

**MODIFICATIONS FOR FUZZY C MEANS CLUSTERING TO ENFORCE ELLIPSE
AND PARABOLA SHAPES ON CLUSTERS**

A THESIS

**SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR
THE DEGREE OF**

MASTER OF ENGINEERING IN BIO-MEDICAL ENGINEERING

JADAVPUR UNIVERSITY

JUNE 2022

BY

ARKABRATA ACHARJEE

2020-2022

CLASS ROLL NUMBER- 002030201001

REGISTRATION NUMBER- 154626 OF 2020-2021

EXAM ROLL NUMBER- M4BMD22001

UNDER THE GUIDANCE OF

DR. AMIT KONAR

ELECTRONICS AND TELECOMMUNICATION ENGINEERING

JADAVPUR UNIVERSITY, KOLKATA- 700032

AND

DR. PIYALI BASAK

SCHOOL OF BIO-SCIENCE AND ENGINEERING

JADAVPUR UNIVERSITY, KOLKATA- 700032

M.E. (Biomedical Engineering) course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata-700032

CERTIFICATE OF RECOMMENDATION

We hereby recommend that the thesis entitled "Modifications for Fuzzy C Means Clustering To Enforce Ellipse And Parabola Shapes On Clusters" carried out under our supervision by Arkabrata Acharjee may be accepted in partial fulfillment of the requirement for awarding the Degree of Master of Engineering in Biomedical Engineering of Jadavpur University. The project, in our opinion, is worthy for its acceptance.

Piyali Basak

DR. PIYALI BASAK
ASSOCIATE PROFESSOR
SCHOOL OF BIOSCIENCE & ENGG.
JADAVPUR UNIVERSITY
KOLKATA - 700032
Dr. Piyali Basak
(Thesis Advisor)
Assistant Professor
School of Bioscience and Engineering
Jadavpur University
Kolkata-700032

Akonar

PROF. AMIT KONAR
CO-ORDINATOR
INTELLIGENT AUTOMATION
& ROBOTICS
E.T.C.E. DEPTT., J.U., KOL-32
Dr. Amit Konar
(Thesis Advisor)
Professor
Electronics And Telecommunication Engineering
Jadavpur University
Kolkata-700032

Piyali Basak

23rd June 2022
DIRECTOR
School of Bioscience and Engineering
Jadavpur University
Kolkata-700032
Dr. Piyali Basak
Director
School of Bioscience & Engineering
Jadavpur University
Kolkata-700 032

Subeay Chakraborty

24/06/2022
Dean
Faculty of Interdisciplinary Studies, Law and Management
Jadavpur University
Kolkata - 700032
Dean
Faculty of Interdisciplinary Studies, Law & Management
Jadavpur University, Kolkata-700032

**M.E. (Biomedical Engineering) course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata-700032**

CERTIFICATE OF APPROVAL

The forgoing thesis is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the thesis only for the purpose for which it is submitted.

**Dr. Piyali Basak
(Thesis Advisor)
Assistant Professor
School of Bioscience and Engineering
Jadavpur University
Kolkata-700032**

**Dr. Amit Konar
(Thesis Advisor)
Professor
Electronics And Telecommunication Engineering
Jadavpur University
Kolkata-700032**

Signature of Examiner

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains literature survey and original research work by the undersigned candidate, as part of his **Master of Engineering in Biomedical Engineering** studies during academic session 2020-2022.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

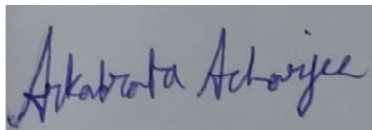
I also declare that, as required by this rules and conduct, I have fully cited and referred all material and results that are not original to this work.

NAME: ARKABRATA ACHARJEE

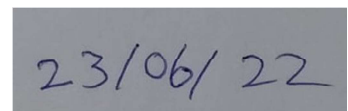
EXAM ROLL NUMBER- M4BMD22001

CLASS ROLL NUMBER- 002030201001

THESIS TITLE: *Modifications for Fuzzy C Means Clustering To Enforce Ellipse And Parabola Shapes On Clusters*



SIGNATURE



DATE

ACKNOWLEDGEMENT

I would like to thank Dr. Amit Konar, Electronics and Telecommunication Engineering, Jadavpur University, and Dr. Piyali Basak, School Of Bio-Science And Engineering, Jadavpur University for giving me the guidance required as and when for preparing this academic work. I would also like to thank all the Authors, Scientists whose works were cited for the purpose of this Thesis. I would also like to thank my Mother Mrs. Mita Acharjee, who constantly supported and encouraged me in my difficult days.

I would like to dedicate this work to my Mother Mrs. Mita Acharjee and in the memory of my Father Late Apurba Kumar Acharjee, it is only because of them and their unconditional sacrifices and blessings I am what I am today.

Arkabrata Acharjee

Arkabrata Acharjee

27/06/22

ABSTRACT

The work presented in this Thesis is mainly focus in the area of clustering, to be more specific in the area of Fuzzy C Means Clustering. The problem is to change the search geometry of the above mentioned algorithm. The shapes which the modified algorithms are required to enforced are Ellipse and Parabola. The task related to shapes were done in [10] and [14]. We try to develop a simpler approach.

The cost function was altered and update equations were derived. The algorithm was also modified and tested on artificial datasets. One of the algorithms is also implemented in image processing problem.

CONTENTS

Chapter No.	Title	Pg. No
1	Artificial intelligence : a brief note	6
2	Machine Learning	8
3	Optimization	11
4	Clustering	14
5	Fuzzy Set	16
6	Fuzzy C Means Clustering	18
7	Norm: Definition, Types, Relation With Fcm And Problem	21
8	Lagrange Multiplier Technique	24
9	Conics	26
10	Ellipse	28
11	Modification Of Fcm Clustering To Enforce Ellipse	30
12	Parabola	61
13	Modification Of Fcm Clustering To Enforce Parabola	63
14	Application	106
15	Performance	122
16	Conclusion	131
17	References	

ILLUSTRATIONS

Fig.No.	Title	Pg.No.
2.1	Machine Learning In a Summary	9
9.1	A Cone from which conic sections can be created	26
9.2	Graph showing a point on the locus of a conic section	27
10.1	Ellipse	28
11.6.1	Graph Showing output of the modified Algorithm	47
11.6.2	Graph showing the elliptical cluster 1	48
11.6.3	Graph Showing elliptical cluster 6	49
11.6.4	Graph showing elliptical cluster 10	50
11.12.1	Graph showing artificial data containing ellipses	58
11.12.2	Graph showing output of the algorithm after the final run	59
12.1	Parabola With directrix as Y axis	61
12.2	Parabola With directrix as $Ax+By+C=0$	62
13.6.1	Graph showing random data points selected using random number generator	80
13.6.2	Graph showing output of the modified algorithm	81
13.6.3	Graph showing one of the clusters	82
13.12.1	Graph showing the artificial data used	89
13.12.2	Graph showing output at the end of 1 st run	90
13.12.3	Graph showing output at the end of 2 nd run	91
13.12.4	Graph showing output at the end of 3 rd run	92
13.12.5	Graph showing output at the end of 4 th run	93
13.12.6	Graph showing output at the end of 5 th run	94
13.12.7	Graph showing output at the end of 6 th run	95
13.12.8	Graph showing output at the end of 7 th run	96
13.12.9	Graph showing output at the end of 8 th run	97
13.12.10	Graph showing output at the end of 9 th run	98
13.12.11	Graph showing output at the end of 10 th run	99
13.12.12	Graph showing output at the end of 11 th run	100
13.12.13	1 st cluster at the end of last run	101
13.12.14	2 nd cluster at the end of last run	102
13.12.15	3 rd cluster at the end of last run	103
13.12.16	The above graph shows membership function at the end of 11 th run	104
14.1	Image used from the mentioned[11] paper	106
14.2	Image portion used from the mentioned [11] paper	106
14.3	Image portion converted to gray scale	107
14.4	Gray Scale image sharpened	107
14.5	Edge detected using Canny	108

14.1.1	Graph Showing Pixel positions having value 1	109
14.1.2	1 st Run Output	110
14.1.3	2 nd Run Output	111
14.1.4	3 rd Run Output	112
14.1.5	4 th Run Output	113
14.1.6	5 th Run Output	114
14.1.7	6 th Run Output	115
14.1.8	7 th Run Output	116
14.1.9	8 th Run Output	117
14.1.10	9 th Run Output	118
14.1.11	10 th Run Output	119
14.2.1	Summary of steps followed in the application	120
15.1	Table showing performance of Hierarchical Fuzzy C Means Ellipse Clustering for a certain instance	122
15.2	Graph showing No of iterations vs Run number	123
15.3	Graph Showing CPU Time consumed Vs run number	124
15.4	Graph showing CPU time consumed in seconds vs Number of Iterations	125
15.5	Table showing performance of Fuzzy C Means Ellipse Clustering of variable size and FCM	126
15.6	Table showing performance of Hierarchical Fuzzy C Means Parabola Clustering for a certain instance	127
15.7	Graph showing No of iterations vs run number	128
15.8	Graph showing CPU time consumed in seconds vs run number	129
15.9	Graph showing CPU time consumed in seconds vs number of iterations	130

CHAPTER 1

ARTIFICIAL INTELLIGENCE : A BRIEF NOTE

Artificial Intelligence [1] is a buzz word in present day science and technology. Almost every other day there are new developments in this field. It is so fast evolving that if we don't keep track on the developments our knowledge will become obsolete. Each and every day new concepts, improvements of old concepts are done. This fast moving pace is the result of continuous strive of humans to develop more intelligent and fast machines which will outperform existing ones. Such is the nature of Artificial Intelligence [1].

Let us understand what Artificial Intelligence [1] is. Artificial Intelligence in its own right is a discipline formed from other well established disciplines. The term Artificial Intelligence was given by John McCarthy. It is an interdisciplinary subject in nature. Artificial Intelligence has no clear cut definition, it has many interpretations but the most widely accepted one is:

The simulation of human intelligence on a machine so as to make it identify and use the right piece of knowledge at a step of problem solving. [1]

To understand this definition we need to define the terms Intelligence and knowledge. Intelligence is the ability to use the right knowledge at the right time [1] and knowledge are rules which are required to solve a problem [1], knowledge can be gained from various sources and can be static as well as dynamically evolving. Thus in simple terms Artificial Intelligence simulates human mind on machine and is involved with mathematical models which makes a machine act in a rational way [1].

Artificial Intelligence has many components so as to make it mimic the way in which humans think. Intelligent system is a rational machine [1] i.e. it can reason its actions, so it is a good planner [1]. Also such a system needs to acquire knowledge which is dependent on perception [1] i.e. acquiring information from the environment via senses also a system should have learning ability [1] for understanding the perceived information and developing knowledge [1].

Thus we can see that artificial intelligent system approximately represents our own understanding of our thinking ability which is based on cognitive science [1]. The subject will become more and more enriched as we will understand how our mind works which is still a mystery hidden behind a thick fog even after so much advancement in science and technology. Even in medical science we have estimated that we still need a lot to understand about human

body even after thousands of years of development. We still have a long way to go, leagues of new discoveries are still there to make, understand and present.

In Artificial Intelligence [1] we solve problems. A problem is a task in hand and we need to find its solution which is valuable for us [1]. A state can be defined as a status of the problem after applying an operator on a previous state [1]. An operator [1] is a rule which is specific to a given problem. Thus we can say solution is a collection of states [1]. We apply operator on a state and continue this until we get the desired state [1]. Also at every state we need to apply the most applicable operator [1]. This approach of problem solving is called state space approach [1]. We generally use Artificial Intelligence [1] in problems for which direct algorithms are not available [1].

There are many state space searching algorithms like

- a) Generate and test [1]
- b) Hill climbing [1]
- c) Heuristic search [1]
- d) Means and End analysis. [1]

Artificial Intelligence is based on various disciplines which are Mathematics, Philosophy and cognitive science, Psychology and computer science. It has been applied in many fields like Robotics [1], Navigation [1], Medicine, Theorem Solving [1], Games, Natural language processing [1], Computer vision [1], Astronomy etc.

CHAPTER 2

MACHINE LEARNING

Artificial Intelligence [1] is not a single domain but comprises of multiple domains for instance

- a) Natural Language Processing [1]
- b) Cognition
- c) Deep learning
- d) Computer Vision and many more. [1]

One such domain is that of Machine Learning [1] [2][3].

Machine Learning [1] [2] [3] is the technology which most technology students want to learn. The term Machine Learning was introduced by Arthur Samuel. According to Arthur Samuel Machine Learning is

The field of study that gives computer the ability to learn without being explicitly programmed.

Thus Machine Learning is a domain of study where we study and develop various learning algorithms which can automatically improve from its learning experience and data using which the experience is developed. The data is task oriented. [2]

Tom Mitchell has defined well posed learning problem. According to him:

A computer program is said to learn from experience E with respect to some task T and performance parameter P if its performance on T as measured by P improves with experience E .

Intelligence [1] has many domains and Machine Learning is said to represent behavioral intelligence [1].

Machine learning is mostly mathematics. One needs to understand linear algebra, vector spaces [6], coordinate geometry, statistics[3], partial differentiation, optimization[7][8] and many more. These are the crux of machine learning [1][2][3].

In machine learning we develop a model [3] also known as hypotheses using data provided and a machine learning algorithm. We basically train the model [3] and after training we use the model to give output which is a prediction on the basis of some input. The entire process is summarized in the diagram.

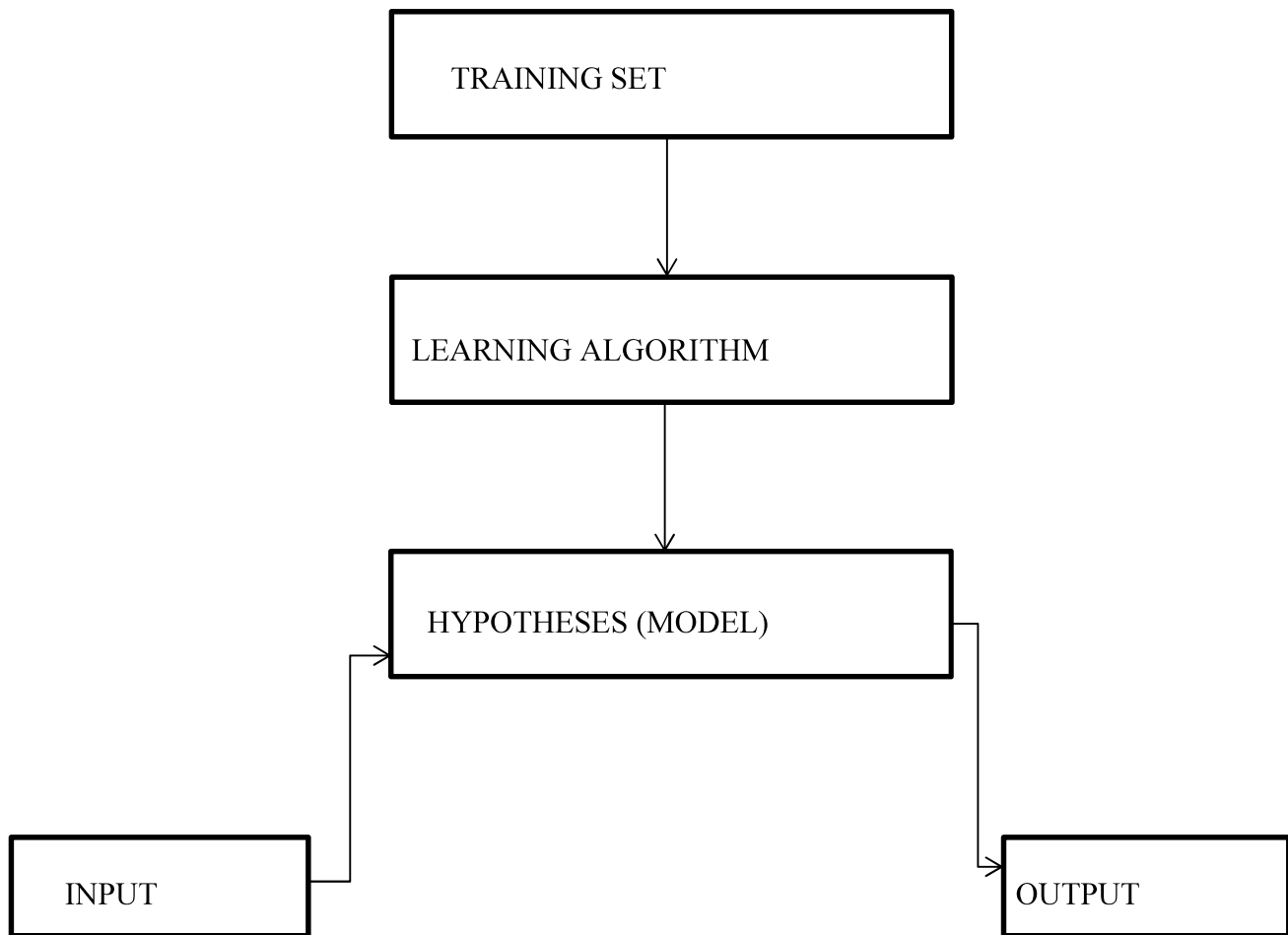


Fig.2.1 Machine Learning In a Summary

The output generated is a prediction and not the exact value. We use the learning algorithm to develop the parameters of the model [3].

Machine Learning can be classified into following broad categories [3]:

a) Supervised learning [3]

b) Unsupervised learning [3]

c) Semi-supervised learning [3]

d) Reinforcement learning [3]

Let us understand these categories in brief.

2.1 Supervised Learning [3]

In Supervised learning [3] the dataset provided has labels which are the output of the model for the corresponding input instance. It is the type of learning where we specifically tell the system(learner) the solution for a problem instance i.e. we supervise. It is like teaching a child about a ball by showing it a ball and telling about it.

Examples of supervised learning [3] are

- a) Regression [3]
- b) Classification [3]

2.2 Unsupervised Learning [3]

In unsupervised learning [3] the dataset provided does not have labels. It is like learning without any guide and let the learner to figure out on its own. For instance, giving a person a hammer without telling him or her about it and its uses. The learner will on its own figure out what the object in hand is what are its uses and many more information.

Some of the unsupervised learning [3] tasks are as follows:

- a) Clustering [3]
- b) Dimensionality reduction [3]
- c) Association rule learning [3]
- d) Anomaly detection [3]

2.3 Semi-supervised learning [3]

This type of learning is used when dataset is partially labeled. It uses mixture of supervised [3] and unsupervised learning algorithms[3].

2.4 Reinforcement learning [3]

This is a reward based learning. If the learner performs well it is rewarded else it is punished.

CHAPTER 3

OPTIMIZATION

Mathematics has mainly two parts namely Pure Mathematics and Applied Mathematics. Pure mathematics is mathematics without any concern for real life application but applied mathematics is quite the opposite. Applied mathematics is the branch of mathematics which is applied to find solutions to real life problems. It is used in physics, chemistry, biology, economics and many more disciplines. In physics, one of its uses is in fluid dynamics. In biology we can use it to model biological systems and find solutions for the system.

Applied Mathematics has many faucets and one such faucet is Optimization Theory [7][8]. Optimization Theory [7][8] is also a part of computational mathematics and operations research[13]. Optimization Theory [7][8] is used in management, science, engineering and defense [8].

Optimization Theory is mainly centered around the task of finding optimal solution for a given problem [7] [8], and so provides us with algorithms which help us to achieve the mentioned goal.

Any Optimization problem can be formulated as stated below

$$\begin{aligned} \min f(x) & \quad \text{eqn(i)} \\ \text{s.t. } x & \in X, \quad [7] \end{aligned}$$

Where $x \in \mathbb{R}^n$ is a decision variable, $f(x)$ is the objective function, $X \subset \mathbb{R}^n$ is the constraint set or feasible region [7]. If $X = \mathbb{R}^n$ then the problem becomes unconstrained optimization problem [7]:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{eqn(ii)} \quad [7]$$

Constrained optimization problem [7] can be written as follows:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} f(x) & \quad \text{eqn(iii)} \\ \text{s.t. } c_i(x) &= 0, i \in E, \\ c_i(x) &\geq 0, i \in I \quad [7] \end{aligned}$$

Where E and I are the index set for equality constraints and inequality constraints [7].

In a summary we can say that in optimization we are given a function and a set of constraints which can be equality and inequality conditions [7] [8]. The function represents the system and the constraints provide the space in \mathbb{R}^n in which solutions to the problem lie [7] [8]. Using a suitable algorithm we find the optimal solution for the problem. We can either maximize a given function or minimize it. In short we need to find value of parameters for which the function is either maximized or minimized depending on our need.

Machine learning [1] [2] [3] is heavily dependent on Optimization Theory [7] [8]. Most of the task in Machine learning is an Optimization Task. In machine learning [1] [2] [3] we are basically trying to find parameters of a cost function which minimizes the value of the cost function, sometimes constraints are also provided. The cost function summarizes the task given in hand and is responsible for making accurate predictions made by the given model responsible for the task. Let us see some examples:

- a) Linear Regression [3]- In linear regression the task is to predict output which is continuous in nature for instance price based on a linear model. Using the linear model and training instances we develop a cost function which is mean squared error, using the cost function and applying the gradient descent [3] algorithm which is the optimizer algorithm based on gradients we find parameters for the model. In gradient descent [3] we basically compute gradient of the cost function with respect to the parameter multiply with it learning rate and try to move towards the parameters for which the cost function is in its global minima. Using the model along with the optimized parameters we make predictions. It is a supervised learning task.
- b) Linear Classification [3]- In Classification, which is a supervised learning task, we aim at assigning a category to the a given data tuple. The output is finite and discrete signifying the fact that there are fixed number of classes. We take a linear model and apply it to a sigmoid function which yields value between 0 and 1 if the yield is large then it is classified else not. Gradient descent is again used to find the parameters of the linear model which was applied in the sigmoid function. Sigmoid function is used because its derivative exists.
- c) Clustering [3]- It is a unsupervised learning task where we aim to group data into clusters depending on some similarity based on a criteria. For instance in fuzzy c means clustering [9] we have a cost function which is to be minimized. We simply use the minima condition for a function to perform 1st order partial derivatives from where we find update equations for the unknowns. Using the update equations iteratively we find the unknowns for which the cost function is minimized.

Thus we see that optimization and machine learning go hand in hand. There are also other tasks in machine learning where optimization is required like in neural networks we use gradient descent and back propagation. Simulated annealing [1] [8], Hill climbing[1], Genetic algorithms [1] [8], ant colony optimization [8], particle swarm optimization[8] in Artificial Intelligence[1] are basically optimization algorithms.

CHAPTER 4

CLUSTERING

Clustering [3] [17] is an unsupervised learning task [3] which is used very frequently. As already stated unsupervised learning [3] doesn't require supervision and the dataset provided doesn't have labels. In clustering we ask the system to find groups in the dataset provided. It groups data points which are similar in some nature.

Clustering [3] [17] is a basic task in machine learning [1] [2] [3] and there are many algorithms and concepts for doing it. Even the clusters can be either hard or soft. To name a few well known algorithms the below given list will do fine:

- a) K means Clustering [15]
- b) Fuzzy C Means Clustering [9]
- c) Hierarchical Clustering [16]
- d) Density based clustering [17]
- e) Partitioning Clustering [19]
- f) Spectral Clustering [18]

Almost every now and then clustering algorithms are developed. Some get accepted and stand the test of time.

Some clustering approach requires us to specify the number of clusters at the beginning of the execution itself as one of the input parameters. For instance K means clustering [15] and Fuzzy C Means Clustering [9]. These techniques are somewhat rigid and have the drawback that number of clusters entered should be close to the real number of clusters in the dataset hence some sort of knowledge about the dataset provided is required, in short we need to study it and we can't stay blindfolded, that is a dreamed luxury of this technique. If number of clusters is greater than the actual number in the dataset then clusters will become divided. If the number is less than the desired number then clusters will combine to form bigger clusters. Both cases are undesirable.

While other clustering algorithms don't require pre-specified number of clusters, for instance hierarchical clustering[16] where we consider each data point to be a potential cluster centroid and find other data points which are within its predefined space. Here actual number of clusters are deduced while executing the algorithm same, DBSCAN also has such property [23]. We can stay blindfolded about this parameter here as it is taken care by the algorithm itself. We just feed the dataset to the system. This is a useful feature of this approach.

Again clusters can be either hard or soft. In hard clusters a data point can belong to only one cluster but in soft clusters a data point may belong to more than one cluster at the same time.

Clustering is used in lot of applications be it for studying trends in the market in corporate world or in medical imaging. For instance if we have a dataset for sales of a product which contains features of age and gender then we can use clustering to find clusters in the dataset and use the results to find deduct some facts like finding the largest cluster then we can make inferences that which group of people use the product most and from the smallest cluster we can say that which group of people uses the product least and accordingly set the marketing strategy to gain more profit.

In medical science if we have a dataset for a particular diseases and we apply clustering on it we may deduce which group of people is more susceptible to the disease and which group is least and then take proper actions to contain which may lead to its total eradication. Uses of Clustering can be seen in [10] [14].

We see that clustering is the basic step behind many novel ideas and is used to study about data and develop inferences from them which may be useful. These inferences help us to develop solutions to solve problem in our hands.

Let us see a real life application of clustering in computational biology and bio-informatics. In gene microarray clustering we cluster about 80-120 dimension data points where each dimension is some parameter of a genes whose value is determined from studies conducted in wet lab. After clustering we find the larger cluster which might be responsible for some health condition. Using gene regulatory network we can find the main genes in the cluster responsible for the condition. Again clustering [3] [17] can be used to find cells from images by the help of some preprocessing and then some clustering algorithm, this application is shown in this work.

CHAPTER 5

FUZZY SET

We know from mathematics that a set is a collection of element/item, the elements can be anything from numbers to objects [6]. For instance the following set is a set of even numbers:

$$S = \{x : x \bmod 2 = 0\}$$

In crisp sets [6], elements are not repeated i.e. if an element exists in a given set then it is unique. But in multi-set [6] this limitation is not allowed. In multi-set, if an element belongs to a given set then it may or may not be unique for the given set under study. This is quite resonant with the real world for instance let us consider a set of students in a class of a school, now suppose there is some student named 'A' in set theory [6] 'A' named student is unique i.e. no other student can't have the same name but naturally this is not the case. In real world we do come across people having same names, they can be either two or more than two or even one we can't say it for sure whether a name is unique or not. In such cases the concept of multi-set do come handy which removes this problem. So if we consider a multi-set then both the students having the name 'A' can be considered in the set. Sets and Multi-sets have different approaches or ways of thinking about the computation of common set operations of Union [6], Intersection [6] and negation [6]. Also some operations are unique for each of the categories for instance sum of multi-sets [6] and Cartesian product [6] of normal sets [6].

Now coming back to the basic property of a set. If an element belongs to a given set then it is completely associated with the given set, if the element is not a member of the given set then it is not at all associated with the given set. The philosophy is hardline, either it belongs to the set or doesn't belongs to the set. If we develop a membership function for the given set which accepts an element then it will output either 0 or 1. 0 means not a member 1 means a member. It can't produce values in between. They are crisp in nature.

$$f_A(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \notin A, \end{cases} \quad \text{eqn(i)}$$

This hardline way of thinking is not present in humans for some cases where there are intermediate values in between 0 and 1 which means a mixture of yes and no instead of either yes or no way of thinking. For instance if we ask someone whether today is hot or not, the person might response in different ways instead of responding with either a simple yes or no answer. The person might say that it is not that hot or it is warm or it is boiling hot. This type of conversations do happen and it represents that we try to introduce some amount of degree of associativity which is in between 0 and 1.

The hardline way of thinking has been utilized in many machine learning unsupervised clustering algorithm in which a data point can either belong to a given cluster or not. Few such algorithms are as follows:

- a) K means clustering [15]
- b) Basically any algorithm which doesn't have the provision for data points to have membership value other than 0 and 1.

Now let us understand the concept of Fuzzy sets [5] [6]. Fuzzy set [5] [6] was developed by Lotfi.A Zadeh. Fuzzy set is unlike normal set. We have already seen that in a crisp set an element can belong to the set completely or not at all. This is where the concept of fuzzy set comes with a spark. In a fuzzy set each element has a membership function value within the interval [0,1]. This analogy is quite similar to some conclusions developed by humans as we have seen in the question of hotness.

Mathematically we define fuzzy set as follows

Let S be a fuzzy set in U and x be an element. We can say that the condition of sets i.e. $x \in S$ or $x \notin S$ need not hold.

Let μ_S denote the fuzzy membership function for set S then

$$\mu_S(x) \in [0,1] \quad \text{eqn(ii)} \quad [6]$$

In crisp set the membership value is taken from two element set i.e. $\{0,1\}$ but for fuzzy sets [5] [6] it is a range that is given by $[0,1]$.

The concept of fuzzy sets is frequently used in computer science particularly in Artificial Intelligence [1]. Fuzzy set is used in the concept of reasoning which replaced Bayesian reasoning as it was intractable and computationally expensive [20]. In regard to this problem the accuracy does decrease but performance increases if we use fuzzy reasoning [20].

Fuzzy sets [5] [6] is also used in clustering. If we use the notion of sets in clustering then a given data point can belong to only one cluster. This notion is used in K means [15] which has already been stated before. But if we use the notion of fuzzy set [5] [6] then a given data point can belong to more than one cluster at the same time where the membership of the data point with the clusters is given by the membership function of the respective clusters. This concept has been used in Fuzzy C Means Clustering [9].

CHAPTER 6

FUZZY C MEANS CLUSTERING

We have already gone through Fuzzy set [5] [6] definition. Let us now come to Fuzzy C means clustering [9] abbreviated as FCM. Fuzzy C Means clustering [9] was developed by James .C Bezdek and others [9]. It uses the concept of fuzzy sets in clustering. It has the property that a data point belongs to all the clusters at the same time having varying degree of associativity. It allows overlapping clusters. Each cluster has their own membership function [9]. It should be stated well before that in Fuzzy C Means clustering [9], number of cluster to be detected in the given dataset is given prior to execution.

Let there be m clusters and n data points, then the cost function of fuzzy c means clustering [9] is given by

$$J = \sum_{i=1}^m \sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha || \vec{x}_p - \vec{v}_i ||^2$$

eqn(i)

With the constraint

$$\sum_{i=1}^m \mu_i(\vec{x}_p) = 1$$

eqn(ii)

Where μ_i is the fuzzy membership function for ith cluster taking \vec{x}_p as input and its range is $0 < \mu_i(\vec{x}_p) < 1$.

\vec{x}_p is a data point of a fixed dimension.

\vec{v}_i is the centroid of ith cluster having same dimensions that of a data point.

α is the fuzziness parameter having the acceptable range of $1 < \alpha < \infty$.

In the cost function we have used the l_2 norm which has been squared, basically it is computing the distance squared between the centroid and the data point.

Using the above cost function and constraint we get the following update equations

$$\vec{v}_i = \frac{\sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha \vec{x}_p}{\sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha}$$

eqn(iii)

$$\mu_i(\vec{x}_p) = \frac{\left[\frac{1}{\|\vec{x}_p - \vec{v}_i\|^{\frac{2}{\alpha-1}}} \right]}{\sum_{i=1}^m \left[\frac{1}{\|\vec{x}_p - \vec{v}_i\|^{\frac{2}{\alpha-1}}} \right]}$$

eqn(iv)

Using the above two update equations we develop the Fuzzy C Means clustering algorithm.

Algorithm: FCM [9]

Input: $\alpha, \vec{x}_p \forall p = 1 \text{ to } n$

Output: $\vec{v}_i \forall i = 1 \text{ to } m$

Step 1. Assign $\mu_i(\vec{x}_p) \in [0,1] \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$.

Step 2. Compute $\vec{v}_i \forall i = 1 \text{ to } m$.

Step 3. Compute $\mu_i(\vec{x}_p) \in [0,1] \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$.

Step 4. Repeat from step 2 until $\vec{v}_i \forall i = 1 \text{ to } m$ doesn't shift much.

Step 5. Exit.

The above algorithm gives centroids of the clusters. Fuzzy C Means clustering[9] does have certain advantages. When compared with K means Clustering [15] we see that FCM does not give crisp clusters but gives clusters which are supported by all the data points with varying

degree of membership this by passes the problem in k means [15] where a data point is equidistant from both the clusters and hence we need to arbitrarily break the tie by assigning the data point to either one of the clusters. It supports the idea that a data point can belong to more than one cluster at the same time. Fuzzy C Means Clustering [9] also has the advantage that it is less likely to get trapped in local optima than that of k means clustering [15]. The concept of fuzzy C means clustering [9] is also supported in real life that a person can belong to more than one social group at the same time. For instance a swimmer can have more than one swimming club membership or a party going person can have more than one club membership also a person is related to more than one family at the same time, the list is endless with such examples.

But with all the pros there has to be some cons. Let us understand how data points belonging to a cluster are extracted. It all depends on the membership function value of the cluster under study with data point as input to the function. We set a threshold value c for the task, if $\mu_i(\vec{x}_p) \geq c$ then \vec{x}_p belongs to the cluster else we remove it. Now following this concept a weakly supporting data point supporting all clusters will be removed from all the clusters. This is not the case for k means clustering. Also we have to set number of clusters to be found in the data set. This will cause a problem if the actual number of clusters is different. Another issue is that we don't have any idea about the proper value of fuzziness parameter. If wrong value is chosen, the algorithm gets trapped in local optima [21]. This problem becomes more prominent when number of dimensions is more [21].

CHAPTER 7

NORM: DEFINITION, TYPES, RELATION WITH FCM AND PROBLEM

In mathematics we come across the term Norm [3] [8] often. It is related with vector spaces [6]. Norm allows us to compute length of a vector [6], distance between two vectors[6].

There is a mathematical definition for norm [8]

A mapping $\|\cdot\|$ is called a norm iff it satisfies the following properties:

- a) $\|x\| \geq 0, \|x\|=0$ iff $x=0$ i.e. a null vector.
- b) $\|ax\|=|a| \|x\|, \forall a \in \mathbb{R}$
- c) $\|x+y\| \leq \|x\| + \|y\|$

Where x and y are vectors such that $\forall x, y \in \mathbb{R}^n$. [8]

So norm is a function which takes vector as inputs and generates a real number greater than or equal to zero.

Generally a l_p norm is given as follows [8]

$$l_p = \|x\|_p = \left[\sum_{i=1}^n |x_i|^p \right]^{\frac{1}{p}}$$

eqn(i)

From the above equation we can see that l_1 norm is given by [8]

$$l_1 = \|x\|_1 = \sum_{i=1}^n |x_i|$$

eqn(ii)

l_2 norm is given by [8]

$$l_2 = \|x\|_2 = \left[\sum_{i=1}^n |x_i|^2 \right]^{\frac{1}{2}}$$

eqn(iii)

l_3 norm is given by [8]

$$l_3 = ||x||_3 = \left[\sum_{i=1}^n |x_i|^3 \right]^{\frac{1}{3}}$$

eqn(iv)

l_∞ norm is given by [8]

$$l_\infty = ||x||_\infty = \max_{1 \leq i \leq n} |x_i|$$

eqn(v)

Larger the value of p more norm is sensitive to larger values and neglects smaller values [3].

l_0 gives number of nonzero elements in the given vector and l_∞ gives maximum absolute element in the vector [8].

Let us study l_2 norm [8] and see its properties.

l_2 norm [8] is also known as Euclidean norm. It is used to find the length of a vector. It is also used to find distance between two points in n dimensional space. Suppose we have point A(x,y) and B(p,q) then the distance between A and B is given by

$$AB = ((x - p)^2 + (y - q)^2)^{\frac{1}{2}} \quad \text{eqn(vi)}$$

If one of the points is origin then l_2 norm [8] gives us the distance between origin and the point under study.

One point to be noted is that l_2 norm [8] predominantly imposes a circular geometry in 2d plane and spherical geometry in 3d plane [4]. It is the inherent property of l_2 norm [3] [8] as we can see that it finds radius with one of the point as the center of the sphere or circle and we know that the equation given below is the equation of the locus of a circle with origin as its center.

$$x^2 + y^2 = r^2 \quad \text{eqn(vii)}$$

This is nothing but l_2 norm [8] squared with one of the point being (x,y) and other being the origin (0,0).

In Machine learning [1] [2] [3] we do frequently come across l_2 norm [3] [8] as it is less sensitive to large values than l_p where $p \geq 3$ and more sensitive than l_1 [3]. Norms are used in clustering like in k means [15], fuzzy c means clustering [9]. In fuzzy c means clustering [9] l_2 norm squared is used in the cost function to find the centroids and clusters [9] [15]. But using l_2 norm has its own draw backs.

l_2 norm [3] [8] is more susceptible to outliers than l_1 norm [3]. An outlier is a data point which is considerably away from other data points. Root mean square error is l_2 norm so it is susceptible to outliers than mean absolute error [3]. When outliers are less in data set then we can use l_2 norm as it gives better performance. Root mean square error is used in the cost function of linear regression which is optimized using gradient descent algorithm.

l_2 norm [3] [8] predominantly imposes a circular geometry [4] which causes a problem in clustering [3] [17] when we try to impose cluster of some different shape than that of a circle or spherical geometry. For instance we can't enforce parabolic or elliptical shape on the cluster because of the inherent nature of l_2 norm to impose spherical geometry.

In the following chapters we try to change the default search geometry of fuzzy c means clustering algorithm [9] to enforce parabola shape and ellipse shape on clusters. Changes in the cost function as well as changes in the iterative nature of the algorithm are required to achieve the goal. The former is based on the concept of shape functions and the later modification is done to make the cost function much simpler hence easier to derive the update equations which will simplify the entire process.

CHAPTER 8

LAGRANGE MULTIPLIER TECHNIQUE

If we are given an objective function $f(x)$ [8] without any constraints(equality/inequality) then we simply compute it's gradient and equate it with zero to minimize it i.e.

$$\frac{df(x)}{dx} = 0$$

If it is an objective function with multiple variables $f(x_1, x_2, x_3 \dots x_n)$ then we perform partial derivative w.r.t. individual variables and then equate them with zero i.e.

$$\frac{\partial f(x_1, x_2, x_3 \dots x_n)}{\partial x_1} = 0$$

$$\frac{\partial f(x_1, x_2, x_3 \dots x_n)}{\partial x_2} = 0$$

$$\frac{\partial f(x_1, x_2, x_3 \dots x_n)}{\partial x_3} = 0$$

.

.

.

$$\frac{\partial f(x_1, x_2, x_3 \dots x_n)}{\partial x_n} = 0$$

We do this because when we try to find minima of a function we try to find the point where gradient value changes it's sign from negative to positive. This is a simple task and is applicable when the function has no constraints.

When the objective function has constraints [8] to be more specific equality constraints [8] the above method can't be used which is quite simple. In short we can't perform derivative operation on constrained optimization problems directly.

To find the minima in such situations we need to convert the constrained optimization problem to its unconstrained form [8]. When the constraints are equality in nature, we apply Lagrange multiplier technique [8] to convert the constrained optimization problem[8] into unconstrained optimization problem [8].

The process is summarized as follows

$$\min \quad f(x_1, x_2, x_3, \dots, x_n)$$

Given constraints

$$c_j(x_1, x_2, x_3, \dots, x_n) = 0 \quad j = 1 \text{ to } p \text{ and } p < n$$

The unconstrained form of the above equality constrained optimization problem is

$$L(x_1, x_2, x_3, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_p) = f(x_1, x_2, x_3, \dots, x_n) + \sum_{j=1}^p \lambda_j c_j(x_1, x_2, x_3, \dots, x_n)$$

The above equation is that of the unconstrained optimization problem [8] where λ_j $j = 1$ to p are the Lagrange Multipliers [8].

Now we can apply the derivative operator to minimize $L(x_1, x_2, x_3, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_p)$ i.e.

$$\frac{\partial L(x_1, x_2, x_3, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_p)}{\partial x_i} = 0 \quad i = 1 \text{ to } n$$

$$\frac{\partial L(x_1, x_2, x_3, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_p)}{\partial \lambda_j} = 0 \quad j = 1 \text{ to } p$$

Now we can use the above set of equations to find value of the unknowns. We use this technique in the derivations done in the text.

CHAPTER 9

CONICS

Ellipse and Parabola, well known topics in high school mathematics, coordinate geometry to be more specific, are conic sections. In order for us to understand these two terms we need to first recapitulate conics.

A Cone is 3-dimensional shape as shown in fig. 9.1. According to Wikipedia a cone is a 3-dimensional geometric shape that tapers smoothly from a flat base to a point called as vertex.

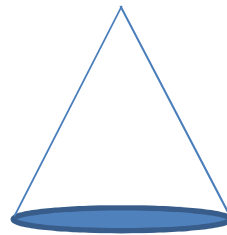


Fig.9.1 A Cone from which conic sections can be created

A conic section [22] can be defined as follows:

A conic section is a curve which is obtained by cutting a right-circular cone with a plane that doesn't pass through the vertex of the cone. [22]

In simple terms it means 2-dimensional faces which are formed by cutting a right-circular cone

Using mathematical jargons it can be defined as:

The locus of point P which moves on a plane so that its distance from a fixed point S on the plane always bears a constant ratio to its perpendicular distance PM from a fixed straight line ZH on the plane is called a conic section. [22]

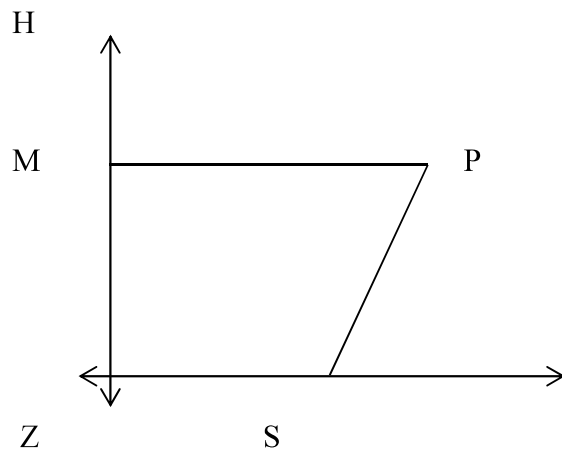


Fig.9.2 Graph showing a point on the locus of a conic section

In the above diagram S is the focus of the Conic,

ZH is the directrix i.e. the fixed straight line,

P is the point on the locus of the conic,

The ratio is known as eccentricity e and is defined as $SP:PM=e$.

Conic section can be divided into three categories which are as follows:

a)Parabola [22]

b)Ellipse [22]

c)Hyperbola [22]

A parabola is obtained when we cut the cone parallel to the generator of the cone. When the cutting plane is inclined to generator we get an ellipse if we have a single cone, a hyperbola if we use double cone whose generators are on the same straight line and have a common vertex.

CHAPTER 10

ELLIPSE

This chapter gives us a brief introduction to ellipse and its properties. It is necessary for us to recap the theory as it will come in handy in the following chapters.

Ellipse is a type of conic which is created when the cutting plane is inclined with respect to the generator of a right circular cone. [22]

It can be defined as follows:

An ellipse is the locus of a point P which moves such that its distance from a fixed point S always bears a constant ratio e ($0 < e < 1$) to its perpendicular distance from a fixed straight line i.e. directrix. [22]

e is the eccentricity of the ellipse under study.

Its standard equation is given by

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad [22]$$

Where

$$b^2 = a^2(1 - e^2) \quad [22]$$

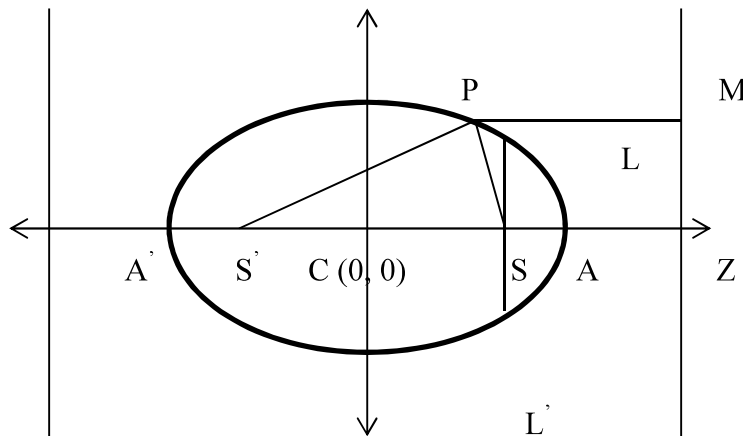


Fig.10.1 Ellipse

(x,y) is a point on the locus of the ellipse.

Coordinates of S is $(ae,0)$.

Coordinates of A is $(a,0)$. [22]

Coordinates of A' is $(-a,0)$. [22]

a is semi major/minor axis length of the ellipse. [22]

b is semi minor/major axis length of the ellipse. [22]

For a to be semi major axis length and b to be semi minor axis length, condition $a > b$ should hold. [22]

For b to be semi major axis length and a to be semi minor axis length, condition $b > a$ should hold. [22]

Major axis length is $2a$ if a is semi major axis then $2b$ is minor axis length vice versa. [22]

A and A' are vertices of the ellipse.

S and S' are focuses of the ellipse.

Length of latus rectum is $LL' = 2b^2/a$ [22]

SP and S'P are focal distances $SP = a - ex$ and $SP' = a + ex$ [22]

CHAPTER 11

MODIFICATION OF FCM CLUSERING TO ENFORCE ELLIPSE

In this part we try to alter the FCM [9] cost function so that the algorithm detects ellipse shaped clusters in 2d space instead of circles which was enforced by the l_2 norm [3] [8] used in the cost function [4]. In order to change the geometry of the clusters which was enforced by the FCM [9] algorithm we need to modify the cost function so that it enforces ellipse shape instead that of a circle. To be more specific we need to alter the l_2 norm [3] [8] squared part of the FCM [9] cost function.

Many cost functions were developed at first but only two of them were easily and properly solvable. At first a norm based approach was developed which tried to enforce ellipse shape on clusters but it had problems of its own as it was essentially enforcing two independent circles whose centers were thought to be as focuses of an ellipse. This function needs some more refinement.

The norm based cost function was given by

$$J = \sum_{i=1}^m \sum_{p=1}^n [\mu_i(\vec{x}_p)^\alpha (\|\vec{x}_p - \vec{v}_i\|^2 + \|\vec{k}_i - \vec{v}_i\|^2 - \|\vec{v}_i - \vec{l}_i\|^2) + \mu_{shape}(i)^\beta (\|\vec{x}_p - \vec{l}_i\|^2 + \|\vec{x}_p - \vec{k}_i\|^2)]$$

With constraints

$$\sum_{i=1}^m \mu_i(\vec{x}_p) = 1$$

$$\sum_{i=1}^m \mu_{shape}(i) = m$$

And the update equations were given by

$$\vec{v}_i = \frac{\sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha (\vec{x}_p + \vec{k}_i - \vec{l}_i)}{\sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha}$$

$$\vec{k}_i = \frac{\sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha \vec{v}_i + \sum_{p=1}^n \mu_{shape}(i)^\beta \vec{x}_p}{\sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha + \sum_{p=1}^n \mu_{shape}(i)^\beta}$$

$$\vec{l}_i = \frac{\sum_{p=1}^n \mu_{shape}(i)^\beta \vec{x}_p - \sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha \vec{v}_i}{\sum_{p=1}^n \mu_{shape}(i)^\beta - \sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha}$$

$$\mu_i(\vec{x}_p) = \frac{\left[\frac{1}{(\|\vec{x}_p - \vec{v}_i\|^2 + \|\vec{k}_i - \vec{v}_i\|^2 - \|\vec{v}_i - \vec{l}_i\|^2)} \right]^{\frac{1}{\alpha-1}}}{\sum_{i=1}^m \left[\frac{1}{(\|\vec{x}_p - \vec{v}_i\|^2 + \|\vec{k}_i - \vec{v}_i\|^2 - \|\vec{v}_i - \vec{l}_i\|^2)} \right]^{\frac{1}{\alpha-1}}}$$

$$\mu_{shape}(i) = \frac{\frac{m}{\left\{ \sum_{p=1}^n (\|\vec{x}_p - \vec{l}_i\|^2 + \|\vec{x}_p - \vec{k}_i\|^2) \right\}^{\frac{1}{\beta-1}}}}{\sum_{i=1}^m \frac{1}{\left\{ \sum_{p=1}^n (\|\vec{x}_p - \vec{l}_i\|^2 + \|\vec{x}_p - \vec{k}_i\|^2) \right\}^{\frac{1}{\beta-1}}}}$$

Here \vec{k}_i and \vec{l}_i are focuses of the i th ellipse where \vec{v}_i is the centroid of the i th ellipse. β and α are fuzzifiers. $\mu_{shape}(i)$ is the fuzzy shape membership function which gives degree of the i th cluster being ellipse shaped and $\mu_i(\vec{x}_p)$ is the fuzzy membership function and \vec{x}_p is the 2d data point.

The second cost function is based on the equation of the ellipse [22] itself and is inspired from [10] the difference being that it requires less number of unknown variables and the function is itself in a much simpler form. All this factors have helped in achieving the task.

11.1 Cost Function and Derivation of update equations

Let $\mu_i(\vec{x}_p)$ be the fuzzy membership function of i^{th} cluster and p^{th} data point. It can have value in the range $0 < \mu_i(\vec{x}_p) < 1$.

a_i is the semi major/minor axis length of i^{th} cluster.

b_i is the semi minor/major axis length of i^{th} cluster.

α is the fuzziness parameter which has the range of permissible values of $1 < \alpha < \infty$.

$\vec{x}_p = (x_{xp}, x_{yp}) \forall p=1$ to n are p 2-dimensional data points.

$\vec{v}_i = (v_{xi}, v_{yi}) \forall i=1$ to m are 2-dimensional centroids of i^{th} cluster.

Let the cost function for enforcing elliptical cluster geometry in the given 2-dimensional dataset be given by:

$$J = \sum_{i=1}^m \sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha \left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right] \dots eqn(i)$$

With the constraint

$$\sum_{i=1}^m \mu_i(\vec{x}_p) = 1 \dots \dots eqn(ii)$$

Converting the above equation from constrained [8] to unconstrained [8] form by applying Lagrange multiplier technique [8] we get:

$$F = \sum_{i=1}^m \sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha \left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right] - \sum_{p=1}^n \lambda_p \left[\sum_{i=1}^m \mu_i(\vec{x}_p) - 1 \right] \dots eqn(iii)$$

Where λ_p is the Lagrange multiplier [8]. F is the unconstrained [8] cost function form of J .

Eqn(i) has been designed on the principle that:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad [22]$$

Is the equation of an ellipse centered at $(0,0)$. The equation below comes handy

$$\frac{(\psi - x)^2}{a^2} + \frac{(\varphi - y)^2}{b^2} = 1 \quad [22]$$

The above equation is that of an ellipse centered at (ψ, φ) . In this modified form we consider (ψ, φ) to be the centroid of ellipse under study. Here the concept of translation is used.

x and y are the x axis and y axis values of (x,y) respectively.

a and b are the lengths of semi major/minor axis.

In order to find the unknowns we need to minimize eqn(iii) we do this by using minimization criterion for a function using derivatives i.e $\frac{d}{dx}f(x) = 0$. Since there are more than one unknown variables in eqn(iii) we will need to use the partial derivative operator.

Our goal here is to find values for the unknown variables such that the unconstrained cost function [8] F is minimized.

Finding v_{xi}

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to v_{xi}

$$\frac{\partial F}{\partial v_{xi}} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} 2 \frac{(v_{xi} - x_{xp})}{a_i^2} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} \frac{(v_{xi} - x_{xp})}{a_i^2} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} (v_{xi} - x_{xp}) = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} v_{xi} - \sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} x_{xp} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} v_{xi} = \sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} x_{xp}$$

$$\Rightarrow v_{xi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} x_{xp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha}} \dots eqn(iv)$$

$$\therefore v_{xi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha} x_{xp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^{\alpha}} \text{ is the update equation for finding } v_{xi}.$$

Finding v_{yi}

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to v_{yi}

$$\frac{\partial F}{\partial v_{yi}} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha \cdot \frac{(v_{yi} - x_{yp})}{b_i^2} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha \frac{(v_{yi} - x_{yp})}{b_i^2} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha (v_{yi} - x_{yp}) = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha v_{yi} - \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha v_{yi} = \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}$$

$$\Rightarrow v_{yi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha} \dots eqn(v)$$

$$v_{yi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha} \text{ is the update equation for finding } v_{yi}.$$

Finding λ_p

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to λ_p

$$\frac{\partial F}{\partial \lambda_p} = 0$$

$$\Rightarrow - \left[\sum_{i=1}^m \mu_i(\vec{x_p}) - 1 \right] = 0$$

$$\Rightarrow \sum_{i=1}^m \mu_i(\vec{x_p}) - 1 = 0$$

$$\Rightarrow \sum_{i=1}^m \mu_i(\vec{x_p}) = 1 \dots eqn(vi)$$

\therefore on solving $\frac{\partial F}{\partial \lambda_p} = 0$ we get $\sum_{i=1}^m \mu_i(\vec{x_p}) = 1$ this is required to find the update equation of the $\mu_i(\vec{x_p})$ unknown.

Finding $\mu_i(\overrightarrow{x_p})$

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to $\mu_i(\overrightarrow{x_p})$

$$\frac{\partial F}{\partial \mu_i(\overrightarrow{x_p})} = 0$$

$$\Rightarrow \alpha \mu_i(\overrightarrow{x_p})^{\alpha-1} \left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right] - \lambda_p = 0$$

$$\Rightarrow \alpha \mu_i(\overrightarrow{x_p})^{\alpha-1} \left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right] = \lambda_p$$

$$\Rightarrow \mu_i(\overrightarrow{x_p})^{\alpha-1} \left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right] = \frac{\lambda_p}{\alpha}$$

$$\Rightarrow \mu_i(\overrightarrow{x_p})^{\alpha-1} = \frac{\lambda_p}{\alpha} \frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]}$$

$$\Rightarrow \mu_i(\overrightarrow{x_p}) = \left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} \frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}} \dots \text{eqn(vii)}$$

Substituting eqn(vii) in eqn(vi) we get:

$$\Rightarrow \sum_{i=1}^m \left[\left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} \frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}} \right] = 1$$

$$\Rightarrow \left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} \sum_{i=1}^m \left[\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}} \right] = 1$$

$$\Rightarrow \left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} = \frac{1}{\sum_{i=1}^m \left[\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}} \right]} \dots eqn(viii)$$

From eqn(viii) we get the value of $\left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}}$ which is one of the terms in eqn(vii). We can substitute eqn(viii) in eqn(vii).

On substituting eqn(viii) in eqn(vii) we get:

$$\Rightarrow \mu_i(\vec{x}_p) = \frac{\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}}}{\sum_{i=1}^m \left[\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}} \right]} \dots eqn(ix)$$

$$\therefore \mu_i(\vec{x}_p) = \frac{\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}}}{\sum_{i=1}^m \left[\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}} \right]} \text{ is the update equation for } \mu_i(\vec{x}_p) \text{ .}$$

In a summary in the above derivations we have developed three update equations which will be used in algorithms to detect ellipse clusters.

The update equations after solving eqn(iii) are as follows:

$$v_{xi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{xp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha}$$

$$v_{yi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha}$$

$$\mu_i(\vec{x_p}) = \frac{\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}}}{\sum_{i=1}^m \left[\frac{1}{\left[\frac{(v_{xi} - x_{xp})^2}{a_i^2} + \frac{(v_{yi} - x_{yp})^2}{b_i^2} \right]^{\frac{1}{\alpha-1}}} \right]}$$

11.2 Modified Algorithm

Using the derived equations for imposing ellipse geometry on clusters, we modify the update equations in the Fuzzy C Means Clustering algorithm [9]. The modified algorithm should impose ellipse geometry on clusters provided the space is 2 dimensional i.e. the data points have x and y coordinates. The modified algorithm is presented below:

Algorithm: Modified Fuzzy C Means Ellipse clustering

Input: $\alpha, \vec{x}_p \forall p = 1 \text{ to } n$

Output: $\vec{v}_i \forall i = 1 \text{ to } m$

Step 1. Assign $\mu_i(\vec{x}_p) \in [0,1] \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$.

Step 2. Select values for a_i and $b_i \forall i = 1 \text{ to } m$

Step 3. Compute v_{xi} and $v_{yi} \forall i = 1 \text{ to } m$

Step 4. Compute $\mu_i(\vec{x}_p) \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$

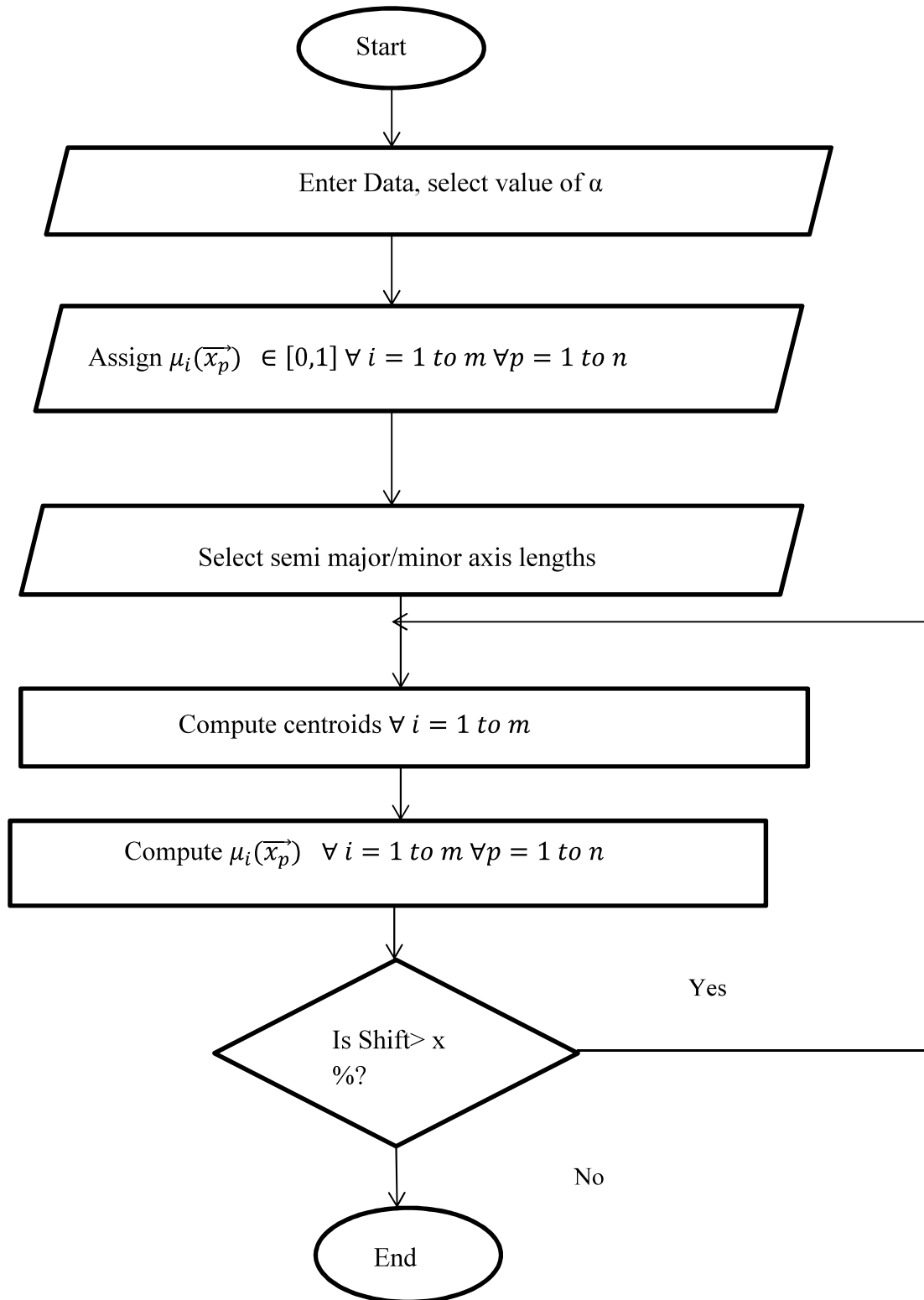
Step 5. Goto Step 3 until v_{xi} and $v_{yi} \forall i = 1 \text{ to } m$ doesn't change much.

Step 6. Print the cluster centroids along with clusters

Step 7. End

The above algorithm does detect ellipses and is shown by the experiments performed. It is to be mentioned that the experiments were performed on artificial dataset which had ellipse shaped clusters in it. Coding was done using GNU Octave. The terms have same meaning as in section 11.1.

11.3 Flowchart



11.4 Code

```
function r=v_modder(mu_x,alpha,data)

d1=reshape(data(:,1),1,100);
d2=reshape(data(:,2),1,100);

vx1=sum((mu_x(1,:).^alpha).*d1)/sum(mu_x(1,:).^alpha);
vy1=sum((mu_x(1,:).^alpha).*d2)/sum(mu_x(1,:).^alpha);
vx2=sum((mu_x(2,:).^alpha).*d1)/sum(mu_x(2,:).^alpha);
vy2= sum((mu_x(2,:).^alpha).*d2)/sum(mu_x(2,:).^alpha);
vx3= sum((mu_x(3,:).^alpha).*d1)/sum(mu_x(3,:).^alpha);
vy3= sum((mu_x(3,:).^alpha).*d2)/sum(mu_x(3,:).^alpha);
vx4= sum((mu_x(4,:).^alpha).*d1)/sum(mu_x(4,:).^alpha);
vy4= sum((mu_x(4,:).^alpha).*d2)/sum(mu_x(4,:).^alpha);
vx5= sum((mu_x(5,:).^alpha).*d1)/sum(mu_x(5,:).^alpha);
vy5= sum((mu_x(5,:).^alpha).*d2)/sum(mu_x(5,:).^alpha);
vx6= sum((mu_x(6,:).^alpha).*d1)/sum(mu_x(6,:).^alpha);
vy6= sum((mu_x(6,:).^alpha).*d2)/sum(mu_x(6,:).^alpha);
vx7= sum((mu_x(7,:).^alpha).*d1)/sum(mu_x(7,:).^alpha);
vy7= sum((mu_x(7,:).^alpha).*d2)/sum(mu_x(7,:).^alpha);
vx8= sum((mu_x(8,:).^alpha).*d1)/sum(mu_x(8,:).^alpha);
vy8= sum((mu_x(8,:).^alpha).*d2)/sum(mu_x(8,:).^alpha);
vx9= sum((mu_x(9,:).^alpha).*d1)/sum(mu_x(9,:).^alpha);
vy9= sum((mu_x(9,:).^alpha).*d2)/sum(mu_x(9,:).^alpha);
vx10= sum((mu_x(10,:).^alpha).*d1)/sum(mu_x(10,:).^alpha);
vy10= sum((mu_x(10,:).^alpha).*d2)/sum(mu_x(10,:).^alpha);

r=[vx1 vy1;vx2 vy2;vx3 vy3;vx4 vy4;vx5 vy5;vx6 vy6;vx7 vy7;vx8 vy8;vx9 vy9;vx10 vy10];
```

```

endfunction

function e=mu_x_modder(v,data,alpha,a,b)
mu=zeros(10,100);

for i= 1: 10
for j= 1: 100
s= (1/(((v(i,1)-data(j,1))/a(i))^2+ ((v(i,2)-data(j,2))/b(i))^2))^(1/(alpha-1));
k=intermediate(j,a,b,v,data,alpha);
mu(i,j)=s/k;
end
end

e=mu;
endfunction

function e=intermediate(j,a,b,v,data,alpha)
s=0;

for i= 1: 10
s=s+ (1/(((v(i,1)-data(j,1))/a(i))^2+ ((v(i,2)-data(j,2))/b(i))^2))^(1/(alpha-1));
end

e=s;
endfunction

```

```

alpha = 4;
a = rand (10,1) * 100;
b = rand (10,1) *10;
i = 0;
mu_x = rand (10,100);
vnew = v_modder ( mu_x,alpha,data );
mu_x = mu_x_modder ( vnew,data,alpha,a,b );
vold = vnew;
i = i + 1
while(1)
    vnew = v_modder ( mu_x,alpha,data );
    mu_x=mu_x_modder(vnew,data,alpha,a,b);
    if((abs(vnew-vold)./vold).*100<=ones(10,2))
        break;
    endif
    vold=vnew;
    i=i+1
endwhile

```

11.5 Variable Description

Variable	Description
r, e	Function output variables
data	Vector of size 100x2 to store data
d1,d2	Vectors of size 1x100 to store x and y values respectively
mu_x	Fuzzy membership function
alpha	Fuzziness parameter i.e. α
a,b	Vector storing semi major/minor axis lengths
i,j	Counters
s,k	To perform summation in mu_x_modder
vx1-vx10	x component of computed centroids
vy1-vy10	y component of computed centroids
vold	To store centroids of previous iteration
vnew	To store centroids of current iteration
mu	To store membership function temporarily while computing it

11.6 Output

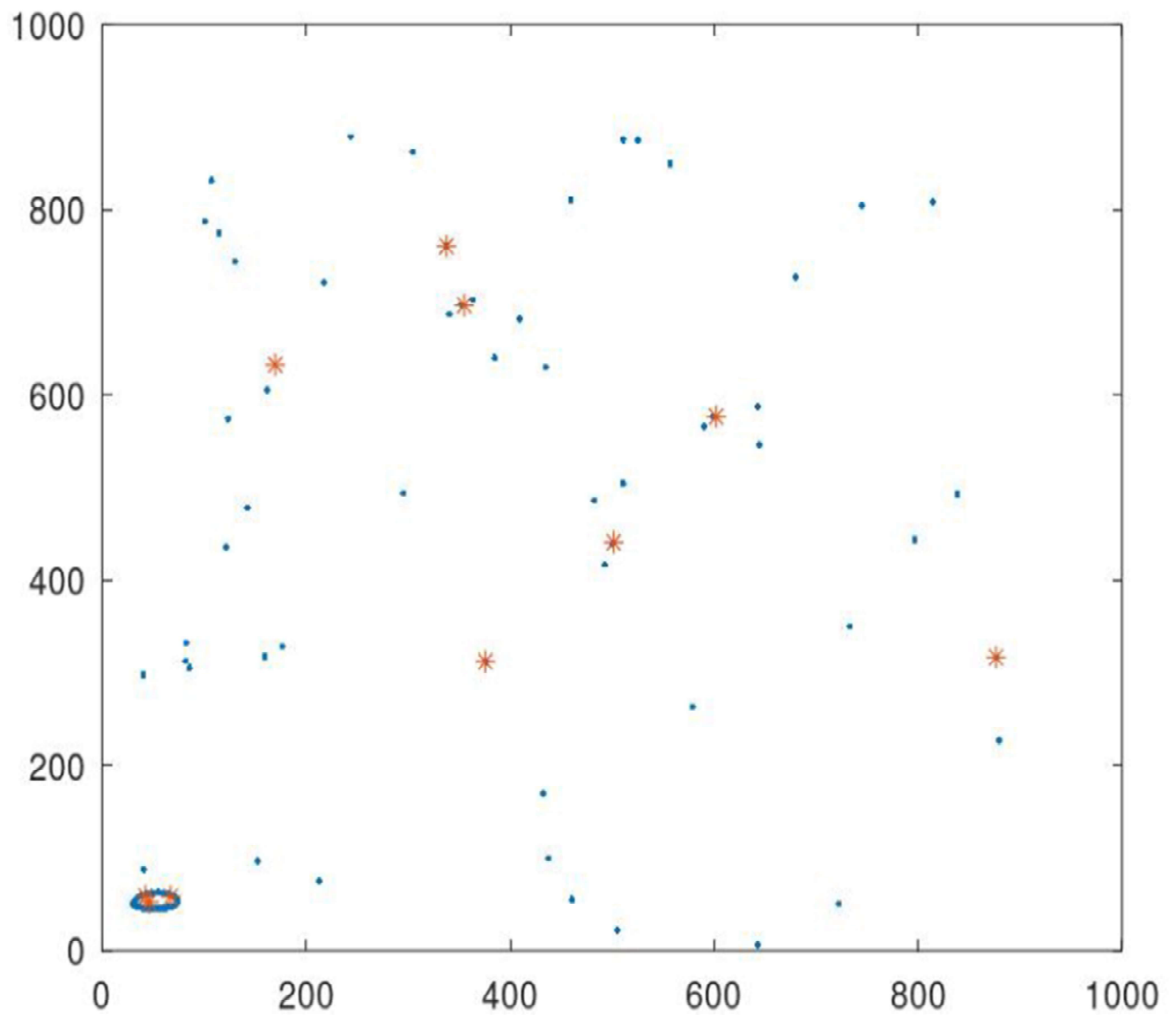


Fig.11.6.1 Graph Showing output of the modified Algorithm

Blue dots are data points and orange asterix are centroids of detected clusters.

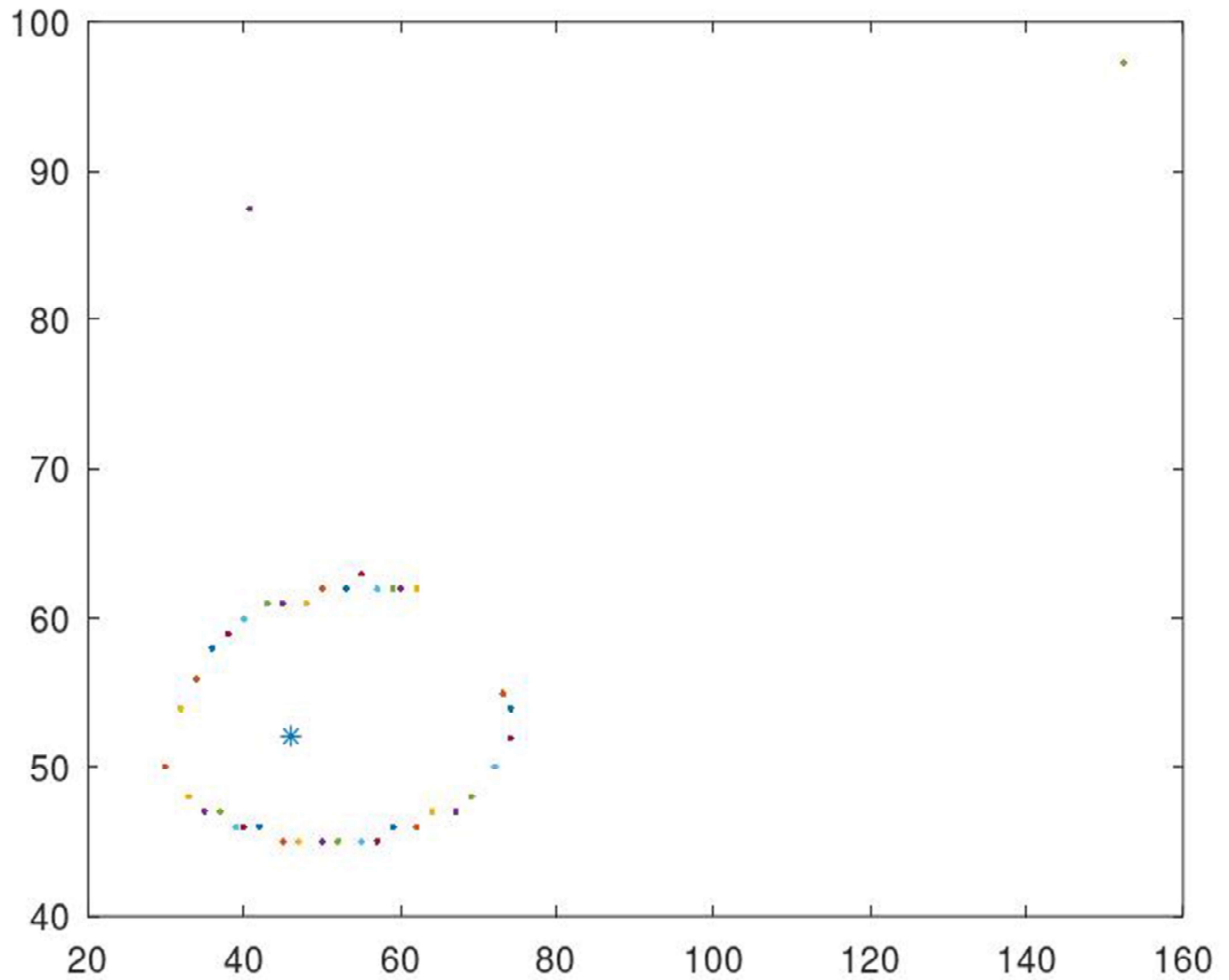


Fig.11.6.2 Graph showing the elliptical cluster 1

Dots represent data points and are selected on the basis of a threshold value. Blue asterix is centroid of the detected cluster.

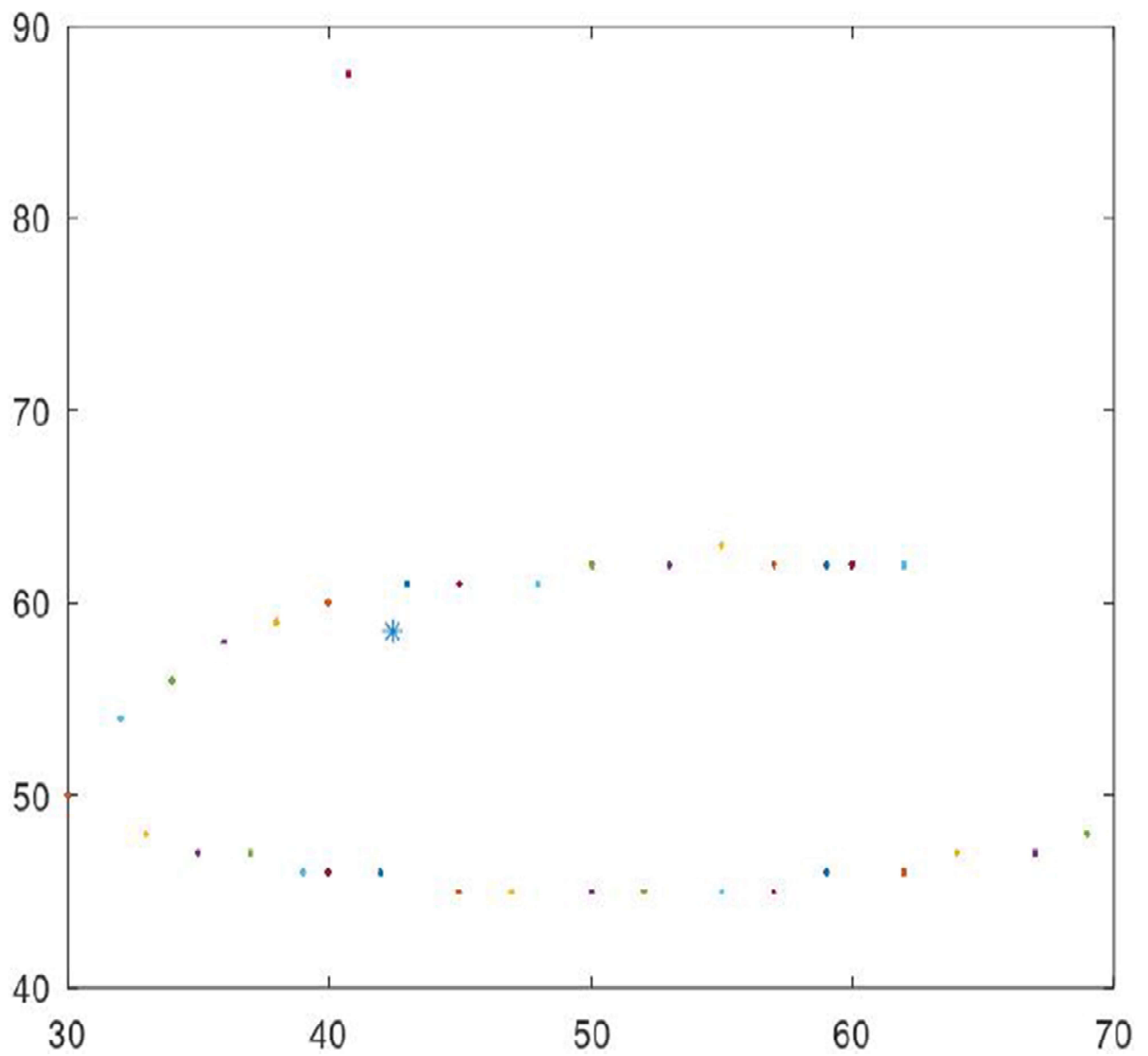


Fig.11.6.3 Graph Showing elliptical cluster 6

Dots represent data points and are selected on the basis of a threshold value. Blue asterix is centroid of the detected cluster.

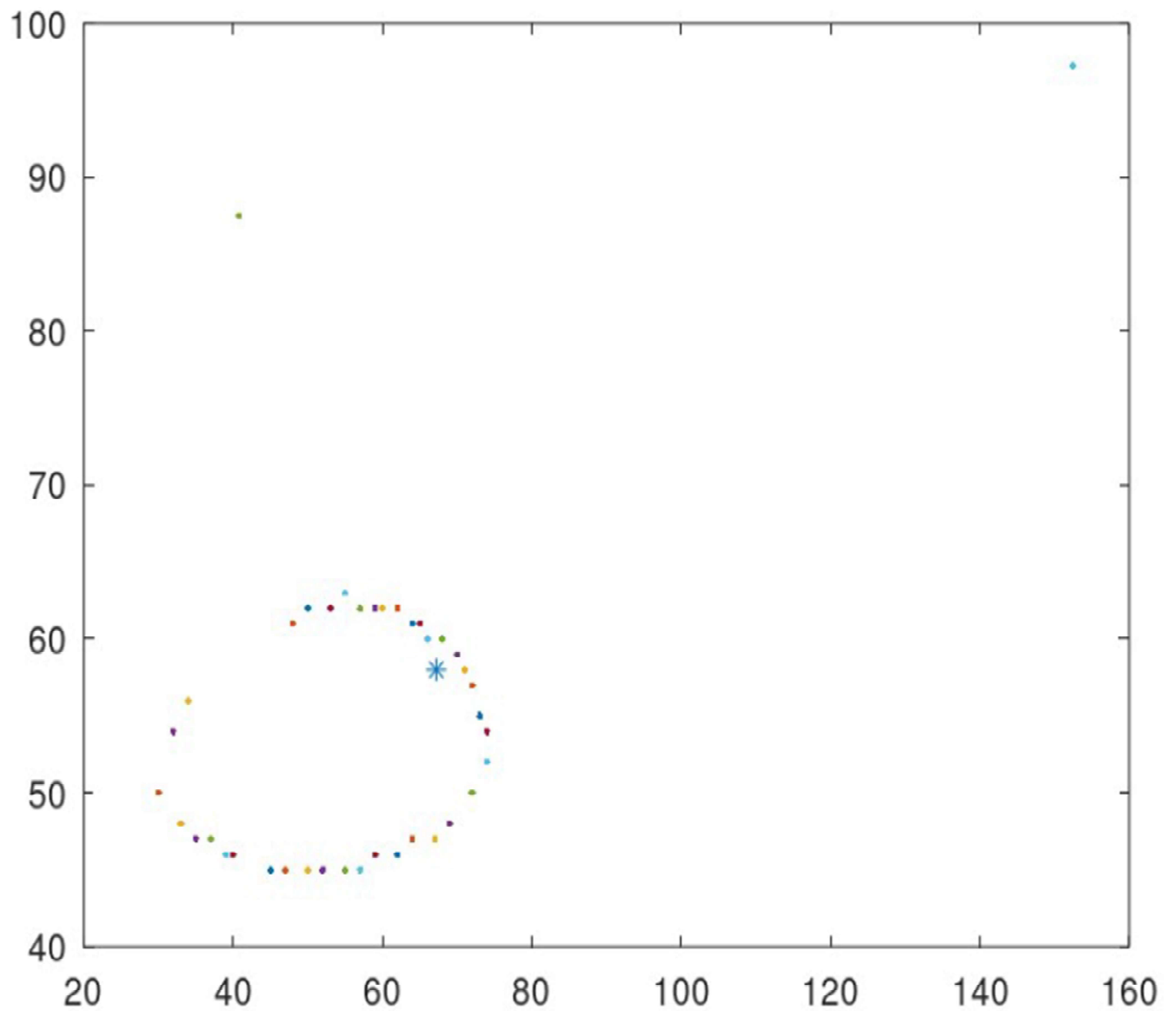


Fig.11.6.4 Graph showing elliptical cluster 10

Dots represent data points and are selected on the basis of a threshold value. Blue asterix is centroid of the detected cluster.

11.7 Discussion Of The Output

From the previous section we have seen that the modified algorithm does detect ellipses and the proof has already been provided as images.

The problem we face now is that the algorithm is somewhat rigid in its approach. It is rigid in the sense that we haven't developed the update equations for a_i and $b_i \forall i = 1 \text{ to } m$. The problem is that the cost function is not in the form that allows to develop their update equations. We bypassed this problem by using predefined values generated using random number generator and multiplying the values with a suitable scale factor. It is to be said again that a_i and $b_i \forall i = 1 \text{ to } m$ are lengths of semi major/minor axes of i^{th} cluster. This parameter is data specific and even though we provide it using random numbers, there still remains the chance of the values being very different than the actual. We need them to be as close as possible to the original for better clustering.

11.8 Fuzzy C Means Ellipse Clustering of Variable Size

Using the derived update equations, Fuzzy C Means clustering [9] algorithm, we develop a new algorithm. The algorithm is given as follows.

Algorithm: Fuzzy C Means Ellipse Clustering of Variable Size

Input: $\alpha, \vec{x}_p \forall p = 1 \text{ to } n$

Output: $\vec{v}_{ik} \forall i = 1 \text{ to } m \forall k = 1 \text{ to } 10$.

Step 1. Select values for a_i and $b_i \forall i = 1 \text{ to } m$.

Step 2. Set $aold_i = a_i * 0.1$ and $bold_i = b_i * 0.1 \forall i = 1 \text{ to } m$.

Step 3. For $k=1$ to 10 repeat step 4 to step 10

Step 4. Assign $\mu_i(\vec{x}_p) \in [0,1] \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$.

Step 5. Compute v_{xi} and $v_{yi} \forall i = 1 \text{ to } m$

Step 6. Compute $\mu_i(\vec{x}_p) \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$

Step 7. Goto Step 4 until v_{xi} and $v_{yi} \forall i = 1 \text{ to } m$ doesn't change much.

Step 8. Print v_{xi} and $v_{yi} \forall i = 1 \text{ to } m$

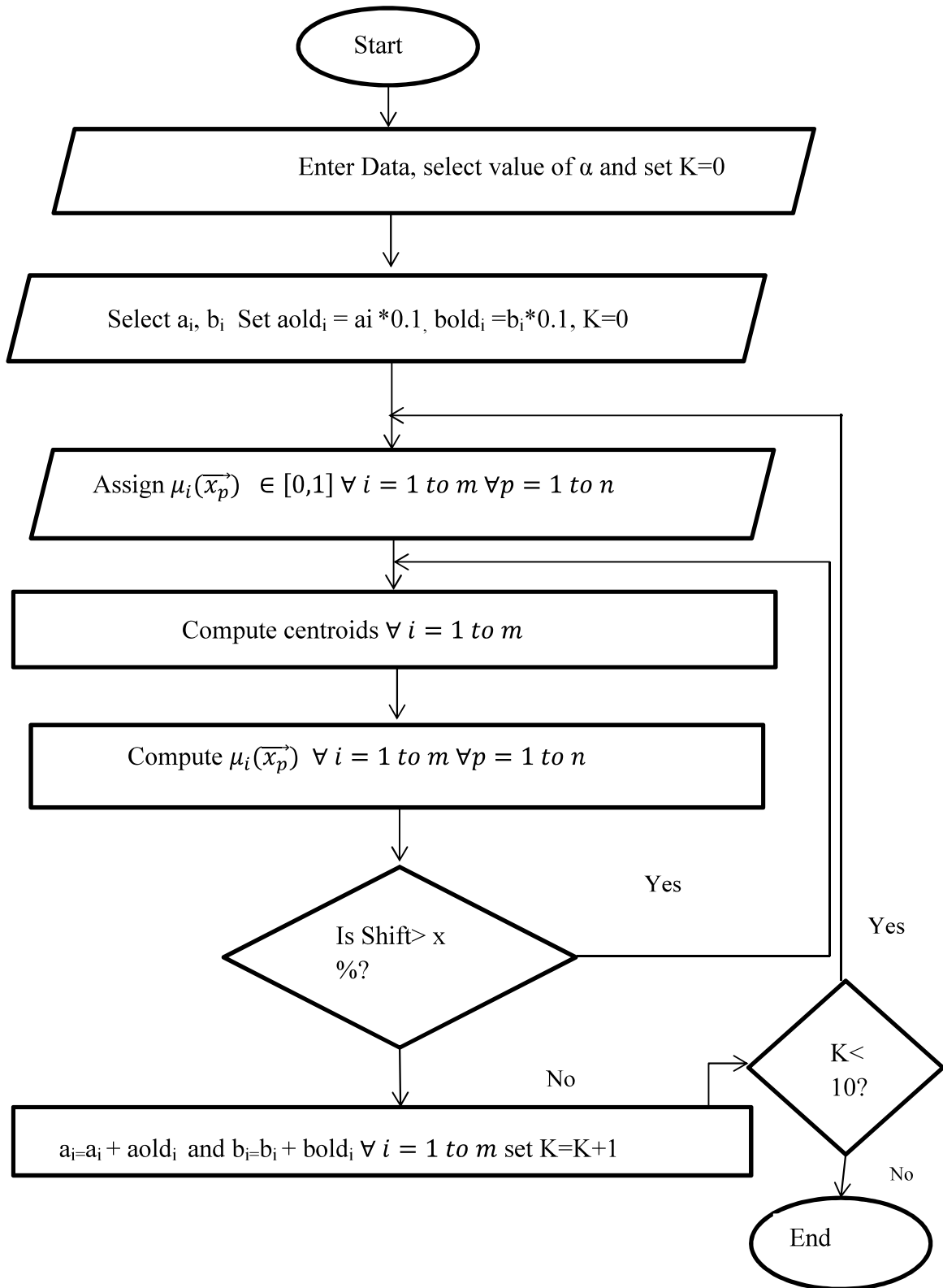
Step 9. Set $a_i = a_i + aold_i$ and $b_i = b_i + bold_i \forall i = 1 \text{ to } m$

Step 10. Set $k=k+1$

Step 11. End

The algorithm has been implemented using GNU Octave. k denotes run number. The above algorithm increases semi axis lengths of clusters by a predefined percentage of the initial values to solve the problem stated in the previous section. Here we increase the size of clusters to be detected in each and every run. Run number and percentage can be varied.

11.9 Flowchart



11.10 Code

```
function r=v_modder(mu_x,alpha,data)

d1=reshape(data(:,1),1,100);
d2=reshape(data(:,2),1,100);

vx1=sum((mu_x(1,:).^alpha).*d1)/sum(mu_x(1,:).^alpha);
vy1=sum((mu_x(1,:).^alpha).*d2)/sum(mu_x(1,:).^alpha);
vx2=sum((mu_x(2,:).^alpha).*d1)/sum(mu_x(2,:).^alpha);
vy2= sum((mu_x(2,:).^alpha).*d2)/sum(mu_x(2,:).^alpha);
vx3= sum((mu_x(3,:).^alpha).*d1)/sum(mu_x(3,:).^alpha);
vy3= sum((mu_x(3,:).^alpha).*d2)/sum(mu_x(3,:).^alpha);
vx4= sum((mu_x(4,:).^alpha).*d1)/sum(mu_x(4,:).^alpha);
vy4= sum((mu_x(4,:).^alpha).*d2)/sum(mu_x(4,:).^alpha);
vx5= sum((mu_x(5,:).^alpha).*d1)/sum(mu_x(5,:).^alpha);
vy5= sum((mu_x(5,:).^alpha).*d2)/sum(mu_x(5,:).^alpha);
vx6= sum((mu_x(6,:).^alpha).*d1)/sum(mu_x(6,:).^alpha);
vy6= sum((mu_x(6,:).^alpha).*d2)/sum(mu_x(6,:).^alpha);
vx7= sum((mu_x(7,:).^alpha).*d1)/sum(mu_x(7,:).^alpha);
vy7= sum((mu_x(7,:).^alpha).*d2)/sum(mu_x(7,:).^alpha);
vx8= sum((mu_x(8,:).^alpha).*d1)/sum(mu_x(8,:).^alpha);
vy8= sum((mu_x(8,:).^alpha).*d2)/sum(mu_x(8,:).^alpha);
vx9= sum((mu_x(9,:).^alpha).*d1)/sum(mu_x(9,:).^alpha);
vy9= sum((mu_x(9,:).^alpha).*d2)/sum(mu_x(9,:).^alpha);
vx10= sum((mu_x(10,:).^alpha).*d1)/sum(mu_x(10,:).^alpha);
vy10= sum((mu_x(10,:).^alpha).*d2)/sum(mu_x(10,:).^alpha);

r=[vx1 vy1;vx2 vy2;vx3 vy3;vx4 vy4;vx5 vy5;vx6 vy6;vx7 vy7;vx8 vy8;vx9 vy9;vx10 vy10];
```

```

endfunction

function e=mu_x_modder(v,data,alpha,a,b)

mu=zeros(10,100);

for i= 1: 10

for j= 1: 100

s= (1/(((v(i,1)-data(j,1))/a(i))^2+ ((v(I,2)-data(j,2))/b(i))^2))^(1/(alpha-1));

k=intermediate(j,a,b,v,data,alpha);

mu(i,j)=s/k;

end

end

e=mu;

endfunction


function e=intermediate(j,a,b,v,data,alpha)

s=0;

for i= 1: 10

s=s+ (1/(((v(i,1)-data(j,1))/a(i))^2+ ((v(I,2)-data(j,2))/b(i))^2))^(1/(alpha-1));

end

e=s;

endfunction

alpha=4;

aold=a=rand(10,1)*100;
bold=b=rand(10,1)*10;
aold1=aold.*(0.1);
bold1=bold.*(0.1);
for K= 1 to 10
i=0;
mu_x=rand(10,100);

```

```

vnew=v_modder(mu_x,alpha,data);
mu_x=mu_x_modder(vnew,data,alpha,a,b);
vold=vnew;
i=i+1
vold
while(1)
vnew=v_modder(mu_x,alpha,data);
mu_x=mu_x_modder(vnew,data,alpha,a,b);
if((abs(vnew-vold)./vold).*100<=ones(10,2))
break;
endif
vold=vnew;
i=i+1
endwhile
vnew
a
b
mu_x=rand(10,100);
a=a+aold1;
b=b+bold1;
end

```

11.11 Variable Description

Variable	Description
r, e	Function output variables
data	Vector of size 100x2 to store data
d1,d2	Vectors of size 1x100 to store x and y values respectively
mu_x	Fuzzy membership function
alpha	Fuzziness parameter i.e. α
a,b	Vector storing semi major/minor axis lengths
i,j,K	Counters
s,k	To perform summation in mu_x_modder
vx1-vx10	x component of computed centroids
vy1-vy10	y component of computed centroids
vold	To store centroids of previous iteration
vnew	To store centroids of current iteration
aold, bold	To store initial semi major/minor axis lengths
aold1,bold1	To store 10% of initial semi major/minor axis lengths
mu	To help in temporary storage of membership function while computing it

11.12 Output

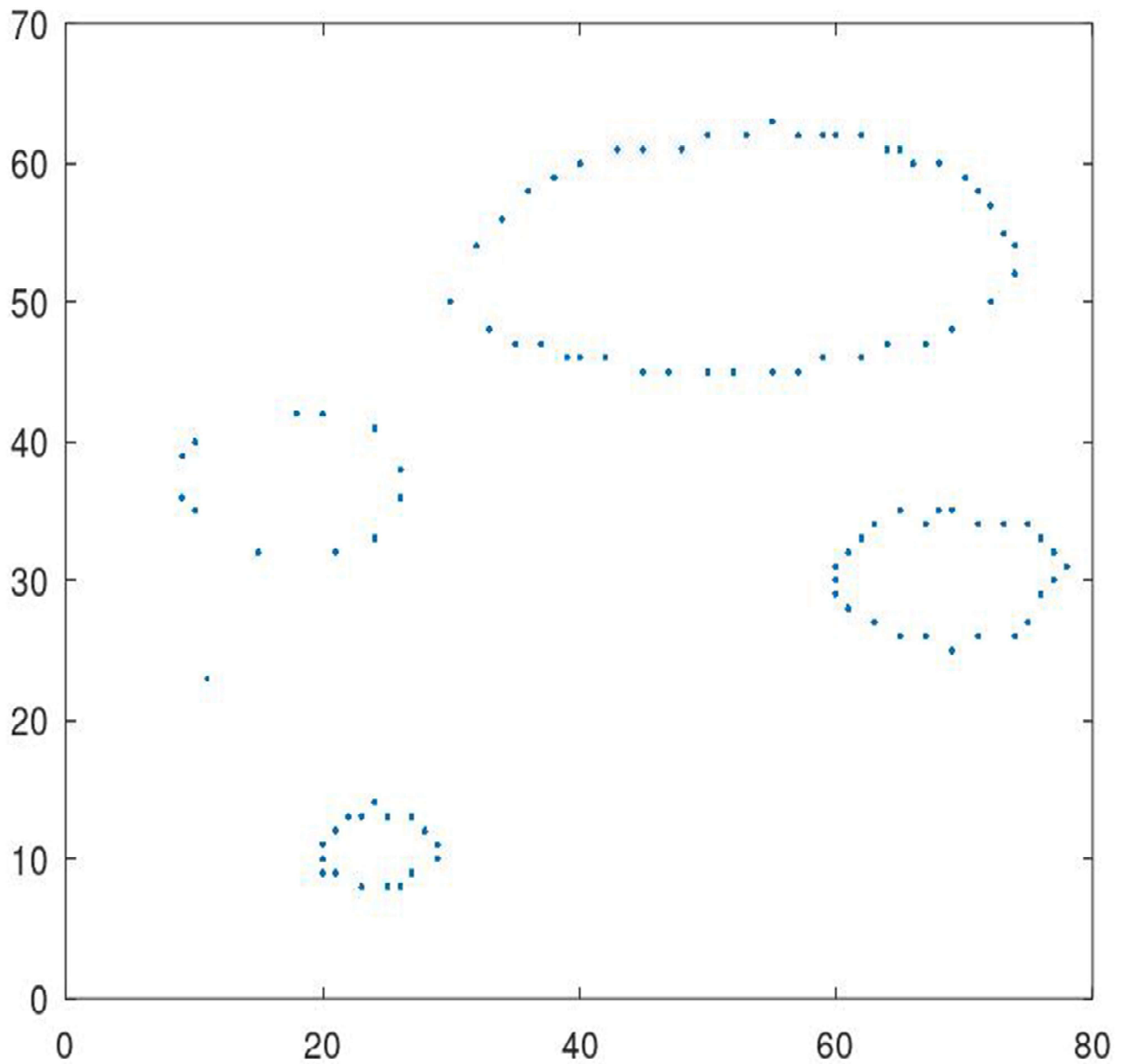


Fig. 11.12.1 Graph showing artificial data containing ellipses

In the graph blue dots represents data points. The data set is artificial and designed under supervision to contain ellipses. From the previously presented algorithm it was shown that the calculations do induce ellipse geometry in 2 D space.

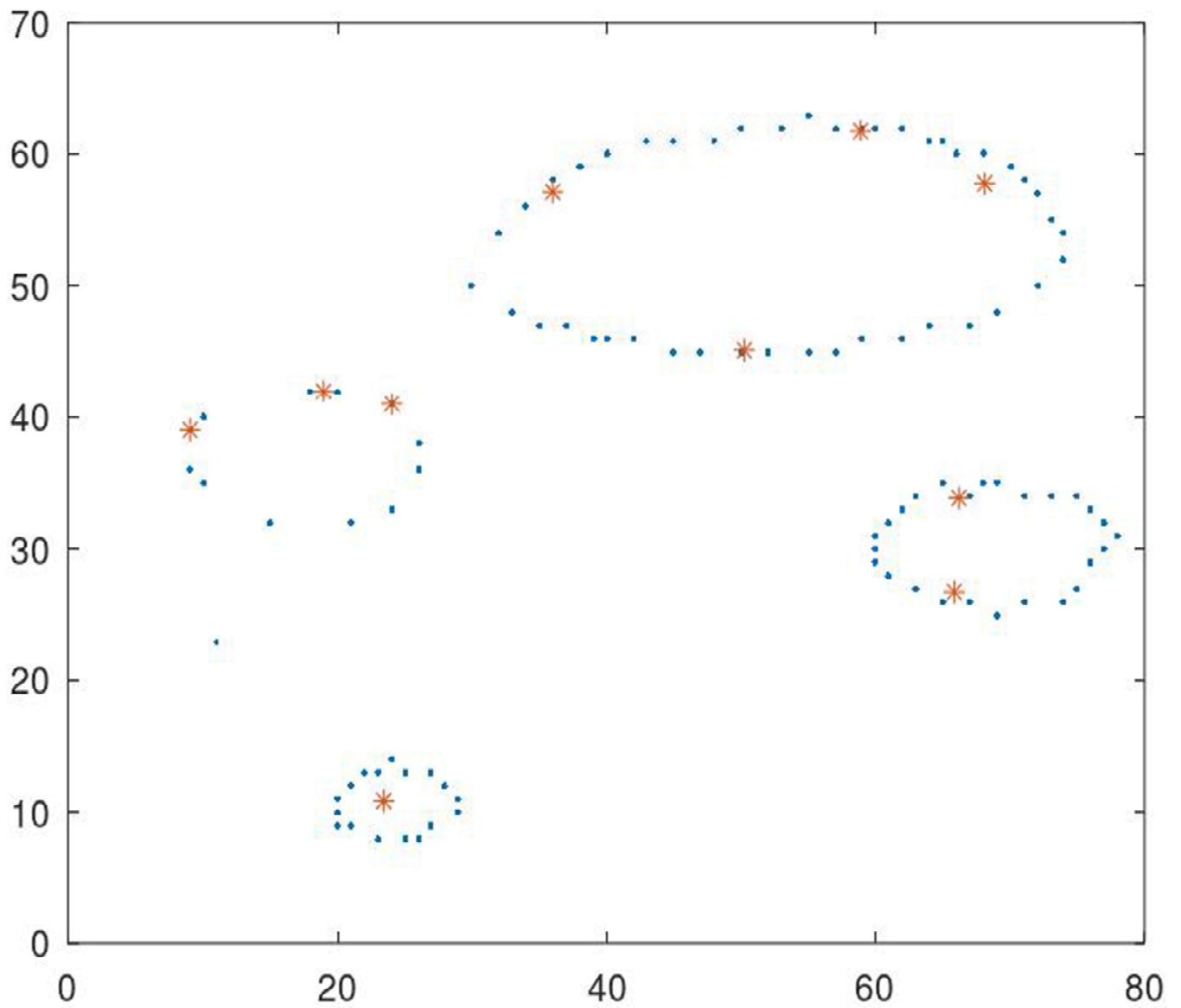


Fig.11.12.2 Graph showing output of the algorithm after the final run

In the graph blue dots represents data points and the orange asterisk represents centroids. The number of clusters to be detected was selected as 10 and all the centroids are within the locus of detected ellipses.

11.13 Discussion Of The Output

The Fuzzy C means Ellipse clustering of variable size algorithm does work. It also solved the problem of the modified algorithm to some extent which was of fixed semi major/minor axis lengths.

The algorithm will perform much better if we somehow can learn the fuzziness parameter value which will yield minimal cost function value else it will lead to local minima [21]. Here we have speculated and selected as per choice.

CHAPTER 12

PARABOLA

Parabola [22] is a conic [22] which is formed when the cutting plane is parallel to the generator of the cone [22].

Parabola can be defined as follows:

A Parabola is the locus of a point which moves in a plane in such a way that its distance from a fixed point S(focus), is always equal to its perpendicular distance from a fixed straight line (directrix) in the plane. [22]

Equation of a parabola having its axis as x axis and y axis as directrix is given by the equation

$$y^2 = 4ax \quad [22]$$

Parabola has eccentricity $e=1$. [22]

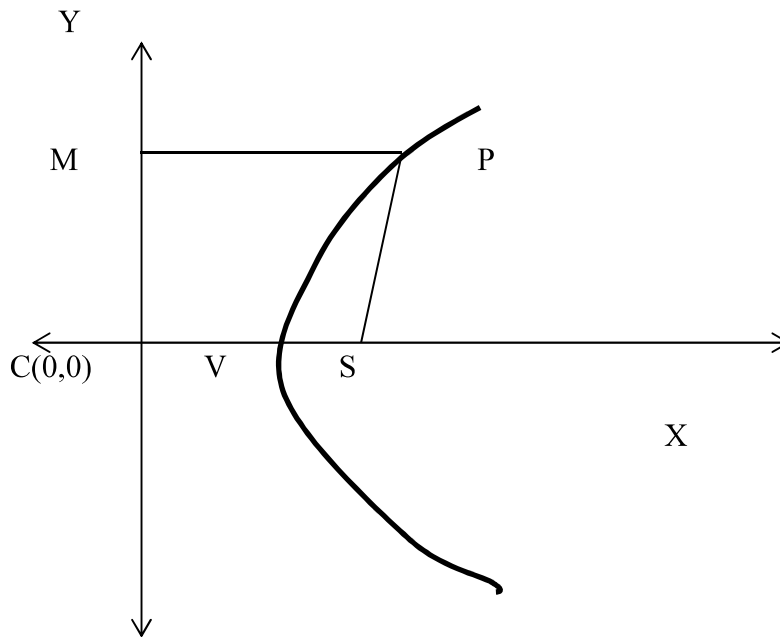
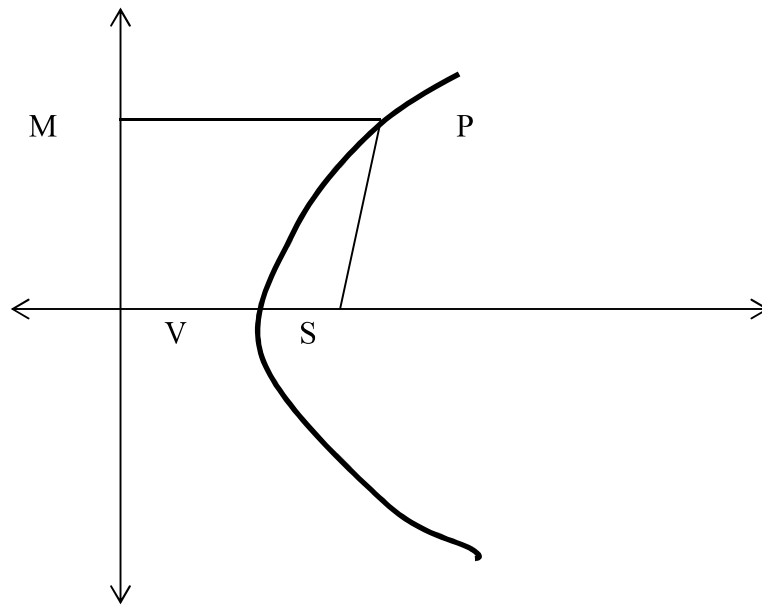


Fig.12.1 Parabola With directrix as Y axis

Equation of a Parabola having $Ax+By+C=0$ as the directrix is given by

$$(x-v_x)^2 + (y-v_y)^2 - \left[\frac{(Ax + By + C)^2}{A^2 + B^2} \right] = 0 \quad [22]$$



$$Ax+By+C=0$$

Fig.12.2 Parabola With directrix as $Ax+By+C=0$

Where $PM^2 = \frac{(Ax+By+C)^2}{A^2+B^2}$ is the perpendicular length squared from point P to point M on the directrix $Ax+By+C=0$. [22]

CHAPTER 13

MODIFICATION OF FCM CLUSTERING TO ENFORCE PARABOLA

The problem in our hand is to modify Fuzzy C Means Clustering [9] algorithm cost function to enforce Parabola shape on clusters in 2 dimensional space. As it was previously stated that Fuzzy C means Clustering [9] algorithm predominantly enforces spherical shape on clusters [4] because of the cost function which predominantly uses l_2 norm [3] [8] to compute distance between centroid and a data point and the norm utilized has the form of equation of a circle's locus which has already been shown and with the help of some simple manipulations it can be understood very well.

In order to solve this problem we take the same approach which we have taken while deriving the update equation for ellipse i.e. using the concept of shape function [10]. We have seen that ellipse has a eccentricity greater than 0 and less than 1 i.e. the distance between focus and data point and perpendicular distance between data point and directrix is not the same, they bear a ratio which is the eccentricity but for parabola eccentricity is 1 i.e. the distances previously are equal. At first a directrix approach where the directrix was chosen as the y axis was developed and the cost function was designed in such a way that the update equation would be easily derivable. His method did not work and membership function values were imaginary. Secondly another cost function was developed where the directrix were chosen as both x axis and y axis to make the function much more independent. In ellipse we have simply used the locus equation lhs without equating with 1, it was done to solve the problem of imaginary number membership values and it was simple and intuitive and was more or less independent because one of the semi axis length is greater than the other, the one greater automatically becomes semi major and the other semi minor and the ellipse automatically orients itself about the x axis or y axis. Any other orientation can be easily detected because of the fuzzy set concept any data point near the enforced ellipse would automatically have higher membership function value this concept helped to reduce the number of variables required which reduced the number of update equations. Such is the utility of fuzzy sets.

Unfortunately the second approach for parabola did not work as well as imaginary number membership value was computed.

The path to choose then was to use the distance concept i.e. distance between focus and data point and perpendicular distance between focus and directrix are equal. Here both side squared form is utilized.

13.1 Cost Function And Derivation Of Update Equations

Let $\mu_i(\vec{x}_p)$ be the fuzzy membership function of i^{th} cluster and p^{th} data point. It can have value in the range $0 < \mu_i(\vec{x}_p) < 1$.

α is the fuzziness parameter which has the range of permissible values of $1 < \alpha < \infty$.

$\vec{x}_p = (x_{xp}, x_{yp}) \forall p=1$ to n are p 2-dimensional data points.

$\vec{v}_i = (v_{xi}, v_{yi}) \forall i=1$ to m are 2-dimensional focuses of i^{th} cluster.

A_i, B_i and C_i are the constants of i^{th} directrix equation $A_i x_{xp} + B_i x_{yp} + C_i = 0$

Let the cost function for enforcing Parabola cluster geometry in the given 2-dimensional dataset be given by:

$$J = \sum_{i=1}^m \sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha \left[(x_{xp} - v_{xi})^2 + (x_{yp} - v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right] \dots eqn(i)$$

With the constraint

$$\sum_{i=1}^m \mu_i(\vec{x}_p) = 1 \dots eqn(ii)$$

Converting the above equation from constrained [8] to unconstrained [8] form by applying Lagrange multiplier technique [8] we get :

$$F = \sum_{i=1}^m \sum_{p=1}^n \mu_i(\vec{x}_p)^\alpha \left[(x_{xp} - v_{xi})^2 + (x_{yp} - v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right] - \sum_{p=1}^n \lambda_p \left[\sum_{i=1}^m \mu_i(\vec{x}_p) - 1 \right] \dots eqn(iii)$$

Where λ_p is the Lagrange multiplier [8]. F is the unconstrained cost function form of J .

In order to find the unknowns we need to minimize eqn(iii) we do this by using minimization criterion for a function using derivatives i.e $\frac{d}{dx}f(x) = 0$. Since there are more than one unknown variables in eqn(iii) we will need to use partial derivative operator.

Our goal here is to find values for the unknown variables such that the unconstrained [8] cost function F is minimized.

Finding v_{xi}

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to v_{xi}

$$\frac{\partial F}{\partial v_{xi}} = 0$$

$$\Rightarrow - \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha 2. (x_{xp} - v_{xi}) = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha (x_{xp} - v_{xi}) = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{xp} - \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha v_{xi} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha v_{xi} = \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{xp}$$

$$\Rightarrow v_{xi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{xp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha} \dots eqn(iv)$$

$\therefore v_{xi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{xp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha}$ is the update equation for finding v_{xi} .

Finding v_{yi}

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to v_{yi}

$$\frac{\partial F}{\partial v_{yi}} = 0$$

$$\Rightarrow - \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha 2. (x_{yp} - v_{yi}) = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha (x_{yp} - v_{yi}) = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp} - \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha v_{yi} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha v_{yi} = \sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}$$

$$\Rightarrow v_{yi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha} \dots eqn(v)$$

$$\therefore v_{yi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha} \text{ is the update equation for finding } v_{yi}.$$

Finding C_i

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to C_i

$$\frac{\partial F}{\partial C_i} = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha \left[-\frac{2(A_i x_{xp} + B_i x_{yp} + C_i)}{A_i^2 + B_i^2} \right] = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)}{A_i^2 + B_i^2} \right] = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha \left[\frac{(A_i x_{xp} + B_i x_{yp})}{A_i^2 + B_i^2} \right] + \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha \left[\frac{C_i}{A_i^2 + B_i^2} \right] = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha (A_i x_{xp} + B_i x_{yp}) + \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha C_i = 0$$

$$\Rightarrow \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha C_i = - \sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha (A_i x_{xp} + B_i x_{yp})$$

$$\Rightarrow C_i = \frac{-\sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha (A_i x_{xp} + B_i x_{yp})}{\sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha} \dots eqn(vi)$$

$$\therefore C_i = \frac{-\sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha (A_i x_{xp} + B_i x_{yp})}{\sum_{p=1}^n \mu_i(\bar{x}_p)^\alpha} \text{ is the update equation for finding } C_i.$$

Finding λ_p

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to λ_p

$$\frac{\partial F}{\partial \lambda_p} = 0$$

$$\Rightarrow - \left[\sum_{i=1}^m \mu_i(\vec{x_p}) - 1 \right] = 0$$

$$\Rightarrow \sum_{i=1}^m \mu_i(\vec{x_p}) - 1 = 0$$

$$\Rightarrow \sum_{i=1}^m \mu_i(\vec{x_p}) = 1 \dots eqn(vii)$$

\therefore on solving $\frac{\partial F}{\partial \lambda_p} = 0$ we get $\sum_{i=1}^m \mu_i(\vec{x_p}) = 1$ this is required to find the update equation of the $\mu_i(\vec{x_p})$ unknown.

Finding $\mu_i(\overrightarrow{x_p})$

Using the concept of minima and partial differentiation of eqn(iii) i.e. F with respect to $\mu_i(\overrightarrow{x_p})$

$$\frac{\partial F}{\partial \mu_i(\overrightarrow{x_p})} = 0$$

$$\Rightarrow \alpha \mu_i(\overrightarrow{x_p})^{\alpha-1} \left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right] - \lambda_p = 0$$

$$\Rightarrow \alpha \mu_i(\overrightarrow{x_p})^{\alpha-1} \left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right] = \lambda_p$$

$$\Rightarrow \mu_i(\overrightarrow{x_p})^{\alpha-1} \left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right] = \frac{\lambda_p}{\alpha}$$

$$\Rightarrow \mu_i(\overrightarrow{x_p})^{\alpha-1} = \frac{\lambda_p}{\alpha} \frac{1}{\left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]}$$

$$\Rightarrow \mu_i(\overline{x_p}) = \left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} \frac{1}{\left[(x_{xp} - v_{xi})^2 + (x_{yp} - v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}} .eqn(viii)$$

Substituting eqn(viii) in eqn(vii) we get:

$$\Rightarrow \sum_{i=1}^m \left[\left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} \frac{1}{\left[(x_{xp} - v_{xi})^2 + (x_{yp} - v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}} \right] = 1$$

$$\Rightarrow \left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} \sum_{i=1}^m \left[\frac{1}{\left[(x_{xp} - v_{xi})^2 + (x_{yp} - v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}} \right] = 1$$

$$\Rightarrow \left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}} = \frac{1}{\sum_{i=1}^m \left[\frac{1}{\left[(x_{xp} - v_{xi})^2 + (x_{yp} - v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}} \right]} \dots eqn(ix)$$

From eqn(ix) we get the value of $\left[\frac{\lambda_p}{\alpha} \right]^{\frac{1}{\alpha-1}}$ which is one of the terms in eqn(viii). We can substitute eqn(ix) in eqn(viii).

On substituting eqn(ix) in eqn(viii) we get:

$$\Rightarrow \mu_i(\vec{x_p}) = \frac{\frac{1}{\left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}}}{\sum_{i=1}^m \left[\frac{1}{\left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}} \right]} \dots eqn(x)$$

$$\therefore \mu_i(\vec{x_p}) = \frac{\frac{1}{\left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}}}{\sum_{i=1}^m \left[\frac{1}{\left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right] \right]^{\frac{1}{\alpha-1}}} \right]} \text{ is the update equation for } \mu_i(\vec{x_p}) .$$

In a summary in the above derivations we have developed four update equations which will be used in the algorithms for parabola cluster detection.

The update equations after solving eqn(iii) are as follows:

$$v_{xi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{xp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha}$$

$$v_{yi} = \frac{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha x_{yp}}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha}$$

$$C_i = \frac{-\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha (A_i x_{xp} + B_i x_{yp})}{\sum_{p=1}^n \mu_i(\vec{x_p})^\alpha}$$

$$\mu_i(\vec{x_p}) = \frac{\frac{1}{\left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right]^{\frac{1}{\alpha-1}} \right]^{\frac{1}{\alpha-1}}}}{\sum_{i=1}^m \left[\frac{1}{\left[(x_{xp}-v_{xi})^2 + (x_{yp}-v_{yi})^2 - \left[\frac{(A_i x_{xp} + B_i x_{yp} + C_i)^2}{A_i^2 + B_i^2} \right]^{\frac{1}{\alpha-1}} \right]^{\frac{1}{\alpha-1}}} \right]}$$

13.2 Modified FCM Algorithm To Detect Parabola

Using the derived update equations from the modified cost function, we develop the modified Fuzzy C Means clustering[9] algorithm to detect parabola. The modified algorithm should impose parabola geometry on clusters provided the space is 2 dimensional i.e. the data points have x and y coordinates. The algorithm is given below

Algorithm: Modified Fuzzy C Means Parabola Clustering

Input: $\alpha, \vec{x}_p \forall p = 1 \text{ to } n$

Output: $\vec{v}_i \forall i = 1 \text{ to } m$

Step 1. Assign $\mu_i(\vec{x}_p) \in [0,1] \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$.

Step 2. Select values for A_i and $B_i \forall i = 1 \text{ to } m$

Step 3. Compute v_{xi} and $v_{yi} \forall i = 1 \text{ to } m$

Step 4. Compute $C_i \forall i = 1 \text{ to } m$

Step 5. Compute $\mu_i(\vec{x}_p) \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$

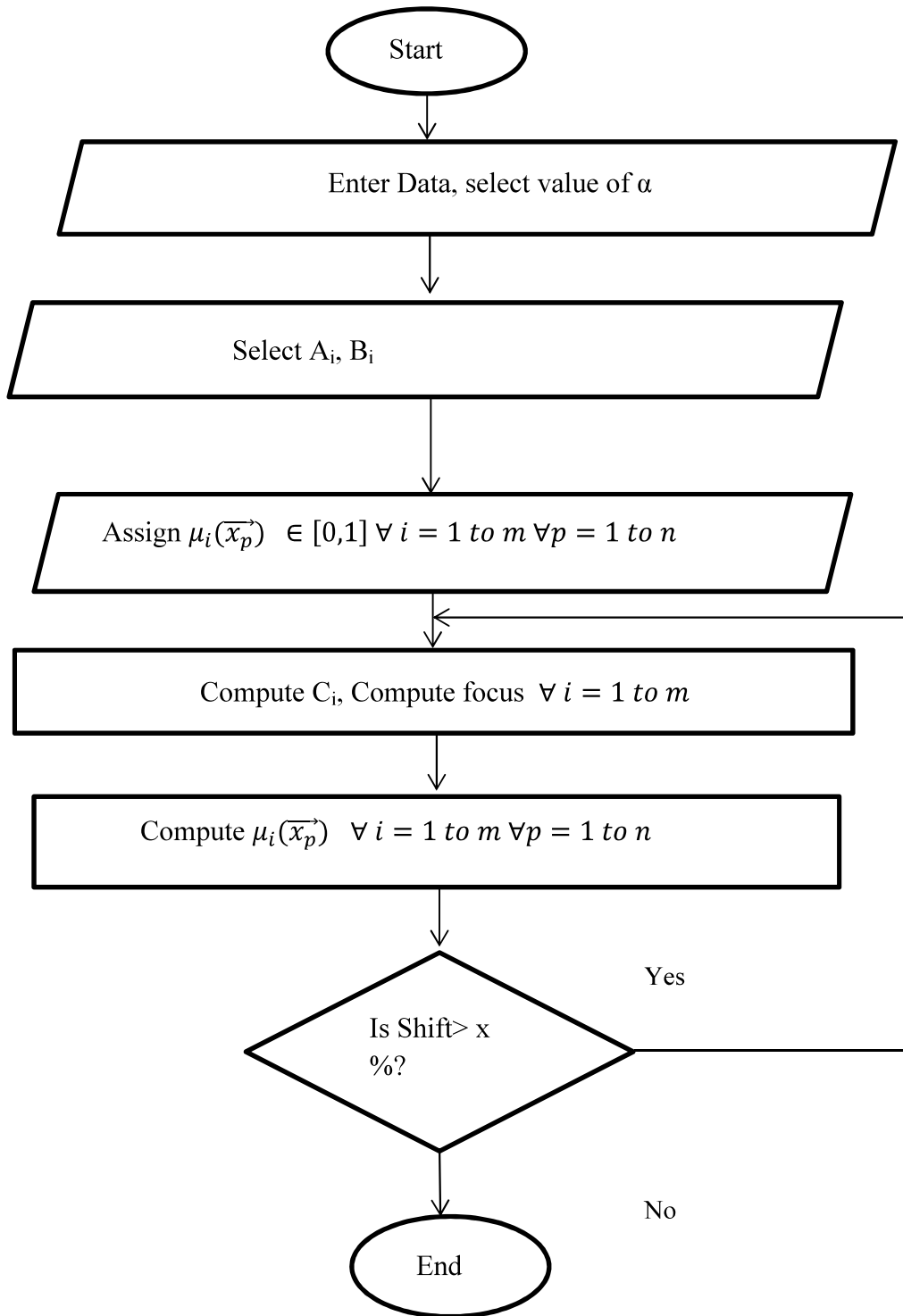
Step 6. Goto Step 3 until v_{xi} and $v_{yi} \forall i = 1 \text{ to } m$ doesn't change much.

Step 7. Print the cluster focuses along with clusters.

Step 8. End

The above algorithm has been implemented on 100 random data points generated using random number generator and multiplied by 100. The coding was done in octave and α was taken as 3. The number of clusters to be detected was 3. Here the variables used have the same meaning as in that of the previous section.

13.3 Flowchart



13.4 Code

```
function e=mu_x_mod(data,alpha,a,b,Y,B,C)

mu=zeros(3,100);

for i= 1: 3

for j= 1: 100

s=(1/(((data(j,1)-a(i))^2+(data(j,2)-b(i))^2-
(Y(i)*data(j,1)+B(i)*data(j,2)+C(i))^2/(Y(i)^2+B(i)^2))))^(1/(alpha-1));

k=intermediate(j,a,b,Y,B,C,data,alpha);

mu(i,j)=s/k;

end

end

e=mu;

endfunction

function e=intermediate(j,a,b,Y,B,C,data,alpha)

s=0;

for i= 1: 3

s=s+(1/(((data(j,1)-a(i))^2+(data(j,2)-b(i))^2-
(Y(i)*data(j,1)+B(i)*data(j,2)+C(i))^2/(Y(i)^2+B(i)^2))))^(1/(alpha-1));

end

e=s;

endfunction

function e=a_mod(mu_x,alpha,data)

d2=reshape(data(:,1),1,100);

a1=sum((mu_x(1, :).^alpha).*d2)/sum(mu_x(1, :).^alpha);

a2=sum((mu_x(2, :).^alpha).*d2)/sum(mu_x(2, :).^alpha);

a3=sum((mu_x(3, :).^alpha).*d2)/sum(mu_x(3, :).^alpha);
```

```

e=[a1;a2;a3];
endfunction

function e=b_mod(mu_x,alpha,data)

d2=reshape(data(:,2),1,100);

b1=sum((mu_x(1, :).^alpha).*d2)/sum(mu_x(1, :).^alpha);
b2=sum((mu_x(2, :).^alpha).*d2)/sum(mu_x(2, :).^alpha);
b3=sum((mu_x(3, :).^alpha).*d2)/sum(mu_x(3, :).^alpha);
e=[b1;b2;b3];
endfunction

```

```

function e=C_mod(mu_x,data,alpha,Y,B)

d1=reshape(data(:,1),1,100);
d2=reshape(data(:,2),1,100);

C1=-sum((mu_x(1, :).^alpha).*(d1*Y(1)+d2*B(1)))/sum(mu_x(1, :).^alpha);
C2=-sum((mu_x(2, :).^alpha).*(d1*Y(2)+d2*B(2)))/sum(mu_x(2, :).^alpha);
C3=-sum((mu_x(3, :).^alpha).*(d1*Y(3)+d2*B(3)))/sum(mu_x(3, :).^alpha);
e=[C1;C2;C3];
endfunction

Y=rand(1,3)*-100;

B=rand(1,3)*100;

a=a_mod(mu_x,alpha,data);
b=b_mod(mu_x,alpha,data);
C=C_mod(mu_x,data,alpha,Y,B);
mu_x=mu_x_mod(data,alpha,a,b,Y,B,C);
while(1)

```

```

a1=a_mod(mu_x,alpha,data);
b1=b_mod(mu_x,alpha,data);
C1=C_mod(mu_x,data,alpha,Y,B);
mu_x=mu_x_mod(data,alpha,a1,b1,Y,B,C1);
i=i+1;
if(abs(a-a1)./a*100<=ones(1,3) && abs(b-b1)./b*100<=ones(1,3))
break;
endif
a=a1;
b=b1;
endwhile
a
b

```

13.5 Variable Description

Variable	Description
r, e	Function output variables
data	Vector of size 100x2 to store data
d1,d2	Vectors of size 1x100 to store x and y values respectively
mu_x	Fuzzy membership function
alpha	Fuzziness parameter i.e. α
a,b	Vectors of size 3x1 storing Focus x component and y component respectively
i,j	Counters
s,k	To perform summation in mu_x_mod
a1-a3	Normal variables to store computed x component values of focuses
b1-b3	Normal variables to store computed y component values of focuses
a1, b1	Vectors of size 3x1 to store current computed focus x and y components respectively
C	Vector of size 3x1 to store directrix equation parameter as in $Ax+By+C=0$
C1	Vector of size 3x1 to compute directrix equation parameter as in $Ax+By+C=0$
C1-C3	Normal variables to store computed directrix equation parameters as in $Ax+By+C=0$
Y	It is the 1x3 vector which stores parameter 'A' values of directrix equation $Ax+By+C=0$
B	It is the 1x3 vector which stores parameter 'B' values of directrix equation $Ax+By+C=0$

13.6 Output

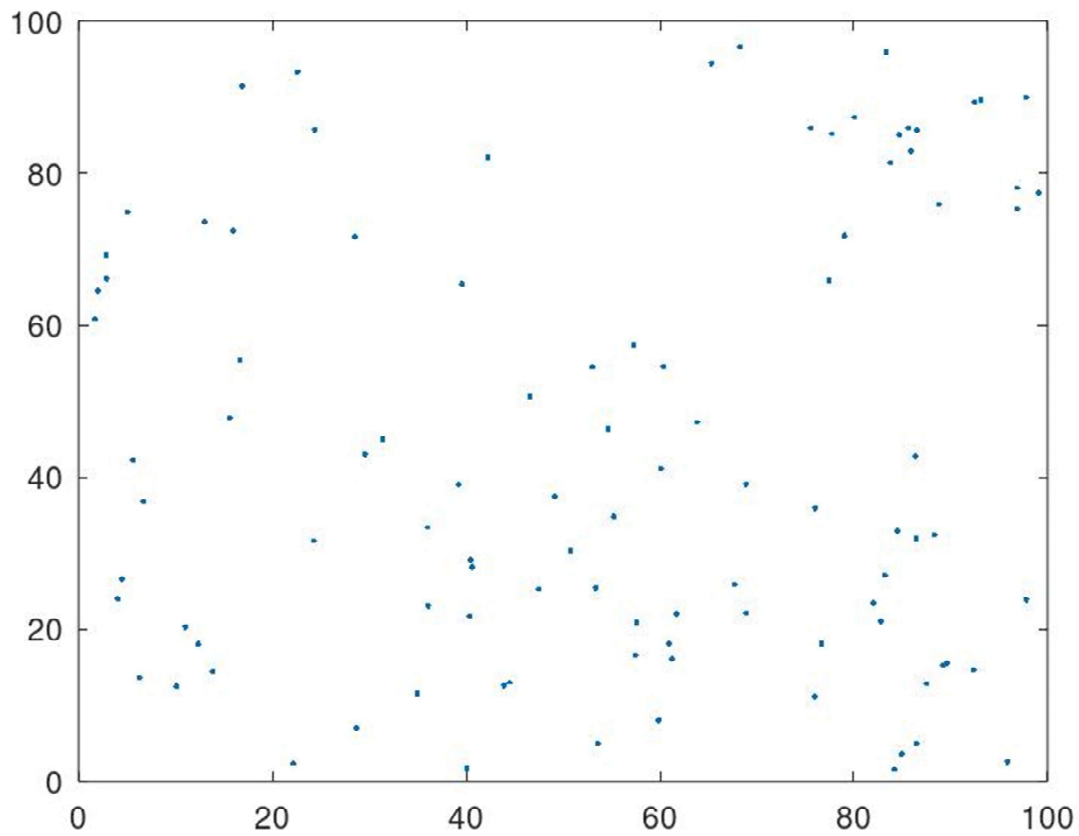


Fig.13.6.1 Graph showing random data points selected using random number generator

In the above graph blue dots represents data points. The data was generated using random number generator and multiplying each value with 100.

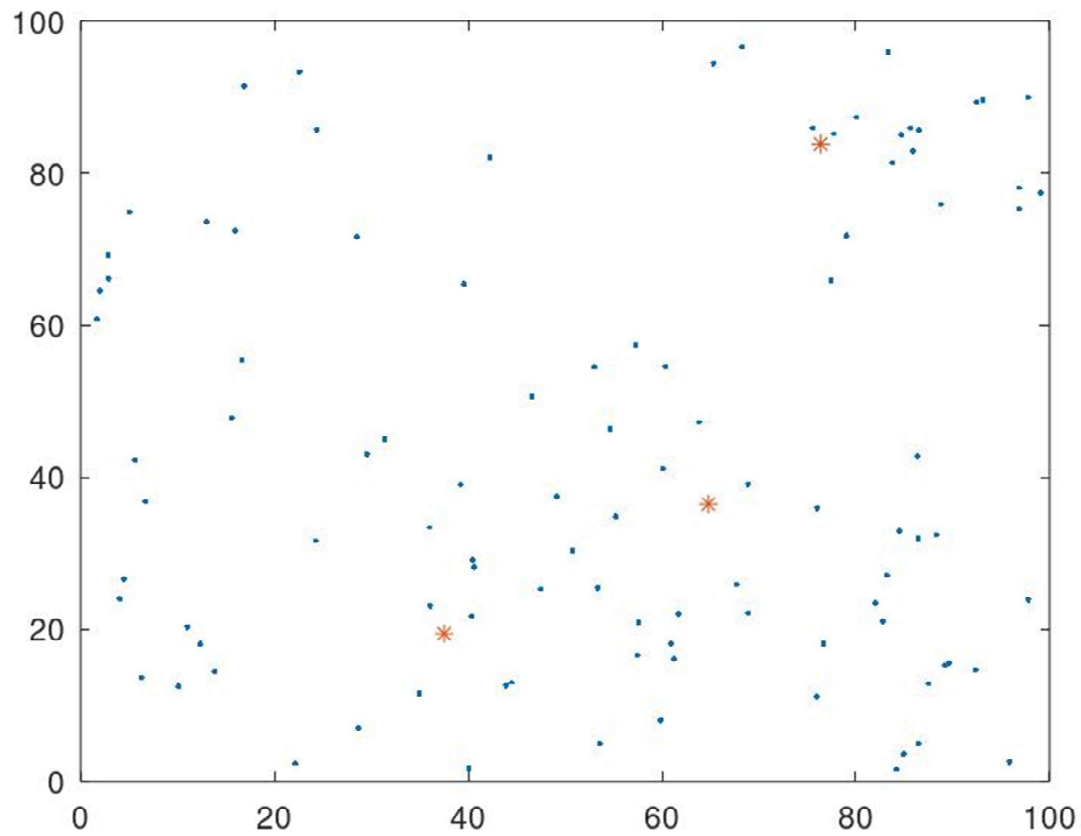


Fig.13.6.2 Graph showing output of the modified algorithm

In the above graph blue dots represents data points. Orange asterix represents focuses of detected clusters.

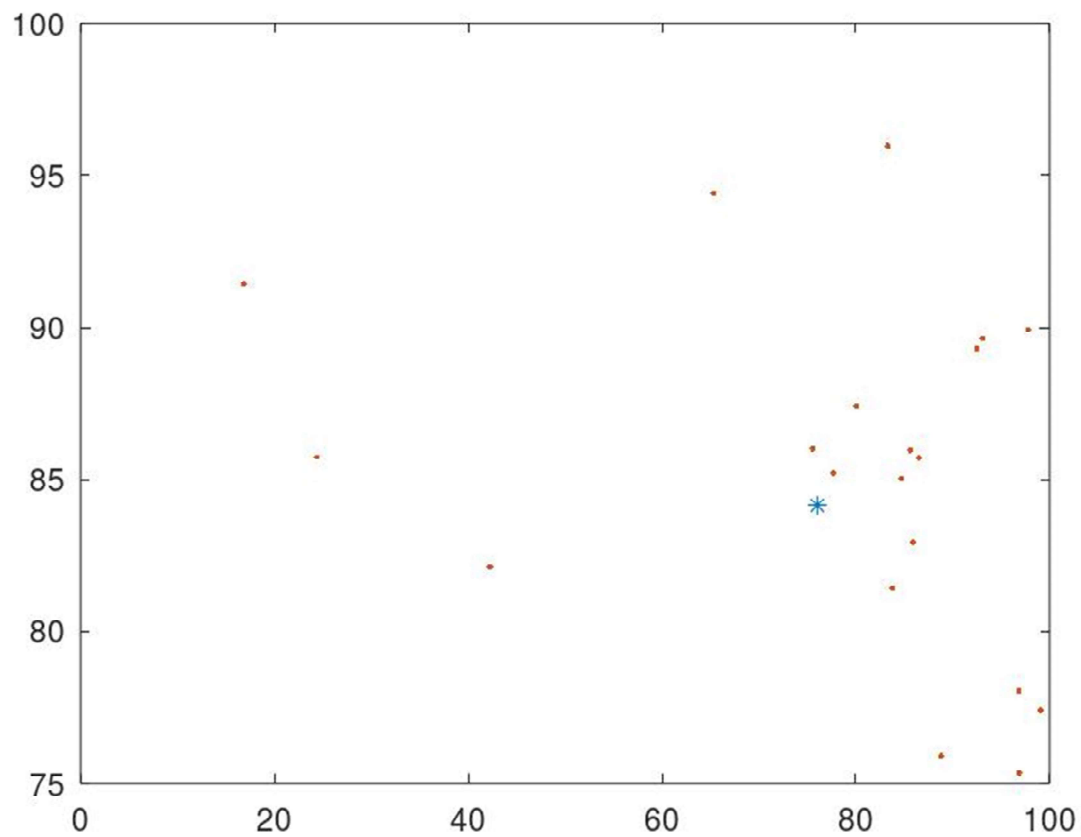


Fig. 13.6.3 Graph showing one of the clusters

In the above graph orange dots represents data points belonging to a cluster selected using a threshold value. The blue asterix represents focus of the cluster.

13.7 Discussion Of Output

The algorithm along with the cost function does in fact tries to impose a parabola shape on clusters in 2 dimensional space. But there are some problems with this approach. The first and foremost problem is that there is no computationally efficient way to determine dataset dependent directrix equations. It was stated in the ellipse section that the semi major/minor axes lengths doesn't have any update equations but the problem here is different. Here directrix equation coefficients can be made completely dataset dependent i.e. it reflects the data. Even though it required some initial values but the update equations derived were extremely complex and would have increased the complexity manifold. So the method was totally bypassed and only one coefficient was made data dependent. This has created the problem that the proper directrix equation may not be used in the process and the used one has a high chance to be drastically different from the real ones.

There is a solution to the problem which will not completely fix the situation but will fix it to a great extent if properly implemented. Here we alter (increment) the coefficients in a step wise manner by a constant value. This will provide the benefit of providing new directrix equation in each run and we might get close to the actual equations which will yield better clusters than that of the previous algorithm. This will give variations in the orientation and positions of parabolas.

13.8 Fuzzy C Means Parabola Clustering of Variable Orientation and position

Using the derived update equations for parabola clustering and then merging the concepts of Fuzzy C Means Clustering, we get the algorithm of Fuzzy C Means Parabola clustering of Variable orientation and position in 2 dimensional space.

Algorithm: Fuzzy C Means Parabola Clustering of variable orientation and positions

Input: $\alpha, \vec{x_p} \forall p = 1 \text{ to } n$

Output: $\vec{v_{iK}} \forall i = 1 \text{ to } m \forall k = 1 \text{ to } 11.$

Step 1. Select values for A_i and $B_i \forall i = 1 \text{ to } m.$

Step 2. Set $Aold_i = A_i * 0.1$ and $Bold_i = B_i * 0.1 \forall i = 1 \text{ to } m.$

Step 3. For $k=1$ to 10 repeat step 4 to step 11

Step 4. Assign $\mu_i(\vec{x_p}) \in [0,1] \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n.$

Step 5. Compute v_{xik} and $v_{yik} \forall i = 1 \text{ to } m$

Step 6. Compute $C_i \forall i = 1 \text{ to } m$

Step 7. Compute $\mu_i(\vec{x_p}) \forall i = 1 \text{ to } m \forall p = 1 \text{ to } n$

Step 8. Goto Step 4 until v_{xiK} and $v_{yiK} \forall i = 1 \text{ to } m$ doesn't change much.

Step 9. Print the cluster focuses along with clusters

Step 10. Set $A_i = A_i + Aold_i$ and $B_i = B_i + Bold_i \forall i = 1 \text{ to } m$

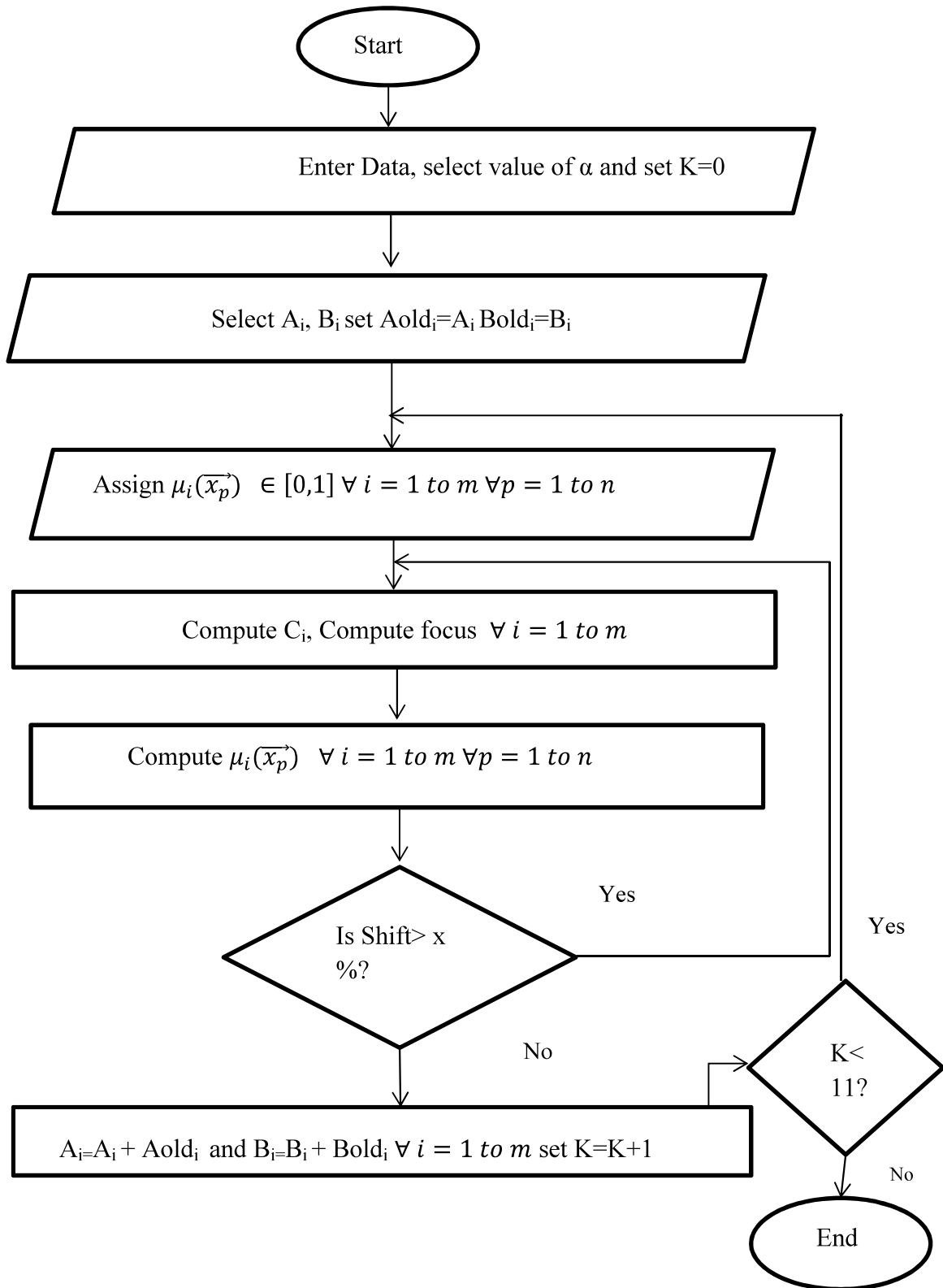
Step 11. $K=K+1$

Step 12. End

The algorithm in effect changes the directrix equations. In the previous algorithm C_i was altered on the basis of dataset but A_i and B_i was not altered i.e. on the basis of A_i, B_i and dataset only C_i was adjusted. In effect, this algorithm adjusts a new directrix for each cluster in every run. So in short the problem of complex update equations and inefficient clustering is solved to some extent.

This code is implemented on artificial datasets where data points are arranged in the form of three parabolas. α was taken as 4.

13.9 Flowchart



13.10 Code

```
function e=a_mod(mu_x,alpha,data)
d2=reshape(data(:,1),1,40);
a1=sum((mu_x(1, :).^alpha).*d2)/sum(mu_x(1, :).^alpha);
a2=sum((mu_x(2, :).^alpha).*d2)/sum(mu_x(2, :).^alpha);
a3=sum((mu_x(3, :).^alpha).*d2)/sum(mu_x(3, :).^alpha);
e=[a1;a2;a3];
endfunction

function e=b_mod(mu_x,alpha,data)
d2=reshape(data(:,2),1,40);
b1=sum((mu_x(1, :).^alpha).*d2)/sum(mu_x(1, :).^alpha);
b2=sum((mu_x(2, :).^alpha).*d2)/sum(mu_x(2, :).^alpha);
b3=sum((mu_x(3, :).^alpha).*d2)/sum(mu_x(3, :).^alpha);
e=[b1;b2;b3];
endfunction

function e=C_mod(mu_x,data,alpha,Y,B)
d1=reshape(data(:,1),1,40);
d2=reshape(data(:,2),1,40);
C1=-sum((mu_x(1, :).^alpha).*(d1*Y(1)+d2*B(1)))/sum(mu_x(1, :).^alpha);
C2=-sum((mu_x(2, :).^alpha).*(d1*Y(2)+d2*B(2)))/sum(mu_x(2, :).^alpha);
C3=-sum((mu_x(3, :).^alpha).*(d1*Y(3)+d2*B(3)))/sum(mu_x(3, :).^alpha);
e=[C1;C2;C3];
endfunction

function e=mu_x_mod(data,alpha,a,b,Y,B,C)
mu=zeros(3,40);
for i= 1: 3
for j= 1: 40
s=(1/((data(j,1)-a(i))^2+(data(j,2)-b(i))^2-
(Y(i)*data(j,1)+B(i)*data(j,2)+C(i))^2/(Y(i)^2+B(i)^2)))^(1/(alpha-
1));
k=intermediate(j,a,b,Y,B,C,data,alpha);
mu(i,j)=s/k;
end
end
e=mu;
endfunction

function e=intermediate(j,a,b,Y,B,C,data,alpha)
s=0;
for i= 1: 3
s=s+ (1/((data(j,1)-a(i))^2+(data(j,2)-b(i))^2-
(Y(i)*data(j,1)+B(i)*data(j,2)+C(i))^2/(Y(i)^2+B(i)^2)))^(1/(alpha-1));
end
e=s;
endfunction
Y=Yo=rand(1,3)*-100
```

```

B=Bo=rand(1,3)*100
j=0;

alpha=4;
i=1;
for K=1:10
mu_x=rand(3,40);
a=a_mod(mu_x,alpha,data);
b=b_mod(mu_x,alpha,data);
C=C_mod(mu_x,data,alpha,Y,B);
mu_x=mu_x_mod(data,alpha,a,b,Y ,B,C);
while(1)
a1=a_mod(mu_x,alpha,data);
b1=b_mod(mu_x,alpha,data);
C1=C_mod(mu_x,data,alpha,Y,B);
mu_x=mu_x_mod(data,alpha,a1,b1,Y,B,C1);
i=i+1;

if(abs(a-a1)./a*100<=ones(1,3) && abs(b-b1)./b*100<=ones(1,3))
break;
endif
a=a1;
b=b1;
endwhile
a
b
Y=Y+0.1*Y0;
B=B+0.1*B0;
end

```

13.11 Variable Description

Variable	Description
r, e	Function output variables
data	Vector of size 40x2 to store data
d1,d2	Vectors of size 1x40 to store x and y values respectively
mu_x	Fuzzy membership function
alpha	Fuzziness parameter i.e. α
a,b	Vectors of size 3x1 storing Focus x component and y component respectively
i,j	Counters
s,k	To perform summation in mu_x_mod
a1-a3	Normal variables to store computed x component values of focuses
b1-b3	Normal variables to store computed y component values of focuses
a1, b1	Vectors of size 3x1 to store current computed focus x and y components respectively
C	Vector of size 3x1 to store directrix equation parameter as in $Ax+By+C=0$
C1	Vector of size 3x1 to compute directrix equation parameter as in $Ax+By+C=0$
C1-C3	Normal variables to store computed directrix equation parameters as in $Ax+By+C=0$
Y	It is the 1x3 vector which stores parameter 'A' values of directrix equation $Ax+By+C=0$
B	It is the 1x3 vector which stores parameter 'B' values of directrix equation $Ax+By+C=0$
Yo	Vector of size 1x3 to store Y variable in it
Bo	Vector of size 1x3 to store B variable in it
K	Counter to maintain run number

13.12 Output

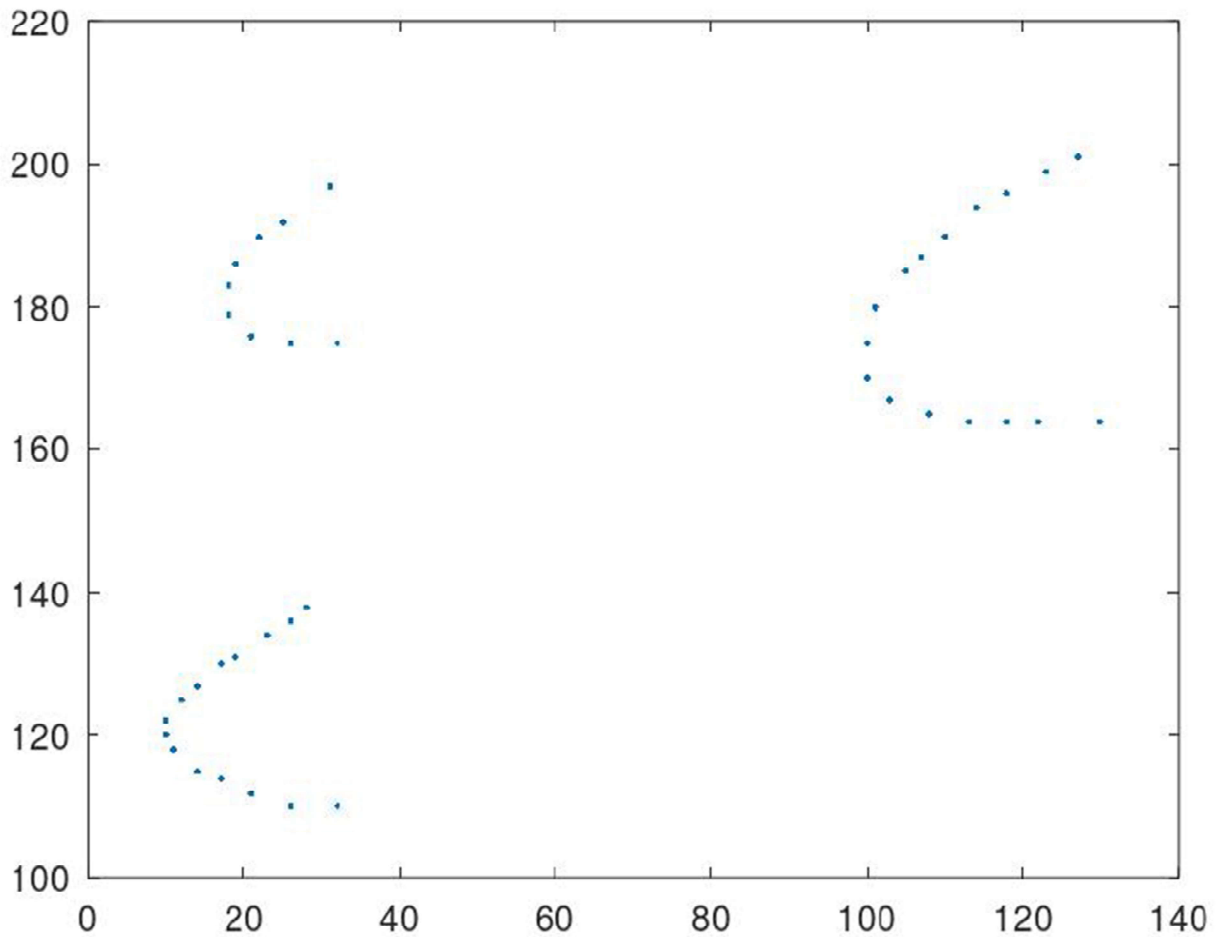


Fig. 13.12.1 Graph showing the artificial data used

The above graph shows data points of the artificial dataset designed intentionally to introduce parabolas. As it is evident from the above graph it does contain three parabolas. The blue dots represent the data points.

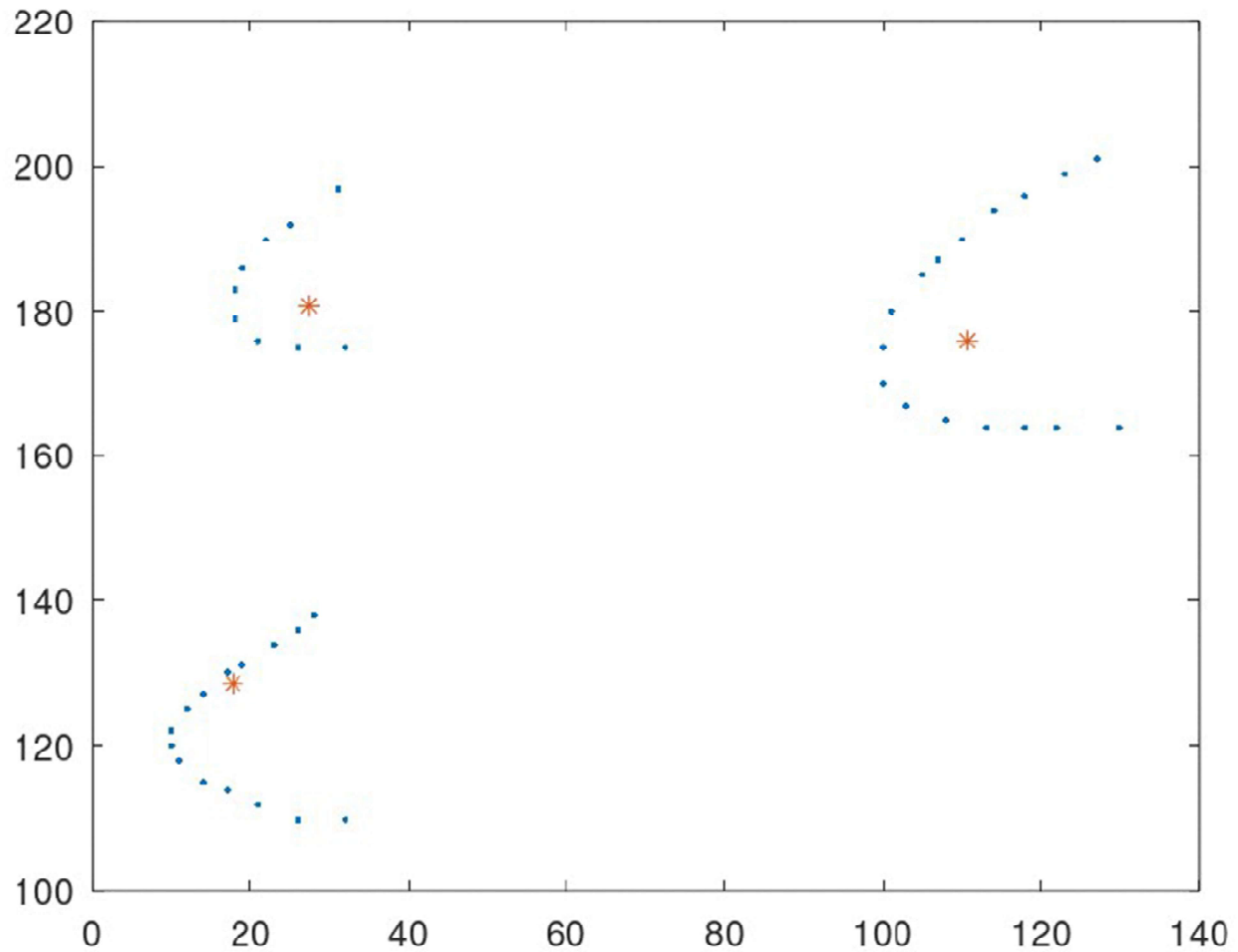


Fig.13.12.2 Graph showing output at the end of 1st run

The above graph corresponds to the output of the 1st run. The blue dot represents data points. The initial coefficients were select randomly and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

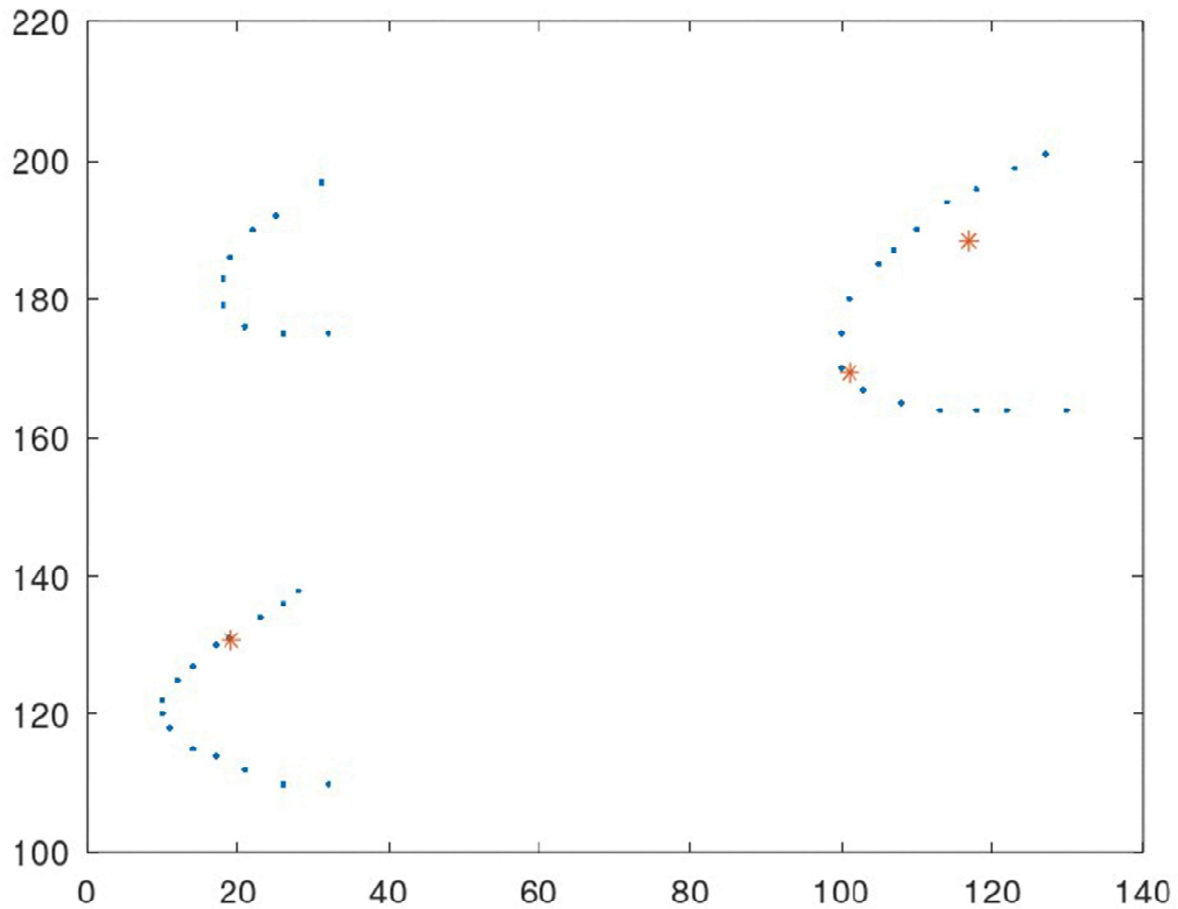


Fig.13.12.3 Graph showing output at the end of 2nd run

The above graph corresponds to the output of the 2nd run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

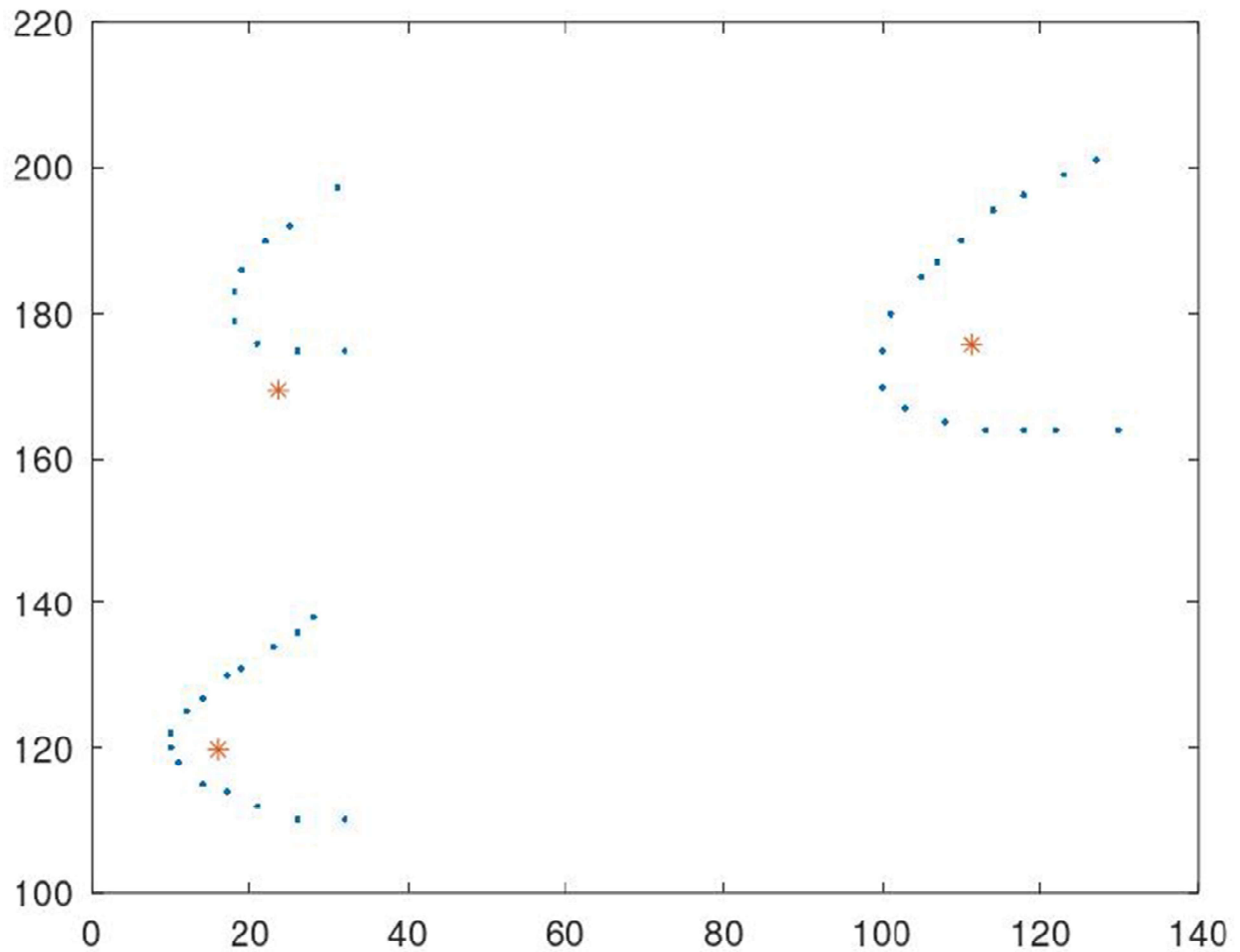


Fig.13.12.4 Graph showing output at the end of 3rd run

The above graph corresponds to the output of the 3rd run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

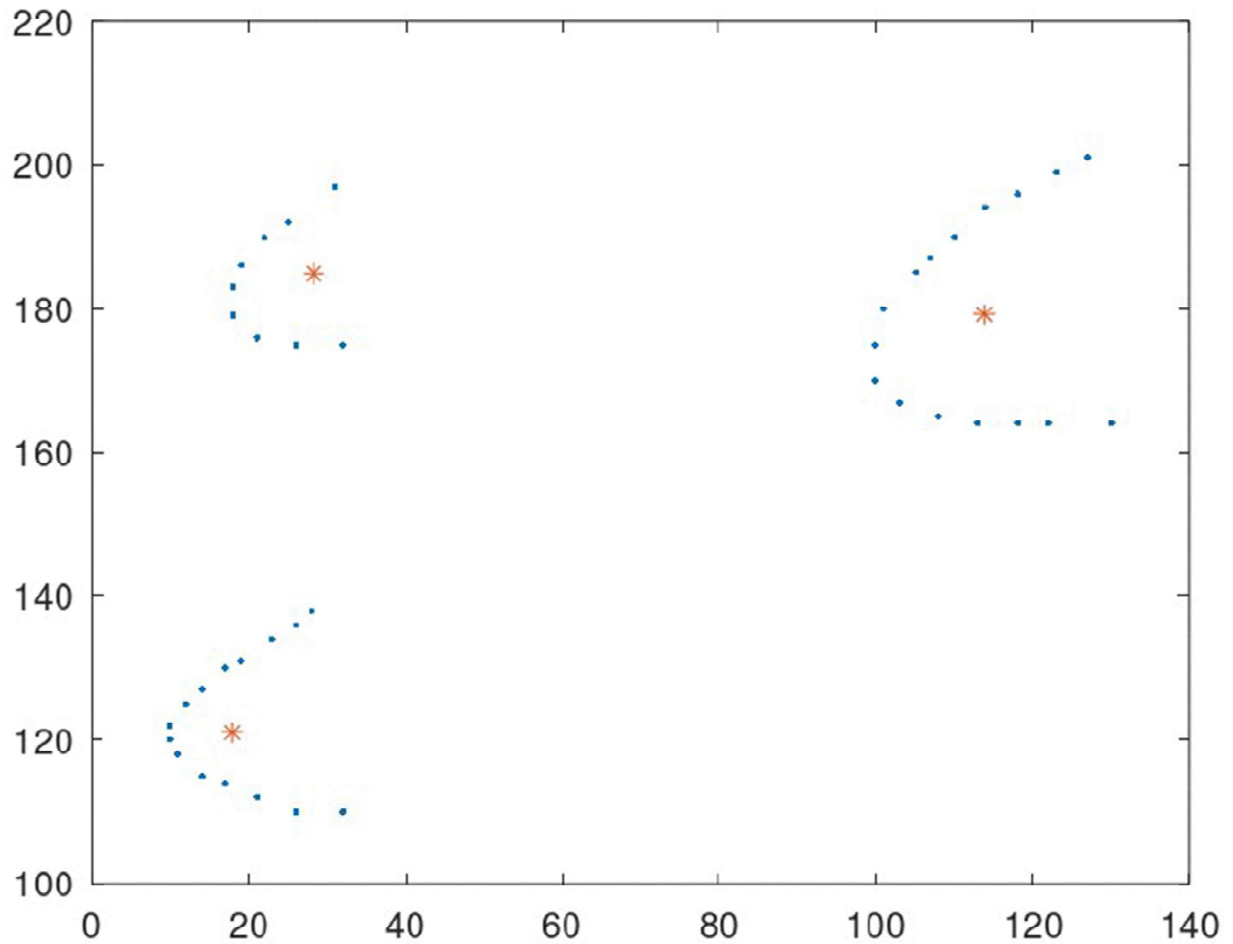


Fig.13.12.5 Graph showing output at the end of 4th run

The above graph corresponds to the output of the 4th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

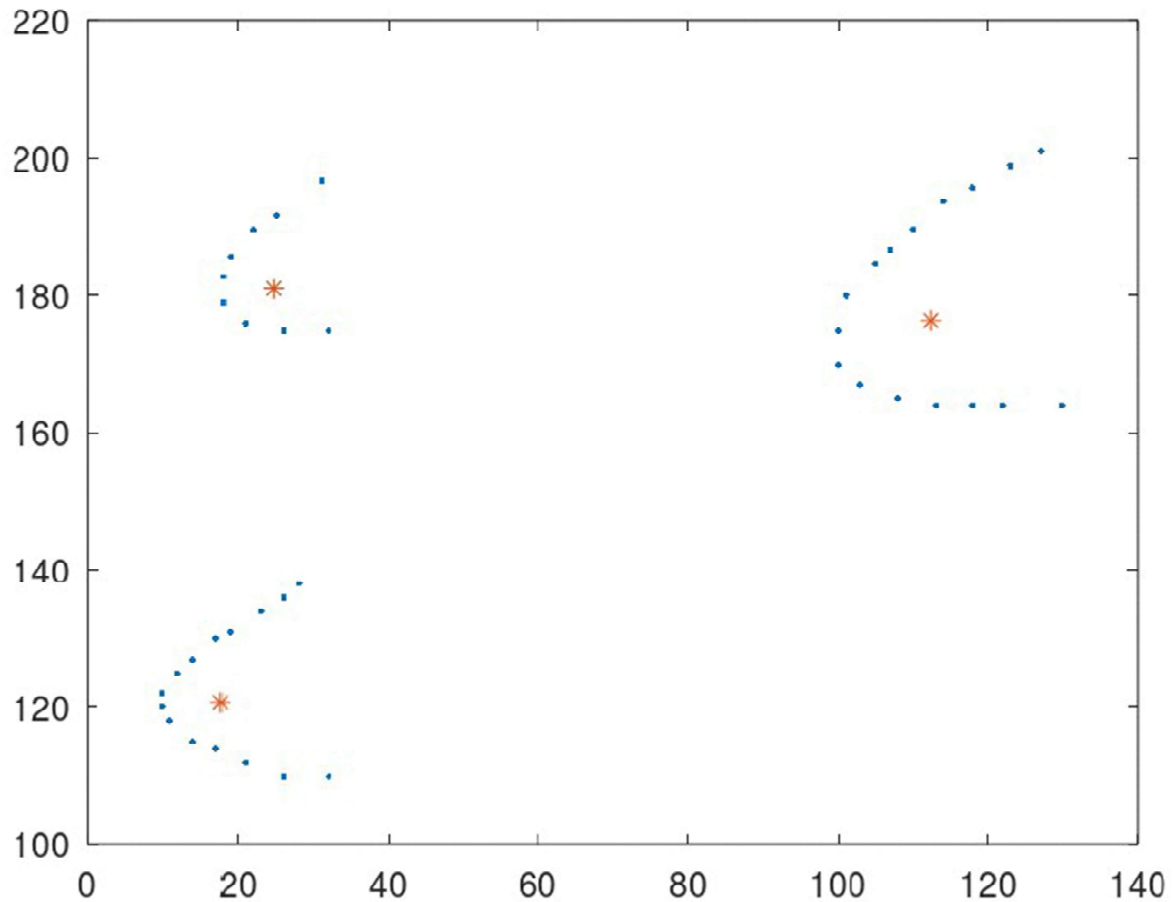


Fig.13.12.6 Graph showing output at the end of 5th run

The above graph corresponds to the output of the 5th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

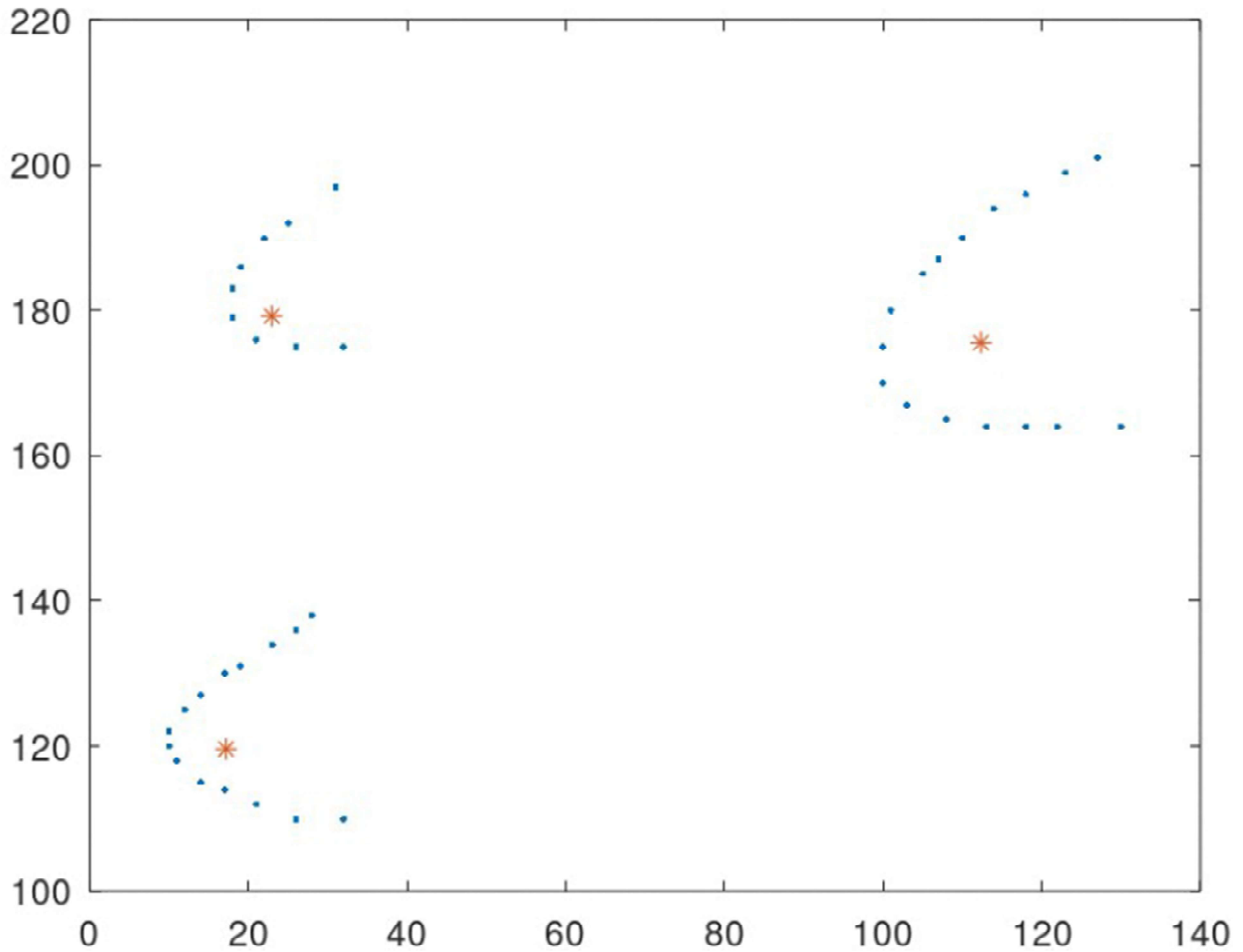


Fig. 13.12.7 Graph showing output at the end of 6th run

The above graph corresponds to the output of the 6th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

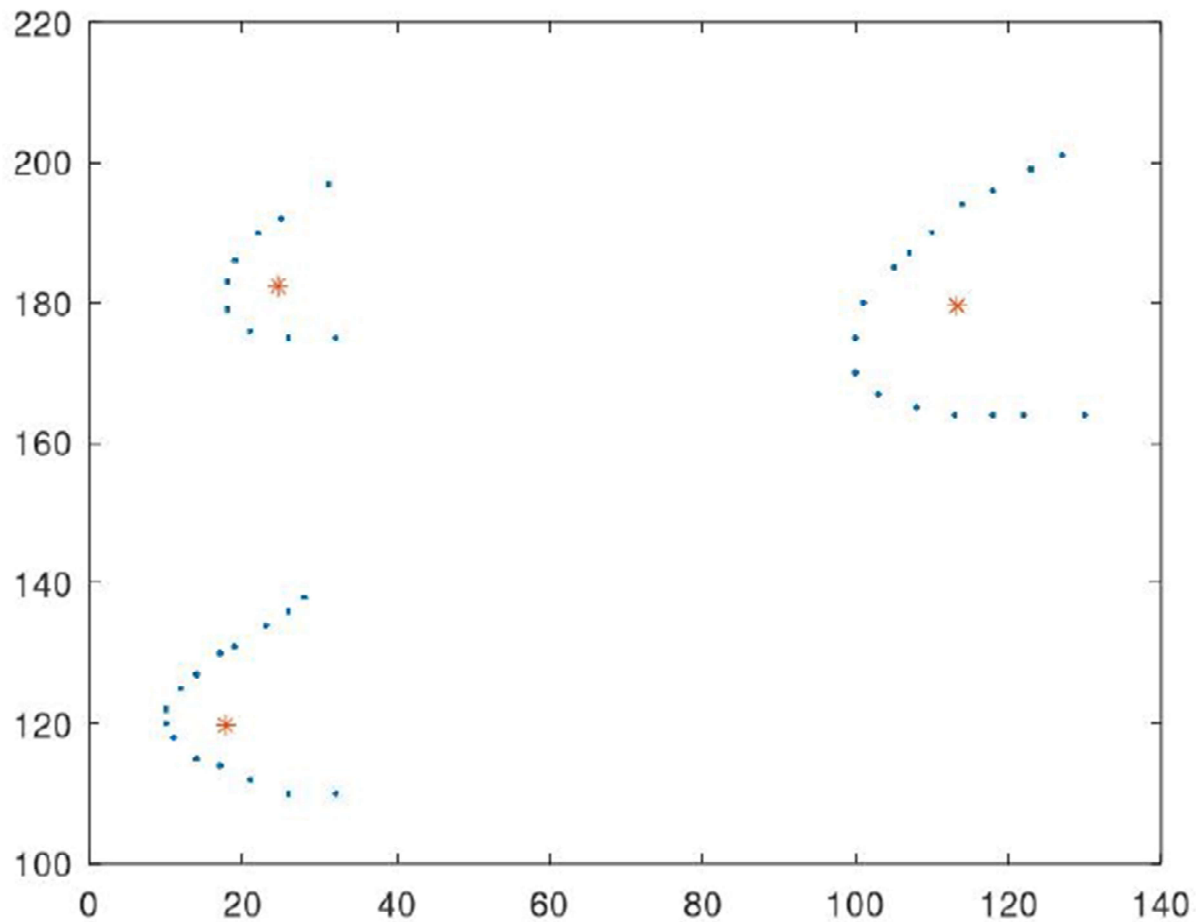


Fig.13.12.8 Graph showing output at the end of 7th run

The above graph corresponds to the output of the 7th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

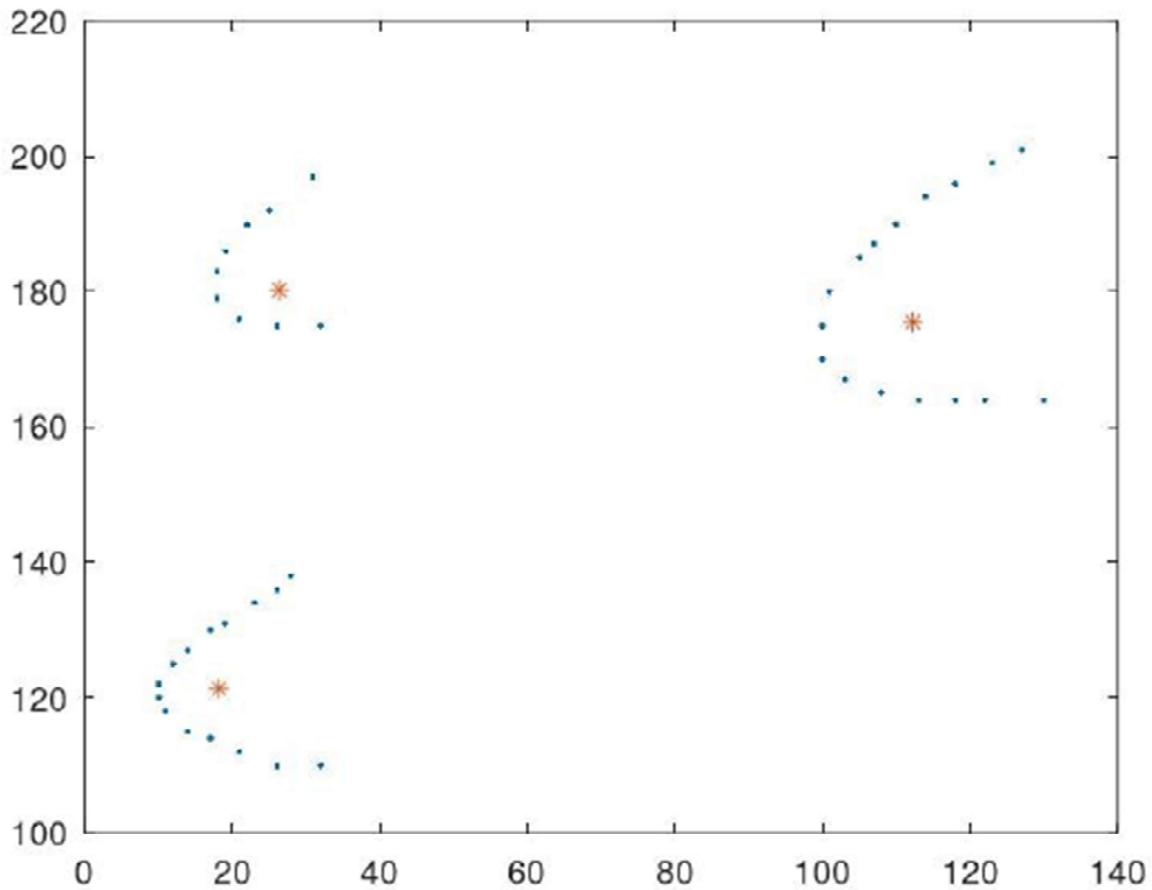


Fig. 13.12.9 Graph showing output at the end of 8th run

The above graph corresponds to the output of the 8th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

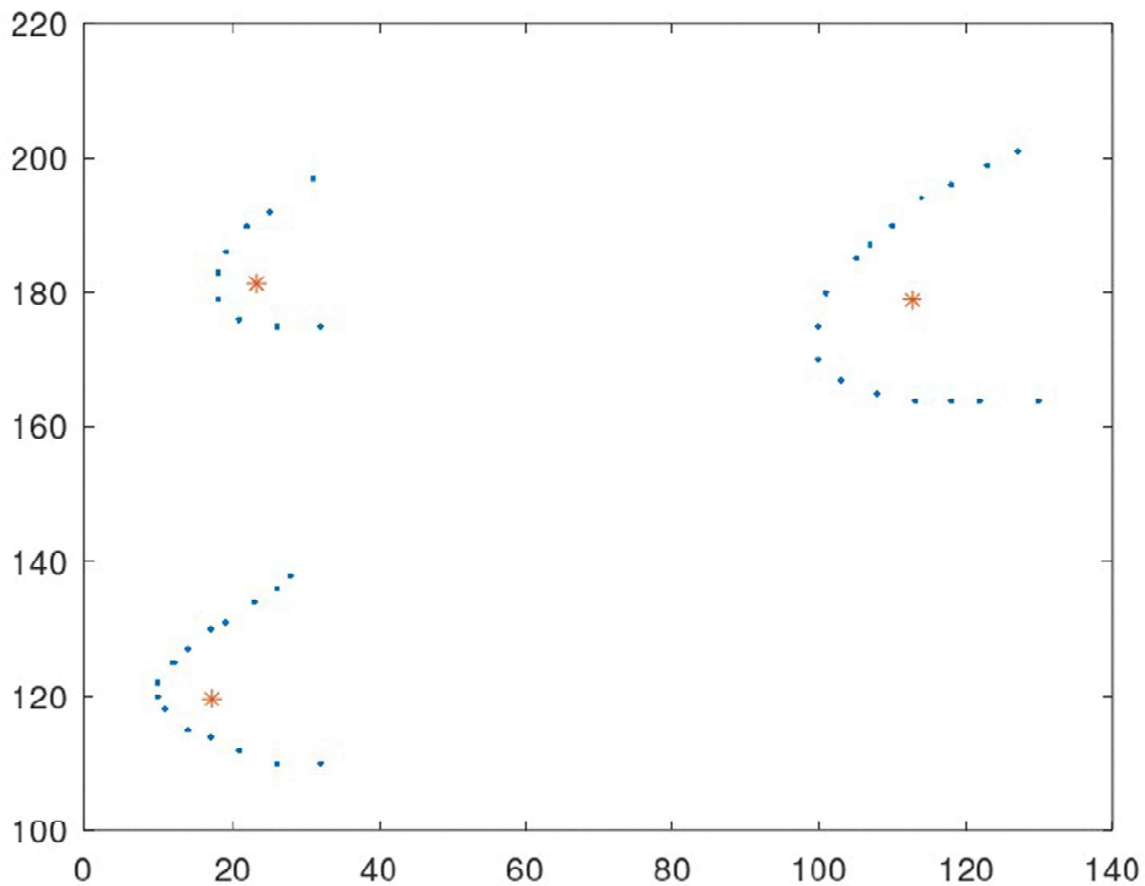


Fig.13.12.10 Graph showing output at the end of 9th run

The above graph corresponds to the output of the 9th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

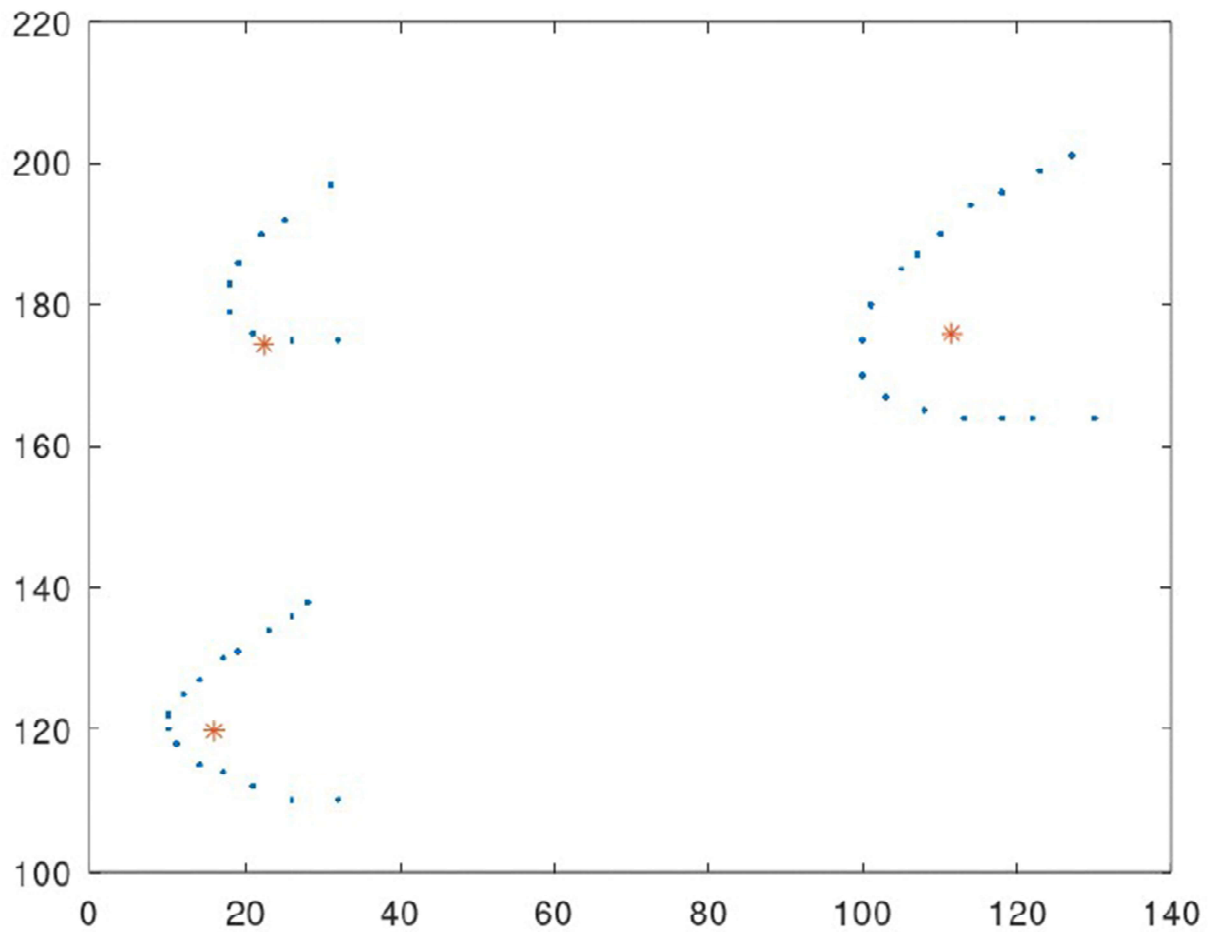


Fig.13.12.11 Graph showing output at the end of 10th run.

The above graph corresponds to the output of the 10th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

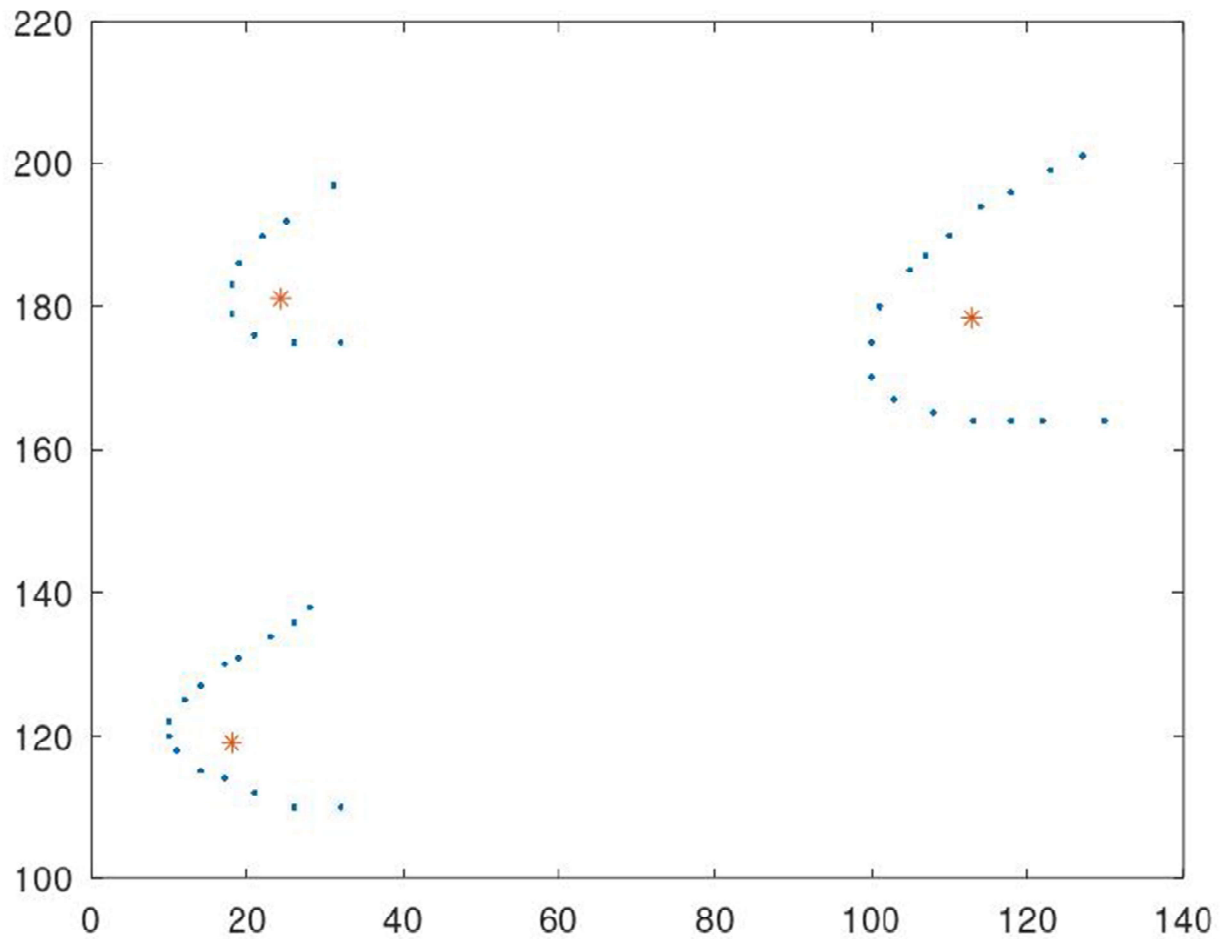


Fig.13.12.12 Graph showing output at the end of 11th run

The above graph corresponds to the output of the 11th run. The blue dot represents data points. The coefficients were formed by adding $0.1 \times \text{initial value}$ to the previous run coefficients and was used along with data set to develop the remaining coefficient and find focuses of clusters. Orange asterix represents focuses of parabola shaped clusters.

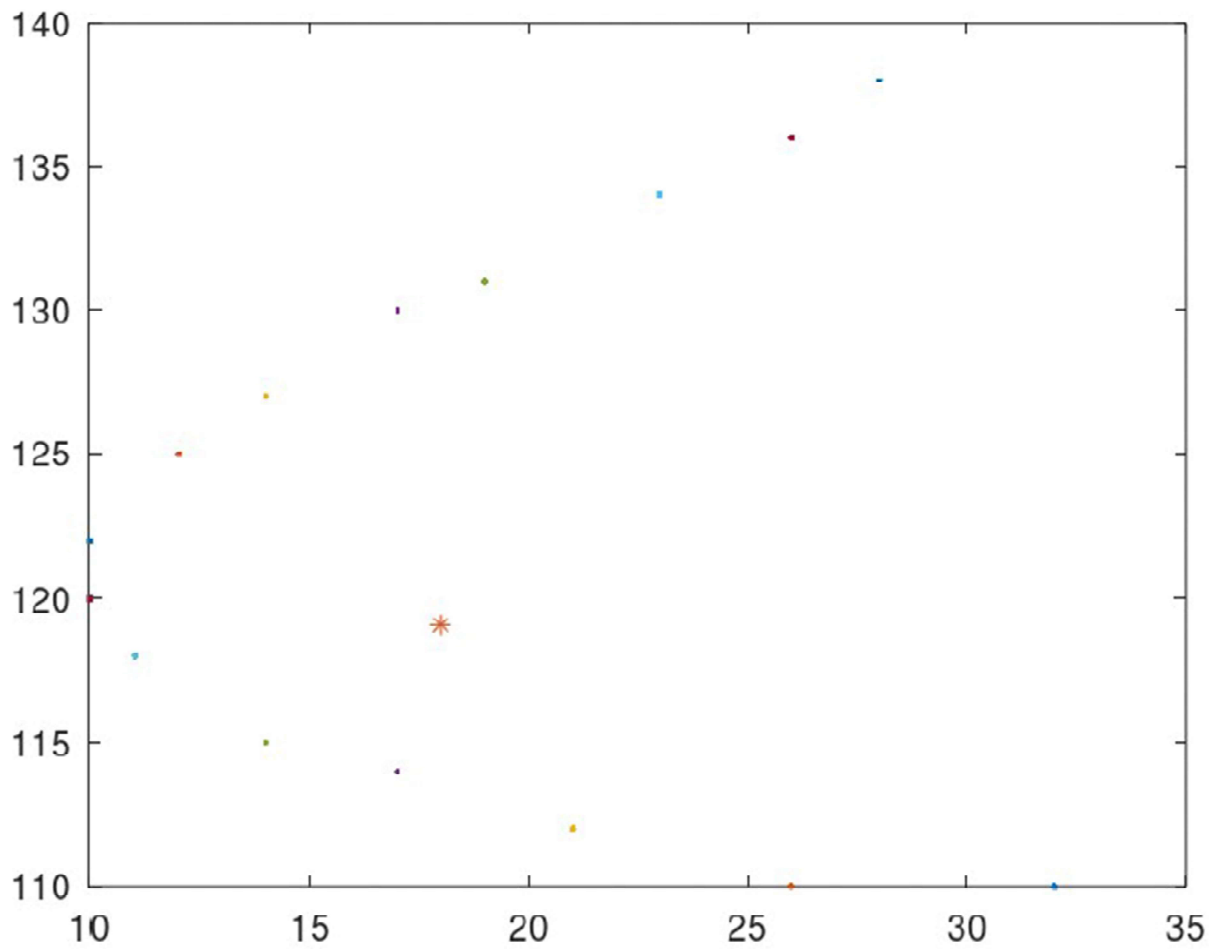


Fig.13.12.13 1st cluster at the end of last run

The graph shows the 1st cluster using a certain cutoff value for membership function. This output corresponds to the 11th run. Purple asterix denotes focus of the parabola.

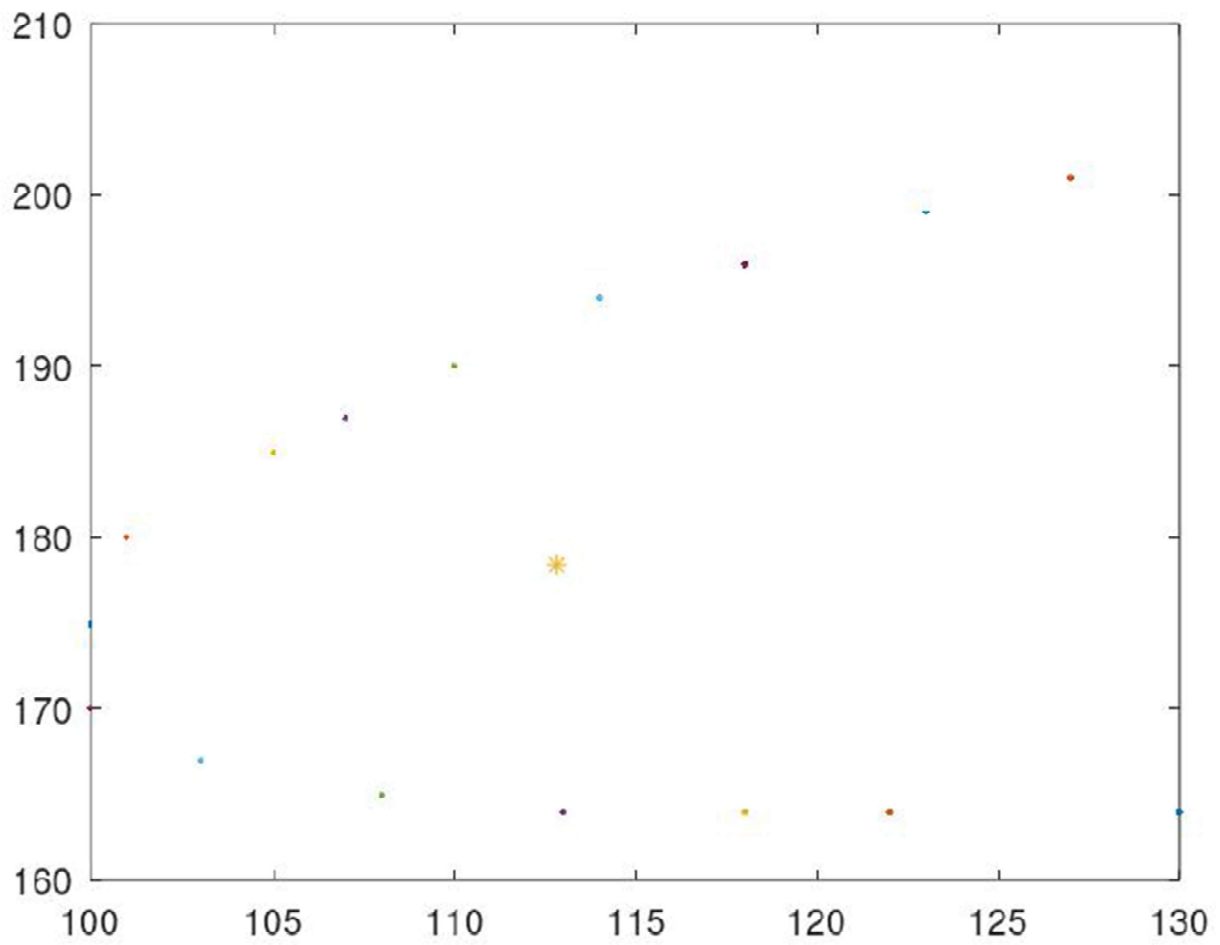


Fig. 13.12.14 2nd cluster at the end of last run

The graph shows the 2nd cluster using a certain cutoff value for membership function. This output corresponds to the 11th run. Purple asterix denotes focus of the parabola.

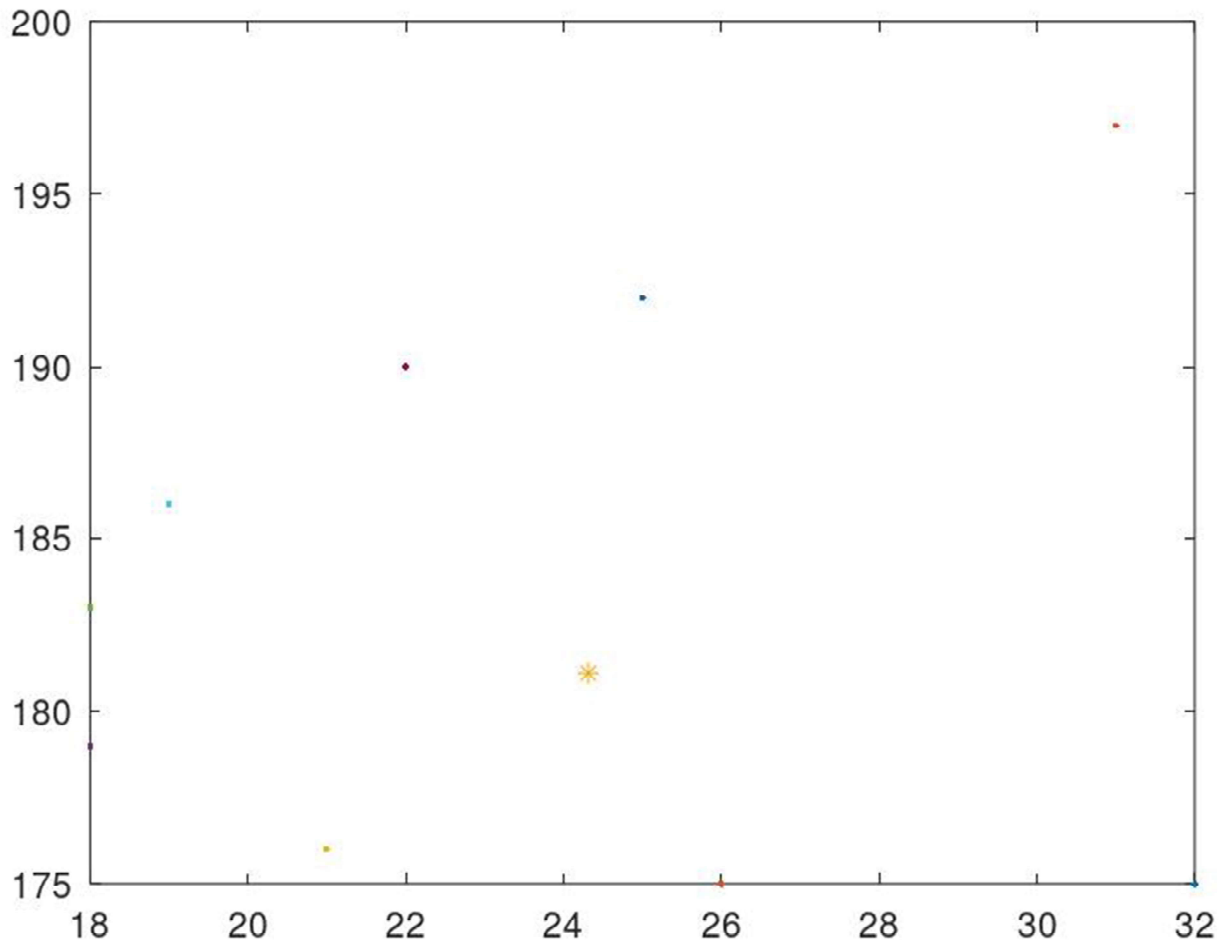


Fig.13.12.15 3rd cluster at the end of last run

The graph shows the 3rd cluster using a certain cutoff value for membership function. This output corresponds to the 11th run. Purple asterisk denotes focus of the parabola.

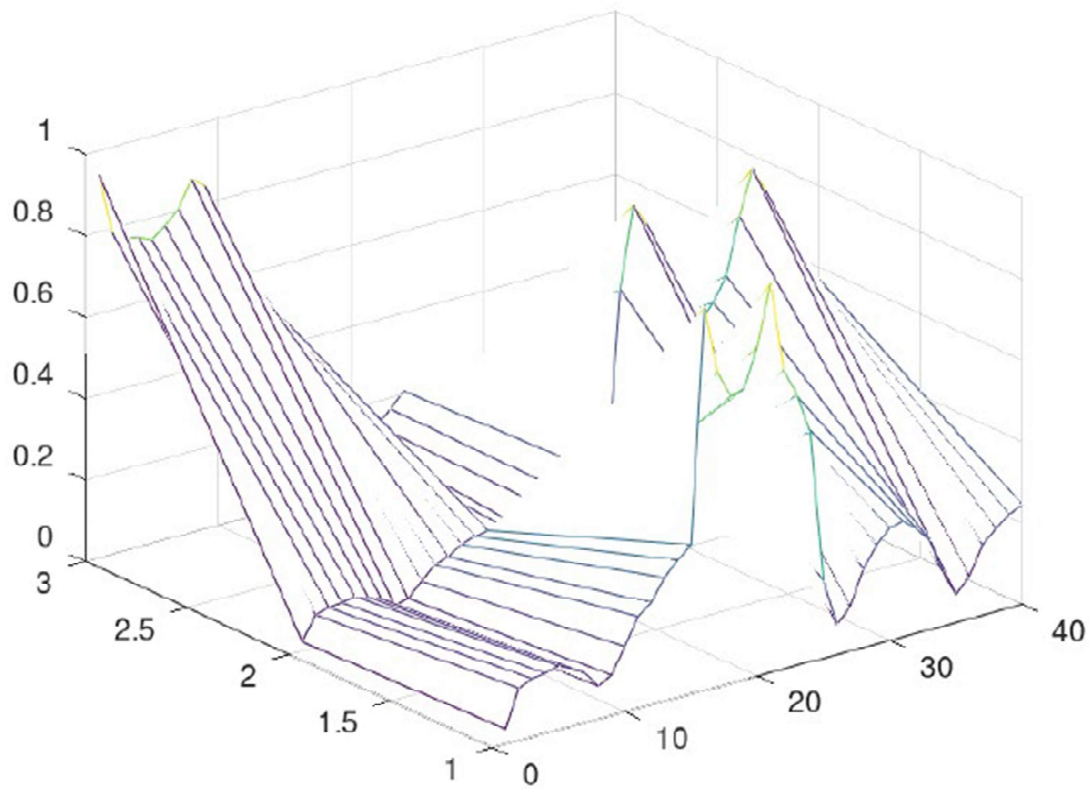


Fig.13.12.16 The above graph shows membership function at the end of 11th run.

The above graph is that of the fuzzy membership function at the end of 11th run. Also sum of membership value of a data point with respect to all the clusters.

13.13 Discussion Of Output

As we see from the outputs of the new algorithm we find that the algorithm detects the focuses of the parabolas with ease. At each run the parameters of the directrix equations are incremented in the steps of 10% of initial values. Using the new values and the data and the fuzziness parameter we get other required parameters i.e. the directrix equation gets altered not only in intercept but also by slope which will be evident if we think about the equation and perform some basic algebraic operations and rewriting the equation in slope intercept form. In effect we use a new equation for each cluster in every new run. We can further modify the increment operations to suit the need of the problem. In short we are assuming that a parabola can be anywhere and oriented freely in 2D space. As same as before Fuzziness parameter should be selected with care [21].

CHAPTER 14

APPLICATION

So far the theoretical and conceptual grounds of our new algorithm and tested on artificial data and random data. Here some potential uses of the algorithms are stated followed by an example.

The main task of the algorithms developed is clustering which is a form of unsupervised learning task. We can use these algorithms where the inherent task is that of clustering. We can use them to find clusters of ellipse, parabola shapes. We can also use them in image processing [12] to find objects of desired shape.

Here we show one application in the field of image processing. The image has been taken from [11].

From the digital image [12] we took a small section from it which was converted from RGB to gray scale image [12]. Then the gray scale image was sharpened [12] and to the sharpened image we apply Canny edge detection algorithm [12] to get the edges now. Then we find the pixel positions of those pixels which have the value 1 [12] i.e. the pixels corresponding to edges. The positions were plotted on the graph and we apply Fuzzy C Means Ellipse Clustering of variable size algorithm. A point to be mentioned the initial values of semi major/minor axis lengths of clusters should be as close to the desired clusters as much as possible. Here the clusters are blood cells Leukocyte to be more specific [11]. Here the centroids belong to the cells in the image slice.

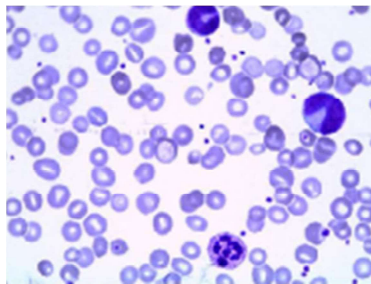


Fig.14.1 Image used from the mentioned [11] paper

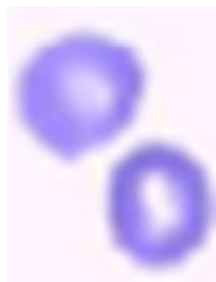


Fig.14.2 Image portion used from the mentioned [11] paper

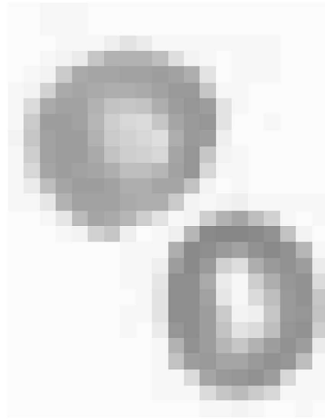


Fig.14.3 Image portion converted to gray scale

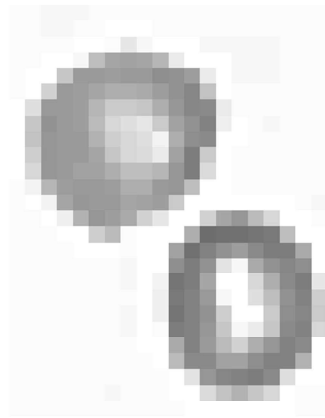


Fig.14.4 Gray Scale image sharpened

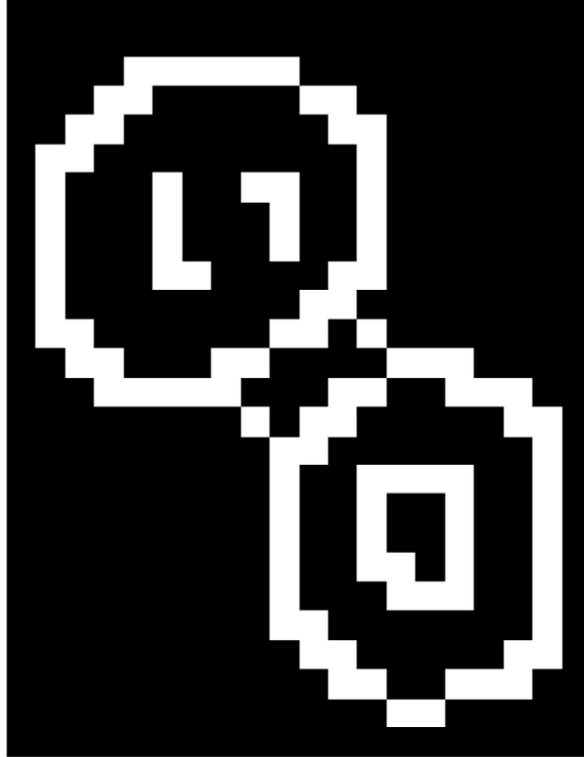


Fig.14.5 Edge detected using Canny

In this example attempt was made to find 2 clusters (cells). Each cluster corresponds to a cell and thus centroid of clusters can be thought of as centroid of cells. The value of α was taken as 2.8.

14.1 Output

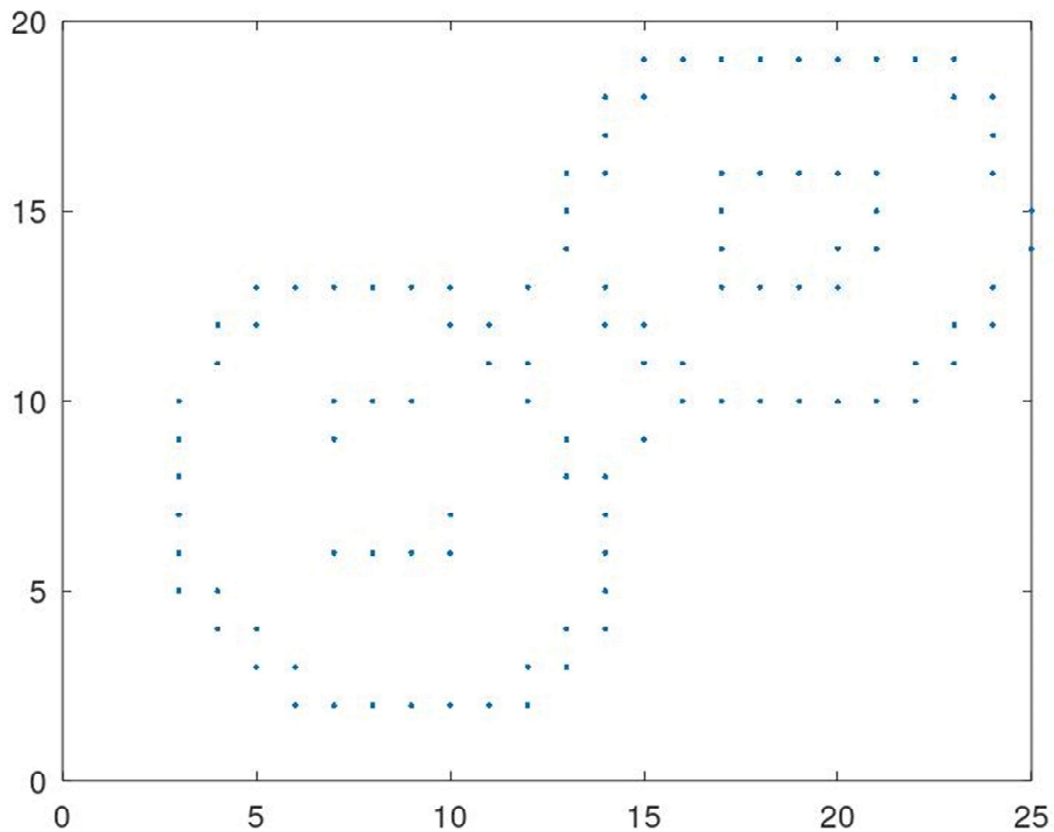


Fig.14.1.1 Graph Showing Pixel positions having value 1

The above graph was developed by converting the digital image from RGB format to gray scale image [12]. Then sharpening [12] it at applying Canny edge detection algorithm [12] on it to find the edges in the image then extracting pixels with value 1[12] and plotting them. A point to be noted that in digital image [12] the top left hand corner is the origin and the ordinate value increases as we move from left to right and top to bottom [12]. In short the above graph is mirror reflection about the x axis..

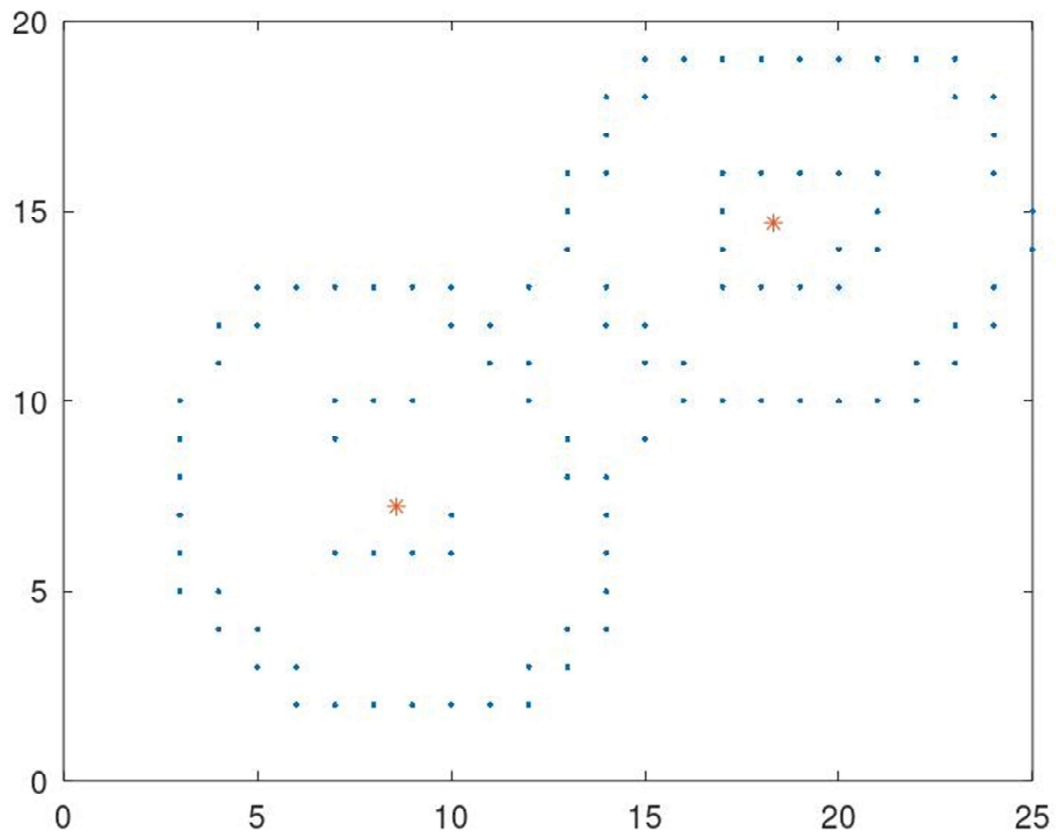


Fig.14.1.2 1st Run Output

The above graph shows the output of the 1st run. We randomly select lengths of semi major/minor axes of clusters but should be close to real values as much as possible. Blue dots are data points and orange asterisk are centroids.

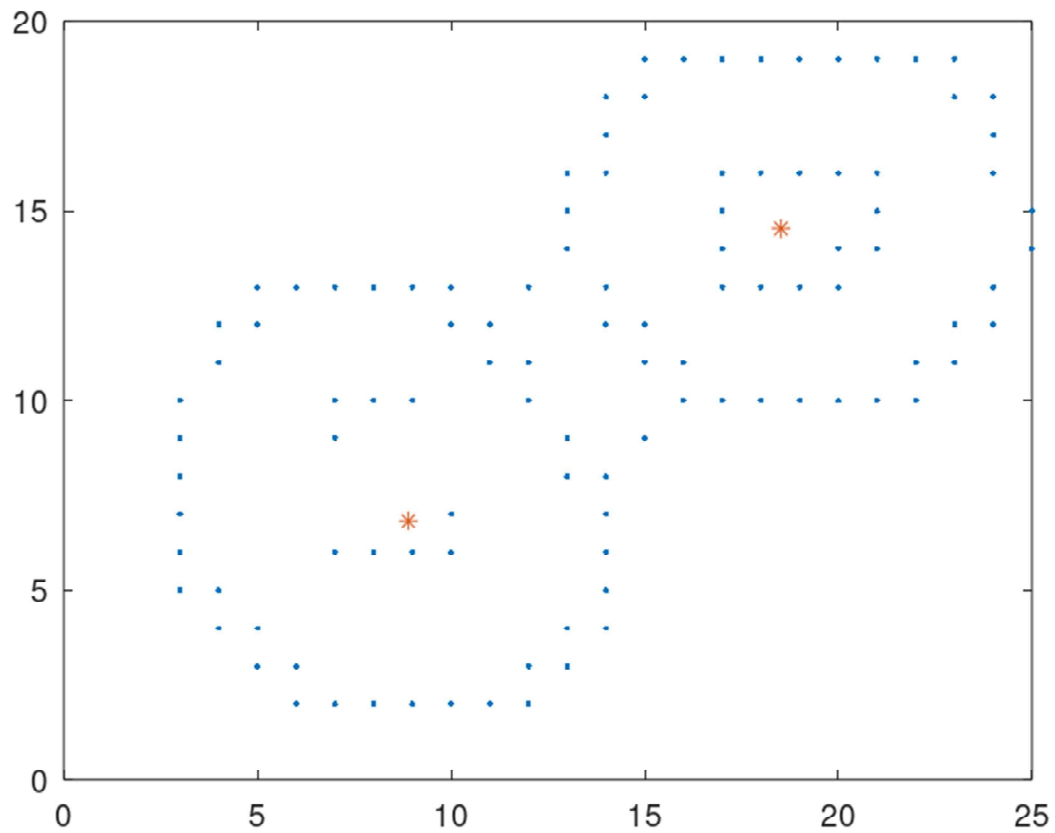


Fig.14.1.3 2nd Run Output

The above graph shows the output of 2nd run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. . Blue dots are data points and orange asterix are centroids.

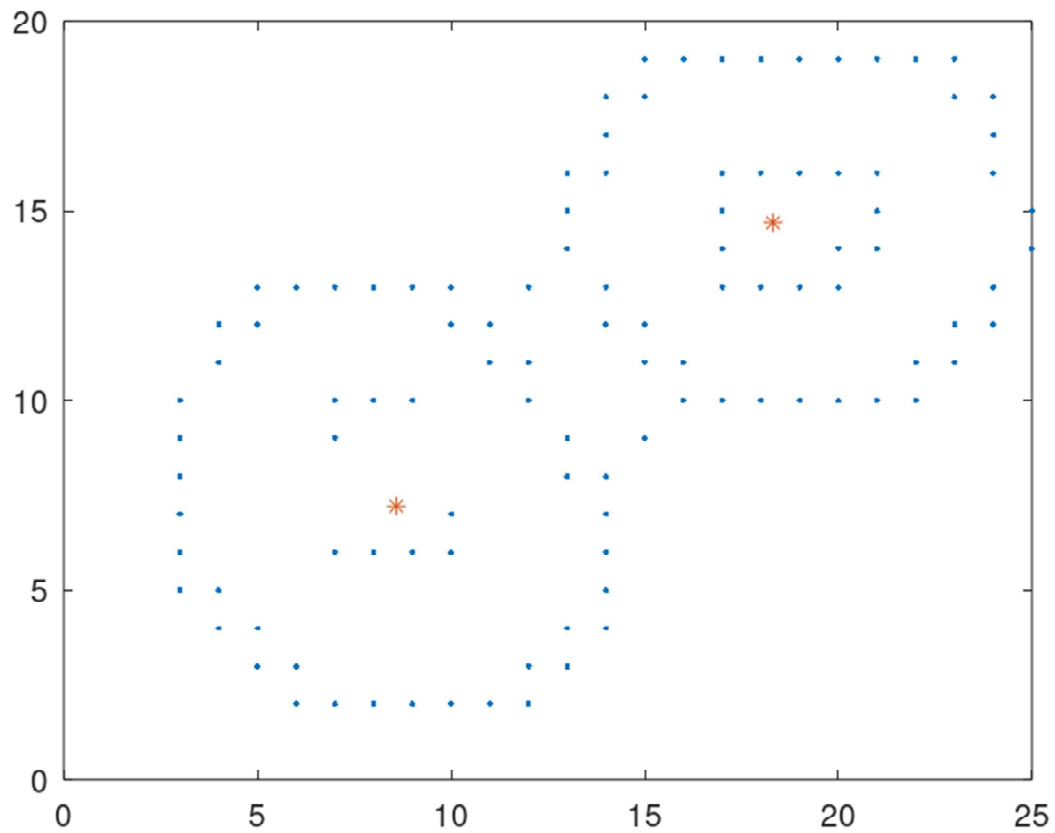


Fig.14.1.4 3rd Run Output

The above graph shows the output of 3rd run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. Blue dots are data points and orange asterix are centroids.

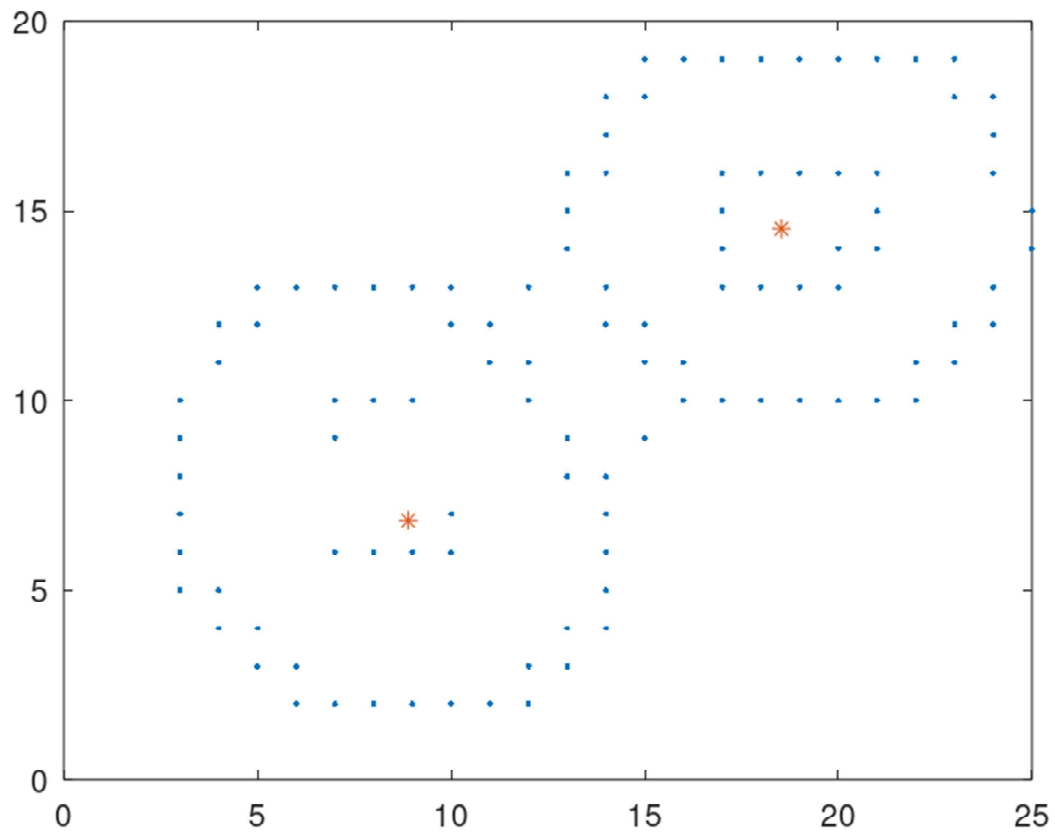


Fig. 14.1.5 4th Run Output

The above graph shows the output of 4th run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. Blue dots are data points and orange asterix are centroids.

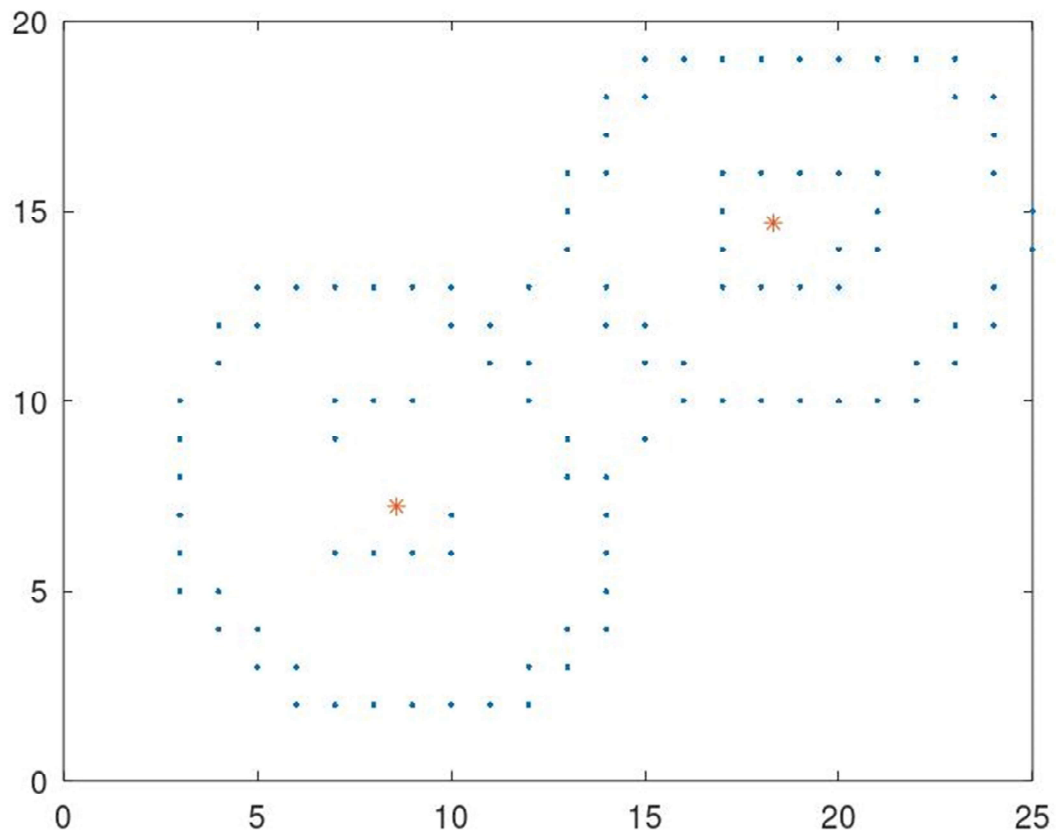


Fig.14.1.6 5th Run Output

The above graph shows the output of 5th run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. Blue dots are data points and orange asterix are centroids.

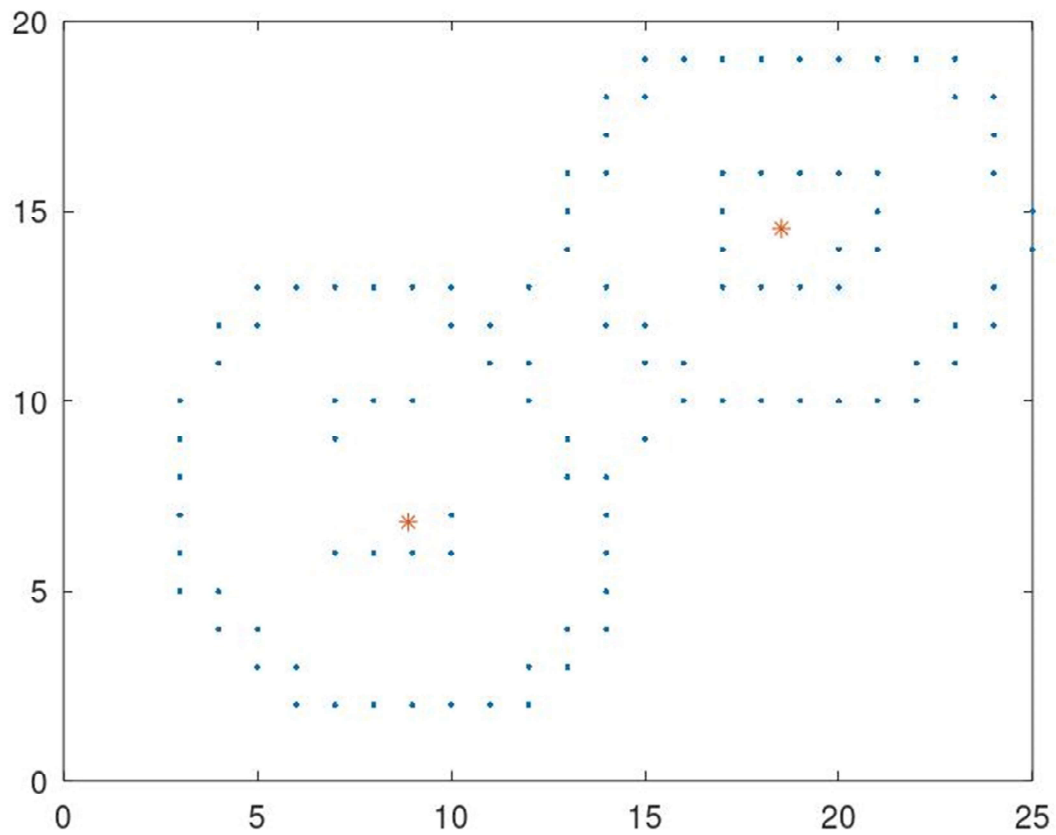


Fig. 14.1.7 6th Run Output

The above graph shows the output of 6th run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. Blue dots are data points and orange asterix are centroids.

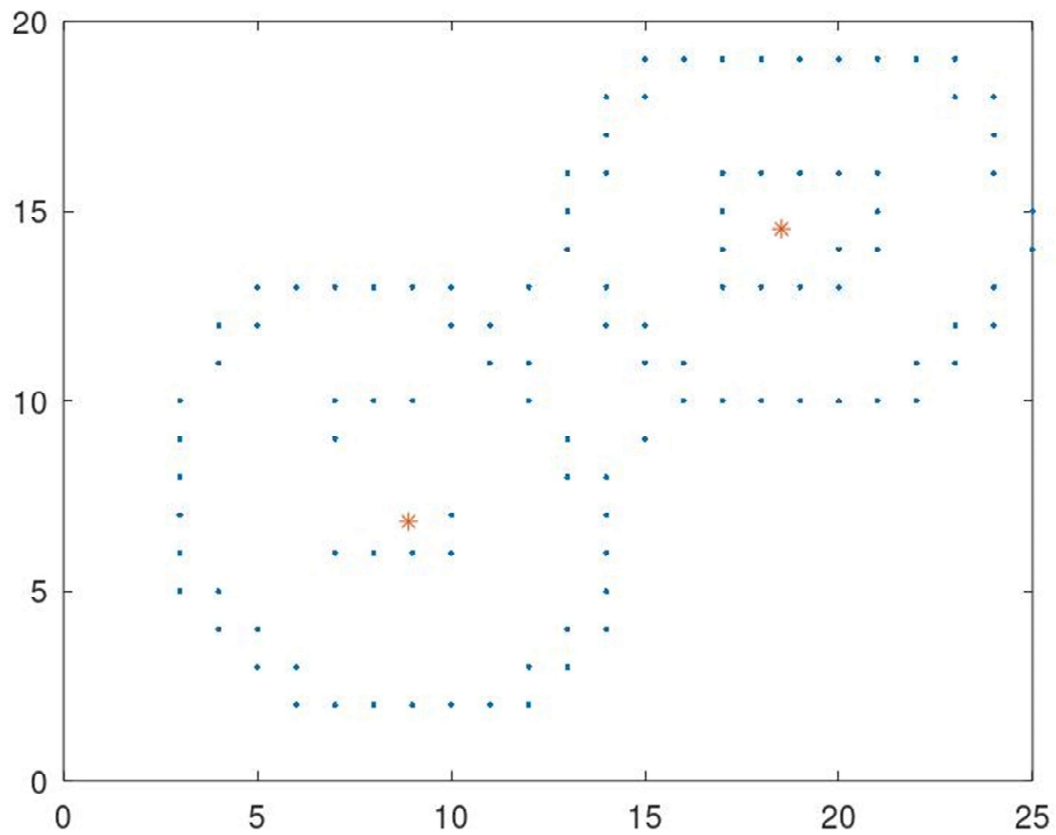


Fig.14.1.8 7th Run Output

The above graph shows the output of 7th run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. Blue dots are data points and orange asterix are centroids.

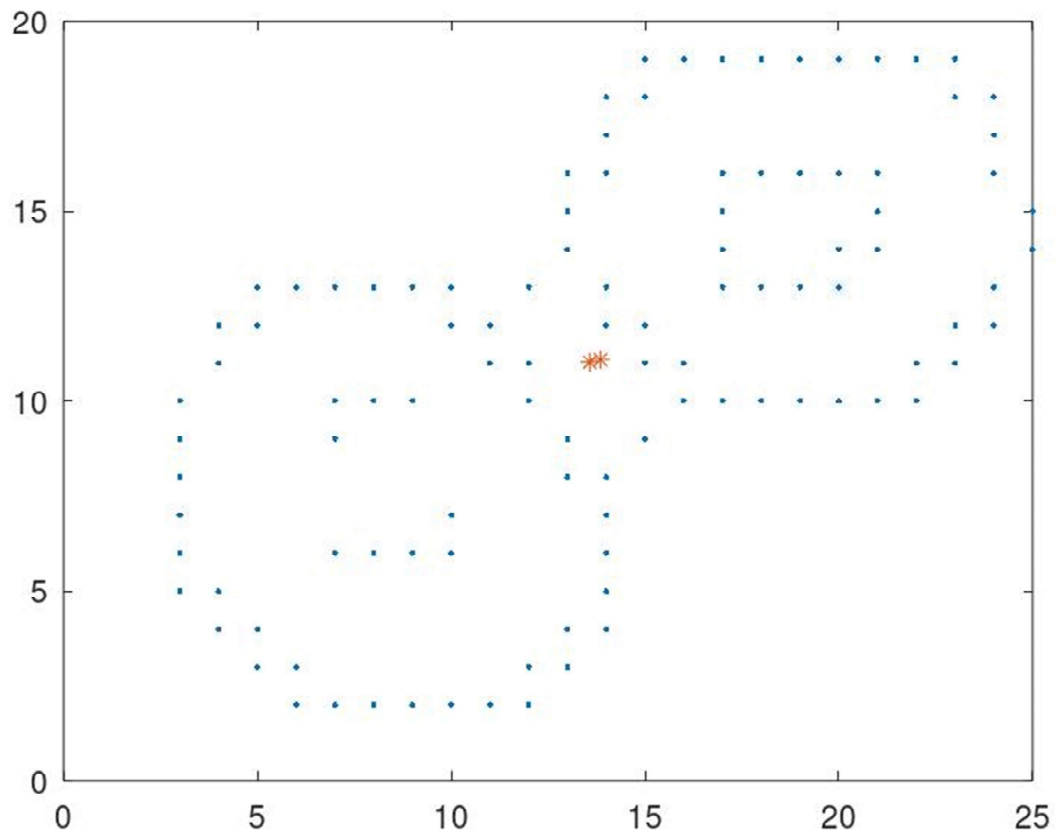


Fig. 14.1.9 8th Run Output

The above graph shows the output of 8th run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. . Blue dots are data points and orange asterix are centroids.

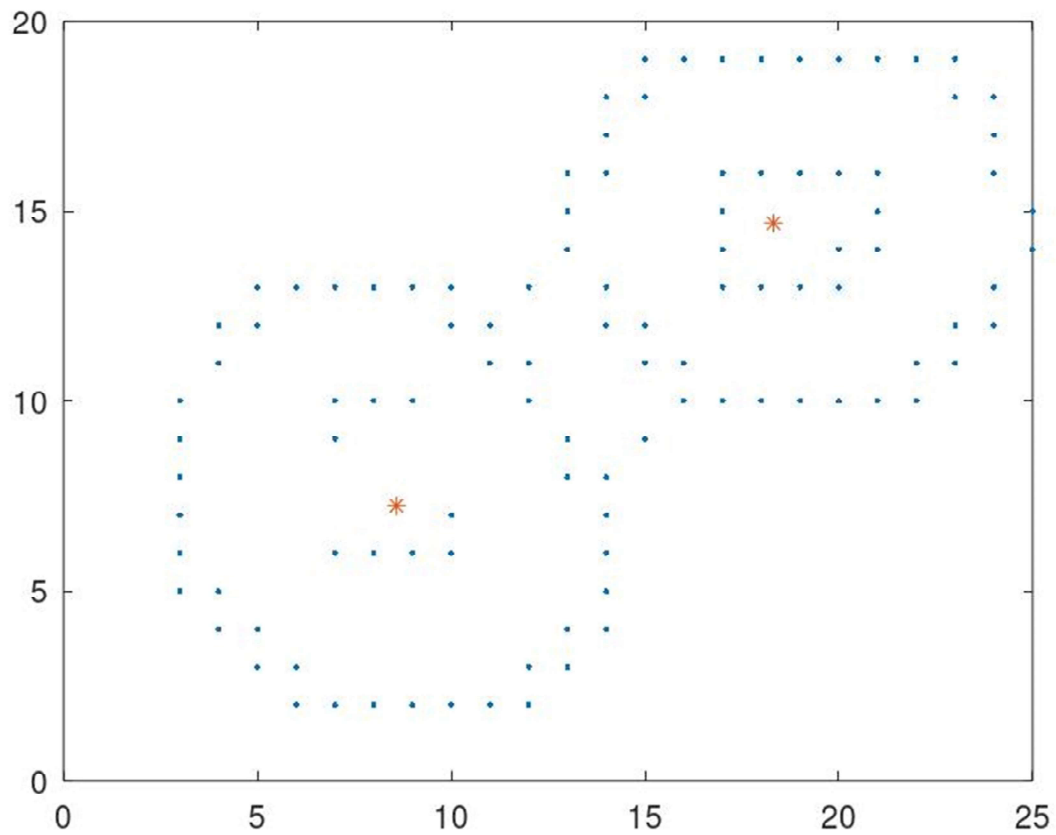


Fig 14.1.10 9th Run Output.

The above graph shows the output of 9th run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. Blue dots are data points and orange asterisk are centroids.

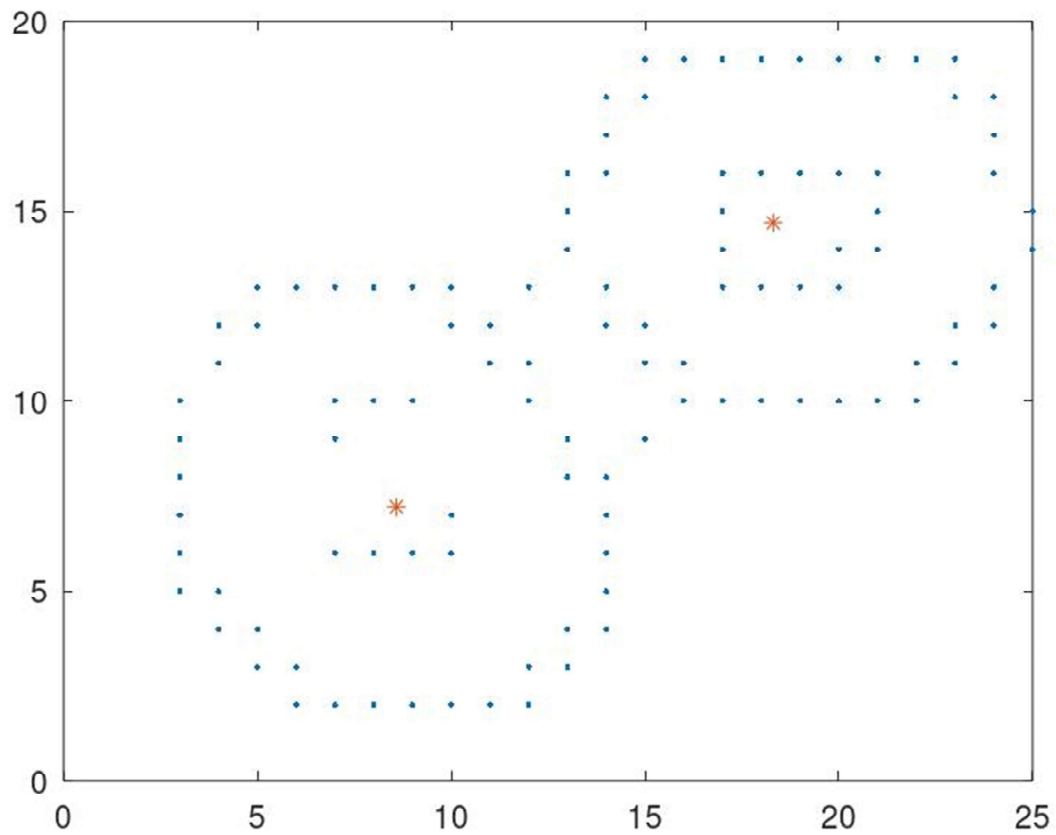


Fig.14.1.11 10th Run Output

The above graph shows the output of 10th run. The semi major/minor axes values are incremented by $0.1 \times \text{initial values}$. Blue dots are data points and orange asterix are centroids.

14.2 Process In A Nutshell

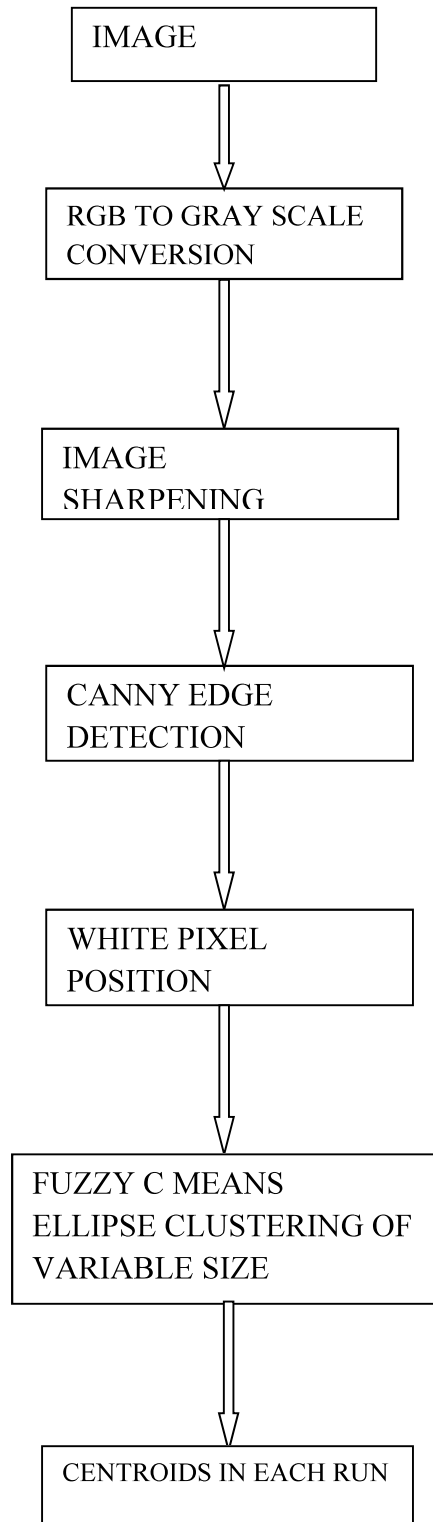


Fig 14.2.1. Summary of steps followed in the application

14.3 Discussion Of Output

From the above images we can see that cell centroids are properly detected in most of the runs. Fuzziness parameter should be properly selected else the algorithm will get stuck at local optima [21] which is inherent property of Fuzzy C Means Clustering [9] algorithm which is the basis of both Fuzzy C Means Ellipse Clustering of variable size and Fuzzy C Means Parabola Clustering of variable orientation and positions .

CHAPTER 15

PERFORMANCE

In this chapter we present the performance of Fuzzy C Means Ellipse Clustering of variable size and Fuzzy C Means Parabola Clustering of variable orientation and positions.

We have chosen the performance parameters to be No of iteration per run and c.p.u. time consumed for the entire run to complete. The value of α was chosen as 2.8.

Run Number	No Of Iterations Required	C.P.U. Time Consumed in seconds
1	10	0.4219
2	8	0.2969
3	9	0.3281
4	7	0.2969
5	8	0.2656
6	9	0.3281
7	11	0.4531
8	8	0.2500
9	7	0.2500
10	12	0.4531

Fig 15.1 Table showing performance of Fuzzy C Means Ellipse Clustering of Variable size for a certain instance

The above table is for the application of Fuzzy C Means Ellipse Clustering of variable size on the formatted image data which are white pixel positions. We see that it takes on an average 8.9 iterations per run and 0.33437s average cpu time per run. CPU time was found out by using `cputime()` function at the beginning and at end of each run and subtracting the values. The values may change.

Let us plot the above performance data

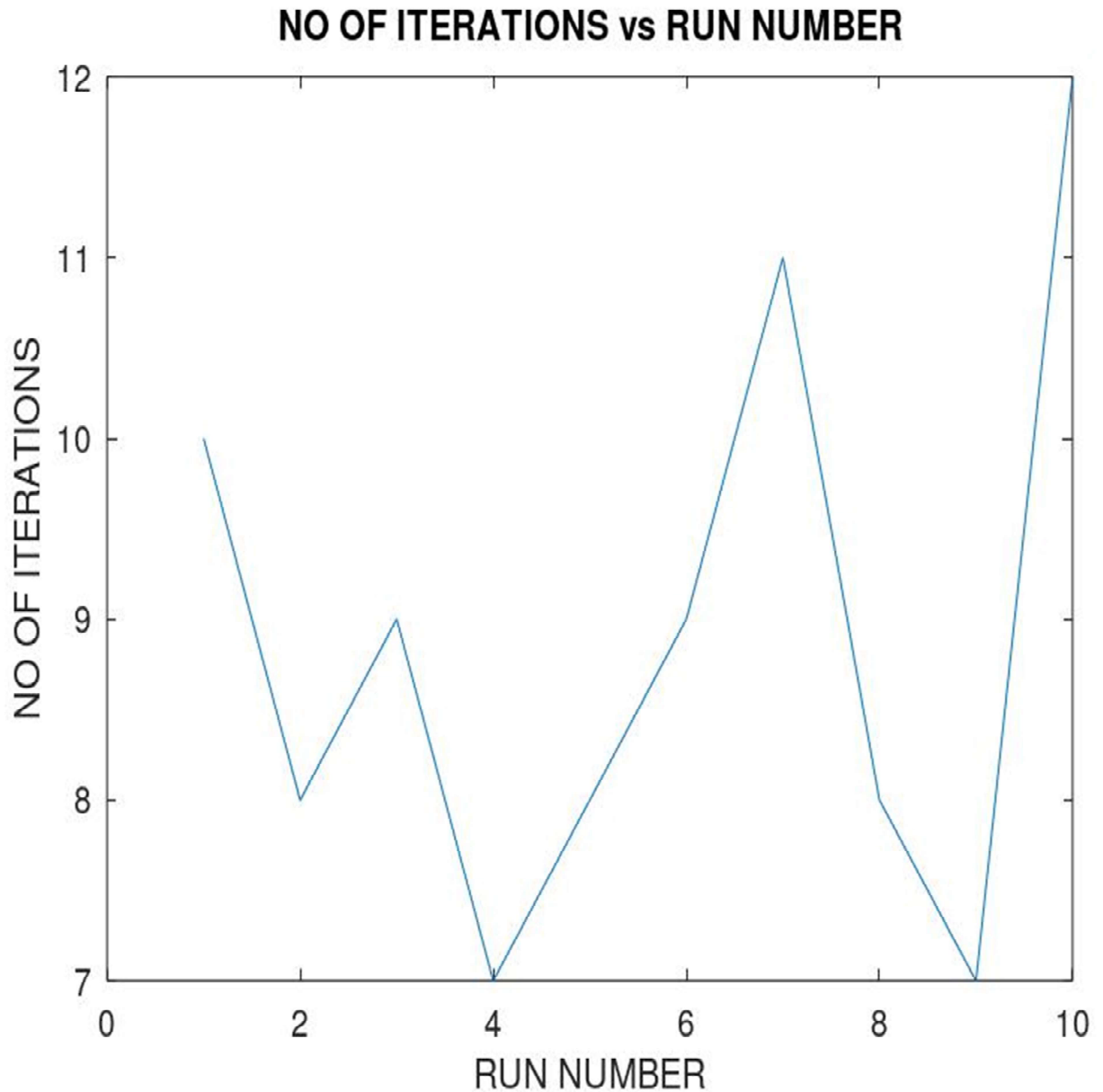


Fig.15.2 Graph showing No of iterations vs Run number

The above graph shows us No of iterations vs Run number of Fuzzy C Means Ellipse Clustering of variable size, the number of clusters (cells) to be searched for was 2 and number of runs were 10. It shows a variation and no fixed relationship is maintained.

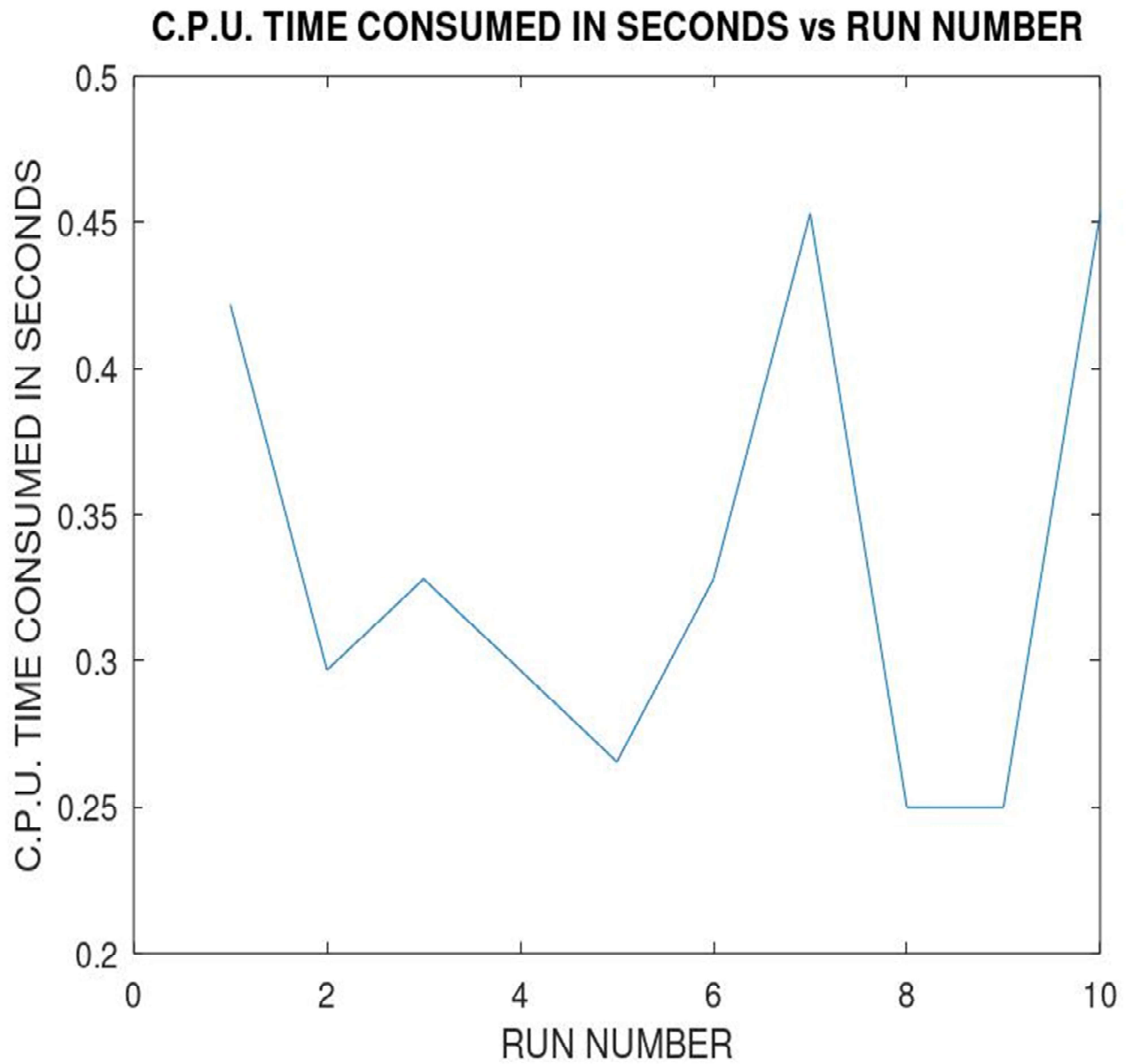


Fig15.3 Graph Showing CPU Time consumed Vs run number

The above graph shows us C.P.U. time consumed vs run number.

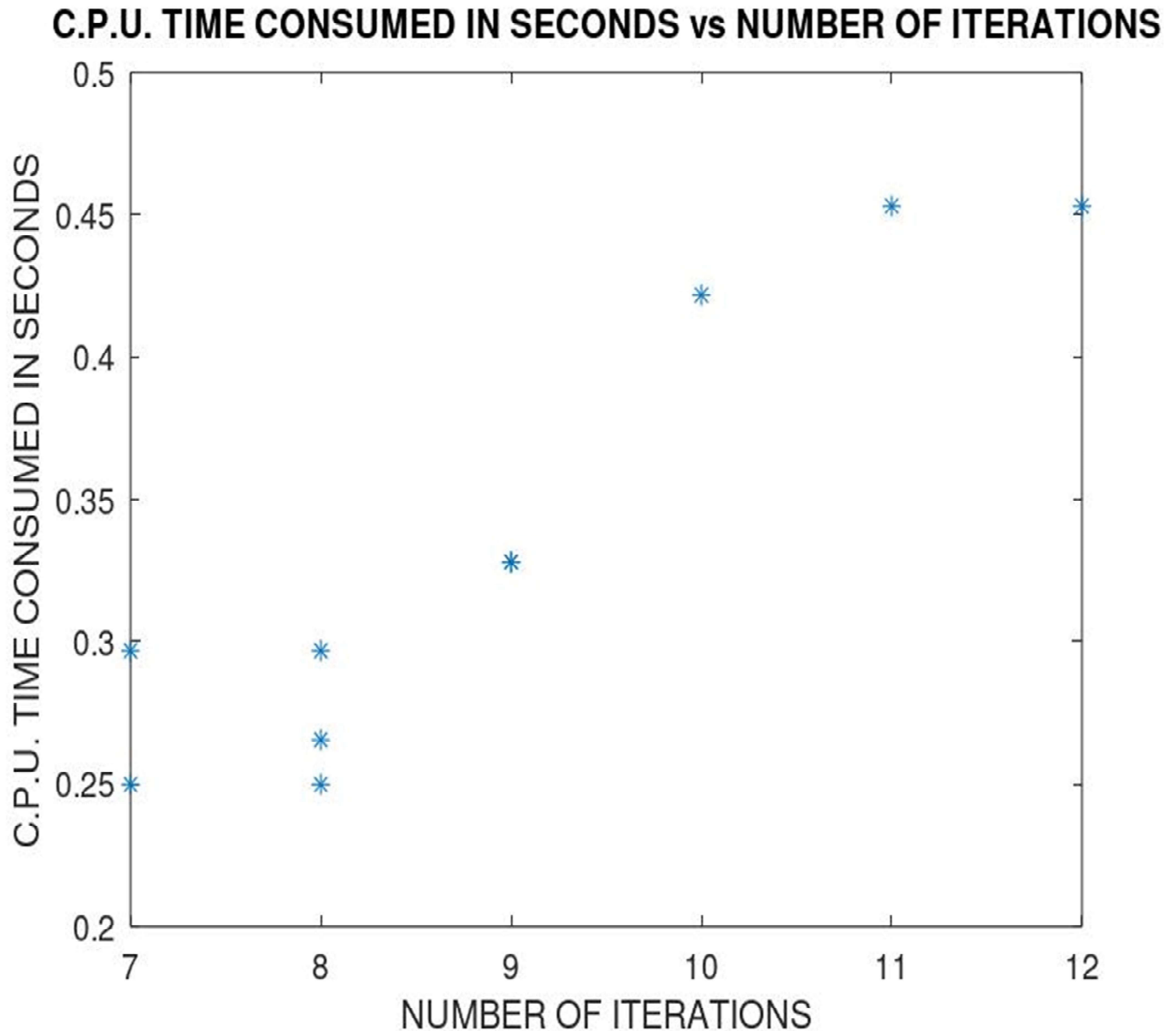


Fig.15.4 Graph showing CPU time consumed in seconds vs Number of Iterations

In the above graph asterix represents points where x value represents number of iterations and y value represents time consumed by the CPU. System mode execution and user mode execution are both considered, it has been used in all performance studies presented here.

Algorithm used	Data Used	α	Avg. Iterations	Average C.P.U. Time Consumed in seconds	Avg. SSE
Fuzzy C Means Clustering	White Pixel Positions in the Leukocyte Image	2.8	8.6	0.17967	(0.60052, 0.68996)
Fuzzy C Means Ellipse Clustering of Variable size	White Pixel Positions in the Leukocyte Image	2.8	8.58 (Per run)	0.270937	(0.4220, 0.6618)

Fig 15.5 Table showing performance of Fuzzy C Means Ellipse Clustering of variable size and FCM

An experiment was performed for comparing Fuzzy C Means Ellipse Clustering of variable size and FCM. FCM was executed 10 times and the values were averaged accordingly. Now Fuzzy C Means Ellipse clustering of variable size was executed 10 times with the same initial conditions every time, in its case runs we divided each quantity by 100 to get average as each instance requires 10 runs and the algorithm was executed 10 times so $10 \times 10 = 100$. We also used SSE parameter which is error sum of squares for each cluster. In case of FCM we divided each result by 10 to get average values. We used top 50 highest membership value data points to find Avg. SSE along with the centroids discovered by the respective algorithms. We see that Fuzzy C Means Ellipse Clustering of Variable Size performs better.

Now we present the performance of an instance of Fuzzy C Means Parabola Clustering of variable orientation and positions. The value of α was taken as 4. We use the same performance parameters as with that of Fuzzy C Means Ellipse Clustering of variable size.

Run Number	Number Of Iterations	C.P.U. Time Consumed In Seconds
1	15	0.5156
2	17	0.6250
3	14	0.4688
4	10	0.3438
5	19	0.6094
6	14	0.5625
7	20	0.6875
8	12	0.5312
9	12	0.4531
10	19	0.6562
11	20	0.6094

Fig 15.6 Table showing performance of Fuzzy C Means Parabola Clustering of variable orientation and positions for a certain instance

The same artificial dataset was used and number of clusters to be detected was 3. The average number of iterations was $15.\overline{63}$ and average cpu time consumed was 0.5511 s.

Let us plot the results.

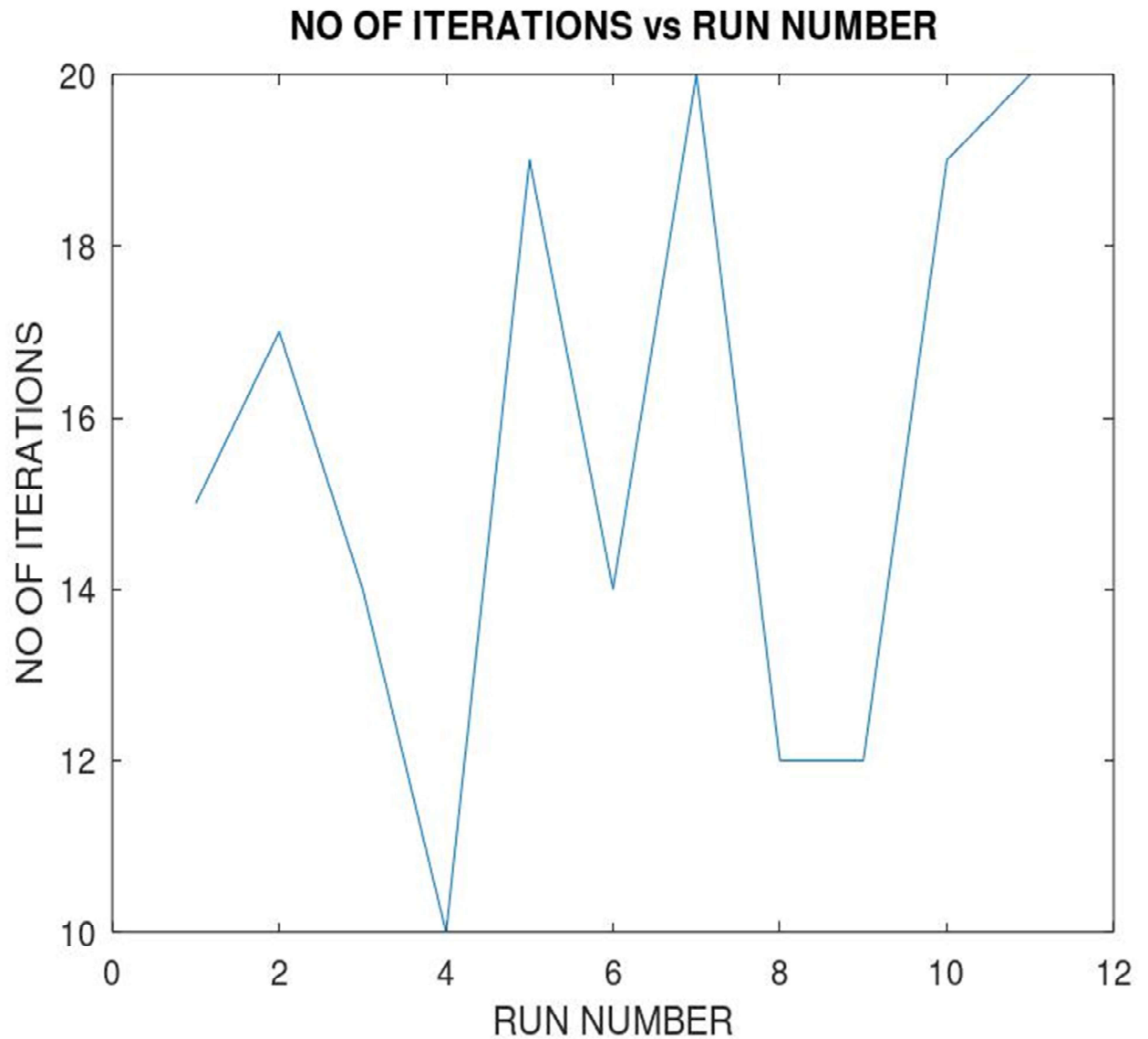


Fig 15.7 Graph showing No of iterations vs run number

The graph shows us no of iterations vs run number of fuzzy c means parabola clustering of variable orientation and positions. As it is evident there is no proper relationship between the two metrics and is highly variable. It looks like it depends on the parameters used in that run.

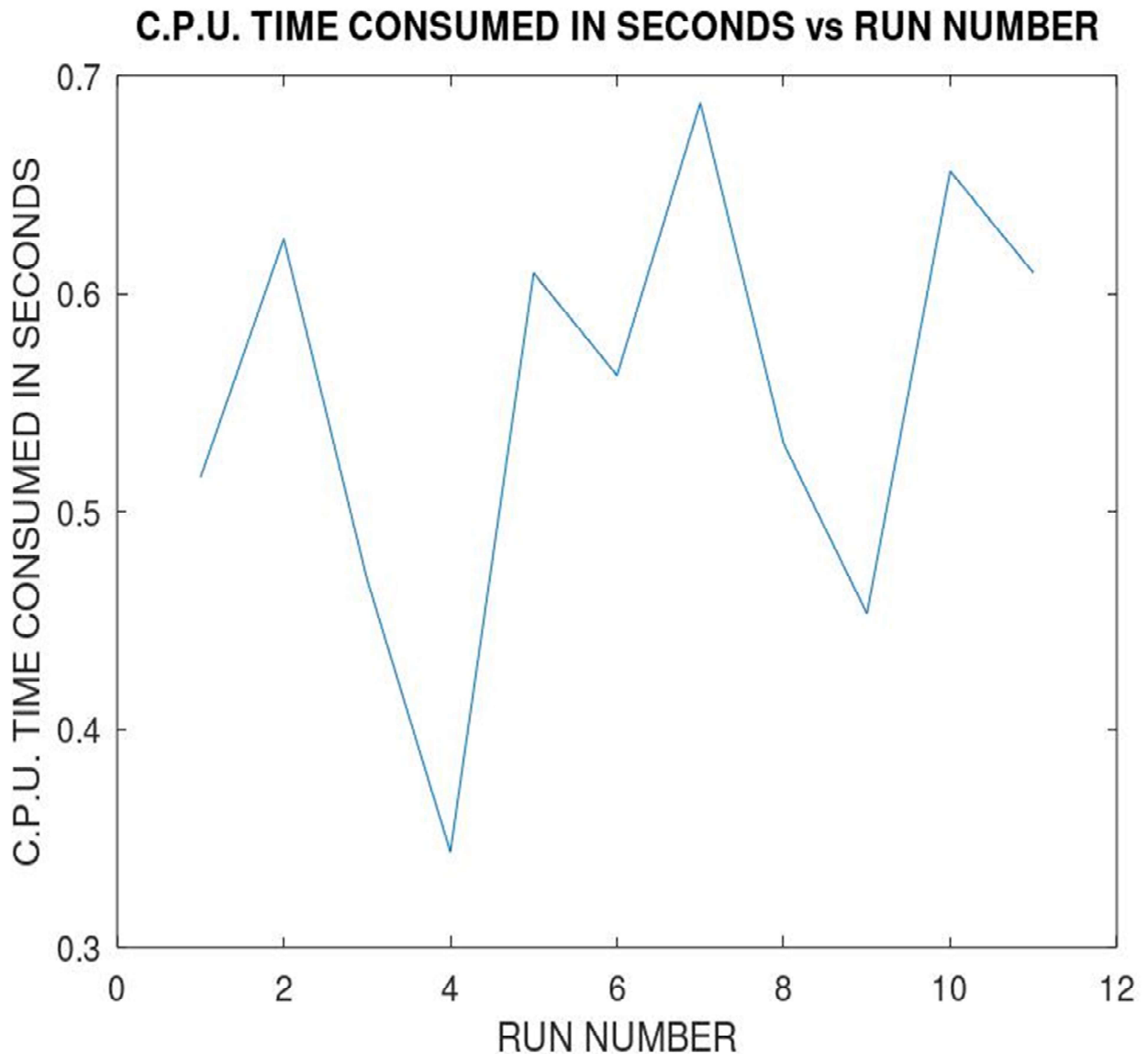


Fig 15.8 Graph showing CPU time consumed in seconds vs run number

The graph shows us CPU time consumed in seconds vs run number of fuzzy c means parabola clustering of variable orientation and positions. As it is evident there is no proper relationship between the two metrics and is highly variable. It looks like it depends on the parameters used in that run. . CPU time was found out by using `cputime()` function at the beginning and at end of each run and subtracting the values. The values may change.

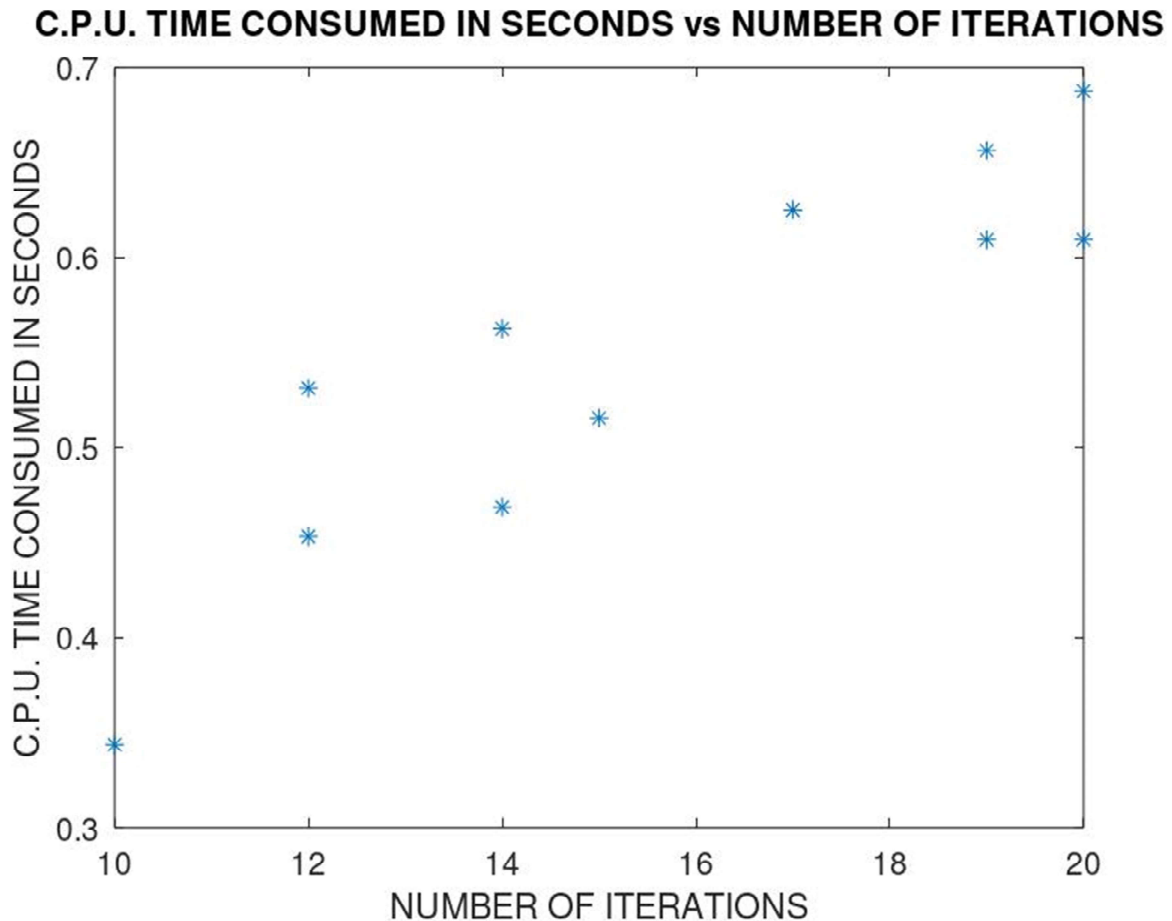


Fig 15.9 Graph showing CPU time consumed in seconds vs number of iterations

In the above graph asterix represents points where x value represents number of iterations and y value represents time consumed by the CPU. System mode execution and user mode execution are both considered. It is for fuzzy c means parabola clustering of variable orientation and positions.

Both the algorithms have the performance comparable to that of Fuzzy C Means clustering [9] algorithm but the accuracy of Fuzzy C Means Ellipse Clustering of Variable size is better than that of FCM as seen in fig 15.5. The relationship between run number and number of iterations is random and highly depends on starting conditions i.e. the values chosen for various parameters.

CHAPTER 16

CONCLUSION

The works presented here does enforce their respective shapes on clusters. Some more refinement might be required which might further improve the applicability of the algorithms introduced.

It should be again stated that the fuzziness parameter value should be selected with proper care so that the cost function yields the minimum value possible [21]. If it is not done properly then the algorithms will get stuck in local optima. This value selection is itself an interesting problem in its own right.

The algorithms can be used in various applications such as image processing [12]. It can be used for organ such as eyes, spleen, tumor e.t.c., cell detection in case of Bio-medical Image processing.

REFERENCES

1. Amit, K. (2018). *Artificial intelligence and soft computing: behavioral and cognitive modeling of the human brain*. CRC press.
2. Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (Eds.). (2013). *Machine learning: An artificial intelligence approach*. Springer Science & Business Media.
3. Géron, A. (2017). Hands-on machine learning with scikit-learn and tensorflow: Concepts, Tools, and Techniques to build intelligent systems.
4. Babuška, R. (2000). Fuzzy clustering algorithms with applications to rule extraction. In *Fuzzy systems in medicine* (pp. 139-173). Physica, Heidelberg.
5. Zadeh, L. A. (1996). Fuzzy sets. In *Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers by Lotfi A Zadeh* (pp. 394-432).
6. Gupta, U. S. (2014). *Discrete Mathematical Structures, 1/e*. Pearson Education India.
7. Sun, W., & Yuan, Y. X. (2006). *Optimization theory and methods: nonlinear programming* (Vol. 1). Springer Science & Business Media.
8. Rao, S. S. (2009). Engineering optimization: theory and practice/Singiresu S. Rao.—.
9. Bezdek, J. C., Ehrlich, R., & Full, W. (1984). FCM: The fuzzy c-means clustering algorithm. *Computers & geosciences*, 10(2-3), 191-203.
10. Leung, S. H., Wang, S. L., & Lau, W. H. (2004). Lip image segmentation using fuzzy clustering incorporating an elliptic shape function. *IEEE transactions on image processing*, 13(1), 51-62.
11. Jati, A., Singh, G., Mukherjee, R., Ghosh, M., Konar, A., Chakraborty, C., & Nagar, A. K. (2014). Automatic leukocyte nucleus segmentation by intuitionistic fuzzy divergence based thresholding. *Micron*, 58, 55-65.
12. Gonzalez, R. C., & Woods, R. E. (2018). Digital image processing 4th edition, global edition.
13. Taha, H. A. (2011). *Operations research: an introduction* (Vol. 790). Upper Saddle River, NJ, USA: Pearson/Prentice Hall.
14. Kim, D. W., Lee, K. H., & Lee, D. (2005). Detecting clusters of different geometrical shapes in microarray gene expression data. *Bioinformatics*, 21(9), 1927-1934.
15. Bandyopadhyay, S., & Maulik, U. (2002). An evolutionary technique based on K-means algorithm for optimal clustering in RN. *Information Sciences*, 146(1-4), 221-237.

16. Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3), 241-254.
17. Kriegel, H. P., Kröger, P., Sander, J., & Zimek, A. (2011). Density-based clustering. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 1(3), 231-240.
18. Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, 17(4), 395-416.
19. Barioni, M. C. N., Razente, H., Marcelino, A. M., Traina, A. J., & Traina Jr, C. (2014). Open issues for partitioning clustering methods: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(3), 161-177.
20. Rich, E., & Knight, K. (1991). Artificial Intelligence TATA McGRAW-HILL.
21. Torra, V. (2015, June). On the selection of m for Fuzzy c-Means. In *IFSA-EUSFLAT*.
22. Saha, S and Saha, S. I.S.C Mathematics Part-II, Kalyani Publishers.
23. Martin Ester, Hans-Peter Kriegel, Jiirg Sander, Xiaowei Xu ,A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, : KDD-96 Proceedings.

Report

ORIGINALITY REPORT

6%

SIMILARITY INDEX

PRIMARY SOURCES

1	louisdl.louislibraries.org Internet	197 words — 1%
2	philippineculturaleducation.com.ph Internet	120 words — 1%
3	Anne-Laure Dalibard, Helge Dietert, David Gérard-Varet, Frédéric Marbach. "High Frequency Analysis of the Unsteady Interactive Boundary Layer Model", SIAM Journal on Mathematical Analysis, 2018 Crossref	66 words — < 1%
4	p.uvarov.ru Internet	57 words — < 1%
5	www.scribd.com Internet	52 words — < 1%
6	epdf.pub Internet	50 words — < 1%
7	www.teses.usp.br Internet	36 words — < 1%
8	K. Morling. "Conic Sections – the Ellipse, the Parabola, the Hyperbola", Geometric and Engineering Drawing, 2010 Crossref	34 words — < 1%

9	webidu.idu.gov.co Internet	30 words — < 1%
10	www.slideshare.net Internet	28 words — < 1%
11	Ramadan, A.A.. "L-fuzzy interior systems", Computers and Mathematics with Applications, 201112 Crossref	24 words — < 1%
12	Sarkar, Sanjib. "Machine Learning Assisted FBG- Based Sensors", The University of Texas at San Antonio, 2021 ProQuest	24 words — < 1%
13	ebin.pub Internet	21 words — < 1%
14	"Modeling and Simulation with Compose and Activate", 'Springer Science and Business Media LLC', 2018 Internet	20 words — < 1%
15	Badarneh, Osamah S.. "The <mml:math altimg="si15.gif" overflow="scroll" xmlns:xocs="http://www.elsevier.com/xml/xocs/dtd" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.elsevier.com/xml/ja/dtd" xmlns:ja="http://www.elsevier.com/xml/ja/dtd" xmlns:mml="http://www.w3.org/1998/Math/MathML" xmlns:tb="http://www.elsevier.com/xml/common/table/dtd" xmlns:sb="http://www.elsevier.com/xml/common/struct- bib/dtd" xmlns:ce="http://www.elsevier.com/xml/common/dtd" xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:cals="http://www.elsevier.com/xml/common/cals/dtd"	20 words — < 1%

xmlns:sa="http://www.elsevier.com/xml/common/struct-aff/dtd"><mml:mrow><mml:mi> α </mml:mi></mml:mrow></mml:math>-<mml:math altimg="si16.gif" overflow="scroll"></mml:math>
 xmlns:xocs="http://www.elsevier.com/xml/xocs/dtd"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://www.elsevier.com/xml/ja/dtd"
 xmlns:ja="http://www.elsevier.com/xml/ja/dtd"
 xmlns:mml="http://www.w3.org/1998/MathML", AEU - International Journal of Electronics and Communications, 2016.

Crossref

- | | | |
|----|--|-----------------|
| 16 | lib.buet.ac.bd:8080
Internet | 19 words — < 1% |
| 17 | wikimili.com
Internet | 19 words — < 1% |
| 18 | edoc.pub
Internet | 17 words — < 1% |
| 19 | Anum Mehmood, M. Usman Akram, Anum Tariq, Ayesha Fatima. "A Comparison of Mean Models and Clustering Techniques for Vertebra Detection and Region Separation from C-Spine X-Rays", Advances in Science, Technology and Engineering Systems Journal, 2017
Crossref | 15 words — < 1% |
| 20 | pees.feit.ukim.edu.mk
Internet | 14 words — < 1% |
| 21 | vdocuments.site
Internet | 14 words — < 1% |
| 22 | Santu Chakraborty, Sandip Bag, Subrata Pal, Alok K. Mukherjee. "Structural and microstructural characterization of bioapatites and synthetic hydroxyapatite | 13 words — < 1% |

using X-ray powder diffraction and Fourier transform infrared techniques", Journal of Applied Crystallography, 2006

Crossref

23 www.absoluteastronomy.com 13 words — < 1%
Internet

24 Aseem V. Borkar, Girish Chowdhary. "Multi-agent Aerial Monitoring of Moving Convoys using Elliptical Orbits", 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021 12 words — < 1%
Crossref

25 Lecture Notes in Computer Science, 2003. 12 words — < 1%
Crossref

26 Lorenzo Quirós Díaz. "Layout Analysis for Handwritten Documents. A Probabilistic Machine Learning Approach", Universitat Politecnica de Valencia, 2022 12 words — < 1%
Crossref Posted Content

27 Studies in Computational Intelligence, 2010. 12 words — < 1%
Crossref

28 www.trillia.com 12 words — < 1%
Internet

29 Baez-Cotto, Carlos M.. "Impact of Oil Loading on Lyotropic Liquid Crystal Phase Behavior of Carboxylate Surfactants.", University of Minnesota, 2020 11 words — < 1%
ProQuest

30 Branco, Cian Anthony. "Conical Orbital Mechanics: A Rework of Classic Orbit Transfer Mechanics", Old Dominion University, 2021 11 words — < 1%
ProQuest

-
- 31 Mellit, A.. "Artificial intelligence techniques for photovoltaic applications: A review", Progress in Energy and Combustion Science, 200810
Crossref 11 words — < 1%
-
- 32 dspace.plymouth.ac.uk
Internet 11 words — < 1%
-
- 33 ir.canterbury.ac.nz
Internet 11 words — < 1%
-
- 34 "Proceedings of the Eighth International Conference on Soft Computing and Pattern Recognition (SoCPaR 2016)", Springer Science and Business Media LLC, 2018
Crossref 10 words — < 1%
-
- 35 Mingqin Liu, Ashok Samal. "A fuzzy clustering approach to delineate agroecozones", Ecological Modelling, 2002
Crossref 10 words — < 1%
-
- 36 Sen, Novonil. "Wave Front Shape-Based Approach to Acoustic Source Localization in a Homogeneous Anisotropic Plate", The University of Arizona, 2020
ProQuest 10 words — < 1%
-
- 37 Souvik Biswas, Amiyangshu De, Amit Konar, Piyali Basak. "Effect of disturbance in working memory performace: An fNIRs study", 2017 Third International Conference on Biosignals, Images and Instrumentation (ICBSII), 2017
Crossref 10 words — < 1%
-
- 38 idoc.pub
Internet 10 words — < 1%
-

EXCLUDE QUOTES	OFF	EXCLUDE SOURCES	OFF
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	< 10 WORDS