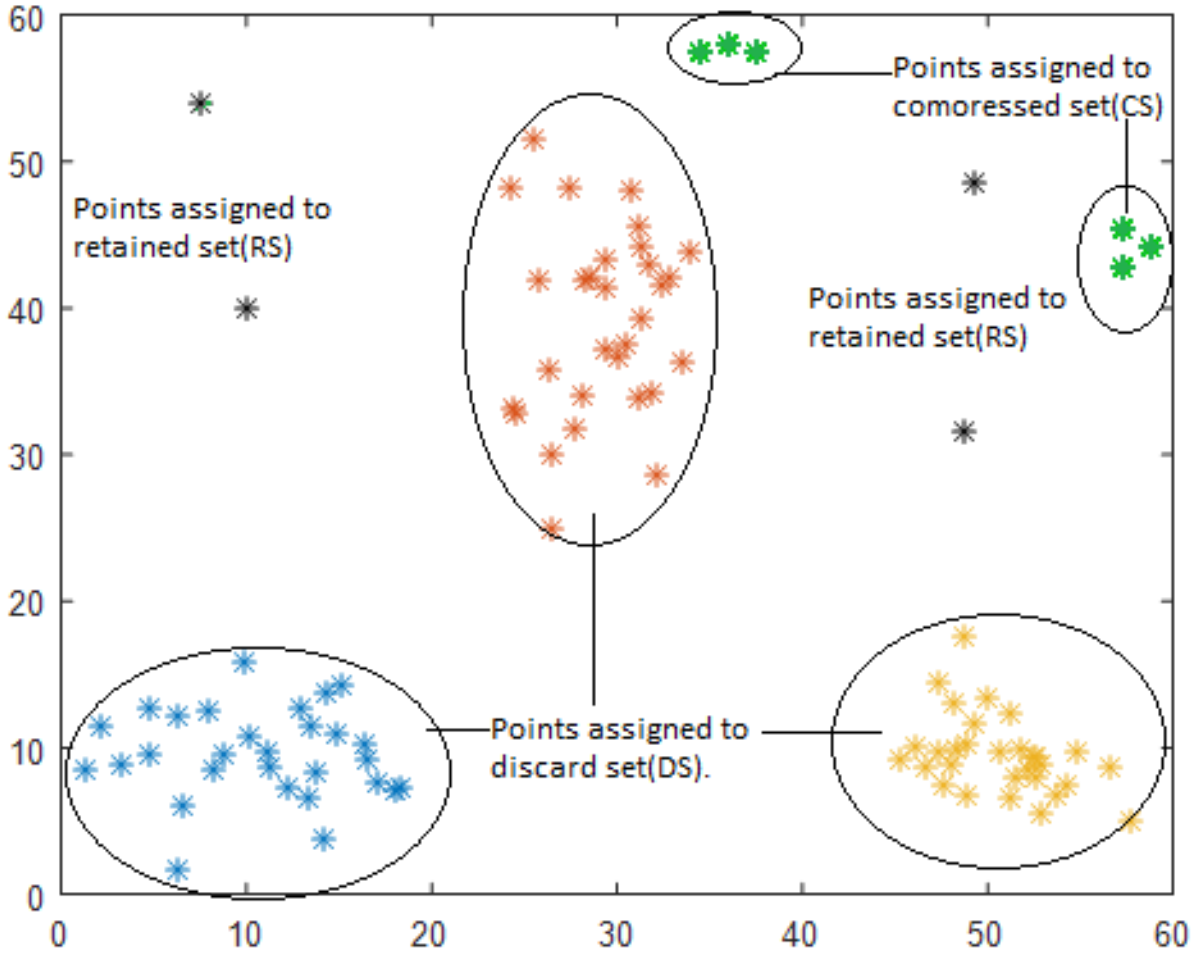


Figure 2.4: Figure showing the procedure to assign an incoming point to different sets in BFR algorithm

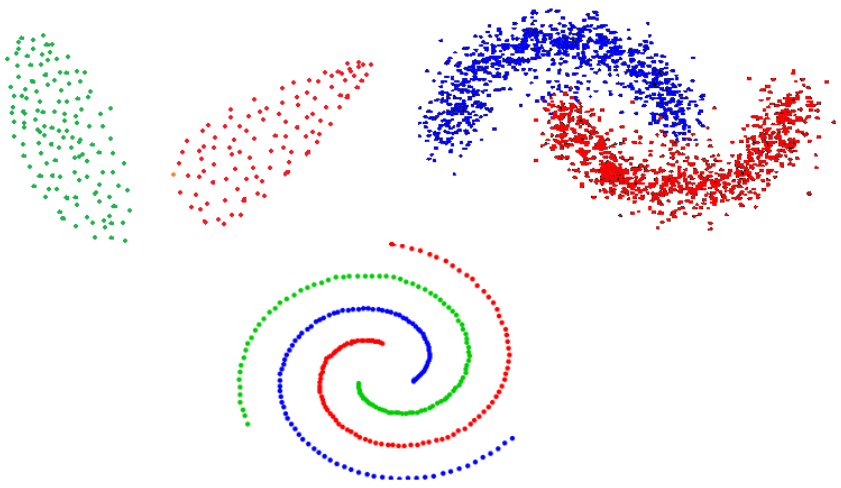


Figure 2.5: Shapes of clusters that cannot be identified using K-means and BFR algorithm

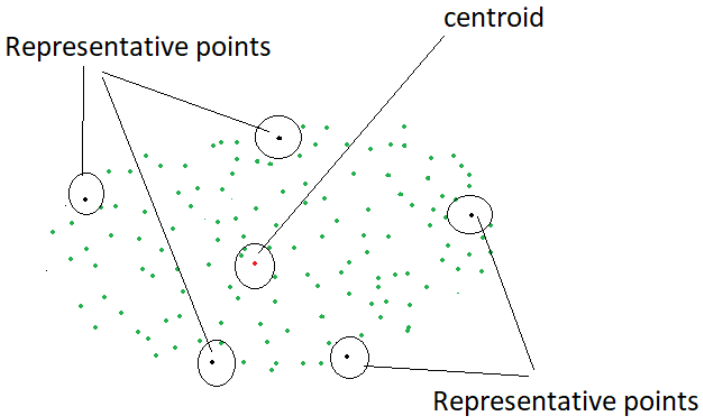
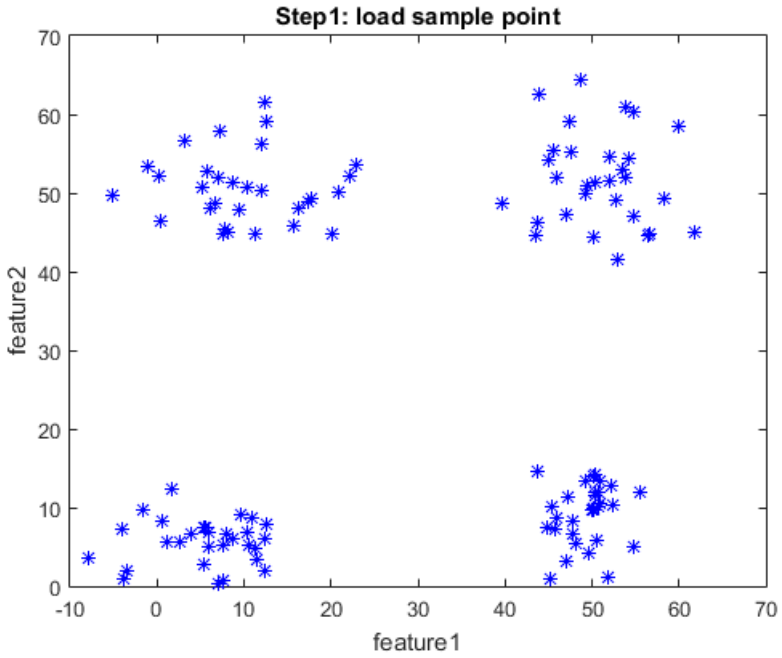


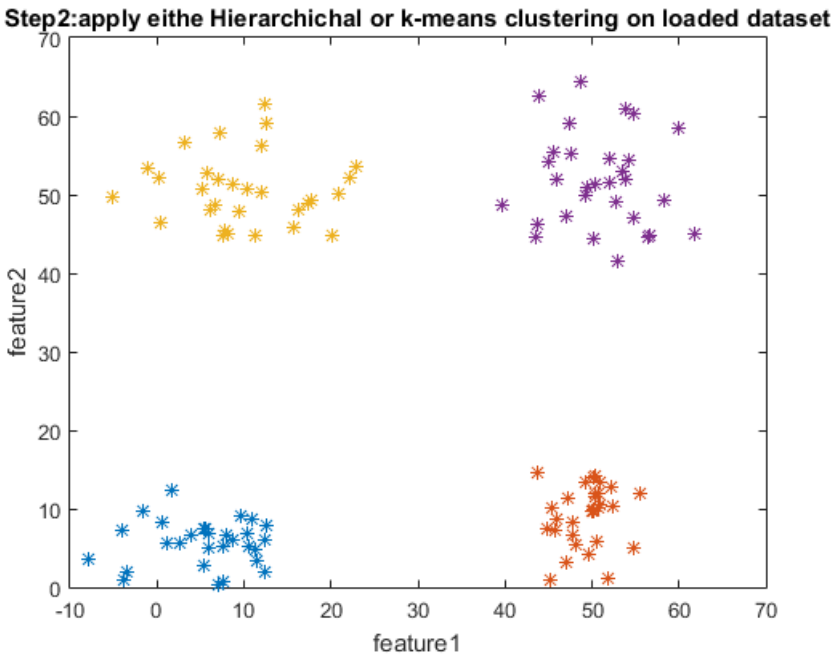
Figure 2.6: Figure showing the representation of a cluster by multiple represebtative points.

Table 2.5: deciding the best combination level of given experiment using OED method [3]

Combinations	A	B	C	fitness
C1	A1	B1	C1	31
C2	A1	B2	C2	54
C3	A1	B3	C3	38
C4	A2	B1	C2	53
C5	A2	B2	C3	49
C6	A2	B3	C1	42
C7	A3	B1	C3	57
C8	A3	B2	C1	62
C9	A3	B3	C2	64
Levels	Factor Analysis			
L1	(F1+F2+F3)/3 = 41	(F1+F4+F7)/3 = 47	(F1+F6+F8)/3 = 45	
L1	(F4+F5+F6)/3 = 48	(F2+F5+F8)/3 = 55	(F2+F4+F9)/3 = 57	
L1	(F7+F8+F9)/3 = 61	(F3+F6+F9)/3 = 48	(F3+F5+F7)/3 = 48	
OED Results	A3	B2	C2	

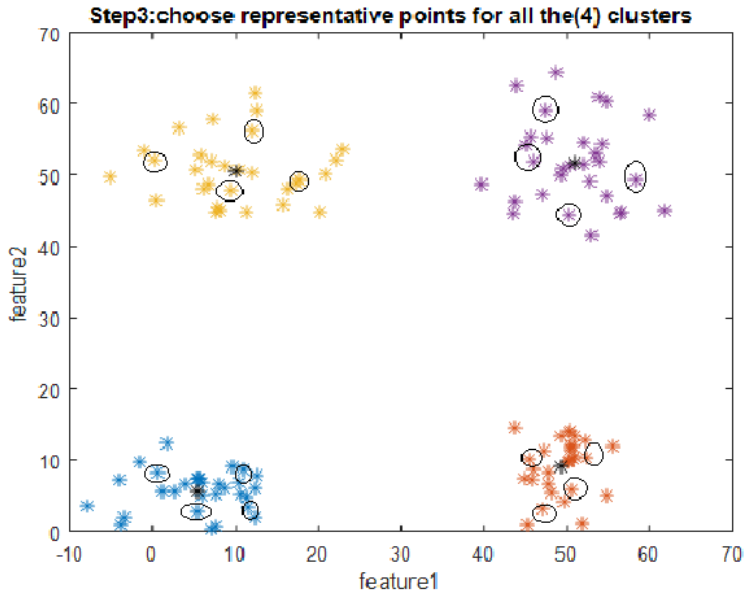


(a)

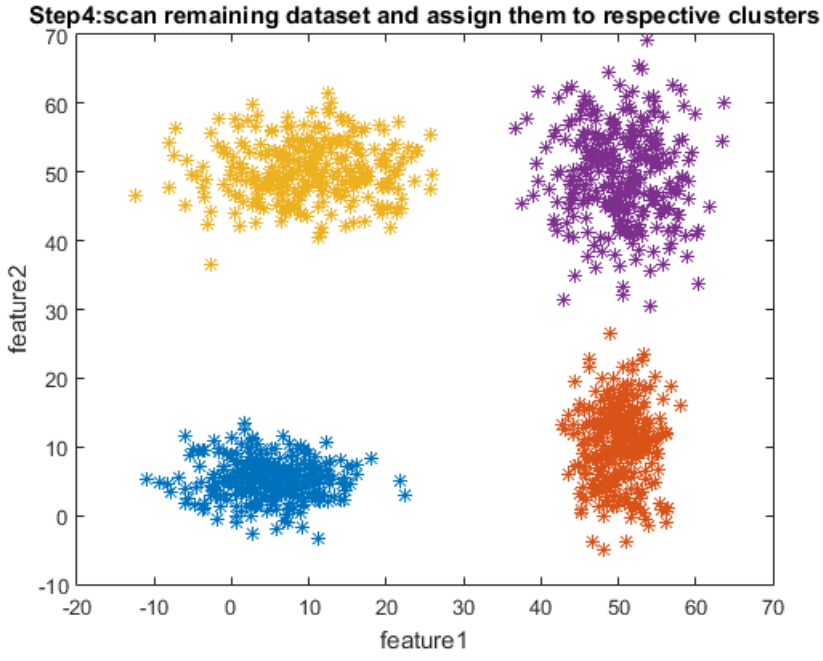


(b)

Figure 2.7: Figure showing (a) initialization and (b) clustering of initially loaded data-points in CURE algorithm



(a)



(b)

Figure 2.8: Figure showing (a) representation of mini cluster and (b) assigning incoming points to cluster in CURE algorithm

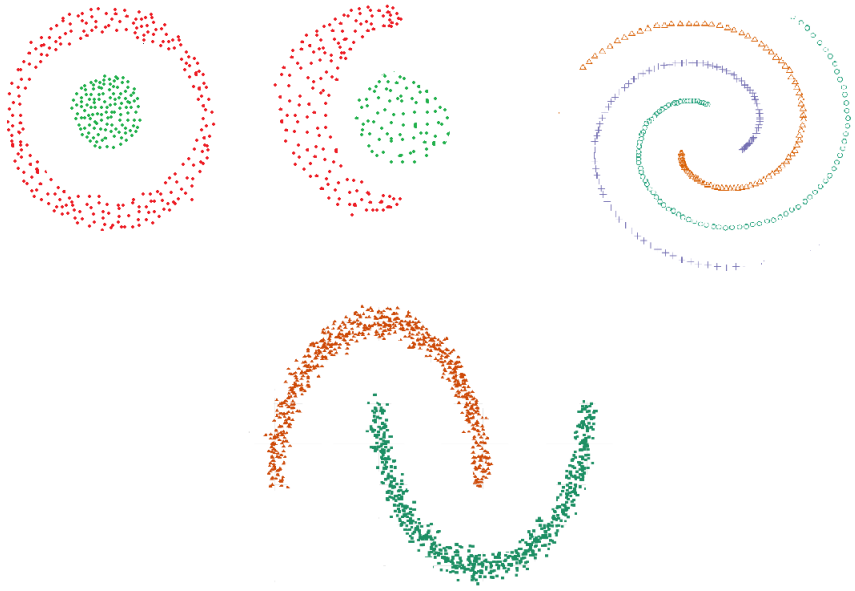


Figure 2.9: Complex shapes that can be clustered using density-based clustering algorithms

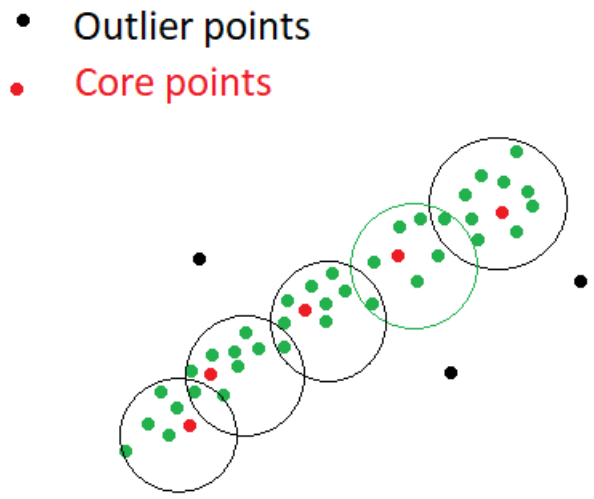


Figure 2.10: figure showing high density core points and outlier points at low density

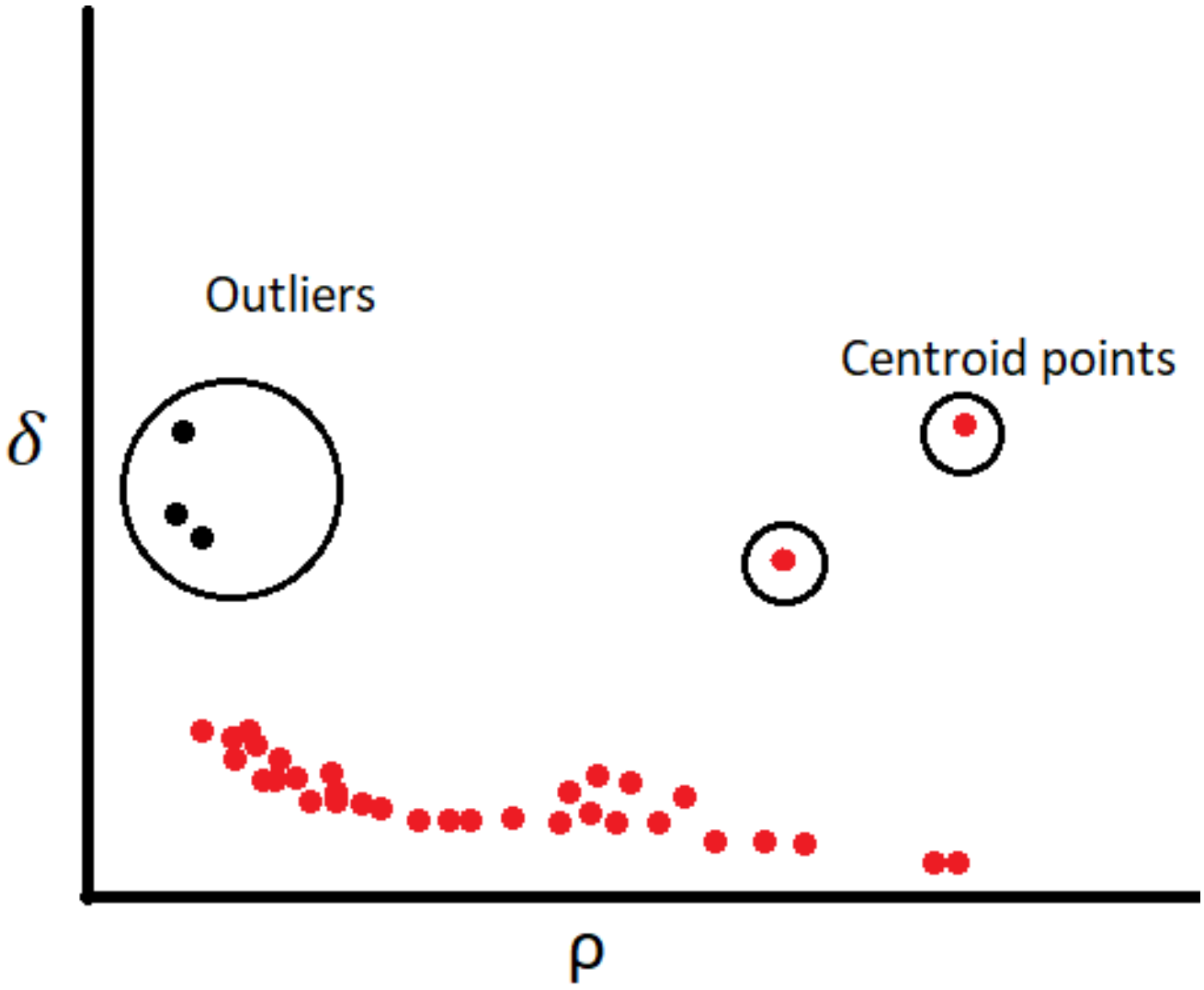
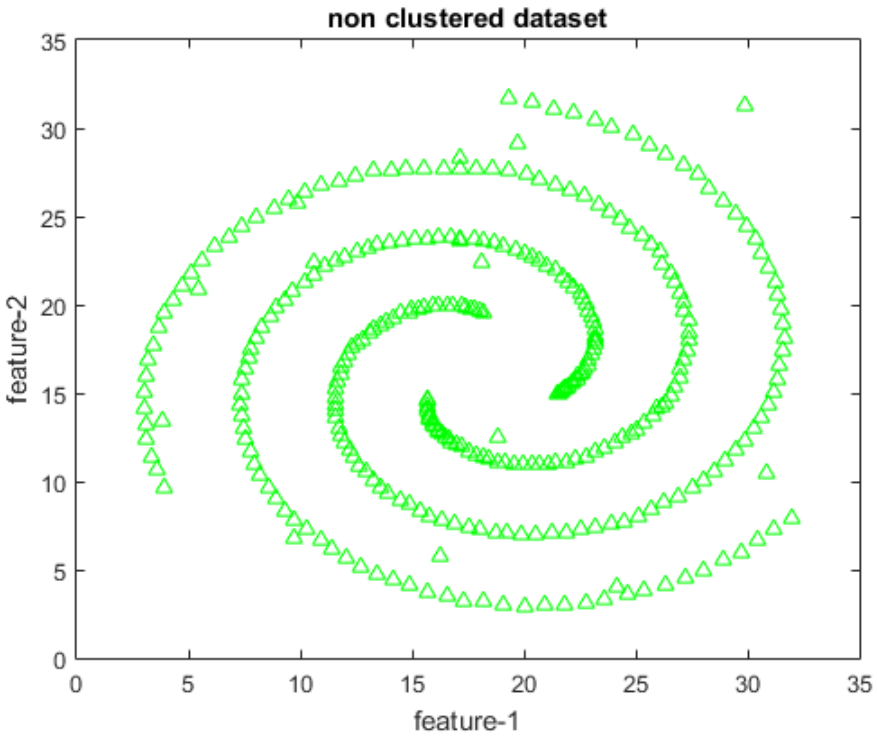
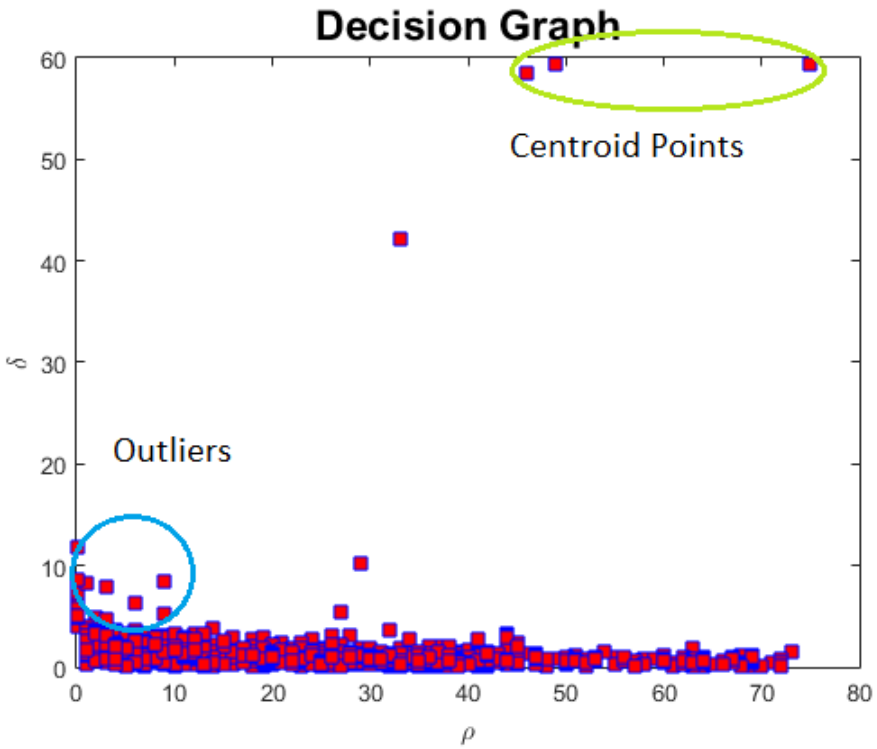


Figure 2.11: figure showing decision graph and poits with high δ values but low ρ value are rejected as outliers and points with high ρ and δ are selected as centroids



(a)



(b)

Figure 2.12: figure showing (a) unclustered dataset and (b) decision graph for the dataset

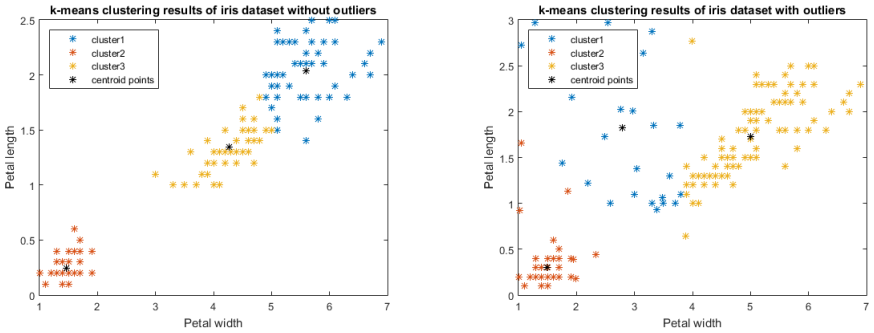
Chapter 3

Robust clustering

Clustering is[31] a method to club a set of similar elements such that similar elements lie in the same group which is also known as a cluster. Elements in the same group or cluster are more identical to each other than the elements in different clusters. The criteria to classify elements is to find out the similarity between them which can be found by analysing their features. Clustering is unsupervised learning that is why it is different from classification. A clustering algorithm should be robust in order to deal with practical datasets.Generally the outcome of an algorithm is affected by the presence of outliers. In this chapter, robust clustering algorithms are used for outliers removal.

3.1 Implementation of Trimmed K-means algorithm

K-means algorithm works on clusters which are normally distributed but the presence of outliers affects the outcome of the algorithm significantly. The deviation of the centroid points from their original position is shown in figure 3.1 where outliers were added in fisheriris-dataset and the results of K-means algorithm are shown for both the cases. The purpose of Trimmed K-means algorithm is to predict the position of centroids correctly. It is evident from figure 3.2 that centroids are identified with high accuracy.



(a) (b)

Figure 3.1: Effect of outliers on the performance of K-means clustering: (a)Result on Iris dataset without outliers (b) Result on Iris dataset after adding outliers

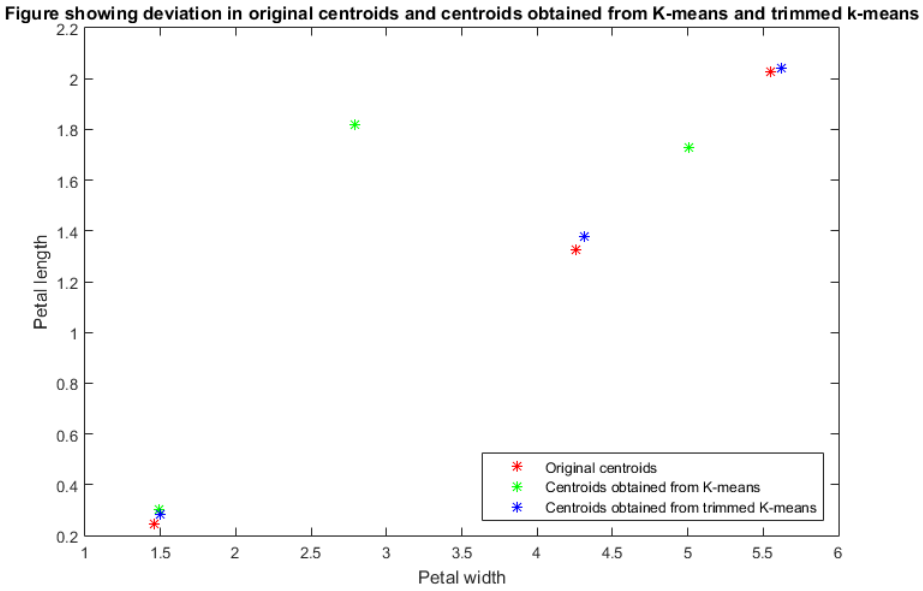


Figure 3.2: figure showing deviation of centroids from original position when outliers are present and correct centroid estimation by trimmed K-means algorithm

3.2 Outliers removal using BFR algorithm

Unlike Trimmed K-means, BFR algorithm can be used to detect as well as remove outliers. The data points in Retained set can be considered as outliers. The results of BFR algorithm are shown in figure-3.3(c). The dataset used is artificial with four normally distributed clusters where cluster-1 has more deviation along x-axis than y, cluster-2 has more deviation along y-axis, cluster-3 and cluster-4 have same deviation along both x and y axes, as shown in figure3.3(a). It can be observed from figure3.3(b) that the outliers which were present along the axes of clusters for which deviation was less are removed

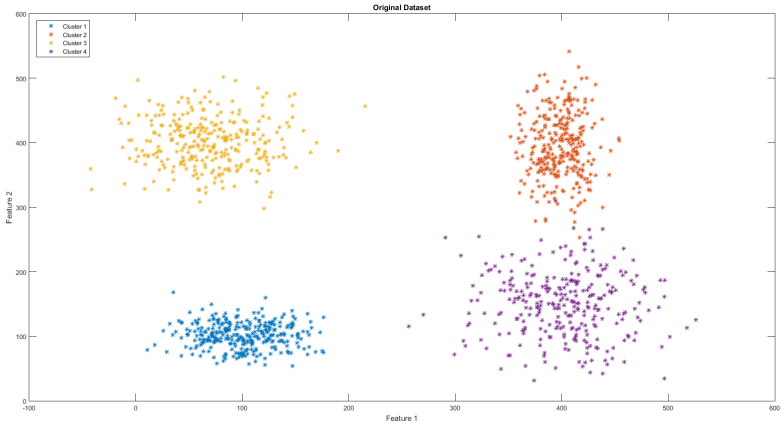
easily whereas the outliers which were present along the boundary for which the deviation was more, cannot be removed completely.

3.3 Outlier removal using density based clustering algorithms

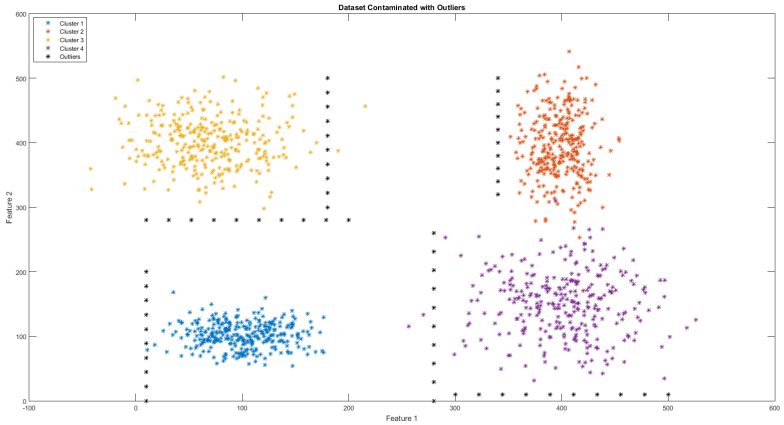
Density-based algorithms treat low density data points as noise or outliers. Using density based clustering algorithms data points which are located away from the cluster , having camparatively low density as campared to those points which are present inside the cluster cloud can be detected as well as removed. DBSCAN algorithm is used to remove the outliers present in the dataset shown in figure 3.4(a) and 3.4(b). Another density based algorithm known as peak density based clustering algorithm is used to detect the outliers from another artificial spiral dataset, shown in figure3.6(a) The results of the algorithm are shown in figure 3.5 and 3.6(c) respectively.

3.4 Important discussions

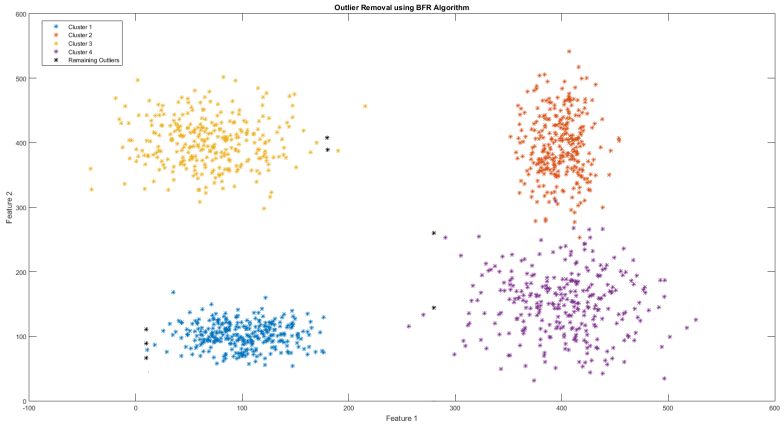
The effect of outliers can be observed from the figure 3.1 on the performance of K-means algorithm. K-means algorithm is not able to detect correct ccentroids and a large deviation can be observed in the position of centroids. This issues can be resolved by using Trimmed K-means algorithm as shown in figure 3.2 where it can be observed that algorithm easily detected original position with almost 99% accuracy in Iris dataset. BFR algorithm was origially made to deal with massive datasets but here it is shown that the algorithm can also be used to detect and remove outliers. In given artificial dataset shown in figure 3.3 outliers are detected and removed with almost 90% accuracy. The density based algorithms also give more than 90% accuracy in detecting and removing outliers.



(a)

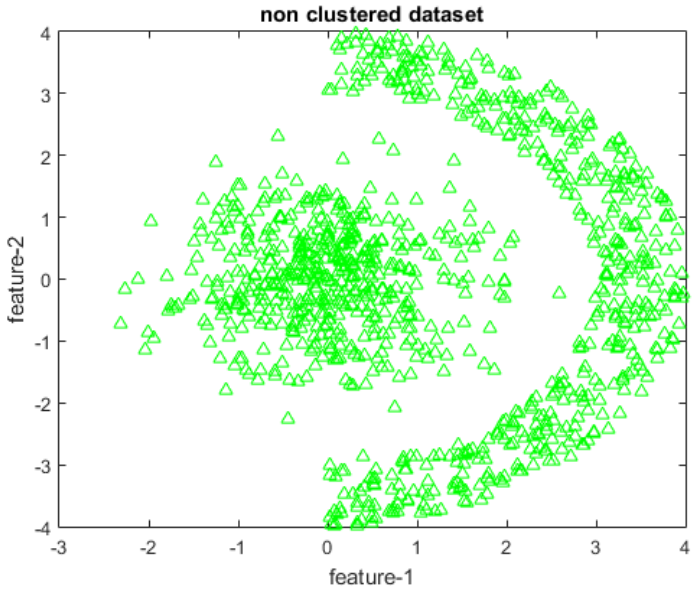


(b)

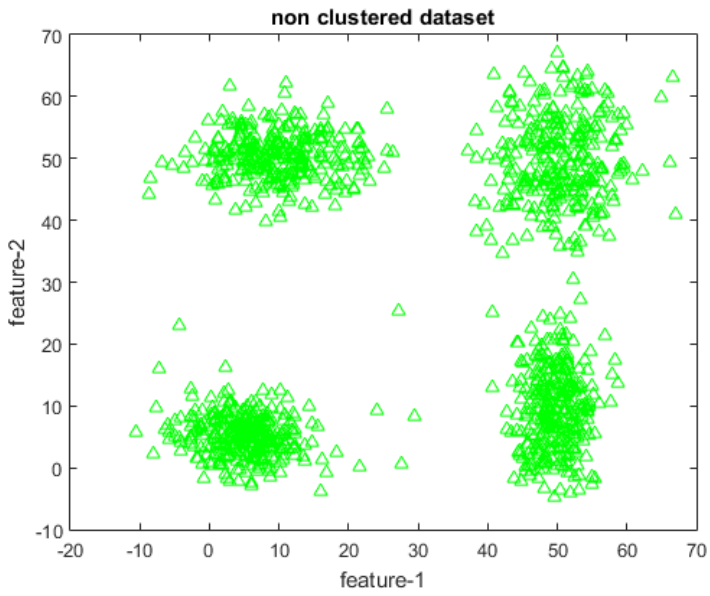


(c)

Figure 3.3: Figure showing: (a) original artificial dataset without outliers (b) dataset with outliers (c) result of BFR algorithm

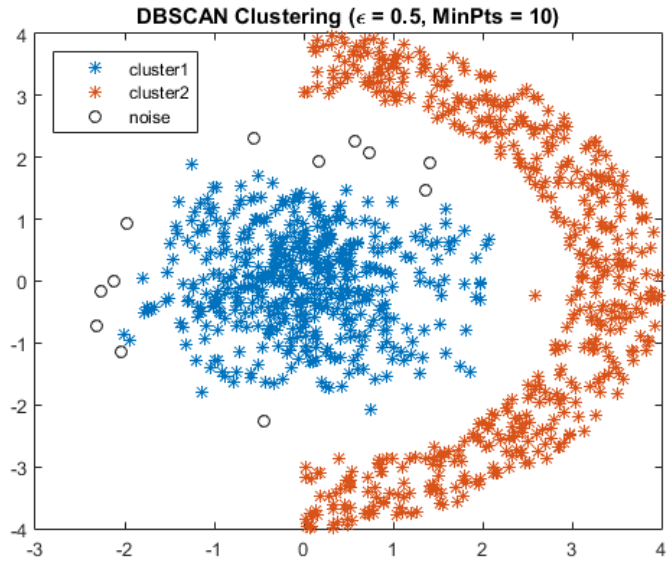


(a)

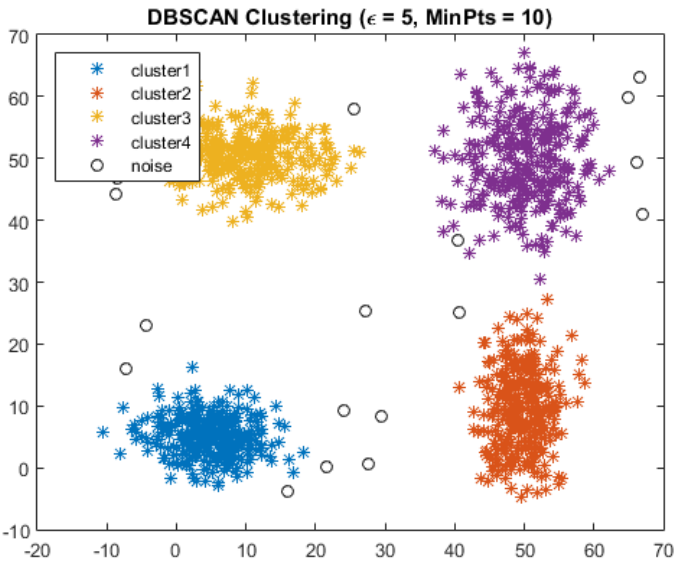


(b)

Figure 3.4: figure showing two differnt non clustered datasets

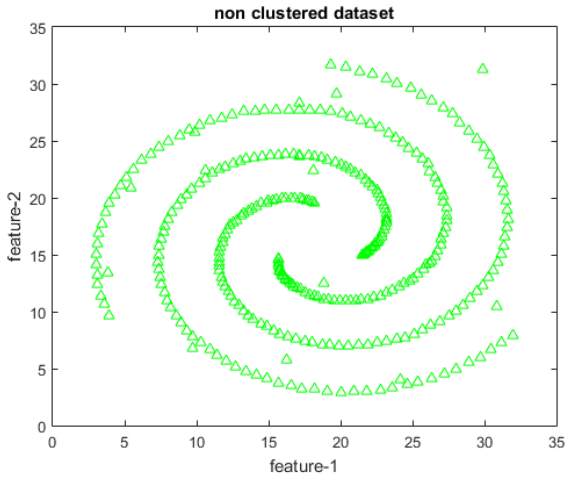


(a)

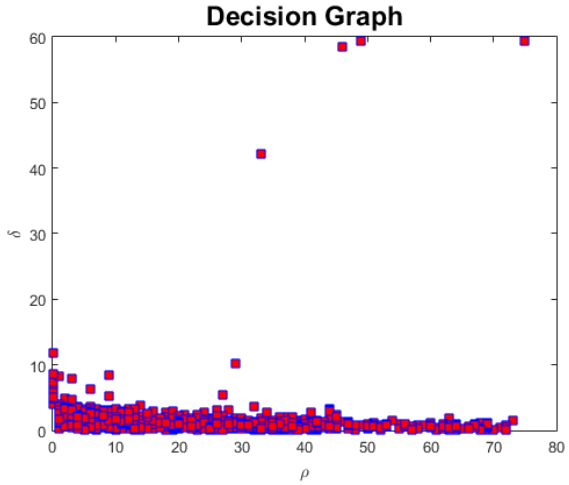


(b)

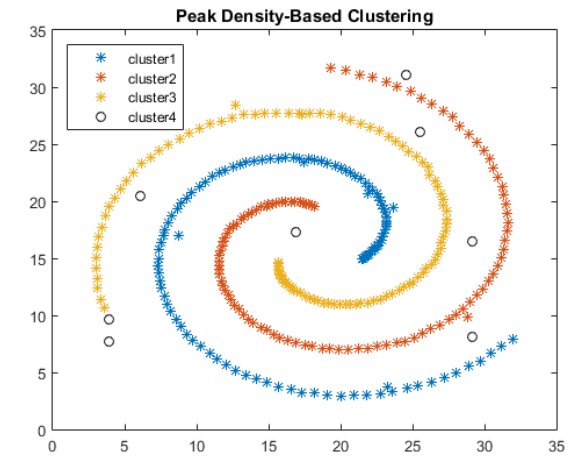
Figure 3.5: figure showing results of dbscan algorithm



(a)



(b)



(c)

Figure 3.6: figure showing (a) unclustered dataset and (b) decision graph for the dataset (c)the result of peak density based clustering

Chapter 4

Orthogonal Grey Wolf Optimization

4.1 Proposed OGWO algorithm

Grey wolf optimization(GWO) is one of the many nature-inspired algorithms that is popular for solving labyrinthine optimization problems. The algorithm uses properties of hunting mechanism and leadership hierarchy of grey wolves. In this chapter a new variant of GWO called the Orthogonal Grey Wolf Optimization(OGWO) in which unlike the previous approach where the position of points are not merely updated by averaging the movement towards three global leaders but is used in combination with the proposed orthogonal methodology to compare and find the best and more robust results. the objective is to obtain the best possible combination of positions from the three global leaders. The simulation analysis on standard benchmark function reveals that the results obtained from the proposed algorithm are more optimal and have lesser standard deviation than the previous approach. In addition to this, the proposed algorithm is also successfully used on cluster analysis and very competent results are obtained when compared to other nature-inspired algorithms like Particle Swarm Optimization(PSO), GWO, Orthogonal PSO(OPSO), Orthogonal Genetic Algorithm with Quantization(OGA)

The final position of each member is updated from equation-2.26 in grey wolf optimization where as in the proposed algorithm it is updated with the help of equation-4.1. where $\oplus$ is orthogonal combination operator. Orthogonal combination can be calculated with the help of orthogonal array. Orthogonal array provides some specific but different

combinations of the features of $\vec{X}1, \vec{X}2$ and $\vec{X}3$

$$X(\vec{t+1}) = best \left\{ \begin{array}{c} \vec{X_1} \oplus \vec{X_2} \oplus \vec{X_3} \\ \\ \\ \frac{\vec{X_1}+\vec{X_2}+\vec{X_3}}{3} \end{array} \right\} \tag{4.1}$$

The benefit of orthogonal combination over averaging can be explained with the help of an example.Suppose that the spherical function is to be optimized and it is considered to be three dimensional for the sake of convenience and let the three-vector movement towards first, second and third global best that is $\vec{X}1, \vec{X}2$ and $\vec{X}3$ be [0 1 2], [1 0 3] and [1 1 2] respectively. Their individual fitness is 5, 10 and 6. when averaged the new position obtained as [[0.67 0.67 2.33] with fitness 6.19. The orthogonal array for this case($N = 3$,$Q = 3$) is obtained as shown in table-1 where it can be seen that one of the nine combinations is [1 2 2] from which the orthogonal combination of vector would be [0 0 1] with fitness 1 and since it is a minimization problem, the result obtained from orthogonal combination is better than the previous one. Thus it is observed that the use of the orthogonal combination gives better results by ensuring the contribution from all the three vectors but still it is not always necessary because although orthogonal array does contain a diverse combinations but not all the possible combinations therefore it is also suggested that average value should also be saved and its fitness should be compared with all the orthogonal combinations before updating the final position of the points. Updating the position orthogonally make the algorithm converge faster. It is

Table 4.1: Table showing orthoganal array for N=3 Q=3 [**?**]

1	1	1
1	2	2
1	3	3
2	1	2
2	2	3
2	3	1
3	1	3
3	2	1
3	3	2

Algorithm 1 Orthogonal Grey Wolf Optimization

Step1. Specify the search space by setting the lower and upper bound.
Step2. Initialize the population in the search space the number of points can vary depending on the problem.
Step3. Initialize alpha, beta and delta positions by calculating the fitness of all the initialized points, the best position is allocated to alpha, second best to the beta and the third best to the delta.
Step4. Generate the orthogonal array for Q=3 and N with the help of Algorithm-1, where Q is the quantization levels and N represents the number of dimensions. Eliminate any row which has all same values.
Step5. Define an array 'z' which can take values from z1 to maximum iteration alternatively after every 'l' where 'l' is more than 2.
Step6. Algorithm proceedes iteratively form 1 to maximum iteration and variable 'i' will keep track of present iteration. If 'i' is equal to a value in 'z' then updating is done by equation-12. In case 'i' is not equal to values in 't' update the position of points by equation 11.
Step7. Update the alpha, beta and delta in case the present position of the point is better then previous position of alpha, beta and delta otherwise retain their previous position.
Step8. Repeat until the count of 'i' reaches to maximum iteration.

also suggested that out of the total number of iterations the orthogonal updating is done either alternatively or after every 'z' iteration. Apart from this it is also necessary to make sure that the new position has a combination from all the movement vectors i.e. $\vec{X}1, \vec{X}2$and $\vec{X}3$ therefore from orthogonal array eliminate any such row which has all the entries same.The complete algorithm is explained in algorithm-2.

4.2 Simulation on benchmark mathematical test function

In this section the proposed algorithm is applied on 20 benchmark functions, first seven (f1-f7) are unimodal benchmark functions, next six(f8-f13) are variable dimensional multimodal benchmark functions remaining functions(f14-f20) are fixed dimensional multimodal benchmark functions. These functins are shown in in table-2 where along with the function expression the dimension, range and absolute minima of all the functions are mentioned. The algorithm was rum on every single function and the results were compared with its original form, Grey Wolf Optimization(GWO) along with Particle Swarm Optimization(PSO). The results obtained are shown in table-3 and it is observed that

the results obtained were much closer to the optimal solution compared to GWO. Also the results are more robust as it can be seen that the standard deviation is much lower compared to obtained from GWO. Comparatively lower standard deviation in results shows that the orthogonal algorithm is more robust.

4.3 Simulation on benchmark clustering dataset

Proposed algorithm was applied on differnt datasets to perform clustering in order to study it's performance. Thirteen datasets were used including real life datasets, gaussian datasets with different standard deviation and highdimensional datasets. Before applying the algorithm on highdimensional datasets their dimensions were reduced using canonical correlation analysis.The detailed information about the datasets is mentioned in the table-4. The fitness function used to solve clustering as an optimization problem is shown in equation-16. The results of the Proposed algorithm is compared with other optimization algorithms like PSO, OPSO, OGA and GWO. The results obtained are mentioned in table-5.

$$f = minimize \sum_{i=1}^{N} \sum_{m=1}^{k} \delta[(r_{i,Q}, c_{m,Q})]^2 \tag{4.2}$$

where, $\delta(r_{i,Q}, c_{m,Q})$ is eucledian distance between cluster centroid $c_{m,Q}$ and $r_{i,Q}$ are the elements of m^{th} cluster. It can be observed that percentage error is low for real time dataset which is concluded from lesser number of overlapping points. Algorithm was also tested on gaussian datasets with decreasing variance,G-8-70 to G-8-40 are those gaussian datasets. G-8-70 dataset has highest variance with a standard deviation of 70 where as dataset G-8-40 has comparatively low variance with a standard deviation of 40, for datasets with higher variance the error percentage is higher due to more overlapping points than the datasets with lesser standard deviation and less overlapping points.

Table 4.2: Results of OGWO algorithm on benchmark functions

Benchmark Function		OGWO		GWO		PSO	
Name	Minima	Ave	Std	Ave	Std	Ave	Std
F1	0	8.633×10^{-31}	1.83×10^{-30}	6.59×10^{-28}	6.34×10^{-05}	0.000136	0.000202
F2	0	9.89×10^{-18}	1.5×10^{-17}	7.18×10^{-17}	0.029014	0.042144	0.045421
F3	0	2.67×10^{-8}	8.26×10^{-8}	3.29×10^{-06}	79.14958	70.12562	22.11924
F4	0	5.3×10^{-16}	1.5×10^{-15}	5.61×10^{-07}	1.315088	1.086481	0.317039
F5	0	1.63×10^{-5}	2.61×10^{-5}	26.81258	69.90499	96.71832	60.11559
F6	0	1.732×10^{-8}	2.01×10^{-8}	0.816579	0.000126	0.000102	8.28×10^{05}
F7	0	5.14×10^{-4}	3.89×10^{-4}	0.002213	0.100286	0.122854	0.044957
F8	2094.9	-1.2×10^{-4}	9×10^{-5}	−6123.1	−4087.44	−4841.29	1152.814
F9	0	0	0	0.310521	47.35612	46.70423	11.62938
F10	0	8.8×10^{-16}	0	1.06×10^{-13}	0.077835	0.276015	0.50901
F11	0	0	0	0.004485	0.006659	0.009215	0.007724
F12	0	1.143×10^{-9}	1.36×10^{-9}	0.053438	0.020734	0.006917	0.026301
F13	0	1.055×10^{-8}	1.36×10^{-9}	0.654464	0.004474	0.006675	0.008907
F14	1	0.9980	2.937×10^{-11}	0.000337	0.000625	3.627168	2.560828
F15	0.00030	3.083×10^{-4}	8.277×10^{-7}	0.000337	0.000625	0.000577	0.000222
F16	−1.0316	−1.0316	2.83×10^{-7}	−1.03163	−1.03163	−1.03163	6.25×10^{-16}
F17	0.398	0.39	2.01×10^{-5}	0.397889	0.397887	0.397887	0
F18	3	3.0028	0.0024	3.000028	3	3	1.33×10^{-15}
F19	−3.86	−3.85	0.0017	−3.86263	−3.86278	−3.86278	2.58×10^{-15}
F20	−3.32	3.2871	0.0541	−3.28654	−3.25056	−3.26634	0.060516

Unimodal benchmark functions

Function	Dim	Range	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j\right)^2$	30	$[-100, 100]$	0
$f_4(x) = \max_i \{\lvert x_i \rvert, 1 \leqslant i \leqslant n\}$	30	$[-100, 100]$	0
$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	0
$f_6(x) = \sum_{i=1}^{n} (\lvert x_i + 0.5 \rvert)^2$	30	$[-100, 100]$	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	0

(a)

Multimodal benchmark function

Function	Dim	Range	f_{min}
$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	$[-500, 500]$	-418.9829×5
$F_9(x) = \sum_{i=1}^{n} \lvert x_i^2 - 10\cos(2\pi x_i) + 10 \rvert$	30	$[-5.12, 5.12]$	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$	30	$[-32, 32]$	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]$	0
$F_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
$F_{13}(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50, 50]$	0
$F_{14}(x) = -\sum_{i=1}^{n} \sin(x_i) \cdot \left(\sin\left(\frac{ix_i^2}{\pi}\right)\right)^{2m}, \; m = 10$	30	$[0, \pi]$	-4.687
$F_{15}(x) = \left[e^{-\sum_{i=1}^{n}(x_i/\beta)^{2m}} - 2e^{-\sum_{i=1}^{n} x_i^2}\right] \cdot \prod_{i=1}^{n} \cos^2 x_i, \; m = 5$	30	$[-20, 20]$	-1
$F_{16}(x) = \{[\sum_{i=1}^{n} \sin^2(x_i)] - \exp(-\sum_{i=1}^{n} x_i^2)\} \cdot \exp[-\sum_{i=1}^{n} \sin^2 \sqrt{\lvert x_i \rvert}]$	30	$[-10, 10]$	-1

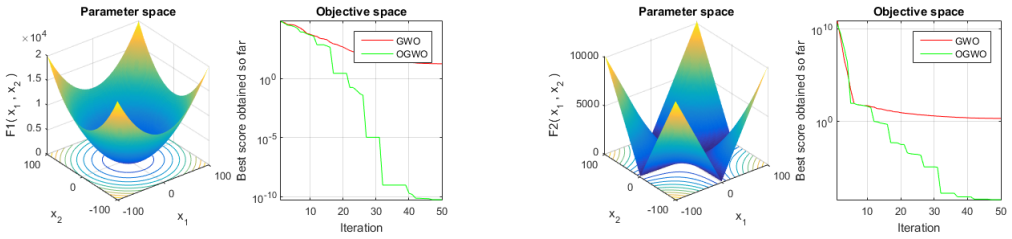
(b)

Fixed dimension multimodal benchmark functions

Function	Dim	Range	f_{min}
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	2	$[-65, 65]$	1
$F_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	$[-5, 5]$	0.00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 5]$	0.398
$F_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2, 2]$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	3	$[1, 3]$	-3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	6	$[0, 1]$	-3.32
$F_{21}(x) = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.5363

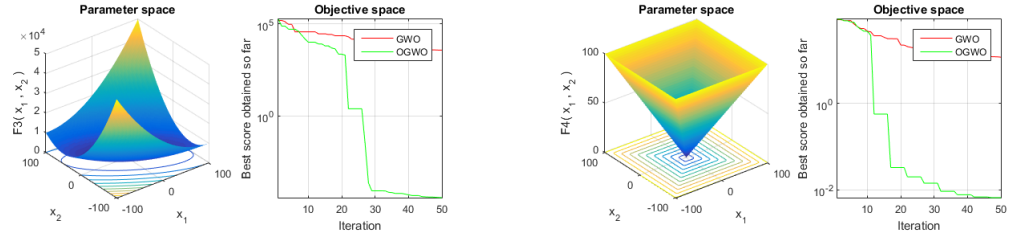
(c)

Figure 4.1: Figure showing benchmark functions used (a) Unimodal (b) Multimodal and (c) Fixed dimensional multimodal benchmark functions[1]



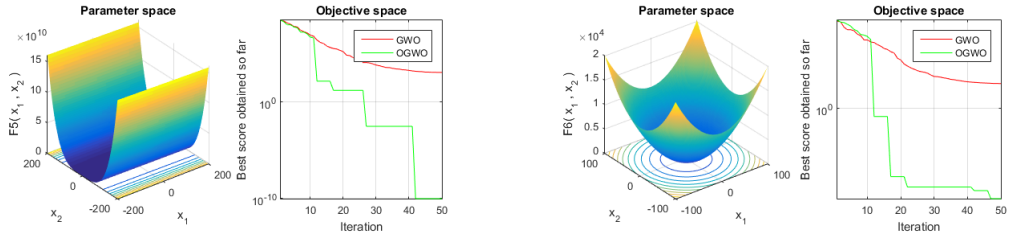
(a) (b)

Figure 4.2: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F1 and (b) F2 unimodal benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms



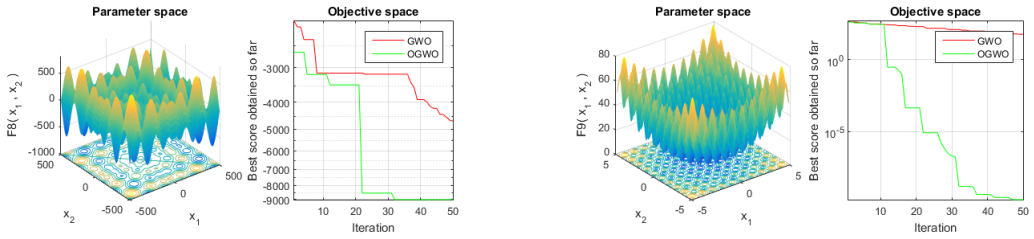
(a) (b)

Figure 4.3: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F3 and (b) F4 unimodal benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms



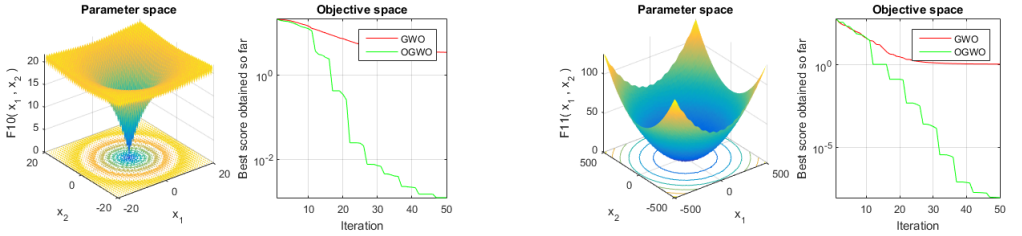
(a) (b)

Figure 4.4: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F5 and (b) F6 unimodal benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms



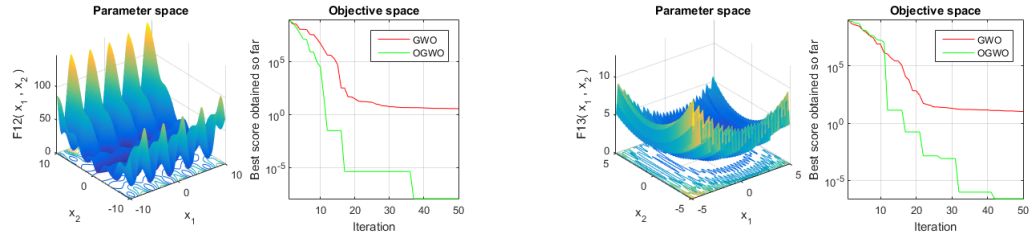
(a) (b)

Figure 4.5: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F8 and (b) F9 multimodal 30-dimensional benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms



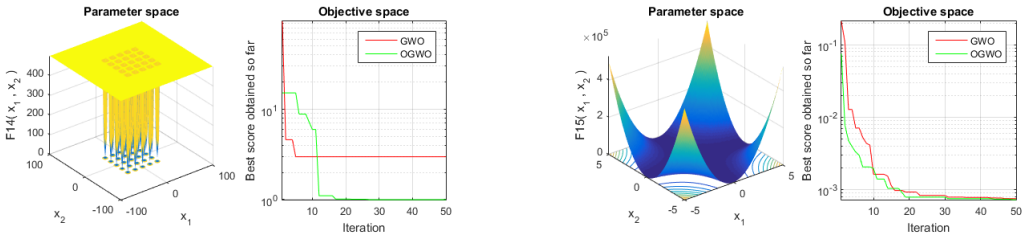
(a) (b)

Figure 4.6: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F10 and (b) F11 multimodal 30-dimensional benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms



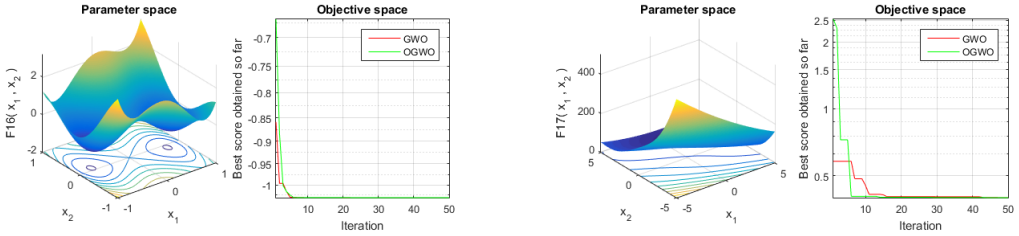
(a) (b)

Figure 4.7: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F12 and (b) F13 multimodal 30-dimensional benchmark functions with 30 dimensions, 50 population size and 50 iterations for both the algorithms



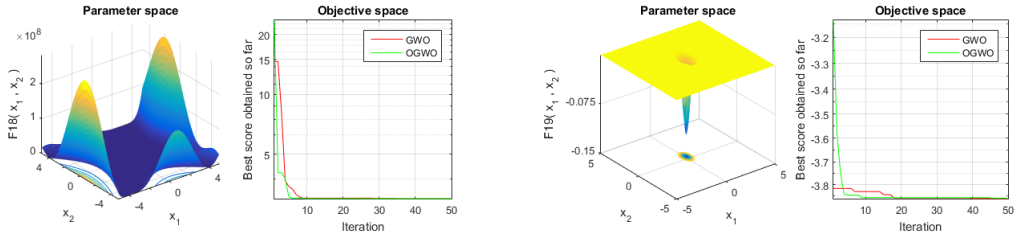
(a) (b)

Figure 4.8: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F14 and (b) F15 fixed dimensional multimodal benchmark functions, with 50 population size and 50 iterations for both the algorithms



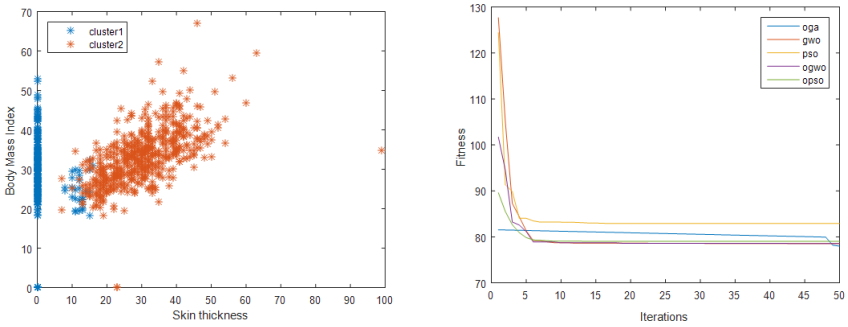
(a) (b)

Figure 4.9: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F16 and (b) F17 fixed dimensional multimodal benchmark functions, with 50 population size and 50 iterations for both the algorithms



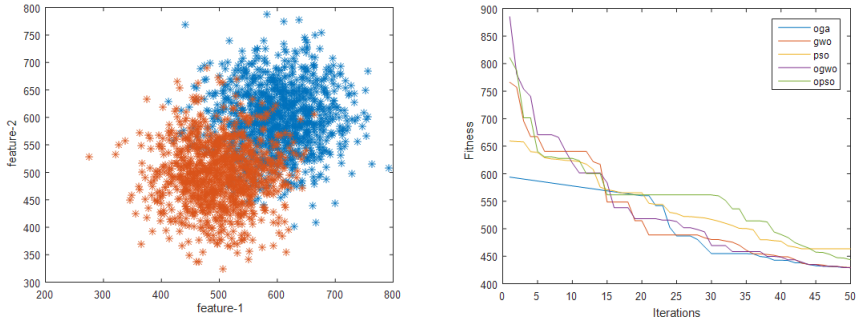
(a) (b)

Figure 4.10: Figure showing the convergence comparison between the proposed algorithm and GWO algorithm on: (a)F18 and (b) F19 fixed dimensional multimodal benchmark functions, with 50 population size and 50 iterations for both the algorithms



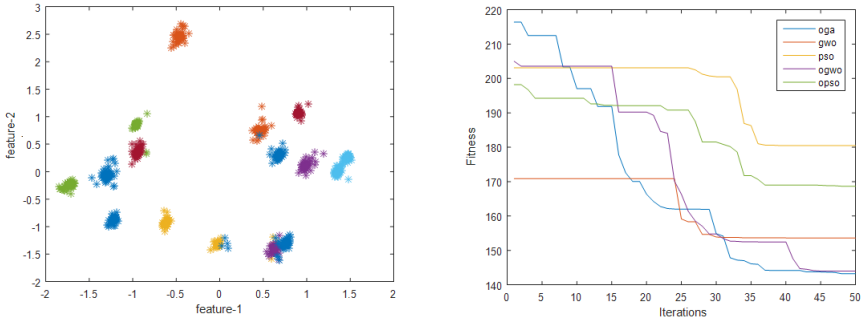
(a) (b)

Figure 4.11: Figure (a) showing clustering results on diabetes dataset using OGWO and (b) showing convergence comparison of OGWO with OGA, PSO, OPSO, and GWO



(a) (b)

Figure 4.12: Figure (a) showing clustering results on gaussian G260 dataset using OGWO and (b) showing convergence comparison of OGWO with OGA, PSO, OPSO, and GWO



(a) (b)

Figure 4.13: Figure (a) showing clustering results on highdimensional highdim-1 dataset using OGWO and (b) showing convergence comparison of OGWO with OGA, PSO, OPSO, and GWO

Table 4.3: table showing details of datasets used

Dataset Name	samples	dimensions	number of clusters
Iris	150	4	3
Wines	178	13	3
Breast cancer	569	31	2
Diabetes	768	8	2
A1	3000	2	20
A2	5250	2	35
G1	2048	5	2
G-8-70	2048	8	2
G-8-60	2048	8	2
G-8-50	2048	8	2
G-8-40	2048	8	2
Highdimensional	1024	32	16
Highdimensiona2	1024	64	16

Table 4.4: Table showing percentage misclassification on different dataset when subjected to differnt clustering algorithms including the proposed algorithm

Dataset Name	pso	opso	oga	gwo	ogwo
Iris	3.3	2	0.67	2	1.33
Wine	6.89	4.68	3.68	4.87	4.87
Breast Cancer	4.23	4.23	1.52	3.97	3.8
Diabetes	5.20	2.69	2.43	2.56	2.56
A-1	13	10.433	10.63	11.67	11.16
A-2	10.05	8.74	8.57	9.75	9.54
G-1	1.36	0.58	0.24	0.34	0.34
G-8-70	4.93	3.369	1.41	1.9	2
G-8-60	1.025	0.6839	0.396	0.43	0.43
G-8-50	0.34	0.23	0.24	0.24	0.24
G-8-40	0	0	0	0	0
Highdimensional	41.01	19.53	12.6	13.8	12.89
Highdimensiona2	31.25	28.32	11.68	14.7	12.9

Chapter 5

Conclusion and future aspects

In this project, ordinary and robust clustering techniques are studied and the robust clustering algorithms are used to classify data points in a dataset correctly in the presence of outliers. In addition to that, a new variant of the Grey Wolf Optimization algorithm(GWO) is proposed which is based on the orthogonal movement of points. The proposed algorithm is used to solve the clustering as an optimization problem by applying the algorithm on thirteen datasets it is also tested on benchmark functions to compare its performance with other algorithms. Four out of thirteen datasets are real-life datasets, seven datasets are gaussian-datasets whereas rest two are high-dimensional datasets. The results obtained from the proposed algorithm are better than its original version in most of the cases, and in some cases, performance is found to be the same. Also, the results obtained are more robust in all the cases because of a very low standard deviation. The computational complexity of the proposed algorithm is more than its previous version.

No work has been done on the initialization of the proposed algorithm, the algorithm follows the random initialization which can be replaced with orthogonal initialization which will make the results of the algorithm more robust. Efforts can be made to reduce the computational complexity of the algorithm as well.

References

[1] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in engineering software*, vol. 69, pp. 46–61, 2014.

[2] Y.-W. Leung and Y. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Transactions on Evolutionary computation*, vol. 5, no. 1, pp. 41–53, 2001.

[3] S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho, "Opso: Orthogonal particle swarm optimization and its application to task assignment problems," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 38, no. 2, pp. 288–298, 2008.

[4] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern recognition letters*, vol. 31, no. 8, pp. 651–666, 2010.

[5] U. P. Shukla and S. J. Nanda, "Parallel social spider clustering algorithm for high dimensional datasets," *Engineering Applications of Artificial Intelligence*, vol. 56, pp. 75–90, 2016.

[6] S. J. Nanda and G. Panda, "A survey on nature inspired metaheuristic algorithms for partitional clustering," *Swarm and Evolutionary computation*, vol. 16, pp. 1–18, 2014.

[7] P. S. Bradley, U. M. Fayyad, C. Reina, *et al.*, "Scaling clustering algorithms to large databases.," in *KDD*, vol. 98, pp. 9–15, 1998.

[8] A. Rajaraman and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press, 2011.

[9] S. J. Nanda and G. Panda, "Automatic clustering algorithm based on multi-objective immunized pso to classify actions of 3d human models," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 5-6, pp. 1429–1441, 2013.

[10] S. Yadav and S. J. Nanda, "League championship algorithm for clustering," in *2015 IEEE Power, Communication and Information Technology Conference (PCITC)*, pp. 321–326, IEEE, 2015.

[11] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, *et al.*, "Constrained k-means clustering with background knowledge," in *Icml*, vol. 1, pp. 577–584, 2001.

[12] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[13] J. A. Cuesta-Albertos, A. Gordaliza, C. Matrán, *et al.*, "Trimmed k-means: An attempt to robustify quantizers," *The Annals of Statistics*, vol. 25, no. 2, pp. 553–576, 1997.

[14] L. A. Garcia-Escudero and A. Gordaliza, "Robustness properties of k means and trimmed k means," *Journal of the American Statistical Association*, vol. 94, no. 447, pp. 956–969, 1999.

[15] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The mahalanobis distance," *Chemometrics and intelligent laboratory systems*, vol. 50, no. 1, pp. 1–18, 2000.

[16] S. Guha, R. Rastogi, and K. Shim, "Cure: an efficient clustering algorithm for large databases," in *ACM Sigmod Record*, vol. 27, pp. 73–84, ACM, 1998.

[17] D. Birant and A. Kut, "St-dbscan: An algorithm for clustering spatial–temporal data," *Data & Knowledge Engineering*, vol. 60, no. 1, pp. 208–221, 2007.

[18] B. Borah and D. Bhattacharyya, "An improved sampling-based dbscan for large spatial databases," in *International Conference on Intelligent Sensing and Information Processing, 2004. Proceedings of*, pp. 92–96, IEEE, 2004.

[19] Y. He, H. Tan, W. Luo, H. Mao, D. Ma, S. Feng, and J. Fan, "Mr-dbscan: an efficient parallel density-based clustering algorithm using mapreduce," in *2011 IEEE 17th International Conference on Parallel and Distributed Systems*, pp. 473–480, IEEE, 2011.

[20] Y. He, H. Tan, W. Luo, S. Feng, and J. Fan, "Mr-dbscan: a scalable mapreduce-based dbscan algorithm for heavily skewed data," *Frontiers of Computer Science*, vol. 8, no. 1, pp. 83–99, 2014.

[21] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[22] M. Du, S. Ding, and H. Jia, "Study on density peaks clustering based on k-nearest neighbors and principal component analysis," *Knowledge-Based Systems*, vol. 99, pp. 135–145, 2016.

[23] J. Xie, H. Gao, W. Xie, X. Liu, and P. W. Grant, "Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors," *Information Sciences*, vol. 354, pp. 19–40, 2016.

[24] J. Xu, G. Wang, and W. Deng, "Denpehc: Density peak based efficient hierarchical clustering," *Information Sciences*, vol. 373, pp. 200–218, 2016.

[25] W.-f. Gao, S.-y. Liu, and L.-l. Huang, "A novel artificial bee colony algorithm based on modified search equation and orthogonal learning," *IEEE Transactions on Cybernetics*, vol. 43, no. 3, pp. 1011–1024, 2013.

[26] X.-M. Hu, J. Zhang, and Y. Li, "Orthogonal methods based ant colony search for solving continuous optimization problems," *Journal of computer science and technology*, vol. 23, no. 1, pp. 2–18, 2008.

[27] S. Chen, Y. Wu, and B. Luk, "Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1239–1243, 1999.

[28] K.-f. Shi, J.-w. Dong, J.-p. Li, S.-n. Qu, and B. Yang, "Orthogonal genetic algorithm," *Acta electronica sinica*, vol. 30, no. 10, pp. 1501–1504, 2002.

[29] R.-E. Precup, R.-C. David, and E. M. Petriu, "Grey wolf optimizer algorithm-based tuning of fuzzy control systems with reduced parametric sensitivity," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 527–534, 2016.

[30] S. Mirjalili, S. Saremi, S. M. Mirjalili, and L. d. S. Coelho, "Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization," *Expert Systems with Applications*, vol. 47, pp. 106–119, 2016.

[31] M. T. M. Silva and V. H. Nascimento, "Improving the tracking capability of adaptive filters via convex combination," *IEEE Transactions on Signal Processing*, vol. 56, pp. 3137–3149, July 2008.

List of Publications

1. Mohit Sharma, Satyasai Jagannath Nanda, Menber IEEE "Orthogonal Grey Wolf Optimization", ICIT 2019
   Status: Communicated