University of Southampton

Faculty of Engineering and Physical Sciences

School of Electronics and Computer Science

FLOPTICS: A novel automated gating technique for flow cytometry data

by

Wiwat Sriphum

Supervisors: Dr Nicolas Green, Dr Gary Wills, Dr Daniel Spencer

January 2023

A thesis submitted in partial fulfilment for the degree of Doctor of Philosophy

University of Southampton

Abstract

Faculty of Engineering and Physical Sciences

School of Electronics and Computer Science

PhD Computer Science

FLOPTICS: A novel automated gating technique for flow cytometry data

by

Wiwat Sriphum

Flow cytometry (FCM) involves the use of optical and fluorescence measurements of the characteristics of individual biological cells, typically in blood samples. It is a widely used standard method of analysing blood samples for the purpose of identifying and quantifying the different types of cells in the sample, the result of which are used in medical diagnoses. The multidimensional dataset obtained from FCM is large and complex, so it is difficult and time-consuming to analyse manually. The main process of differentiation and therefore labelling of the populations in the data which represent types of cells is referred to as Gating: gating is the first step of FCM data analysis and highly subjective. Significant amounts of research have focussed on reducing this subjectivity, however a faster standard gating technique is still needed. Existing automated gating techniques are time-consuming or need many user-defined parameters which affect the differentiation to different clustering results. This thesis presents and discusses FLOPTICS: a novel automated gating technique that is a combination of density-based and grid-based clustering algorithms. FLOPTICS has an ability to classify cells on FCM data faster and with fewer user-defined parameters than many state-of-the-art techniques, such as FlowGrid, FlowPeaks, and FLOCK.

Acknowledgements

I would like to express my sincere gratitude to following people, without them this thesis would not be possible. I am extremely grateful to the great patience and valuable feedback from Dr Gary Wills and Dr Nicolas Green, who are my advisors. I am also deeply indebted to the generous support from the Royal Thai Government, who provided me the scholarship.

I am also thankful to my colleagues, for their moral support, motivation, and feedback. Thanks should also extend to the staff from the university, especially the School of Electronics and Computer Science, for their facilities and assistance.

Finally, I would like to mention my lovely family, parents, spouse and child. I would be remiss in not mentioning my father, who had waited for my graduation day, unfortunately, he has passed away during this process. Their confidence in me has kept my spirits and inspiration high throughout my study.

Declaration of Authorship

I, Wiwat Sriphum, declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

Title of thesis: FLOPTICS: A novel automated gating technique for cytometry data

I confirmed that:

1. This work was done wholly or mainly while in candidature for a research degree at the University of Southampton;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this always clearly attributed;
4. I have acknowledged all main sources of help;
5. Where the thesis is based on work done by myself;
6. Parts of this work have been published as:
    - Sriphum, Wiwat, Wills, Gary and Green, Nicolas (2020) FLOPTICS: A novel automated gating technique for flow cytometry data. 5th International Conference on Complexity, Future Information Systems and Risk, Vienna House Diplomat Prague Evropska 15 16041 Prague, Czech Republic, Prague, Czech Republic. 08 - 09 May 2020. pp. 96-102
    - Sriphum, Wiwat, Wills, Gary and Green, Nicolas (2022) FLOPTICS: A novel automated gating technique for flow cytometry data. International Journal of Organizational and Collective Intelligence (IJOCI), pp.1-21

Signed : _____

Date : 11 January 2023

Table of Contents

Table of Tables

Table of Figures

List of Abbreviations

1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional
AI	Artificial Intelligent
AIC	Akaike Information Criterion
AIDS	Acquired Immune Deficiency Syndrome
ART	Adaptive Resonance Theory
BIC	Bayesian Information Criterion
CD	Cluster of Differentiation
CFU	A Colony-Format Unit
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
EM	Expectation-Maximization
FCM	Flow Cytometry data
FLAME	Flow analysis with automated multivariate
FLOCK	Flow Clustering without K
FSC	Forward Scatter
GMM	Gaussian Mixture Model
HIV	Human Immunodeficiency Virus
ICL	Integrated Complete Likelihood
MinPts	Minimum Points
ML	Machine Learning
OPTICS	Ordering Points To Identify Cluster Structure
PDF	Probability Density Function
RBC	Red Blood Cell
SOM	Self-Organizing Map
SSC	Side Scatter
SSE	Sum Squared Error
WBC	White Blood Cell

Chapter 1 Introduction

Today's computer technology has become an essential technology that is involved in many aspects of human life. Artificial Intelligence (AI) and Machine Learning (ML) are technologies that have played an important role in various fields of science. In the health and medical field, it is another field that is taking advantage of AI and ML technology. Recently, these technologies are inevitably being used in the field of health and medicine. Machine learning technology assists doctors to diagnose faster and more accurately. Blood testing is an important process for diagnosis because blood is an important medium for carrying water, nutrients, germs, toxins, and other foreign substances into the body. Therefore, blood testing is another method that can detect contaminants that invade the body with high accuracy. Flow cytometry, which is discussed in detail in Chapter 2, is a technology used to measure and analyse the physical characteristics of cells in blood sample such as the size of the cell, the complexity of the elements within the cell. The author therefore had the idea to develop a machine learning method to assist diagnose blood data more accurately and higher time efficiency.

Early detection of some categories of infection is able to increase the chance of successful treatment. In 2015, more than 360,000 cancer cases were diagnosed in the UK, representing some 183,000 males and 177,000 females (Cancer Research, 2018). Besides cancer, many serious infections attack people, such as HIV and leukaemia. The World Health Organization revealed in 2016 that more than half of worldwide deaths are accounted for by ten causes divided into 3 categories as follow:

1) Non communicable
2) Communicable, maternal, neonatal and nutritional conditions
3) Road injuries

The first category includes with ischaemic heart disease, stroke, chronic obstructive pulmonary diseases, Alzheimer's disease and other dementias, bronchial cancers, and diabetes mellitus. The second category includes lower respiratory infections, diarrhoeal diseases and tuberculosis. For low-income countries, the top five causes of death in 2016 were lower respiratory infections, diarrhoeal diseases, ischaemic heart disease, HIV/AIDS, and stroke (WHO, 2018). Successful treatment depends on many factors. Early detection of the diseases is one of the factors that can increase the chance to save patient's life.

In order to detect those disease, the most two important parts are medical instruments and expertise. Sometimes, even if the powerful instruments or high performance technologies are provided, the disease might not be detected by staffs with not enough experience and knowledge. On the other hand, when there are many high potential medical expertise but lacking of proper mechanical instruments, the disease might also not be discovered. For example, flow cytometry is

a powerful medical technology that is used for measuring the characteristics of cell in blood sample such as the number of individual type of cells, size of cells, complexity, which are components in cell such as different types of proteins, and granularity, which is a measurement of the internal complexity of cell (Müller and Nebe-Von-Caron, 2010). Although this technology provides high quality of examination results, the staff and specialist analysing this information obtaining from the flow cytometer must have advance training and experience. Therefore, in case of requiring high experience and advanced knowledge for analysing data, it would be great if there is an application that can support staff analyse more convenience and detect infections from health examination results obtaining from a medical instrument. Additionally, if this application is developed, it could help a lot of patients by detecting conditions early and warning them, before they lead to mortality.

Blood analysis is an important process in order to detect conditions and diagnose, as blood is an important intermediary for carrying water, nutrients, germs, toxins, and other substances into the body (Sun *et al.*, 2017). At the same time, it also serves to receive and carrying various substances that the body releases. These substances may enter the body by drinking, breathing, penetrating through the wound, through the skin into the bloodstream (Kuhn *et al.*, 2017). Therefore, blood analysis is a method to detect contaminated substances that enter the body correctly and promptly. Normally, when feeling unwell, most people tend to see a doctor to discuss or find the cause of the illness. In which the doctor will perform history taking and basic physical examination, which includes a blood test to determine the root cause. For these reasons, the benefits of blood analysis are as follows:

Diagnosis of certain diseases that cannot be diagnosed using conventional methods or external physical examination but it can be diagnosed by using blood tests (Tassinari *et al.*, 2018). As well as it has an ability to distinguish the illness behaving similar symptoms, such as pancreatitis, food poisoning, gastritis, and enteritis.

The data obtained from blood tests can be used to monitor treatments such as, monitoring the amount of medication in the blood in order to control the medicine level appropriately, including, use for diet control by detect the number of glucose in blood (Razzak *et al.*, 2018; Shaked *et al.*, 2019).

In some cases, patients infected with virus or disease have no symptoms. If the patients have a blood test, it may be able to identify underlying diseases such as AIDS in the initial stages. Including with various diseases that do not show symptoms in the initial stages (Alexander, 2016).

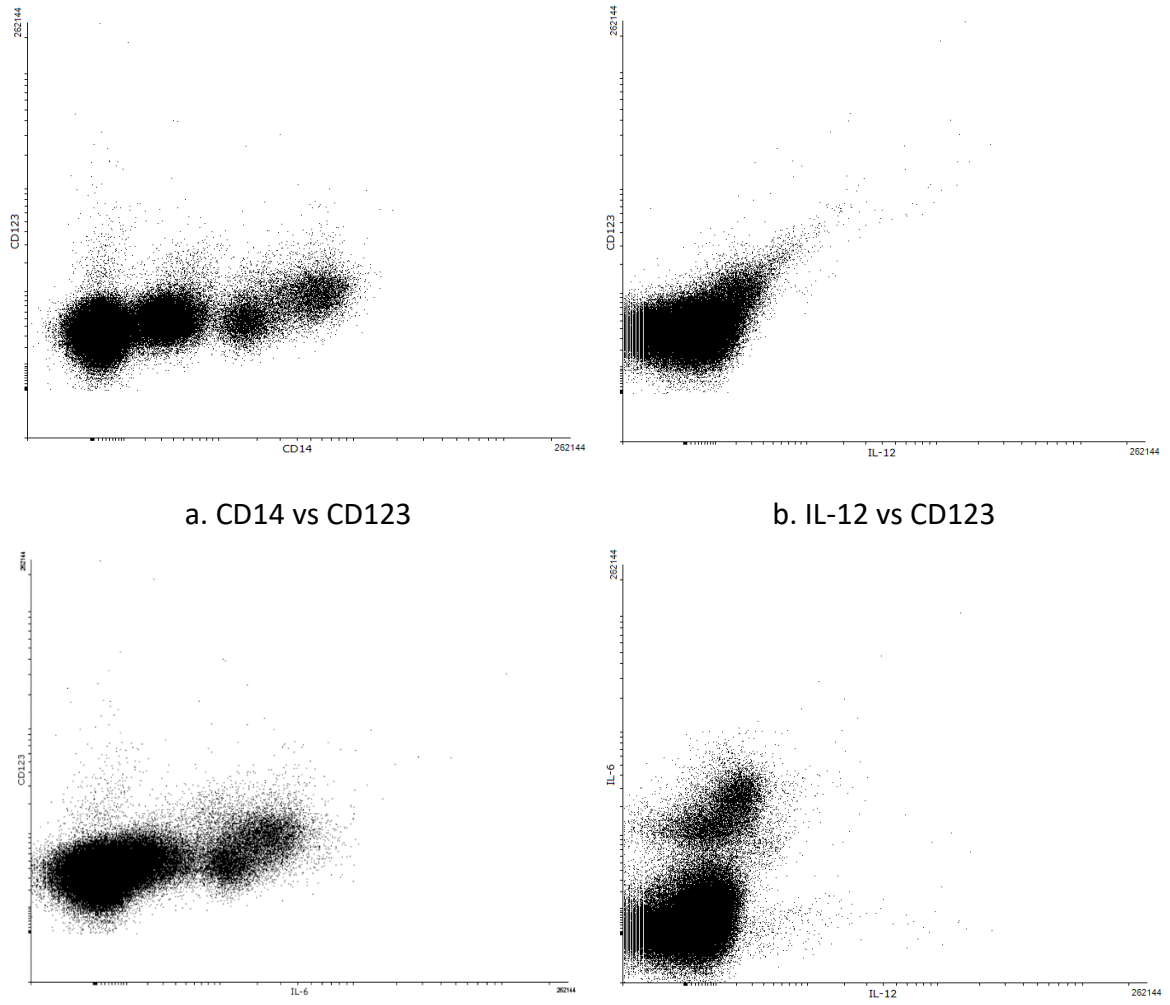
In addition to detecting contaminated substances in the blood sample, blood cell count can also identify donors' state of health.

Blood cell count is to examine the quantity and characteristics of all three types of blood cells, including red blood cells, white blood cells and platelets. Red blood cells (RBC) have a shape of doughnuts (Kuhn *et al.*, 2017). Since the centre has a dimpled but not penetrating, approximately 7.5 to 8.7 microns in size (Diez-Silva *et al.*, 2010). They are created in bone marrow, which is the core of the bones throughout the body. Haemoglobin, which is a substance in RBC, acts to carry oxygen from lungs to different parts of the body and remove carbon dioxide from different parts of the body to be excreted at the lungs (Diez-Silva *et al.*, 2010). White blood cells (WBCs) are larger than red blood cells. Normally, WBCs count is between 4,500 and 10,000 cells per 1 microliter (Sautbine, Voslar and Bancud, 2011). In the body has many types of WBCs that serves to resist and destroy germs or foreign objects entering the body so the number of WBCs fluctuates depending on the inflammation and infection in the body. Lower WBCs is resulting in a chance of being infected easily, on the other hands, if the body has more WBCs than usual it means that the body responds to inflammation or infection in the body (GEORGE and PANOS, 2005; Sautbine, Voslar and Bancud, 2011). The increased amount is usually not very high, generally not more than 30,000 cells per microliter. Platelets is a part of the blood cell, which their size is very small compared to RBC and WBCs. They are responsible for preventing bleeding from blood vessels. The number of platelets is approximately 100,000 to 400,000 cells per microliter. According to individual cell types have different role, therefore, blood cell count can be used to detect and diagnose the patient's state of health. Normally, physicians count individual blood cells by naked eyes through a microscope. Nowadays, there is a technology called flow cytometry that was developed to assist the counting of blood cells in blood sample.

Flow cytometry (FCM) technique uses the concept of cell-scatter measurement and light emission after receiving laser beam stimulation (Bio-Rad, 2018a). It has been widely applied in medical research, especially in haematology and immunology (Jahan-Tigh *et al.*, 2012). It has been broadly adopted in the clinical environment in order to diagnose and monitor treatments such as: leukaemia diagnosis and chemical healing responsiveness, stem cell transplantation monitoring, supporting the decision to distribute anti-viral medicines to HIV patients, allergic substance diagnosis using a basophil activation testing, and detection of anomalous T cells and B cells by measuring individual lymphocytes. FCM evaluates individual cells based on the measurement of photon scattering and light emission in response to an external laser beam (Wood, 2021). This technique was developed more than 40 years ago. At first, there were many limitations because of the huge instrument, its complicated use, and expensive maintenance (Shapiro, 2013). FCM technique has greatly improved and is more convenient to use than earlier and now it has an ability to evaluate a variety of cell features, currently 20 from one measurement (Vembadi, Menachery and Qasaimeh, 2019). When doctors suspect a patient has some category of disease, such as

cancers, diabetes mellitus, HIV/AIDS and leukaemia, they may test the patient's blood sample to measure the number of red blood cells, white blood cells and platelets, which is the traditional method of detecting cancer (Fleisher and Oliveira, 2019).The main objective of FCM is to identify proportion of individual cells type because the proportion and blood cell count allows expertise to recognise patient's state of health. The details of this technology and blood cell analysis are explained in Chapter 3. However, the data obtains from this instrument is normally large and complex that make difficulty for expertise to analyse them.

The FCM technique provides multi-dimensional data, including relative size, relative granularity, and relative fluorescence intensity (McKinnon, 2018). The data is so complicated that it is difficult to manually detect anomalous patterns of cells. Regularly, FCM data analysis can be divided into two primary processes including classification of homogeneous cells into clusters called "Gating" process and interpretation of the information providing from gating process to identify a donor's state of health. This research focuses on gating process because it is highly subjective and is based on the expert's experience working with this data, and is manually time consuming (Lo, Brinkman and Gottardo, 2008; Verschoor *et al.*, 2015). The objective of the gating process being the separation of cells with similar descriptions or homogeneous cells into the same clusters, and selection of cells of interest from a 1D plot, which is called histogram, and a 2D scatter plot. Basically, FCM data is performed as a multi-dimensional of large amount of individual cells in each sample so it can cause a difficulty in manual gating process (Verschoor *et al.*, 2015). When clinician start to create gates, they have to select two features of cells represented in scatter plot and draw polygons on them. However, for gating cells, more than two features may be considered which means that clinician have to switch back and forth between many features, as can be seen in Figure 1.1. Because of this difficulty, many automated gating techniques have been developed but these applications still have some troubles such as time consuming and high dependence on user-defined parameters. The details and limitations of some existing automated gating technique are explained in Chapter 4.



a. CD14 vs CD123

b. IL-12 vs CD123

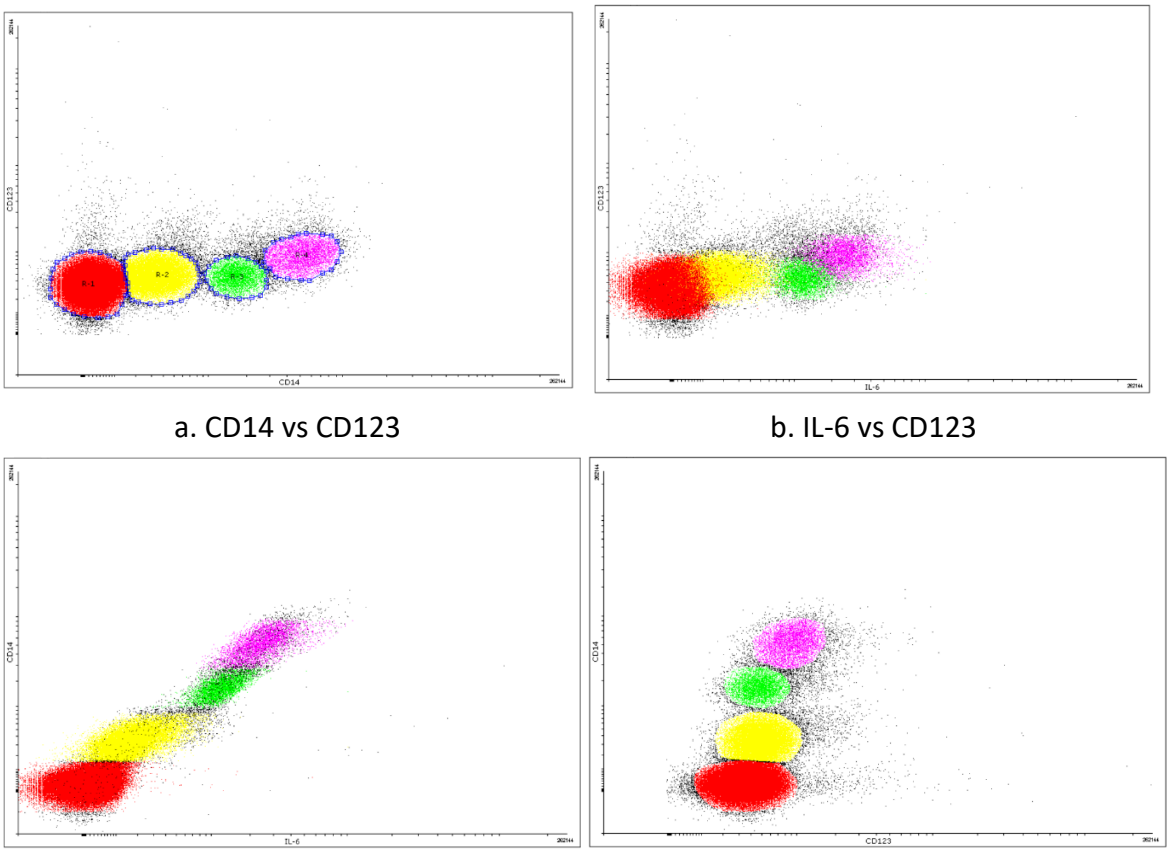
c. IL-6 vs CD123

d. IL-12 vs IL-6

Figure 1-1: Scatter plot with different parameters of FCM data by using file a2006_O1T2pb05i_A5_A05.fcs from https://flowrepository.org

After obtaining the FCM data from flow cytometer, the first step that clinicians have to do is gating. Manual gating is to create gates by drawing polygon on 2D scatter plot in order to separate the different types of cells and discover the proportion of each cell type. For example, if the clinicians would like to create gates from Figure 1.1-a, which represents the scatter plot of CD14 against CD123, cells might be separated into 4 clusters, as in Figure 1.2a. If parameter IL-6 need to be considered, clinicians have to change parameters to see a new perspective, as can be seen in Figure 1.2b., 1.2c., and 1.2d. For this reason, many automated gating applications have been developed for supporting clinicians but there is no application is able to properly create gates or separate individual cells so far. This research is expected to result in a novel clustering technique that will support physicians and experts in identifying cell clusters that could reduce time consuming. Therefore, this research aims to improve clustering techniques by combination the advantages of OPTICS, which requires a few number of parameters, and FLOCK, which is faster than many state

of art algorithms and is composed for the flow cytometry data. These two techniques that are explained in Chapter 3 and Chapter 4.



a. CD14 vs CD123 b. IL-6 vs CD123

c. IL-6 vs CD14 d. CD123 vs CD14

Figure 1-2 Example of manual gating by using file a2006_O1T2pb05i_A5_A05.fcs from https://flowrepository.org

1.1 Problem Statement

The challenge of this research is that clinical agreement on the result from the flow cytometry analysis is inconsistent. The flow cytometry analysis is based on the clinician's experience of working on flow cytometry data analysis, giving rise to different interpretations of results.

The first step in the analysis is the process of manually creating 'gates' by drawing shapes around cells in a 2D scatter plot to categorise cells of interest, i.e., identify homogeneous cell populations that satisfy a distinct function. The manually gating process is thus highly subjective (Bashashati and Brinkman, 2009). Therefore, the diagnosis can be reported differently and lead to different clinical decisions, depending on the clinician's experience of working with these data.

Furthermore, the data will vary according to the flow cytometry measurement and experiment design. For example, when clinicians are interested in different infections, they will stain the same

sample with different fluorophores to highlight the surface molecules of a cell, called markers, which results in different data.

Although many automated gating techniques have been developed, there are still problems, the main one being the many user-defined parameters that need to be entered, which again are based on the clinician's experience in selecting these parameters. This results in a time-consuming process as the analysis is rerun under different parameters.

This research focuses on the data analysis from the flow cytometry. The conjecture is that, through the use of machine learning, it is possible to provide the efficiency gating result in the data from a flow cytometer less time and requires fewer user-defined parameters than current techniques allow.

The research question is therefore: *How can machine learning techniques be applied to discover appropriate clusters of cell in flow cytometry data?*

1.2 Aim and Objectives of Project

Aim

To improve the existing automated gating techniques in order to increase the performance that are fast, powerful, and simple to use by allowing the clinicians to enter fewer parameters so they need less time to obtain results than current techniques.

Objectives

1. To identify the performance of clustering techniques , especially, the techniques that are often applied to flow cytometry data
2. To review the performance (accuracy, running time, the number of user-defined parameters) of current automated gating techniques
3. To select the current technique that provide the best performance as a candidate for improvement
4. To extract the core clustering techniques that are used in the candidate automated gating technique
5. To ascertain the strength and weakness of the core clustering techniques
6. To generate synthesis datasets, they are generated with the shape of clusters that is similar to the shapes that often found on blood cell sample obtaining from flow cytometry data, such as mouse dataset (spherical shape with different radius), barcode dataset (long sticks shape), and imitative dataset (imitate from real sample cluster shape).

7. To evaluate the performance of each core clustering techniques and the candidate automate gating technique by conducting base-line experiments with synthesis datasets that often found on FCM data.
8. To implement a novel clustering technique that is faster and requires less user-defined parameters than current techniques.

1.3 Publications

The following journal articles and peer-reviewed conference papers based on this thesis had been accepted for publication.

1. Sriphum, Wiwat, Wills, Gary and Green, Nicolas (2020) FLOPTICS: A novel automated gating technique for flow cytometry data. 5th International Conference on Complexity, Future Information Systems and Risk, Vienna House Diplomat Prague Evropska 15 16041 Prague, Czech Republic, Prague, Czech Republic. 08 - 09 May 2020. pp. 96-102.
2. Sriphum, Wiwat, Wills, Gary and Green, Nicolas (2022) FLOPTICS: A novel automated gating technique for flow cytometry data. International Journal of Organizational and Collective Intelligence (IJOCI), pp.1-21

1.4 Thesis Outline

Chapter 2 reviews the literature about the techniques use for blood cell analysis and blood cell count that used to evaluate the overall health and detect wide range of infections. It also discussed about manual gating technique that is the first step of flow cytometry data analysis.

Chapter 3 discusses the traditional data clustering techniques, compares drawbacks and benefits of each techniques.

Chapter 4 discusses the automated gating techniques and compares drawbacks and benefits of each technique.

Chapter 5 shows the experimental results of various clustering techniques applying to different datasets in order to discover the most appropriate technique to improve further.

Chapter 6 presents a new flow cytometry analysis technique which is called FLOPTICS.

Chapter 7 summarise and draws conclusions, suggesting future work.

Chapter 2 Blood Cell Analysis

Blood is a vitally necessary fluid in the body, and it can be used for identifying the state of health of the person. Blood cells perform a number of important functions which are implemented by different types of blood cell. Normally, blood is about 7% of the body's weight, at roughly 5 litres, and circulates to service every cell (Albert *et al.*, 2002; Robinson *et al.*, 2018). Blood cells are produced in bone marrow and they are valuable constituents that prevent every cell in the body from being infected, and preserve every cell in the body (Rye *et al.*, 2017). Human bodies are elaborate multicellular organisms that require a transporting mechanism to deliver oxygen and nutrients to tissues, and remove body waste such as carbon dioxide and toxins from the entire body (Rye *et al.*, 2017). If the number of blood cells is significantly lower or higher than normal for that individual, it means that the body is facing some health problems. This is why the circulatory system is so important and can be used as a factor in identifying human well-being.

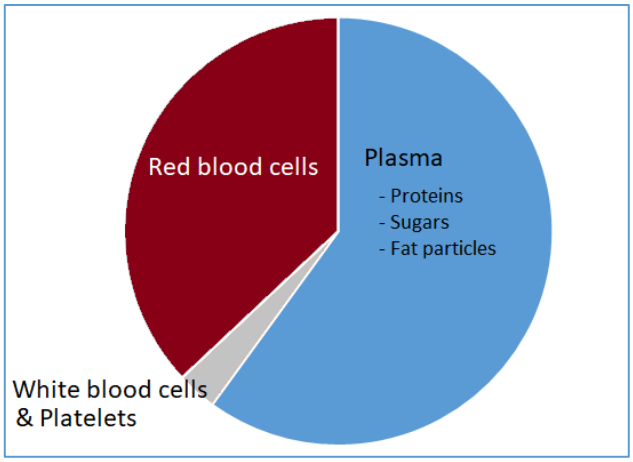


Figure 2-1: Proportion of blood constituents

2.1 Blood components

The blood components come in three units (Figure 2.1): plasma, which is mainly water (about 55% of the blood), red blood cells (also called erythrocytes), and white blood cells (WBCs – also called leukocytes) and platelets (ASH, 2009; Robinson *et al.*, 2018). Plasma is the most common constituent in blood, and contains an abundance of essential substances such as proteins, sugars, fat particles, and nutrients that will be delivered to every cell in the body (Dean L., 2005; Farley, Hendry and McLafferty, 2012). Erythrocytes make up approximately 45% of all blood composition. The main responsibility of erythrocytes is carrying oxygen to every cell in the body. Only around 5%

of the blood is WBCs and platelets. The major responsibility of white blood cells is fighting infections, diseases and other foreign harmful substances that invade the body, by ingesting and destroying them. If the number of WBCs in blood is significantly different from normal, it can be assumed that there is an invasion of disease. The WBC count, therefore, is a key indicator for the immune system in the body. The responsibility of platelets is to reduce blood loss from the body in the case of injury by supporting the body to form clots in order to stop bleeding. Although there are three types of blood cell, the main factor that physicians use for diagnosis is WBC count because they play such an important role in the process of protecting the body against disease and are a part of the immune system.

Physicians generally use the WBC count in a blood sample to diagnose illness, because WBCs are a key component in the immune system so it can be used to detect the person's well-being (King, Toler and Woodell-May, 2018). WBCs make up approximately 1% of blood volume and have 5 different types: Basophil, Eosinophil, Neutrophil, Monocyte, and Lymphocyte. The normal range of the WBC count is between 4,000 – 11,000 cells/µL (Kabat *et al.*, 2017). Each type of WBC has an ability to tackle a specific infection, so the physician usually diagnoses the illness by evaluating the WBCs, especially those involving immune diseases. In other words, if a number of any type of WBC is less than normal, it could mean that the body might not be able to fight the invading infections as well, which could have been eliminated by that type of WBC. Conversely, if a WBC count of any type is higher than normal, it could mean that the body is currently fighting the infections, which is the responsibility of that type of WBC (Kabat *et al.*, 2017). Since WBCs are the body's main defence against infection, it needs to have enough WBCs to fight infections and keep the body healthy. However, if the number of WBCs in the body is higher than normal, it may indicate that the body is infected because the immune system is producing more WBCs than usual. The WBCs make antibodies for attacking a specific infection to dispose of it. For that reason, a high volume of WBCs can be a sign of infection attacking the body. However, manually counting WBCs is a time-consuming and highly subjective in the process of classifying types of WBC using flow cytometry data. The application of machine learning, to support physicians classifying homogeneous cells and analysing the relative numbers of individual blood cells by type in blood sample, could possibly allow them to diagnose some kind of illness faster and more conveniently, and reveal the donor's state of health.

2.2 Blood cell classification

In order to count individual cells, specific properties of the cell type are required, such as size, shape, and complexity. However, some types of cell cannot be differentiated morphologically because they are similar in appearance, such as T-cells and B-cells (Dickinson, 2002; Nagel *et al.*, 2014). Fortunately, these lymphocytes express different types of cell surface molecules, called *cell surface markers*. For example, T-cells and other lymphocytes, such as natural killer cells and B-cells, cannot be discriminated by physical appearance, but T-cells have a unique receptor on the T-cell surface and this receptor is not found in other lymphocytes. This is why scientists developed cell staining for the detection of a single cell under the microscope. These particular molecules on individual cells are called *markers* because they are used to identify different cell populations. The widely-used method for identifying these cells surface markers is thus called Cluster of Differentiation.

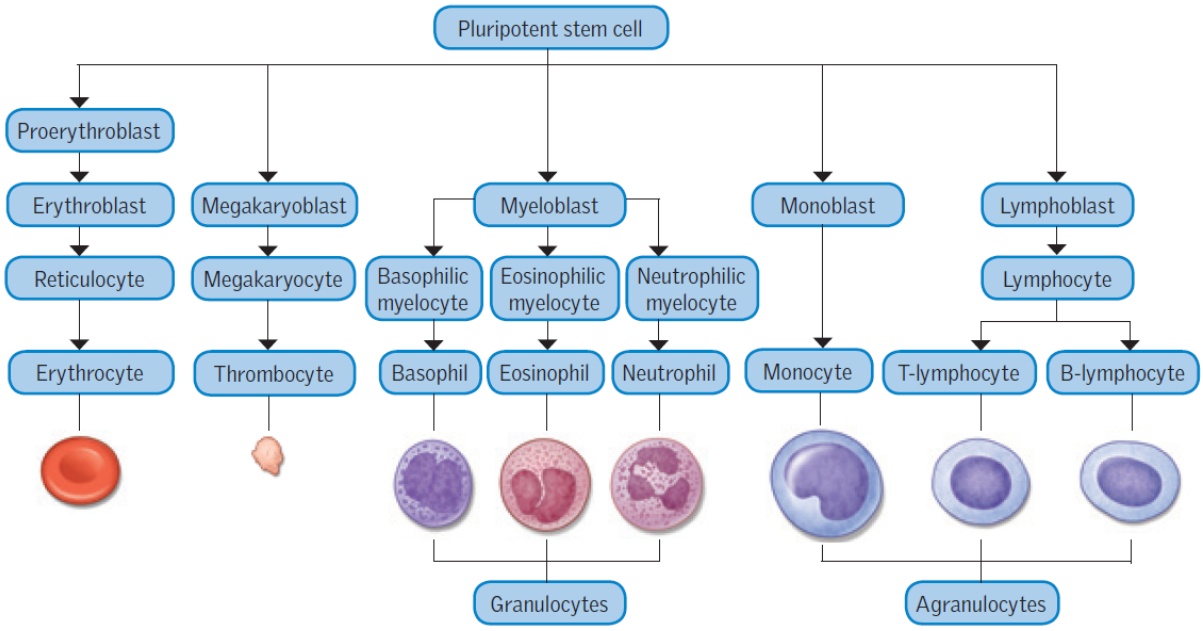


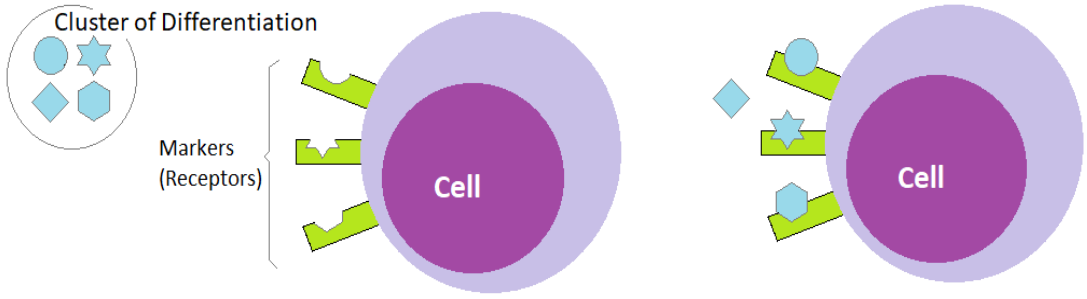
Figure 2-2: Stem cells can develop into many different types of cell (Farley, Hendry and McLafferty, 2012)

Figure 2-2 presents the development structure of blood cells. Every cell originates from a stem cell then become red blood cells (Erythrocytes), white blood cells (Lymphocytes, Monocytes, Neutrophils, Basophils, and Eosinophils), and Thrombocyte or platelets.

2.2.1 Cluster of Differentiation marker

Cluster of Differentiation (CD) marker is a group of monoclonal antibodies, which are used for the identification of cells by staining them on the cell surface, and which can then be classified by

scientists. The monoclonal antibodies are specific for each different epitope on a protein molecule, which is also known as marker or receptor, that belongs to a cell and is found on the surface of individual cells (Kalina *et al.*, 2019). The CD marker (or CD antigen or CD molecule) is combined with the same protein molecule on the cell surface, as seen in Figure 2-3, and can then be detected by biological instruments, such as a microscope. Figure 2-3 was created on the principle that markers can only bind to identical proteins on the cell surface in order to presents the basic concept of combination between receptors on the cell surface and antibodies or CD. Only the receptor and the CD marker with a compatible structure are combined together. Most of these molecules are proteins and they include a large number of cell surface receptors, signalling molecules, and adhesion molecules. These cell surface markers are designated by the prefix CD follow by a number. The number represents the order in which the cell surface marker was discovered. CD markers are useful in identifying and characterising many different types of leukocytes or WBCs. They are particularly useful for the identification of WBCs using flow cytometry (FCM), as explained in Section 2.3 below.



a. before combination

b. after combination

Figure 2-3 : The concept of combination of monoclonal antibodies (CD) with cell surface markers (Kalina *et al.*, 2019; Shahrabi *et al.*, 2020)

2.2.2 White blood cell classification using CD Markers

CD markers are very useful for the identification of many different types of cell, and can identify cell lineage, and also their stage of maturation. Markers can identify a cell lineage, e.g. CD3 is a cell surface marker which is expressed only on a T-cell, whereas CD19 is expressed only on a B-cell (Dickinson, 2002). For stage of maturity, CD1 is found on thymocytes (immature T-cells) but not on mature T-cells. In this way, scientists have used CD markers to count particular cell types and analyse them to identify state of health from the blood sampled. However, there is usually more than one marker on a cell, while the same markers might appear in other types of cell, as shown in Table 2-2 (abcam, 2019).

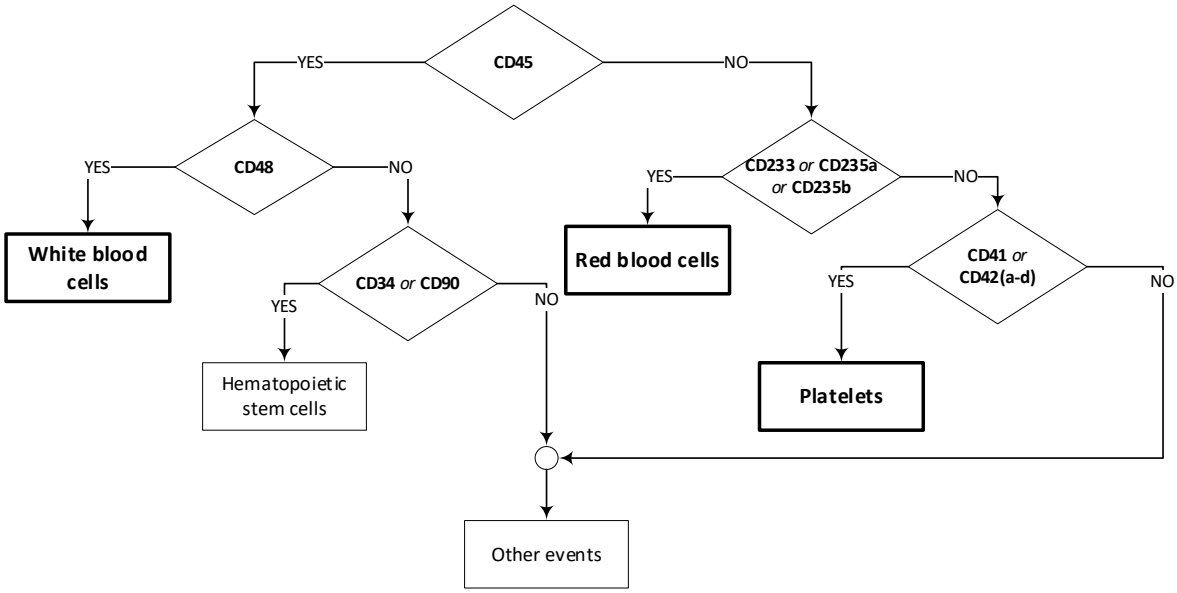


Figure 2-4 : 3-part classification using CD markers based on part-information in Table 2-2 (abcam, 2019)

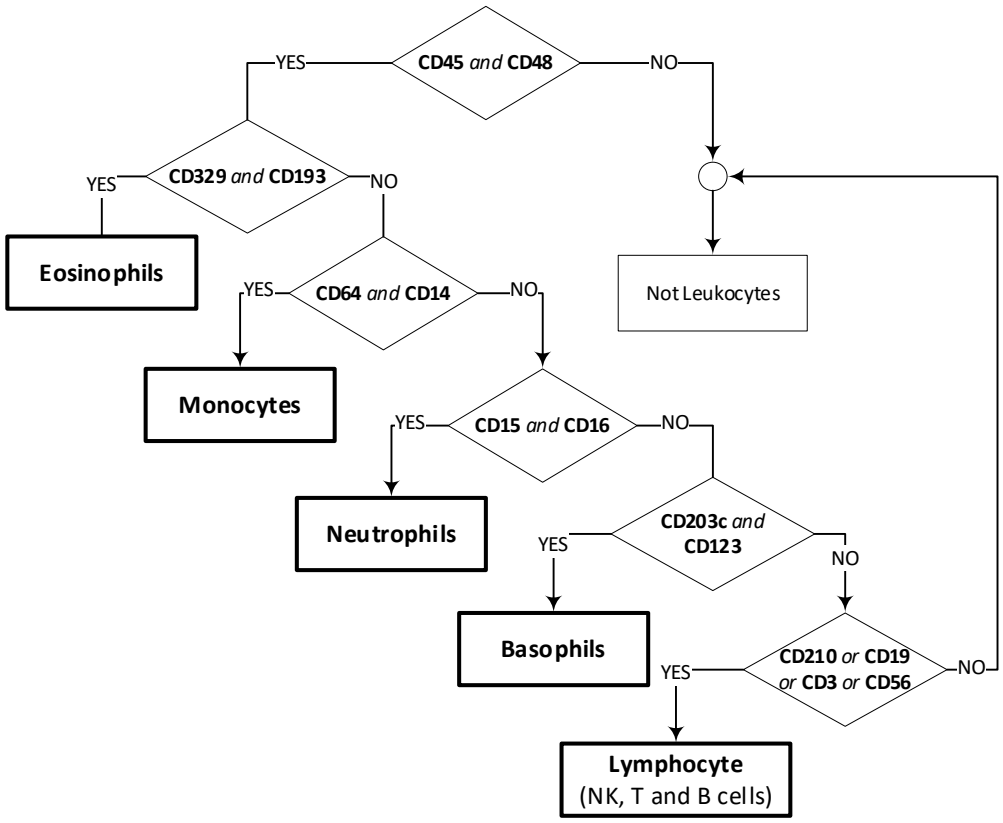


Figure 2-5 : 5-part classification using CD markers based on part-information in Table 2-2 (abcam, 2019)

Table 2-2 shows that cells will be considered as stem cells when at least CD34 is expressed, while other cells are found to be a bit complicated in classification. If CD45 is found, it can be identified as a WBC, but of what kind cannot be determined until the specific markers are expressed. Figure 2-4 gives an example of a 3-part classification (red blood cell, white blood cell, and platelet), while Figure 2-5 gives a 5-part classification (all white blood cells including T-cell, B-cell, NK cell, thrombocyte, and granulocyte) using CD markers.

2.3 Fluorescence

Scientists have developed CD or monoclonal antibodies that have different reactions to external light. These monoclonal antibodies combine with the specific cell surface markers, and enable scientists to differentiate cells by reflecting light from them. This technique is called fluorescence and is the result of a process by which molecules absorb light and emit energy as photons (Caarls *et al.*, 2010). Molecules that have the ability to absorb and emit light energy are also called fluorophores or fluorescent dyes. In other words, the fluorophore is a molecule that is capable of fluorescing. Figure 2-6(a) shows the molecule in its ground state, a relatively low-energy and stable configuration. In Figure 2-6(b), the fluorophore molecule is hit by light from an external source and its energy is absorbed. When enough light energy has been absorbed, the molecule will reach a higher energy-state, called its excited state. The fluorophore is unstable in this excited state, so it repositions from the excited state to the ground state by releasing the excess energy as light. In consequence, scientists can interpret the light emission and thus identify the type of cell. In this way, fluorescence and CD markers have been applied together in a technique called *flow cytometry*, to determine the type of blood cell.

2.4 Flow cytometry (FCM)

FCM is one of the techniques use for cell identification. It is the combination of fluorescence with cell surface markers. Researchers need to analyse the relative number of distinct cell types to identify particular infections, because the proportion of individual cell types can be a function of the infection state, especially the 5 types of WBC. Normally, cell counting can be undertaken manually counting cells (e.g. haemocytometer, plating and CFU counting), by automated cell counting (e.g. electrical resistance, image analysis, and FCM), and by indirect counting (spectrophotometry and impedance microbiology). However, this work focuses on FCM, which provides much useful data, such as the shape of cells, their internal and external structure, and other characteristics.

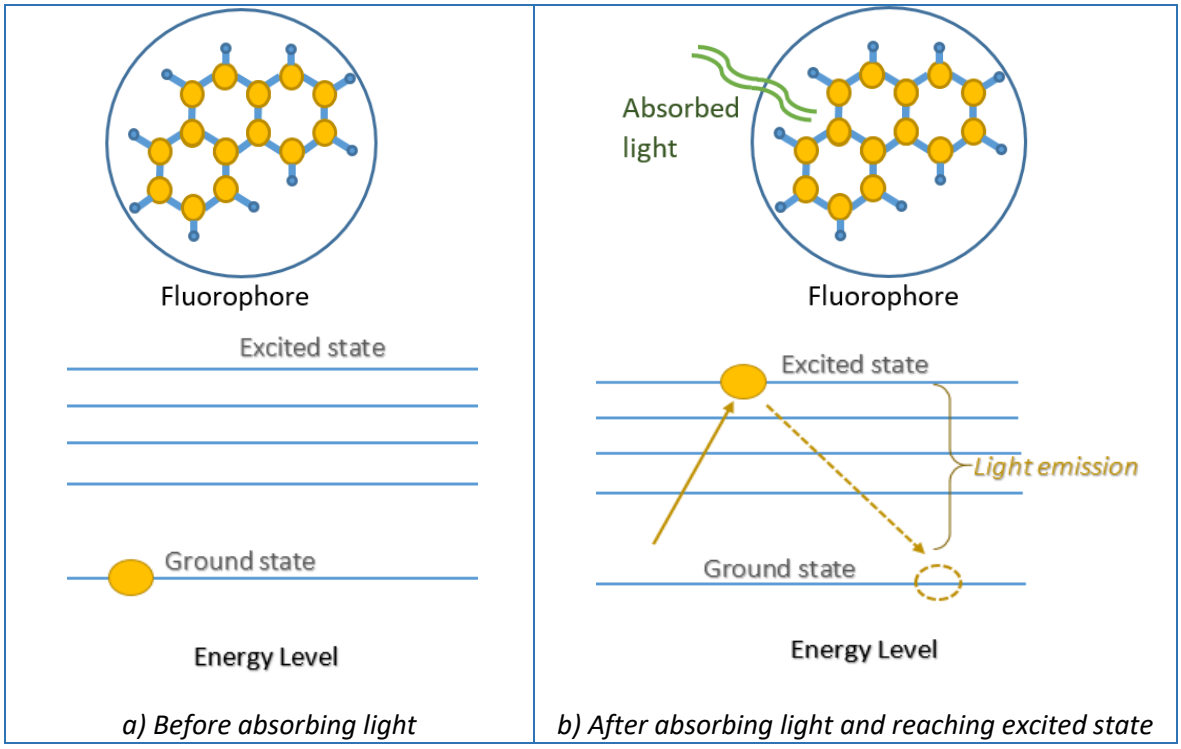


Figure 2-6 : Concept of fluorescence (Caarls *et al.*, 2010)

Flow cytometry measures individual cells by passing them into a stream of fluid and flowing that through at least one laser beam. Cell properties that can be measured include relative size, relative granularity, and relative fluorescence. This technique was developed over 40 years ago, but was limited initially because the cytometer was too large, difficult to maintain, and was an expensive instrument. As with much measuring equipment, FCM is now more accurate, cheaper, and more convenient to use, and hence is widely applied in clinical research, particularly haematology and immunology. In addition, FCM has also been used to diagnose and monitor patient treatment procedures such as cancer, leukaemia, HIV, allergies, and heart disease.

2.4.1 Flow cytometry technique

Flow cytometry is able to detect cells from 0.2 microns to 150 microns in diameter, but the actual capability depends on the equipment used (Rowley, 2019). In the process, cells are suspended in a fluid and driven into the centre of a narrow nozzle, whence they pass through one or more laser beams, a single cell at a time. Light scattering and fluorescent emission from every cell is sensed by a detector. Light scattered by less than 5 degrees is called forward scatter (FSC) and is used to identify the size of cells, while larger deflections are called side scatter (SSC) and are used to measure granularity, and membrane roughness, also known as cell complexity (World Health Organization, 2009). Since cell particles are tagged by known specific monoclonal antibodies on the cell surface marker, the result of the light emission will be read and analysed by the flow cytometer.

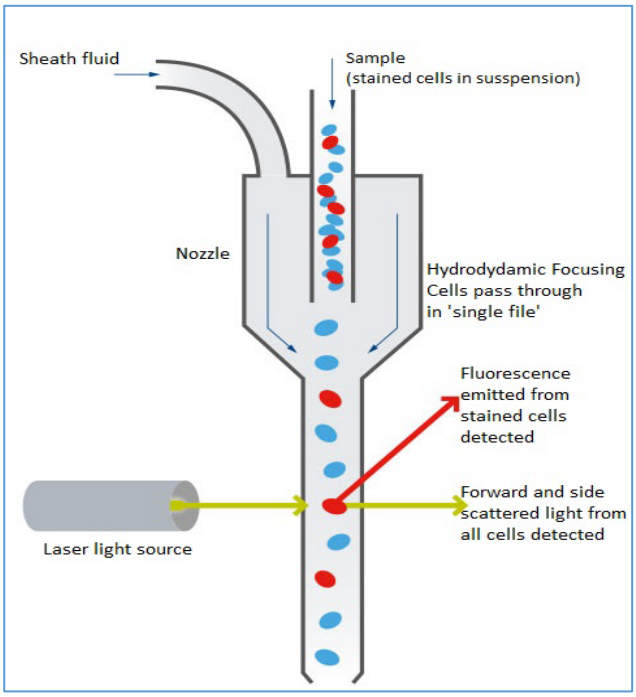


Figure 2-7 : Flow cytometer concept (abcam, 2019)

When cells with different fluorophores pass through the laser beam, SSC is reflected through a lens and a filter, and is detected by a photomultiplier, which converts it into a digital signal. The digital signals are then processed and interpreted, allowing cells of interest to be distinguished from the others.

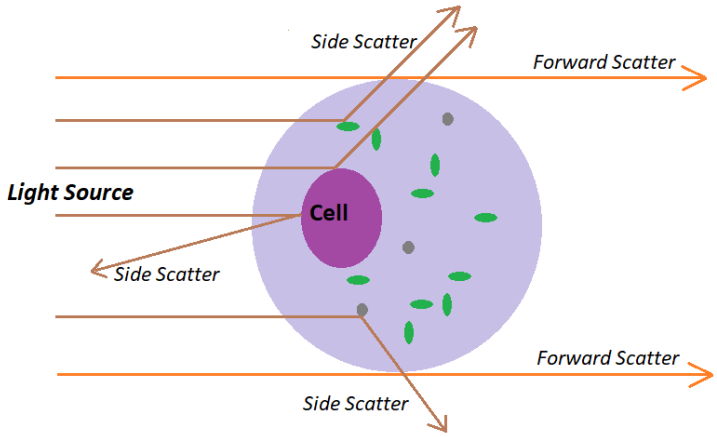


Figure 2-8 : Forward scatter and Side scatter (Rowley, 2019)

The selection of fluorophores to analyse cell characteristics depends on the flow cytometer's ability to measure the wavelength of light. Therefore, a flow cytometer that is able to detect the specific wavelength and differentiate individual cells, is considered the most useful (Bio-Rad, 2018b). The process of cell differentiation is called *gating*, and is based on their forward and side scatter

characteristics. An elementary cytometer can differentiate only 2-3 colour wavelengths, and that is adequate for simple clinical analysis such as measuring T-helper cells, by staining lymphocytes with the 3 markers CD3, CD4 and CD8 (Dickinson, 2002). Higher specification instruments are able to detect a wider range of wavelengths and allow more than 13 colours to be examined.

2.4.2 Characteristics of the data

After blood had been passed into the flow cytometer for measurement, a dataset of cells features is provided, which are represented by a large collection of numbers. This dataset follows the Flow Cytometry Standard (FCS) file format. An example of a flow cytometry dataset (FCM) is shown in Table 2-1. Each row of the table corresponds to each cell processed, and each column corresponds to its parameters. Up to 50 cell parameters may be provided, but the number depends on the capacity of the flow cytometer and the experiment design (Lee *et al.*, 2017).

The numbers in Table 2-1 refer to the intensity of fluorescence of the markers that had been combined on an individual cell surface. The signal intensity can be used to identify the type of cell, and the number of cells for each cell type. Obviously, FCM data is a high-dimension dataset, and the data generally exhibits only one or two parameters for analysis because it is really hard for people to analyse objects in three-dimensional space, and they have difficulty imagining what more than three dimensions looks like. Therefore, the data is displayed in 1D or 2D format in order to make analysis easier.

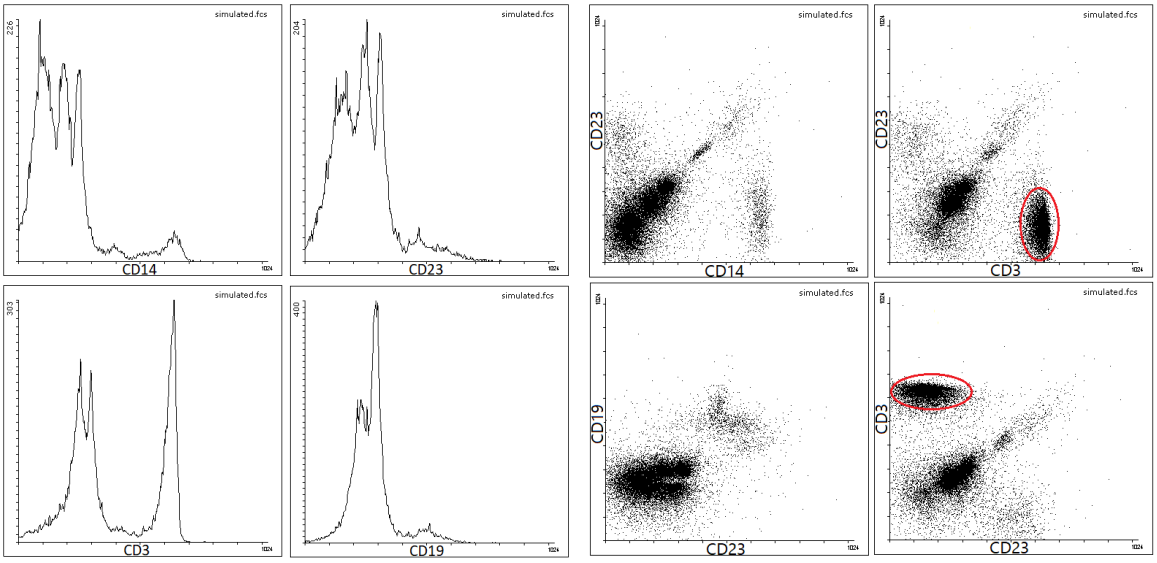
The data in Table 2-1, which were plotted and showed in Figure 2-9, was downloaded from https://flowrepository.org/public_experiment_representations and file id FR-FCM-ZZKA (by Richard Scheuermann) was used as an example.

Table 2-1: Flow cytometry data example

Cell numbers	CD14	CD3	CD23	CD19
1	50	244	38	254
2	132	72	173	206
3	129	257	142	271
…	…	…	…	…
20000	397	571	341	315

A single parameter plot may be displayed as a histogram, which shows the number of cells at a specific measurement value. The y-axis corresponds to the intensity of the signal of the chosen parameter while the x-axis shows the number of cells, as shown in Figure 2-9(a), which represents the number of cells and signal intensity of responses to markers CD14, CD23, CD3 and CD19.

Alternatively, two parameters can be displayed simultaneously as a scatter plot, with the two parameters being displayed on the two axes. Each axis shows the intensity of one parameter and each dot in the space corresponds to one cell. Figure 2-9(b) shows the relationship between two parameters; for the scatter plot between CD23 and CD3, the cells or the events that lie within the red ellipse boundary might be considered as mature T-cells and thymocytes because they express high intensity of CD3 based on information in Table 2-2 (abcam, 2019).



(a) Histogram (b) Scatter plot

Figure 2-9 : FCS data representation using Flowing Software 2

Since the FCM data is high dimension dataset, it is really hard to analyse manually. The clinician has to select only 2 or 3 parameters for viewing and analysing, but sometimes needs to compare more than 3 parameters, which could be a cause of confusion and lead to a time delay and a less accurate result. For this reason, many researchers are applying machine learning to FCM data analysis.

2.5 The Problem of Manual Gating

There are three main problems with manually gating FCM data. First, manual cell labelling or clustering is highly subjective (Bashashati and Brinkman, 2009). Secondly, sometimes the number of abnormal cell populations is so low that they are very difficult to detect. Thirdly, manual FCM data analysis is time-consuming (Bashashati and Brinkman, 2009). These problems are expanded below.

Table 2-2: Some human CD antigens (abcam, 2019)

Target name	Alternative name	Cellular expression
CD3d	T3D, OKT3	Mature T-cells and thymocytes
CD3e	T3E, TCRE	Mature T-cells and thymocytes
CD3g	T3G	Mature T-cells and thymocytes
CD14	-	Monocytes, macrophages (myelomonocytic cells), Langerhans cells and granulocytes
CD15	3-FAL, ELAM 1 ligand, fucosyltransferase, Leu M1	Granulocytes, monocytes, neutrophils and eosinophils
CD16a	Fc-gamma RIII-alpha, FCGR3A, FCG3, FCGR3, IGFR3	Neutrophils, natural killer cells, and macrophages
CD16b	Fc-gamma RIII-beta, FCGR3B, FCG3, FCGR3, IGFR3	Neutrophils and stimulated eosinophils
CD19	B-lymphocyte surface antigen B4, T-cell surface antigen Leu-12	B-cells (but not plasma cells) and follicular dendritic cells
CD23	Low affinity immunoglobulin epsilon Fc receptor, BLAST-2, Fc epsilon RII, FCER2, CLEC4J	Mature B-cells, activated macrophages, eosinophils, follicular dendritic cells and platelets
CD34	-	Hematopoietic stem cells and progenitors and capillary endothelial cells
CD41	Integrin alpha-IIb, GPalpha IIb, ITGA2B, GP2B	Platelets and megakaryocytes
CD42a	Platelet glycoprotein IX, GP9	Platelets and megakaryocytes
CD42b	Platelet glycoprotein Ib alpha chain, GP1BA	Platelets and megakaryocytes
CD42c	Platelet glycoprotein Ib beta chain, GPBB	Platelets and megakaryocytes
CD42d	Platelet glycoprotein V, GP5	Platelets and megakaryocytes
CD45	Leukocyte common antigen, L-CA, PTPRC	Hematopoietic cells (not erythrocytes and platelets)
CD48	BLAST1, BCM1	Leukocytes
CD56	NCAM1	Neural tissues, natural killer cells, T-cell subsets, small cell lung carcinoma
CD64	IgG Fc receptor I, FCGR1A	Monocytes and macrophages
CD66d	CEACAM3, CGM1	Neutrophils
CD90	THY1	Hematopoietic stem cells, neuronal cells, fibroblasts, stromal cells and activated endothelial cells
CD123	IL3RA	Basophils, eosinophils, hematopoietic progenitors, macrophages, dendritic cells, endothelial cells and a small subset of lymphocytes
CD193	CCR3	Eosinophils, trace amounts in neutrophils and monocytes
CD203c	ENPP3	Basophils, mast cells and their progenitors
CD210	IL10RA, IL10R1, CDw210a	Monocytes, B- and T-cells and large granular lymphocytes
CD233	Band 3, AE1	Erythrocytes
CD234	Duffy antigen receptor, DARC	Erythrocytes, endothelial cells and epithelial cells
CD235a	Glycophorin-A	Erythrocytes
CD235b	Glycophorin-B	Erythrocytes
CD329	Siglec-8, SAF-2	Eosinophils

2.5.1 High subjectivity

Gating is the process of classifying a group of interesting and similar types of cell by drawing a boundary round the group on the scatter plot. It is one of a hierarchy of methods that depend strongly on expertise or experience with flow cytometry. Given the same dataset, even experts may analyse it differently. This large dataset is complicated, so it often takes a long time to analyse. Moreover, as the number of parameters or markers to be considered increases, so the gating process will increase as well. Automated gating or computational assistance can therefore provide uniformity and reduce subjectivity.

2.5.2 Few cells have key cell expression

It is hard to detect the key cell expression manually because, sometimes, there are very few abnormal cells in the FCM data. Many machine learning techniques have been applied to help the clinician detect abnormal cells. The number of cells considered abnormal, or exhibiting key events such as leukaemia, are very few (often fewer than 0.1%), as a result of difficulty in detection (Groeneveld-Krentz *et al.*, 2016). Several recent techniques have aimed to automatically analyse the FCM data (Bashashati and Brinkman, 2009). Sometimes the number of abnormal cells is relatively small, so the main objective of algorithms is to automatically label each cell into biologically-meaningful populations (Rota *et al.*, 2016). Applying and improving machine learning approaches to detect those relatively few cell events may achieve more accurate analysis. A number of machine learning approaches have been used, but it is not clear which of them is an appropriate one to detect rare events in the FCM data and how to separate noise from key events, which is quite small within enormous volumes of data.

2.5.3 Gating is time-consuming

The current manual gating of FCM data involves selecting cell populations by drawing geometric shapes on two-dimensional dot plots. These cell populations are sequentially selected for further analysis, based on gate areas drawn. The number of possible 2-dimensional graphs is quite large, which depends on the number of parameters; for example, for 10 parameters there are 45 graphs, which leads to a time-consuming process for manually gating high dimensional data (Rahim *et al.*, 2018). The number of parameters, up to 50, and a huge number of cells, between a thousand and hundreds of thousands, in the FCM data, makes it especially difficult for clinicians to manually identify cells of interest and homogeneous cell populations (Rahim *et al.*, 2018). Many automated

gating techniques have been proposed, and some techniques focus on improving runtime efficiency.

2.6 Conclusion

Blood cell analysis is an important process to identify state of health of the donor. Flow cytometer a technology that provides multi-parametric analysis of single cells in blood sample. Flow cytometry data is quite large and complex and hard to manual analyse. Manual gating is a process of grouping homogenous cells together by drawing boundary on 1D or 2D scatter plot so scientists have to analyse those using their expertise. The analysis, moreover, is highly subjective, the number of key events may be really few, and the manual analysis is time-consuming. Although many automated gating techniques have been produced, a more accurate and faster technique is still required. Most of automated gating techniques were modified from machine learning technique, which were discussed in Chapter3 and Chapter4.

Chapter 3 Clustering Techniques

The previous chapter explained how the types of cell are classified by using fluorescence and CD markers. After obtaining the dataset from a flow cytometer, the next stage is to group similar or interesting events into clusters. This is called the gating process and the problems of the process are high subjectivity and time taken. The FCM dataset is multidimensional and the number of features depends on what category of infection is to be diagnosed, which leads to different experiment designs. These events can be displayed in a one-dimensional histogram, or a two-dimensional scatter plot, and the features exhibited are selected by the clinician. A limitation of manual gating is the naked eye identification of events displayed in these displays. People cannot view more than two dimensions at the moment so they have to switch back and forth between many features, resulting in confusion and leading to time delay. This has led to modification to, and combinations of, clustering methods applied to the gating process. The main objective of data clustering is to minimise intra-cluster distances while maximising inter-cluster distance, as shown Figure 3-1. The many approaches to clustering can be divided into: partitioning methods, hierarchical clustering, density-based clustering, and model-based clustering (Jain, Murty and Flynn, 1999).

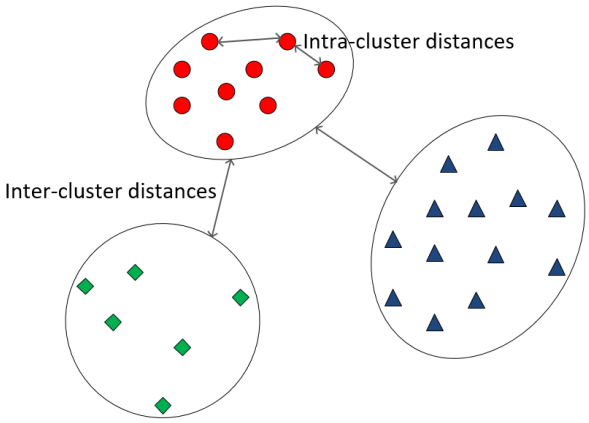


Figure 3-1 : Intra-cluster and inter-cluster distances

3.1 Partitioning methods

The partitioning clustering method discovers the clusters in the dataset by optimising a particular objective function, e.g. the sum of squared error (SSE), and repetitively improving the partition's quality (Rokach and Maimon, 2005). In other words, this method tries to construct partitions so that each object is in only one cluster, but the number of clusters (k) need to be set in advance (Jain, Murty and Flynn, 1999). Suppose that the number of partitions is K. The partitioning algorithms will

start by partitioning all n elements of dataset D into a set of K clusters ($K \leq n$) and will then evaluate the quality of the partitions by optimising an objective function, e.g. minimising the SSE, in order to improve them. For this method, each object must belong to one cluster and each cluster must have at least one element. Typical clustering methods include K-means, K-Medians, and K-Medoids (Rokach and Maimon, 2005).

3.1.1 K-Means

K-Means clustering is widely used and is based on distance between objects and centroids. The final result of K-means clustering is that each object is assigned to the cluster with the nearest centroid (Rokach and Maimon, 2005). The goal of K-Means is trying to minimise a criterion function or objective function as Equation 3.1.

$$J = \sum_{h=1}^{k} \sum_{x_i \epsilon C_h} \|x_i - c_h\|^2 \qquad (3.1)$$

where

k	number of clusters
C_h	set of objects in cluster *h*
c_h	centroid of cluster *h*
x_i	an object in C_h
$\|\| \ldots \|\|$	distance function, such as Euclidian distance

The algorithm of K-Means clustering operates in four steps.

Step 1: Randomly initialise the first K cluster centroids. $c_1, c_2, \ldots, c_k \in \mathbb{R}^d$

where

c centroid vector

d number of data dimensions

Step 2: Assign each object into the cluster whose centroid is nearest. The nearest centroid can be measured using Equation 3.2.

$$\min_{c_h} \sqrt{\sum_{d=1}^{D} (x_{id} - c_{hd})^2}, i = 1,2, \ldots n \qquad (3.2)$$

where

n number of objects in the dataset

x the object

D number of data dimensions

c_h centroid of cluster h, $h = 1, 2, \ldots, k$

Step 3: Re-compute new centroids for all clusters by using the mean value of all objects in individual clusters as new centroids:

$$c_h(new) = \frac{\sum_{i=1}^{n} x_i}{m_h}, h = 1,..,k \qquad (3.3)$$

where

m_h number of objects in cluster h

n number of all objects in the dataset

Step 4: Repeat Step 2, until the centroids do not change

A strength of K-Means is that it is faster than hierarchical clustering if the value of K is small, and is well-known and extensively use in clustering tasks. The relative efficiency is $O(ktn)$ where k is the number of clusters, t is the number of iterations, and n is the number of objects (Jain, Murty and Flynn, 1999). Another strength is that it can produce a better fit cluster than hierarchical clustering, particularly when the clusters appear spherically-shaped. Weaknesses of K-Means are that it is not suitable for noisy data, the number of clusters must be defined in advance, it is not suitable for a dataset with non-spherical shapes, the results are highly dependent on initialisation, and it is not applicable to category data.

3.1.2 K-Medians

K-Medians is less sensitive to noise than K-Means. Based on K-Means, it was enhanced by replacing means with medians. K-Medians can deal with noisy data better than K-Means (Whelan, Harrell and Wang, 2015). Mean and median are averages in statistics; median is the middle value in the ordering from least to greatest in the dataset. Apparently, when some noise is added to a large dataset, median value is more likely to be stable than mean value. For K-Medians clustering, medians are used as cluster centroids. Therefore, the K-medians objective function is

$$J = \sum_{h=1}^{k} \sum_{x_i \epsilon C_h} \|x_i - med_h\|^2 \qquad (3.4)$$

where

k number of clusters

C_h set of objects in cluster h

med_h median of data elements in cluster h

x_i object in C_h

The algorithm of K-Medians clustering operates in four steps:

Step 1: Randomly initialise the first K cluster centroids. $c_1, c_2, \dots, c_k \in \mathbb{R}^d$

where

c centroid vector

d number of data dimensions

Step 2: Assign each object to the cluster with whose centroid is nearest. The nearest centroid can be measured by using equation:

$$\min_{c_h} \sqrt{\sum_{d=1}^{D} (x_{id} - c_{hd})^2}, i = 1,2, \dots n \tag{3.5}$$

where

n number of objects in the dataset

x the object

D number of data dimensions

c_h centroid of cluster h, $h = 1, 2, \dots, k$

Step 3: Re-compute new centroids for the clusters by using the median of each individual feature

Step 4: Repeat Step 2, until the centroids do not change

K-Medians is modified from K-Means by using median instead of mean, in order to increase robustness against noise, especially when the number of noisy items is small, but their values are very much larger or smaller than normal data in the dataset. K-Medians provides faster run time for the identification of noise compared with K-Means clustering (Dalatu, 2016).

3.1.3 K-Medoids

K-Medoids is another method that is modified from K-Means in order to reduce the influence of noise, by replacing mean value with medoid. The medoid is an object that is the most centrally located element in a cluster (Bernábe-Loranca *et al.*, 2014). Noise is dealt with in this technique

similarly to K-Medians because it just uses the most central element instead of median value, and the objective function is the same as K-Means' objective function.

The algorithm for K-Medoids clustering has six steps:

Step 1: Randomly initialise the first K cluster centroids. $c_1, c_2, \dots, c_k \in \mathbb{R}^d$

where

c centroids vector, selected from objects in the dataset as data representative

d number of data dimensions

Step 2: Assign each object into the cluster whose centroid is nearest. The nearest centroid is measured by using Equation 3.6.

$$\min_{c_h} \sqrt{\sum_{d=1}^{D} (x_{id} - c_{hd})^2}, i = 1,2, \dots n \tag{3.6}$$

where

n number of objects in the dataset

x the object

D number of data dimensions

c_h centroid of cluster h, $h = 1, 2, \dots, k$

Step 3: Compute the total cost S of these medoids by using the objective function (SSE)

Step 4: Randomly select non-representative data as new medoids

Step 5: Compute total cost S of swapping the old medoids with new ones

Step 6: If the new S value is less than the old value, then replace old with new medoids

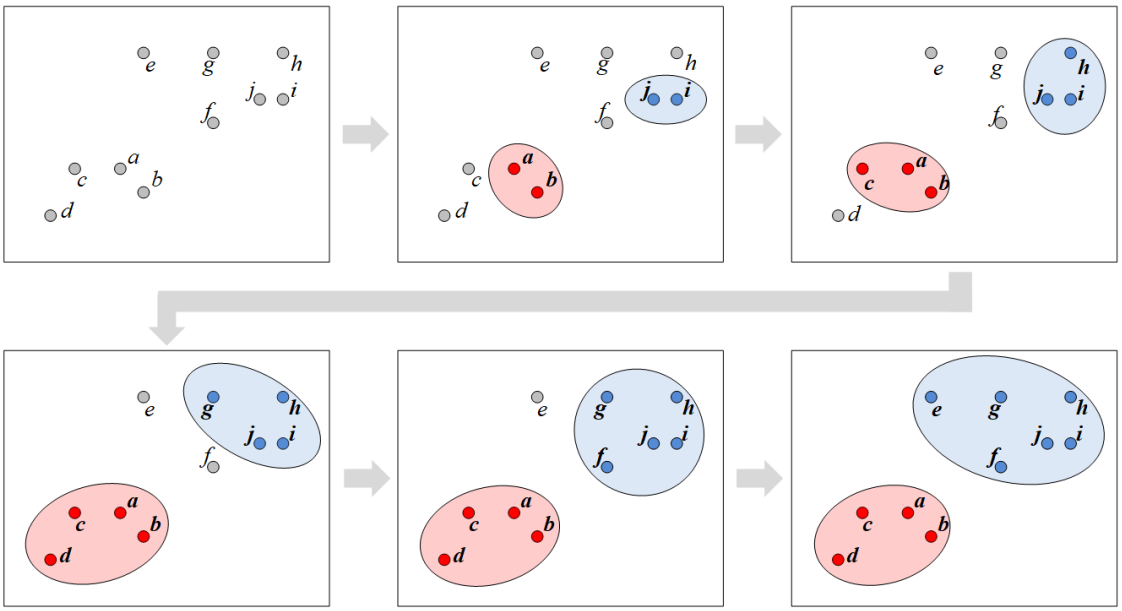
Although the K-Medoids clustering method can effectively handle noise and outliers, it is not suitable for a large dataset because of the computational complexity (Bernábe-Loranca *et al.*, 2014). Therefore, some clustering methods have been proposed that modify this method, such as PAM, CLARA, and CLARANS (Rokach and Maimon, 2005).

3.2 Hierarchical clustering

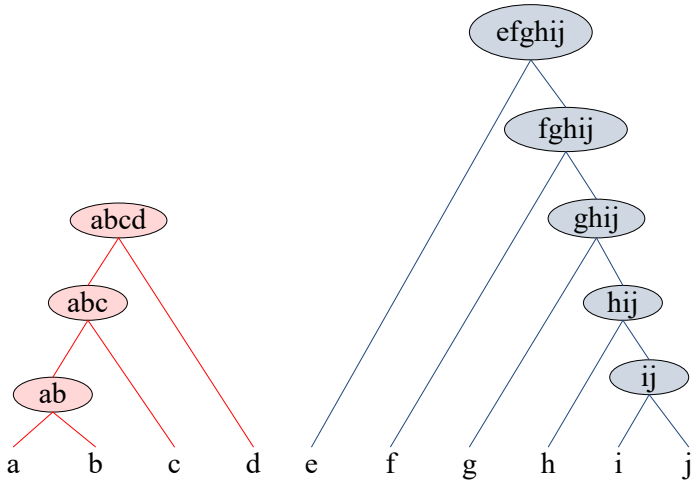
Hierarchical clustering can be divided into agglomerative and divisive (Rokach and Maimon, 2005). Agglomerative is a bottom-up method starting by defining each object as a cluster, then iteratively merging the two nearest clusters into one cluster until the clusters have a reasonable coverage. Hence, the key operation of the agglomerative method is the repeated combination of the two nearest clusters. The divisive method does this in the opposite way, top-down, by starting with one cluster and recursively splitting it into separate clusters.

3.2.1 Agglomerative clustering

The basic concept of this method is quite straightforward, as shown in Figure 3-2. Starting with computation of the distance or similarity matrix, and defining each object as a cluster, the two nearest clusters are repeatedly combined and the distance matrix updated, until it is found that the clusters are appropriate or only one cluster remains (Day and Edelsbrunner, 1984). Three questions arise with this approach. First, what is used as the representative or reference object for a cluster with more than one object? Secondly, what measurement is used to determine the nearest cluster? Thirdly, when should combining clusters stop? The first question can be answered by using the mean of all data points in each cluster. The median can also be used, or the mode (the number that is the most often repeated in the cluster). To answer the second question, after the centroids have been calculated, the distances between the centroids can be used. The most commonly used measure is Euclidian distance. To answer the third question, if the number of clusters (k) can be defined to begin with, merging can stop when there are k clusters.



a) Example of step-by-step agglomerative clustering method



b) Dendrogram of the agglomerative clustering method example

Figure 3-2 : Agglomerative clustering method

On the other hand, if k cannot be identified before clustering starts, some threshold for stopping is needed. This stopping threshold can be distance-based or density-based. For the threshold based on distances, radius (the maximum distance between the centroid and data points) or diameter (the maximum distance between two data points in the cluster), can be used. The merging repetition will stop when new merging would exceed the distance compared to the threshold. For the threshold based on density, the merging repetition will stop when new merging provides a lower density compared to the threshold.

The algorithm for agglomerative clustering has four steps:

Step 1: Identify each data point as a cluster

Step 2: Compute the distance matrix of clusters

Step 3: Merge the closest clusters together and compute the centroids

3.2.2 Divisive clustering

This is the opposite of the agglomerative method, so it is called a top-down method. It is less widely used than agglomerative because it is computationally intensive (Jain, Murty and Flynn, 1999). Divisive clustering starts with a single cluster encompassing the whole dataset and recursively splits it into smaller clusters until it results in n clusters, where n is the number of objects in the dataset or it provide appropriate clusters. This method uses distance as in the agglomerative method, and K-Means clustering is also commonly use. Normally, this method starts with running the K-Means algorithm, where k is 2, on the original dataset and for each of the resulting clusters, K-Means is recursively run on individual sub-populations.

The algorithm for divisive clustering can be summarised as:

Step 1: Identify all data points as a cluster

Step 2: Recursively split all clusters until each cluster contains only one point or the result is appropriate

The hierarchical clustering methods do not have to identify the number of clusters beforehand but any desired number of clusters can be defined by using distance threshold or a density threshold. However, this clustering approach still has problems with complex computation and leads to spending too much time on clustering (Jain, Murty and Flynn, 1999). Moreover, if k is not identified at the start, different thresholds can lead to different results.

3.3 Density-based clustering

Density is a number of data points within a specified radius, so the concept of density-based clustering is that sparser areas in the space are used for separating one cluster from another. Data points that lie in lower density areas than any of the clusters will be identified as noise or outliers. The outstanding strength of density-based clustering is its ability to identify arbitrary or non-spherical shapes. Another strength is its determination of noise, especially when the density of noise is low.

However, there are some weaknesses including time taken when applying this technique to large quantities of data, while its ability to detect noise is decreased when the dataset contains intensive noise (Breitkreutz and Casey, 2008). Many methods have been developed from this concept. The most common and widely used density-based algorithms, DBSCAN and OPTICS are discussed below.

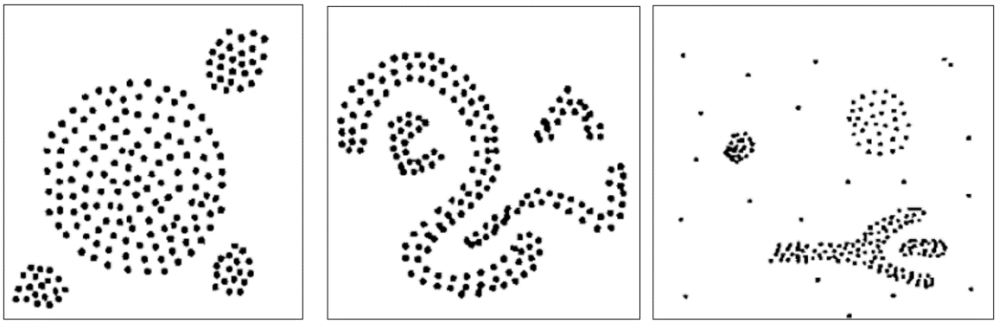


Figure 3-3 : Example of non-convex shapes of clusters (Ester *et al.*, 1996)

3.3.1 DBSCAN

Density-Based Spatial Clustering and Application with Noise (DBSCAN) was proposed by Ester et al., 1996. DBSCAN is an algorithm that actually generates clusters based on density, so it can identify non-convex shapes. Methods based on density have to define some parameters beforehand. For DBSCAN, two parameters are identified by the user: Eps (ε) and MinPts. A point that is considered as a member of a cluster needs to have at least one neighbour (other data point), where the distance between the pair is closer than ε. In other words, data point p is a neighbour of point q when the distance between p and q is less than or equal to ε. $MinPts$ is the minimum number of neighbours of a data point.

- ***Core point*** is a point that has more than or equal to $MinPts$ neighbours.
- ***Border point*** is a point that has fewer than $MinPts$ neighbours but is a neighbour of at least one core point.
- ***Noise point*** is a point that is not a neighbour of any core point.

The algorithm for DBSCAN clustering can be summarised as:

Step 1: Label all data points as core point, border point, and noise point.
Step 2: Treat a core point as the centre of a group.
Step 3: Merge each group together if they have at least an overlap neighbour.

To demonstrate the DBSCAN algorithm step-by-step, a sample of dataset has been created and clustered by this algorithm, as shown Appendix A.

Although DBSCAN is able to identify non-convex shapes, it does not require the number of clusters before attempting the clustering process, and has the ability to identify noise that leads it to be

more robust than partition-based clustering. It works properly only for datasets with uniform densities and needs the user to define some parameters before clustering starts.

3.3.2 OPTICS

Ordering Points To Identify the Clustering Structure (OPTICS) was proposed by (Ankerst *et al.*, 1999). The algorithm is derived from DBSCAN in order to deal with the need for the user to provide two parameters that give different clustering results for different density thresholds. However, OPTICS does not produce an explicit clustering; instead, it generates an ordering density clustering structure as explained below. The main idea behind this algorithm is that higher density clusters are completely contained within a lower density one, as shown in Figure 3-4. Local higher density, therefore, should be processed first.

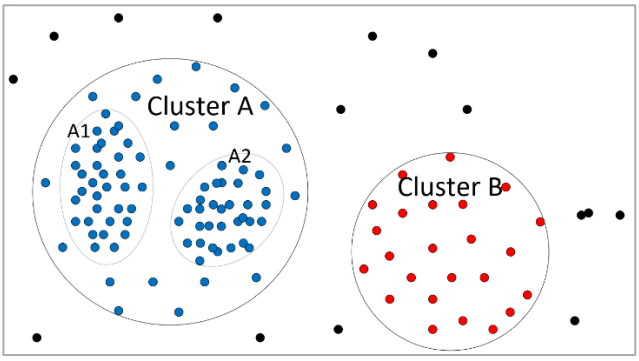


Figure 3-4 : Concept of OPTICS clustering(Ankerst *et al.*, 1999)

- Core-distance

  Let p be an object in the dataset, ε the distance between two objects, $N_\varepsilon(p)$ the set of neighbours of object p, and $MinPts$ the minimum number of neighbours. Then core-distance$_{\varepsilon,MinPts}(p)$ is equal to

  - Infinity or undefined, if the cardinality of $N_\varepsilon(p)$ is less than $MinPts$
  - the minimum distance from p to its neighbour that can cover at least $MinPts$ members, otherwise.

- Reachability-distance

  Let p and o be objects in the dataset, ε the distance between two objects, $N_\varepsilon(p)$ the set of neighbours of object p, and $MinPts$ the minimum number of neighbours. Then reachability-distance$_{\varepsilon,MinPts}(o,p)$ is equal to

  - Infinity or undefined, if the cardinality of $N_\varepsilon(p)$ is less than $MinPts$
  - Max (core-distance (p), distance (o, p)), otherwise.

In accordance with these definitions, therefore, reachability-distance must be equal to or greater than core-distance, as shown in Figure 3-5. From Figure 3-5, reachability-distance(p, q) is equal to reachability-distance(p, r) and equal to core-distance(p), while reachability-distance(p, o) is greater than core-distance(p).

The algorithm for OPTICS clustering can be summarised as follow:

Step 1: Read an unprocessed object (p) from dataset

Step 2: If p is a core-object, update core-distance of p

for each $q \in N\varepsilon(p)$

update reachability of object q

Update OrderSeeds list, which contains the objects ordered by reachability-distance from smallest

Mark p as processed

Step 3: Read an unprocessed object p from OrderSeeds list if the list is not empty, otherwise, read next unprocessed object from the dataset

Step 4: Repeat Steps 2-3 until the end of the dataset

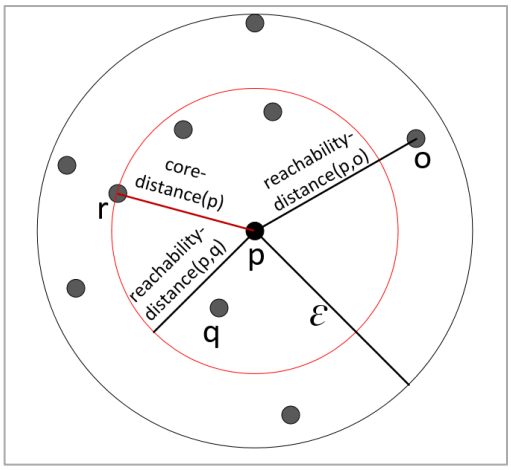


Figure 3-5 : The difference between core-distance and reachability-distance, given ε and *MinPts* = 5

The OPTICS clustering technique process is demonstrated with 26-objects of synthesised data in Appendix A.

Although the most of density-based methods, such as DBSCAN and OPTICS, have an ability to detect non-convex cluster shapes, identify noises and automatically identify the number of clusters, the

density threshold and other parameters have to be carefully defined because different parameters yield surprisingly different results for clustering.

3.4 Model-based clustering

The assumption behind model-based clustering is that the data derives from a finite mixture model of probability distributions (Rokach and Maimon, 2005). In other words, the main idea of model-based clustering is to discover a mathematical model that fits the data. Generally, there are two methods: hard clustering and soft clustering. Hard clustering assumes that there is no overlap between clusters, so every object is identified as a member of only one cluster. Soft clustering is more flexible, since the objects can belong to more than one cluster with different probabilities. Most model-based clustering uses the soft clustering method.

The model-based clustering algorithm itself comes as two main types. The statistical learning model, such as Gaussian Mixture Models (GMMs) and t mixture models, and the neural network learning model, such as SOM (Bullinaria, 2004) and ART (Carpenter and Grossberg, 2002). However, this work focuses on clustering as a supervised learning method, so only the statistical learning model will be discussed here.

3.4.1 Gaussian Mixture Models

Gaussian distribution, also called normal distribution, is exhibited by many datasets in real life. Therefore, the assumption of GMMs is that all objects in a dataset are modelled by several Gaussian distributions that have different means and variances (Yang, Lai and Lin, 2012). For one dimension, the probability density function (pdf) is given by Equation 3.7 and for multivariate (more than one dimension), the pdf is given by Equation 3.8.

$$G(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.7}$$

where

μ mean of all values in a dataset

σ^2 variance of distribution in the dataset

x the data object

$$G(x|\mu,\Sigma) = \frac{1}{\sqrt{2\pi|\Sigma|}} e^{-\frac{(x-\mu)^T(x-\mu)}{2\Sigma}} \tag{3.8}$$

where

μ mean of all values in the dataset

Σ covariance matrix

$(x-\mu)^T$ transpose matrix of $x-\mu$

x the data object

Figure 3-6 (a) shows an example of Gaussian distribution with mean = 5.0 and different variances of 0.5, 1.0, 2.0, and 4.0. Figure 3-6 (b) shows the distribution with variance = 1.0, and different means 3.0, 3.5, 5.0, and 8.0.

A clustering method based on GMMs assumes that the number of models or clusters are specified at the start, following which an algorithm is applied to identify the appropriate value of parameters, such as the Expectation-maximisation algorithm (Yang, Lai and Lin, 2012), which is widely used with GMM-based clustering. Figure 3-7 shows the result of using GMM-based clustering with a one-dimensional dataset, using two Gaussian models with parameters μ = 1.5 and σ^2 = 2.5 for the first model, and μ = 4.5 and σ^2 = 3.5 for the second. Some objects overlap the two models such as objects a, b, and c. However, this method is soft clustering, so each object is identified as belonging to each model by its probability. Thus, object a belongs to the first and second model with probabilities 0.7 and 0.3 respectively, while object b belongs to the first and second model with probabilities 0.5 and 0.5 respectively, etc.

The probability function is $P(c_j|x_i)$, which means that the probability of belonging to cluster c_j of object x_i, is defined as

$$P(c_j|x_i) = \frac{P(x_i|c_j)P(c_j)}{\sum_{h=1}^{k} P(x_i|c_h)P(c_h)} \tag{3.9}$$

where

$$P(x_i|c_j) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu_j)^2}{2\sigma_j^2}} \tag{3.10}$$

and

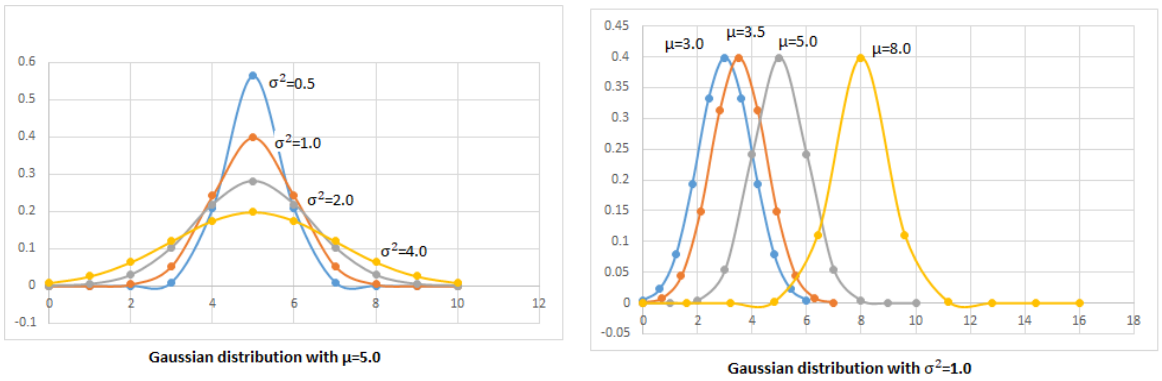
$P(c_j)$ probability of cluster j

k number of clusters or models

μ mean of all values in a dataset

σ^2 variance of distribution in the dataset

x the data object



(a) Mean = 5.0 (b) Variance = 1.0

Figure 3-6 : Gaussian distribution with different parameters

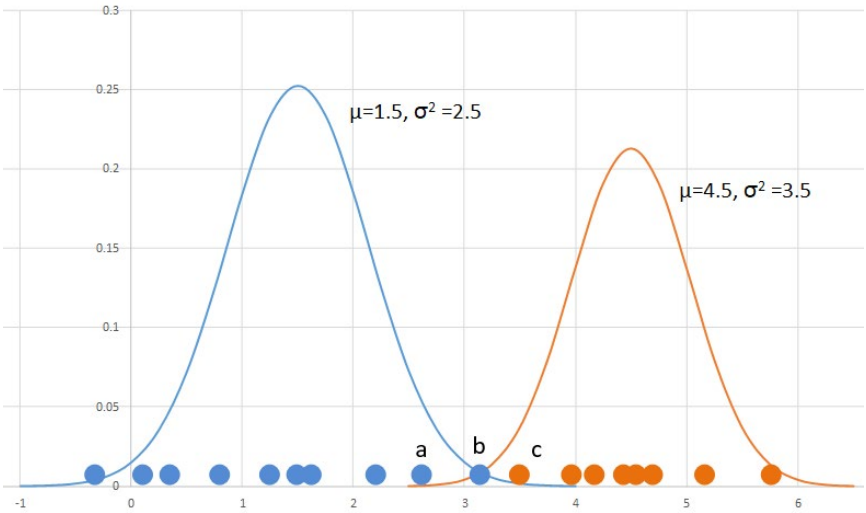


Figure 3-7 : Results of clustering using GMMs with two sets of parameters

GMMs have the ability to identify non-spherical shapes, and each object is assigned a probability in a certain cluster, so this algorithm is more flexible than K-Means. However, the optimisation of the loss functions for the model is not trivial, so the Expectation-maximisation algorithm is applied for this method. Furthermore, the Gaussian distribution model is very sensitive to noise and outliers. Hence, other distribution models have been applied in order to reduce the influence of outliers.

3.4.2 t Mixture models

The GMMs clustering method is widely used but is not suitable for noisy datasets, because the Gaussian distribution model is very sensitive to outliers (Roettger *et al.*, 2009). In order to deal with noisy data, t mixture models are used instead. t distribution is symmetric but it has heavier tails that gives it more robustness to outliers than GMMs, as shown in Figure 3-9. The concept of t mixture models is similar to that of GMMs but the probability density function of t distribution is used instead, defined as follows.

$$f(x) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)\sigma}{\sqrt{\pi\nu}\Gamma\left(\frac{\nu}{2}\right)}\left(1 + \frac{(x-\mu)^2}{\sigma^2\nu}\right)^{-\frac{\nu+1}{2}} \tag{3.11}$$

where

- x the data object
- Γ Gamma function
- ν degrees of freedom
- σ variance of distribution in the dataset
- μ mean of the dataset

Equation 3.11 is the function for one-dimensional dataset. For a *p*-dimensional dataset, the distribution is given by

$$f(x) = \frac{\Gamma\left(\frac{\nu+p}{2}\right)|\Sigma|^{-\frac{1}{2}}}{(\pi\nu)^{\frac{p}{2}}\Gamma\left(\frac{\nu}{2}\right)\left\{1+\frac{(y-\mu)^T\Sigma^{-1}(y-\mu)}{\nu}\right\}^{\frac{\nu+p}{2}}} \tag{3.12}$$

where

- x the data object
- Γ Gamma function
- ν degrees of freedom
- Σ variance matrix
- μ mean of the dataset
- p number of dimensions

Figure 3-8, with the same variance, shows that as the degrees of freedom increase, so the distribution covers more objects both by width and by height.

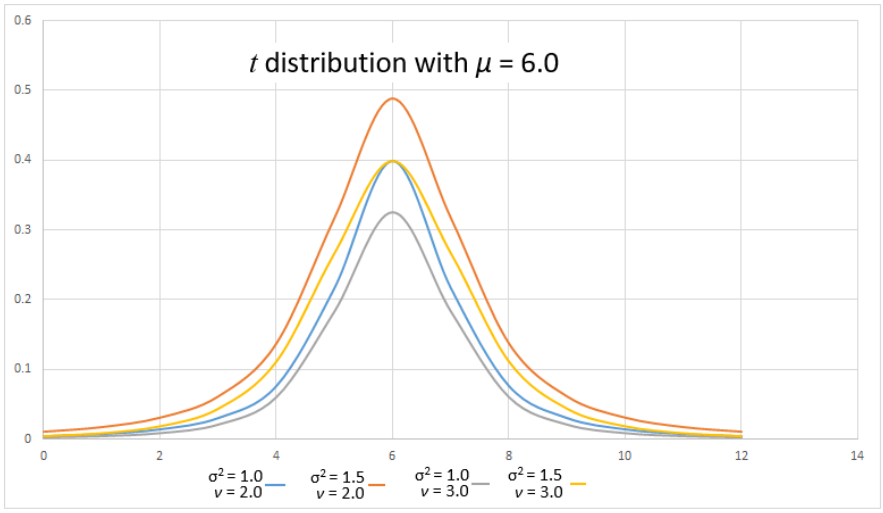


Figure 3-8 : t distribution with different variances and degrees of freedom

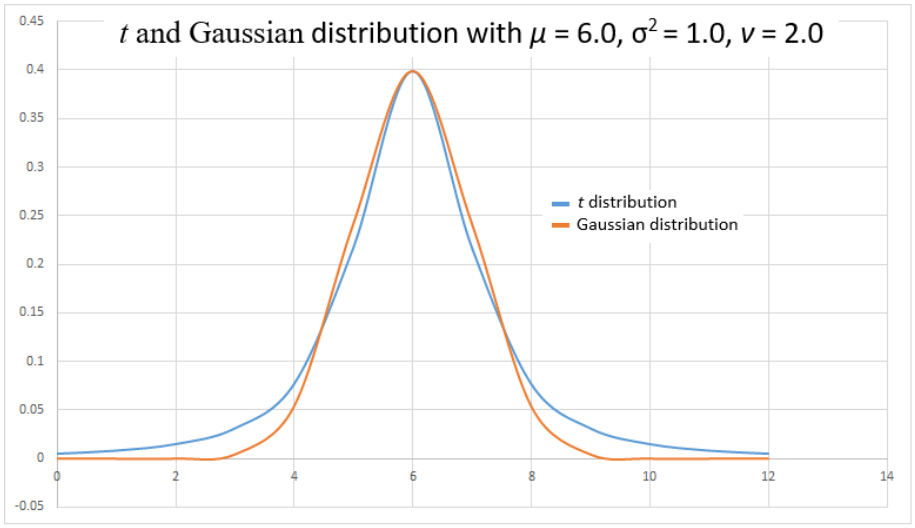


Figure 3-9 : t distribution and Gaussian distribution with the same parameters

Figure 3-9 shows that *t* distribution has heavier tails than Gaussian distribution, i.e. its tails can contain more outliers without any change of the area covering most objects. Therefore it has more robustness to outliers than the Gaussian distribution.

Table 3-1: Comparison of clustering techniques (Rokach and Maimon, 2005; Roettger *et al.*, 2009)

Clustering techniques	User-defined parameters	Non-spherical shape identification	Dealing with noise	Various density identification
Partition method	- k	No	No	Yes
Hierarchical clustering	- k	Yes	No	Yes
Density-based clustering	- ε - $MinPts$	Yes	Auto detection	No
Model-based clustering	- k - model parameters	Yes	Soft clustering (A probability of object to be in each clusters is assigned	Yes

3.5 Discussion

According to the previous sections, the traditional clustering techniques can be divided into: partitioning methods, hierarchical clustering, density-based clustering, and model-based clustering. All techniques require the user to define the number of cluster (k) before start the clustering process except density-based clustering, as can be seen in Table 3-1. Although the density-based clustering has an ability to automatically identify the number of cluster, it needs two user-defined parameters, which are (epsilon) ε and (Minimum points) $MinPts$. ε is distance threshold which greatly effect to the result so the ε becomes challenging to estimate. The partition method is distance-based clustering so it cannot identify the clusters with non-spherical shape, while, others techniques can. Noises are often found in FCM data. From Table 3-1, the density-based clustering has an ability to detect noise so this technique is often modified and applied to analyse FCM data. However, the main drawback of the density-based clustering is that it doesn't perform as well as others when the clusters are of varying density.

3.6 Conclusion

Many clustering methods have been proposed, with different strengths and weaknesses. First, partition-based clustering can identify different density clusters, but is highly sensitive to noise and the number of clusters needs to be known at the start. Secondly, hierarchical-based clustering can identify the number of clusters automatically, but still has a problem with time efficiency. Thirdly, density-based clustering can identify non-concave shape of clusters, is robust to noise, and can automatically identify the number of clusters, but gives a poor clustering result if populations in the dataset have different densities and user must define some parameters before the clustering

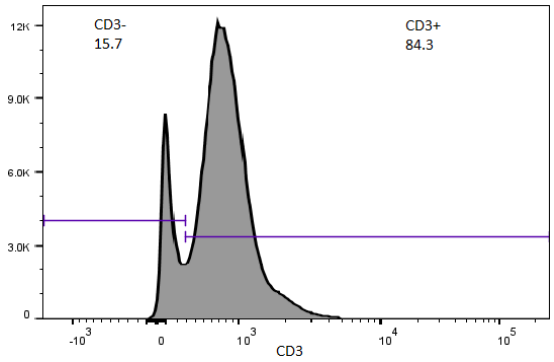
process. Fourthly, model-based clustering provides good results if the populations comply with the models, otherwise, the clustering results can vary. However, most of them are also modified, combined, and can be applied to flow cytometry data for the gating process to better fit the data.

In this Chapter, the well-known traditional clustering techniques have been explained. These clustering techniques had been modified and applied to FCM data that is called automated gating techniques. Therefore, the method for analysis FCM data or automated gating techniques are discussed in the next chapter.

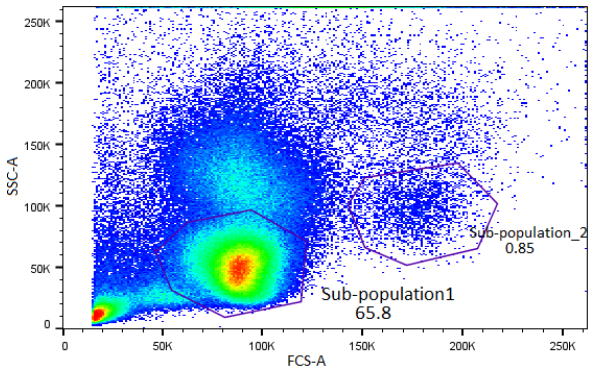
Chapter 4 Method for Analysis of Flow Cytometry Data

4.1 "Gating" – a classification process

Flow cytometry is based on fluorescence. The data obtained from a flow cytometer provides characteristics of hundreds to millions of cells in a blood sample. Generally, a multidimensional dataset received from this method is so large and complex that it is difficult to manually analyse, even with clinical expertise, especially for the clustering process. Gating is the process of selecting homogeneous or interesting cells by drawing geometrical shapes around populations of cells, represented in 1D by a histogram and in 2D by a scatter plot, in order to group them together, as shown in Figure 4-1. Obviously, the expert needs to have experience in these matters and also be familiar with this kind of data in order to select the interesting cells, and to discover sub-populations of cells as far as possible, before entering the next analysis process. Manual gating is highly subjective and time-consuming, which is why automated gating methods are being developed. Machine learning techniques, both of supervised and unsupervised, have been applied to automated gating to develop a general and standard gating model. The expectation from these developments is a performance increase through increased accuracy, less time-consumed, better outlier detection, and novel cell type detection. Although a many automated gating methods have been developed, there have some limitations, so this research area is still challenging.



a) Gating of 1D histogram, y-axis is number of cells, x-axis is intensity of CD3 marker



b) Gating of 2D scatter plot, SSC-A is granularity or complexity of each cell, FCS-A is the size of individual cells

Figure 4-1 : Manual gating examples

4.2 Challenges for automation

Manual gating of FCM data is highly subjective and excessively time-consuming so the research community has made an effort to increase competence by developing several automated gating

methods based on machine learning techniques. However, many factors make the analysis of this data still very challenging.

1) The FCM dataset is a description of millions of cells with multi-dimensions, which depends on the experimental design such as selection of markers to be used or the performance of the flow cytometer. Some of the research applies several techniques to dimensionality reduction to cut evaluation time and complexity of the data. Some techniques need to transform the data for scaling or constructing data more symmetrically. However, information can be lost through dimensionality reduction and the transformation process.
2) Sometimes the number of keys or interesting cells are very rare. Flow cytometry data is able to report cell populations at frequencies as low as 0.0001% (Hedley and Keeney, 2013). Therefore, a clustering approach that can detect very small populations is still needed.
3) The recognition of an appropriate number of clusters before the clustering process starts is still very challenging. Automatic calculation of the number of clusters is included in some techniques, but several do not perform well, while some methods leave it to the user to generate direct and indirect input by passing other parameters, which are difficult to tune (Weber and Robinson, 2016).
4) FCM data is very noisy. Current improvements to the flow cytometer can provide up to 20 features for millions of individual cells (Pyne *et al.*, 2009). The more equipment improves, the more information it will provide. Basically, there are always a lot of outliers in an enormous dataset; therefore these outliers need to be handled in order to provide more accurate gating.

4.3 Applying clustering techniques to FCM data

Many clustering techniques have been applied to FCM data automated gating but almost all the approaches are combinations or modifications of the four main methods of machine learning: partition or K-Means clustering, statistical model clustering, density-based clustering, and others such as grid-based and histogram-based. Every single method has advantages and disadvantages as explained here.

4.3.1 Partition-based or K-Means-based

K-Means clustering is an unsupervised learning algorithm, widely used in many tasks. However, there are some problems applying it to FCM. This technique aims to assign data with similar characteristics to the same group. Basically, the number of clusters, *k,* has to be defined beforehand. To estimate this number, some methods for finding *k* have been produced, such as

Bayesian Information Criterion (BIC) (Schwarz, 1978), Akaike Information Criterion (AIC) (Akaike, 1974), Integrated Complete Likelihood (ICL) (Biernacki, Celeux and Govaert, 2010), and change point detection (Aghaeepour *et al.*, 2010). K-Means clustering is highly sensitive to noise because every point in the dataset is counted, to iteratively calculate the new centroids, which is why so many automated gating methods attempt to reduce the influence of noise. Although K-Means can be used to cluster general datasets, K-Means clustering cannot identify non-concave clusters, which can appear in many real life datasets, because this method uses the distance between centroid and data point to measure the similarity.

4.3.2 Statistical model-based

Model based clustering can deal with non-concave shape distributions and the influence of noise. For this technique, data points that follow the same statistical distribution model are considered as belonging to the same group (Rokach and Maimon, 2005). Obviously, this method can reduce the influence of noise because data points not belonging to any model are ignored. Moreover, the technique can effectively identify rare populations. This technique can also identify non-concave distributions efficiently because the shape of the cluster depends on the model, which can be any shape. However, it is really difficult to define a general model or assumptions regarding cluster distributions that are appropriate for every FCM dataset obtained from a blood sample.

4.3.3 Density-based

In density-based clustering, the number of clusters, *k*, does not need to be defined because the number of clusters can be detected from the distribution and density of the dataset. Moreover, density-based clustering is efficient in identifying non-concave shapes because a data point that has enough similarity to another data point, which is already assigned to a cluster, is assigned to the same cluster (Rokach and Maimon, 2005). In other words, this technique measures the similarity of two data points rather than the similarity between a data point and a centroid. However, some parameters and similarity thresholds have to be set before the iterative clustering process. Therefore, different parameters or different thresholds can lead to different results.

4.3.4 Others

Other techniques such as grid-based and histogram-based have been used for automated gating. The histogram-based approach is based on the one-dimension density plot. It is not widely used in general datasets because it only considers one dimension at the moment. Some automated gating

techniques have applied histogram-based method to FCM data (Sugár and Sealfon, 2010). The number of cells is represented by the y-axis and the intensity of markers is shown on the x-axis. Local peaks of the histogram are considered as the centre of each cluster. This technique cannot be applied directly to high-dimension data. Although the approach has good time performance, it has a problem with information loss because the data dimensions have to be reduced beforehand. Grid-based clusters data by using a grid with multi-resolution in order to explore the data structure in the dataset. STING (Wang, Yang and Muntz, 1997) and CLIQUE (Yadav and Kumar, 2014) use this concept for data clustering.

Many automated gating techniques are trying to use the strength of individual clustering methods by modification or in combination.

4.4 Automated gating techniques

Many approaches have been developed by applying machine learning to automated gating of FCM data. The advantages and disadvantages for each approach are discussed below.

4.4.1 flowClust

flowClust is a model-based clustering method presented by Lo et al. in 2008. This is a statistical model-based approach using t mixture models with Box-Cox transformation (Lo, Brinkman and Gottardo, 2008). They used a t mixture model because t distribution has heavier tails and lower peaks than a Gaussian distribution. They claimed that their methods provide more robustness to noise and also provide a better gating result compared with K-Means and Gaussian mixture model, or even a Gaussian mixture model with Box-Cox transformation, because the latter ones are very sensitive to outliers (Lo, Brinkman and Gottardo, 2008). Box-Cox transformation is used to make the dataset more symmetric and this transformation method is powerful for data with high distribution. Another technique used with flowClust is the Expectation-Maximization algorithm (Yang, Lai and Lin, 2012), which handles parameter estimation and also transformation selection. Although flowClust provides a more standard model for the gating process, this method assumes normal distributions that do not fit well in FCM data, and there are a lot of overlapping clusters in the gating results. Moreover, time taken and estimation of the number of clusters are still a problem (Finak *et al.*, 2009).

The standard model-based clustering is normally based on finite Gaussian mixture models and each cluster can be represented by a separate Gaussian model function. Equation 4.1 is the likelihood for a mixture model with *k* clusters.

$$L(\mu_1, \dots, \mu_K; \Sigma_1, \dots \Sigma_K; w_1, \dots, w_K | \boldsymbol{x}) = \prod_{i=1}^{n} \sum_{k=1}^{K} w_k \, G_p(x_i|\mu_k, \Sigma_k), \sum_{k=1}^{K} w_k = 1 \quad (4.1)$$

where

$\boldsymbol{x}$ is a given data point with independent p-dimensional multivariate data sample

$G_p(x_i|\mu_k, \Sigma_k)$ multivariate Gaussian distribution with the p-dimensional data

μ_k mean of cluster k

Σ_k covariance matrix of cluster k

w_k probability of each cluster k

n number of data points

The Gaussian distribution model is very sensitive to outliers, so the *t*-mixture model is used instead. The *t*-mixture likelihood function (L) is the same as the Gaussian model (Equation 4.1) but the Gaussian density function (Equations 3.1 and 3.2) is replaced by the t density function with mean μ, covariance matrix Σ, and ν degrees of freedom, as shown in Equation 4.2.

(4.2)

where

$\varphi_p(y|\mu, \Sigma, \nu)$ multivariate t distribution with the p-dimensional

ν multivariate Gaussian distribution with the p-dimensional

Γ Gamma function

For the E-step in the Expectation-Maximization algorithm, after random identification of means (μ) and variances (Σ) for each cluster. The membership function is given by Equation 4.3.

$$z_{ik} = \frac{w_k \varphi_p(x_i|\mu_k, \Sigma_k, \nu)}{\sum_{k=1}^{K} w_k \varphi_p(x_i|\mu_k, \Sigma_k, \nu)} \quad (4.3)$$

For the M-step in the **E**xpectation-**Maximization** algorithm, some of the parameters are recalculated, including the proportion (w'_k), mean (μ'_k), and variance matrix (Σ'_k) for each cluster, and are given by Equations 4.4, 4.5, and 4.6.

$$w'_k = \frac{n_k}{n} \quad (4.4)$$

$$\mu'_k = \frac{\sum_{i=1}^{n} z_{ik} x_i}{n_k} \tag{4.5}$$

$$\Sigma'_k = \frac{\sum_{i=1}^{n} z_{ik} (x_i - \mu'_k)(x_i - \mu'_k)^T}{n_k} \tag{4.6}$$

For the Box-Cox transformation, in order to provide the data with more symmetry, the transformation for object x is given by Equation 4.7.

$$x^{(\lambda)} = \begin{cases} \frac{x^{\lambda}-1}{\lambda}, \lambda \neq 0 \\ \log(x), \lambda = 0 \end{cases} \tag{4.7}$$

where

λ is the Box-Cox parameter

Obviously, this method is *t*-mixture model based, so it should perform well with only the datasets whose distribution of objects follow a *t* distribution. Moreover, the datasets have to be transformed before entering the clustering process, so could cause information loss and lead to poor clustering results.

4.4.2 FLAME

Flow analysis with automated multivariate (FLAME) was introduced by Pyne et al. in 2009. FLAME is based on a finite mixture model (mixture of skewed and heavy-tailed multivariate probability distributions) using a scale-free weighted ratio (SWR) in order to identify the number of clusters k (Pyne *et al.*, 2009). They claim that this approach has can detect rare population elements, reduce the influence of noise, estimate an appropriate number of clusters, and identify nonconvex cell populations (Pyne *et al.*, 2009). Generally, FCM datasets tend to be noisy, so this approach seems to be suitable for FCM data, and sometimes the interesting cell populations are very rare. Moreover, they also suggested that the FLAME technique can be directly applied to multi-dimensional data without dimensionality reduction or transformation, which can cause information loss. Although transformation can decrease skewness, it may cause inaccurate clustering, while some other methods need to project the original dataset into a lower dimension before starting the clustering or gating process (Pyne *et al.*, 2009).

FLAME is based on a finite mixture model and skew *t* distributions are used by default. It is claimed that skew *t* distribution provides a better fit to the actual distribution of FCM data than flowClust, as shown in Figure 4-2. However, since FLAME is a finite mixture model based clustering, some

populations may not follow the model or assumptions (Qian *et al.*, 2010). Further, this technique takes excessive computational time. Ge and Sealfon (2012) showed that FLAME takes around 35 times longer than flowPeaks, 3 times longer than Misty mountain, and 1000 times longer than flowMeans, when applying these techniques with a concave dataset. Therefore FLAME apparently still has a problem with time efficiency (Ge and Sealfon, 2012).

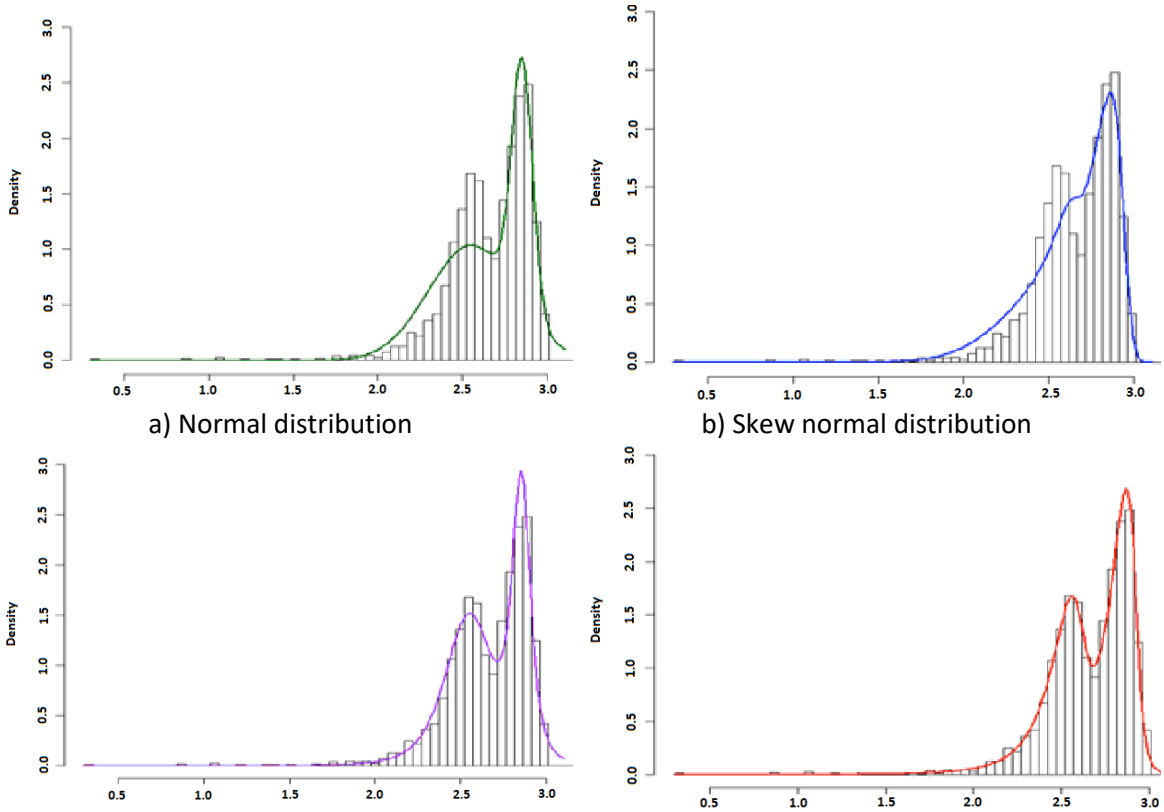


Figure 4-2 : Mixture modelling with different distributions (Pyne *et al.*, 2009)

Equation 4.8 is the probability density function that is modelled for a multivariate data sample X_i with *p* dimensions and unknown parameters Ψ.

$$f(X, \Psi) = \sum_{k=1}^{K} w_k \, f_k(x; \mu_k, \Sigma_k, D_k, \nu_k) \tag{4.8}$$

$f_k(x; \mu_k, \Sigma_k, D_k, \nu_k)$ multivariate skew *t* probability density function for k^{th} cluster

μ mean or location vector

Σ covariance or scale matrix

D skew parameter

ν degrees of freedom

x data sample

Ψ unknown parameters estimated by the EM algorithm

w_k proportion of cluster k

Equation 4.9 is the multivariate skew *t* probability density function, which is used by default for this approach.

$$f(x;\mu,\Sigma,D,\nu) = 2\varphi_{p,\nu}(x;\mu,\Omega)T_{\nu+p}\left(\frac{\tau}{\sigma}\right), x\epsilon R^p \quad (4.9)$$

$\varphi_{p,\nu}(x,\mu,\Omega)$ probability density function of *p*-dimensional t distribution with degrees of freedom ν, scale covariance Ω, and mean μ

Ω = $\Sigma + DD^T$

τ = $D^T\Omega^{-1}(x-\mu)$

σ^2 = $\frac{\nu+\eta}{\nu+p}(1 - D^T\Omega^{-1}D)$

η = $(x-\mu)^T\Omega^{-1}(x-\mu)$

x data sample

For unknown parameter $\Psi = (\mu,\Sigma,D,\nu)$, an estimation of proportions of each cluster $w_1, \dots w_K$, using the Expectation-Maximization algorithm, has been applied. Another main function of this approach is identifying the number of clusters k. To identify of the optimal number of clusters for each sample, every data point is assigned to the cluster with maximum posterior probability p_i, then the optimal number of clusters is set to the value of k that minimises the scale-free weighed ratio (SWR), as can be seen in Equation 4.10.

$$SWR = \frac{\sqrt{\sum_{i,j\epsilon C} p_i p_j d_M^2(i,j) \,/\, \sum_{i,j\epsilon C} p_i p_j}}{\sqrt{\sum_{i\epsilon C, j\epsilon C', C\neq C'} p_i p_j d_M^2(i,j) \,/\, \sum_{i\epsilon C, j\epsilon C', C\neq C'} p_i p_j}} \quad (4.10)$$

Where $d_M(i,j)$ is the scale-free Mahalanobis distance between data point i and data point j, calculated from Equation 4.11.

$$d_M(i,j) = \begin{cases} \sqrt{(i-j)^T \Sigma_C^{-1} (i-j)}\,, i,j \in C \\ \sqrt{(i-j)^T (\Sigma_C + \Sigma_{C'})^{-1} (i-j)}\,, i \in C, j \in C', C \neq C' \end{cases} \quad (4.11)$$

p_i posterior probability of point i belonging to cluster C

p_j posterior probability of point j belonging to cluster C', $C \neq C'$

Σ_c variance matrix of cluster C

This method can be directly applied to high-dimensional data (*the number of dimensions was not given in the original paper*) and is more flexible than flowClust because it uses both symmetric and asymmetric distributions, by changing some parameters for the skew t distributions model. However, since FLAME is finite mixture model based, it can provide good results if the objects in the datasets arise from the models, otherwise, it can provide an opposite clustering result. Also, FLAME still has a problem with time efficiency.

4.4.3 flowMerge

Finak et al. (2009) presented the flowMerge approach, which is a finite mixture model framework that was extended from flowClust by applying a merging algorithm. It was claimed that this framework can provide a better estimate of the number of clusters than GMMs or flowClust, especially as the FCM data distribution is so complicated (Finak *et al.*, 2009). They improved the performance of flowClust by developing a merging algorithm, which extends and modifies the work of Baudry et al. (2008) in order to make it suitable for use with FCM data. The basic concept of this merging algorithm is that two clusters with significant overlapping will be merged into one cluster in order to reduce the entropy of the data after new cluster assignments (Baudry *et al.*, 2008). The entropy function for mixture model of a K cluster is define as

$$\mathrm{ENT}(K) = -2 \sum_{k=1}^{K} \sum_{i=1}^{N} p_{ik} log_2(p_{ik}) \quad (4.12)$$

where

p_{ik} probability of data point i belonging to cluster k

For the two overlapping clusters, the entropy is high when p_i is non-zero, which means data point i is in the overlapping area and belongs to either cluster. On the other hand, the entropy is low or near zero if the two clusters do not overlap or only very little. As a consequence, merging overlapping clusters iteratively leads to a reduction in the entropy of clustering. For each iteration, two clusters are merged. The estimate of the merged cluster mean and covariance matrix, based

on a t distribution, are given by Equations 4.13 and 4.14. The optimal number of clusters k can be considered from the plot of entropy against the number of clusters, shown in Figure 4-3.

$$\mu_* = \frac{(p_i\mu_i + p_j\mu_j)}{p_*} \tag{4.13}$$

$$\_ \frac{((\nu_* - 2)p_*\mu_*^2)}{p_*\nu_*} \tag{4.14}$$

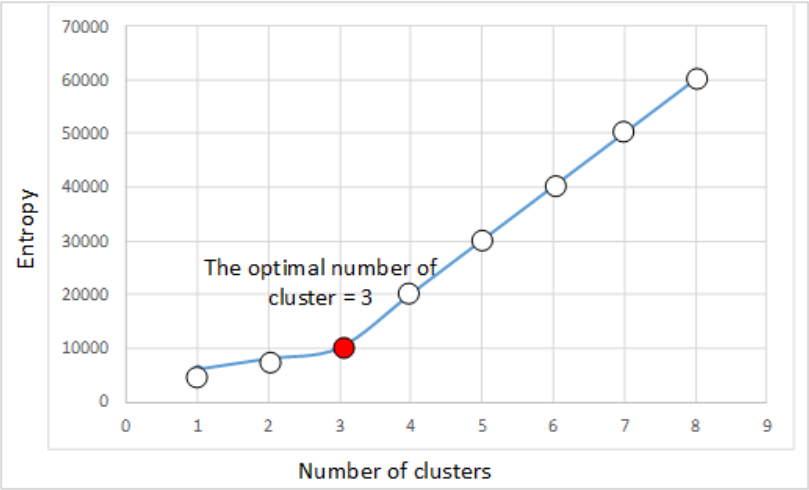


Figure 4-3 Plot of the entropy against number of clusters

The flowMerge framework allows multiple mixture distribution components to model the identical subpopulations. The strengths of this framework are that it is suitable for complicated distributions and provides a better estimate of the number of clusters. However, flowMerge also has the same problem with time efficiency as the FLAME approach. flowMerge is around 1.6 and 2.2 times slower than FLAME when applied to a barcode and a concave dataset, respectively (Ge and Sealfon, 2012).

4.4.4 FLOCK

Flow clustering without K (FLOCK) is another framework that attempts to increase the performance of FCM automated gating (Qian *et al.*, 2010). FLOCK is a combination of density-based, grid-based, and partition-based clustering algorithms. The original paper exhibited two experiments: B-cell subset analysis, and tetanus study. Before starting the clustering process, the experimental datasets had to be normalised because they had different value ranges in each dimension, and could not directly use the same similarity measure for clustering. Z-score normalisation was used to normalise the B-cell subset analysis, and min-max normalisation was used for the tetanus study. The function

formula for Z-score normalisation is given by Equation 4.15, while min-max normalisation is given by Equation 4.16.

$$x' = \frac{x - \mu}{\sigma} \tag{4.15}$$

$$x' = \frac{x - min}{max - min} \tag{4.16}$$

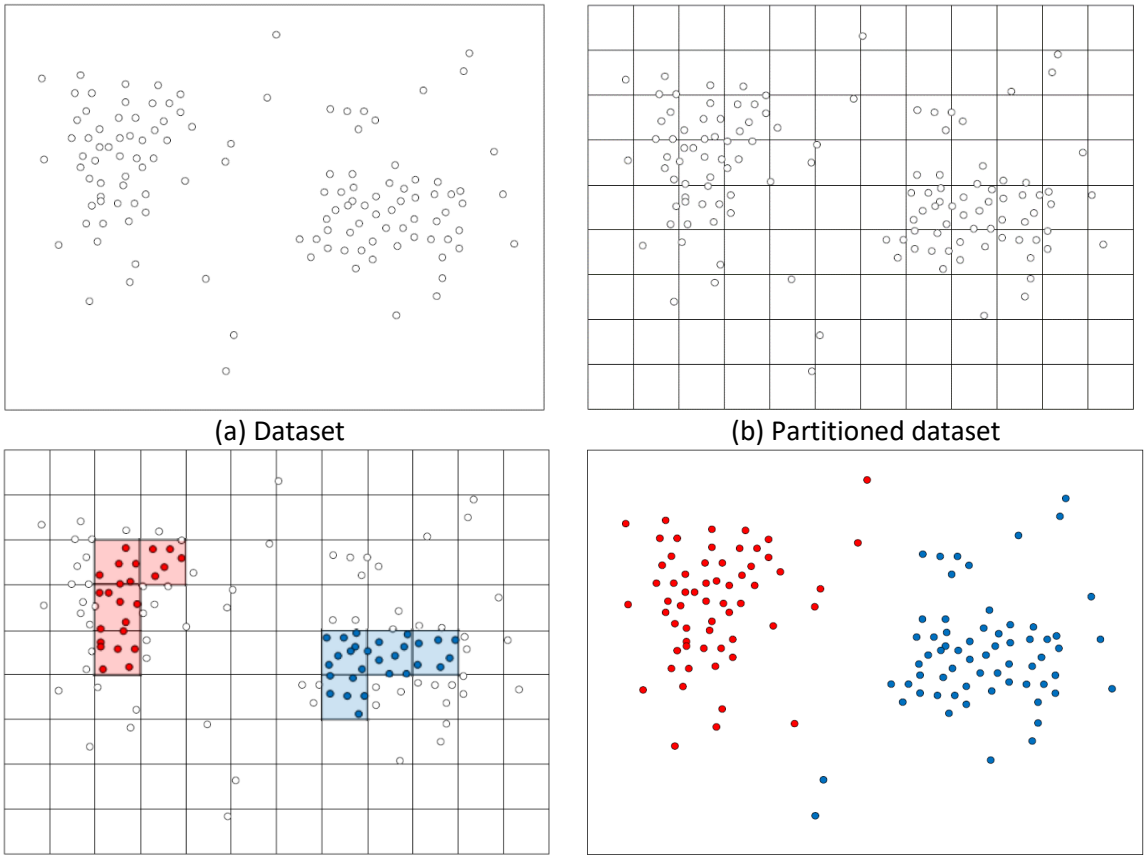
where

x	original data value
x'	normalised data value
μ	mean value of the dataset
σ	variance of the dataset
min	minimum data value in the dataset
max	maximum data value in the dataset

After the dataset has been normalised, the clustering process is started with the FLOCK algorithm given here.

Step 1:	The data, Figure 4-4 (a) is partitioned into equal sized bins for each dimension, Figure 4-4 (b).
Step 2:	Identify dense bins by using the density threshold. Those bins where the number of data points is more than the density threshold are identified as dense bins.
Step 3:	Combine adjacent dense bins into a cluster, Figure 4-4 (c).
Step 4:	Calculate mean value of each cluster and use it as a centroid.
Step 5:	Run the K-Means algorithm to identify each data point belong to a cluster, Figure 4-4 (d).

In a test blood sample, FLOCK identified 17 distinct B-cell subpopulations and identified novel plasmablast subpopulations responding to tetanus vaccinations (Qian et al., 2010). They claimed that the FLOCK approach is more objective and effective than manual gating, especially that the algorithm can automatically define the number of clusters, and is faster than the K-Mean algorithm (Ge and Sealfon, 2012). However, the data has to be normalised before clustering and can cause information loss. Moreover, the result is highly dependent on the two parameters: density threshold and size of bin, and different parameter values can lead to different clustering results. Another weakness is that FLOCK uses K-Mean to allocate every data point into a cluster, but this algorithm cannot deal well with noise, which is prevalent in FCM data.



(a) Dataset (b) Partitioned dataset

(c) Clustered dense bins (d) Each point allocated to a cluster

Figure 4-4 : Result of running FLOCK with a 2-dimensional dataset

4.4.5 flowMeans

Aghaeepour *et al.* (2010) proposed flowMeans as an automated gating framework that was based on K-Means clustering. Unlike the traditional K-Means technique, their new approach can identify the non-spherical shape of clusters by combining multi-clusters into a single subpopulation. flowMeans extends the flowMerge framework by using the statistical model with a clustering algorithm that is more time efficient. Besides replacing the statistical model, the merging criterion in flowMerge was also improved, and it estimates an appropriate number of clusters by using a change point detection approach. The algorithm comprises three procedures.

1.) *Identification of initial number of clusters*. The objective of the first procedure is to identify the reasonable maximum number of clusters, rather than using the number of all events, which results in excessive computational time. Mode-detection algorithms are used (Duong *et al.*, 2008) that have proved suitable for FCM data.

2.) *Merging of significant overlapping clusters*. After running the K-Means clustering algorithm, the two significant overlapping clusters or the two closest clusters are merged into one cluster. This procedure can be separated into two steps,

(i) calculate and update distance matrix for every cluster

(ii) merge the two clusters that are closest.

The similarity measurement used in this procedure is based on the Mahalanobis distance (McLachlan, 1999) and a symmetric semi-distance function between two clusters is defined as follows:

$$D(X_1, X_2) = min \begin{cases} \sqrt{(\mu_{x1} - \mu_{x2})\Sigma_{x1}^{-1}(\mu_{x1} - \mu_{x2})^T} \\ \sqrt{(\mu_{x1} - \mu_{x2})\Sigma_{x2}^{-1}(\mu_{x1} - \mu_{x2})^T} \end{cases} \quad (4.17)$$

3.) *Consideration of final number of clusters*. After several iteratively merging steps, the question is: how many clusters provide the most appropriate result? Segmented regression (piecewise regression) is used to identify the most appropriate number of clusters by detection of the change point. Segmented regression is the algorithm that divides data into two subsets, which are fitted to two different linear equations. The change point is then found by consideration of a plot of the number of clusters against the symmetric Mahalanobis semi-distance, as shown in Figure 4-5.

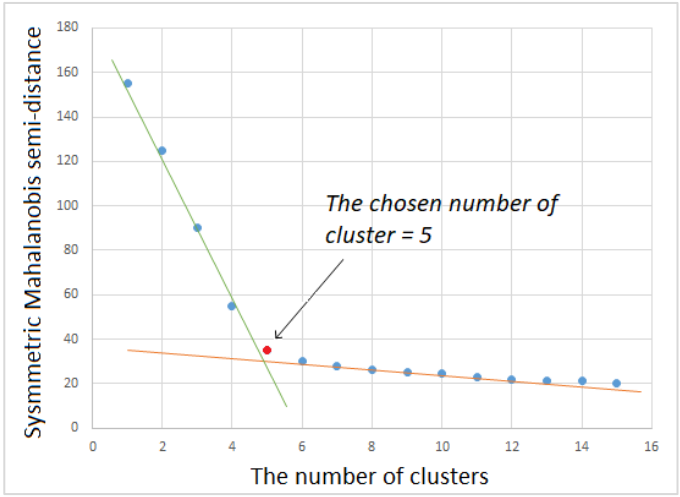


Figure 4-5 : Plot of the number of clusters against the symmetric Mahalanobis semi-distance

The flowMeans framework attempts to overcome the two main problems of the K-Means clustering algorithm: having to predefine the number of clusters, and being unable to identify non-convex shapes. The first shortcoming is attacked with change point detection based on segmented

regression used to find the most appropriate number of clusters. The second is attacked by merging together the significant overlapping clusters using a merging algorithm. However, the flowMeans framework is based on K-Means clustering so it is still sensitive to outliers, which are often found in FCM data, and it does not a really detect non-convex shapes.

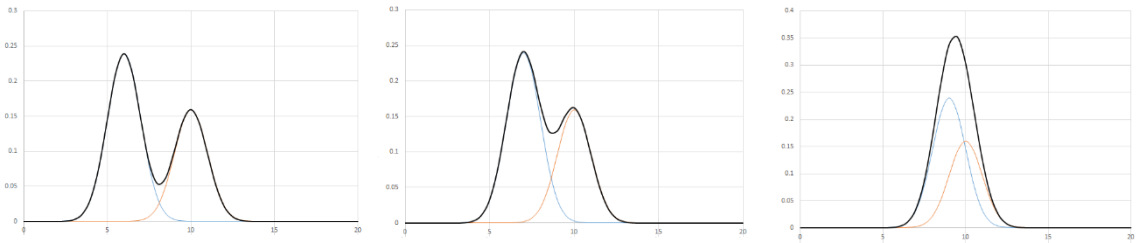
4.4.6 flowPeaks

The objective of the flowPeaks framework is to address the two main problems found in many FCM automated gating approaches. The first is relying on a finite mixture model, which can give poor results if the data distribution does not follow the assumptions or statistical models, such as flowMeans and FLAME. The second is an inability to apply to high-dimensioned data, such as Misty mountain (Sugár and Sealfon, 2010) and FLOCK. flowPeaks was proposed by Ge and Sealfon (2012), who claimed that their framework could be directly applied to high-dimensioned data and can identify arbitrarily shaped clusters. flowPeaks is a combination of finite mixture model and histogram exploration, as seen in Figure 4-6, where the black curve represents the histogram of two Gaussian models and the overall density of those two models. The overall density function is given by

$$f(x) = \sum_{i=1}^{K} w_i \, G(x|\mu_i, \sigma_i) \tag{4.18}$$

where

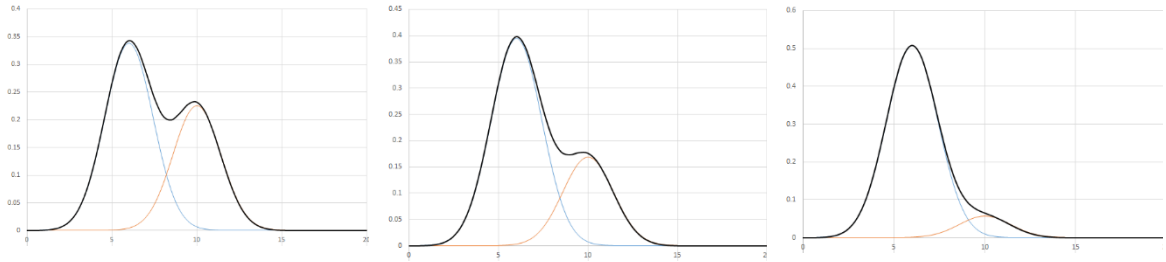
G	Gaussian function
w_i	proportion of the i^{th} model
μ_i	mean value of the i^{th} model
σ_i	variance of the i^{th} model
K	number of clusters



a) $\mu_1 = 6.0, \mu_2 = 10.0$
$w_1 = 0.6, w_2 = 0.4$
$\sigma_1^2 = \sigma_2^2 = 1.0$

b) $\mu_1 = 7.0, \mu_2 = 10.0$
$w_1 = 0.6, w_2 = 0.4$
$\sigma_1^2 = \sigma_2^2 = 1.0$

c) $\mu_1 = 9.0, \mu_2 = 10.0$
$w_1 = 0.6, w_2 = 0.4$
$\sigma_1^2 = \sigma_2^2 = 1.0$



d) $\mu_1 = 6.0, \mu_2 = 10.0$
$w_1 = 0.6, w_2 = 0.4$
$\sigma_1^2 = \sigma_2^2 = 0.5$

e) $\mu_1 = 6.0, \mu_2 = 10.0$
$w_1 = 0.7, w_2 = 0.3$
$\sigma_1^2 = \sigma_2^2 = 0.5$

f) $\mu_1 = 6.0, \mu_2 = 10.0$
$w_1 = 0.9, w_2 = 0.1$
$\sigma_1^2 = \sigma_2^2 = 0.5$

Figure 4-6 : flowPeaks framework demonstration

Figures 4-6a – 4-6c show two Gaussian distribution models with different means. Figures 4-6a and 4-6b have two distinct peaks that should be considered as two clusters, but Figure 4-6c has only one cluster because the overall density of the two models shows only one peak, in other words, the two Gaussian models of Figure 4-6c should be merged together. In the same way, Figures 4-6d – 4-6f show two models with different proportions, while Figure 4-6f exhibits only one cluster.

The flowPeaks framework can be roughly summarised in these steps.

1) Partition data by K-Means with large K in order to construct a large number of small clusters.
2) Model the Gaussian distribution model for individual clusters.
3) Merge local peaks together if the overall density within the models shows only a single peak.

In Step 2, they stated that the estimation of Σ_k may be excessively noisy, so they adjusted the variance matrix to make the density function smoother by

$$\tilde{\Sigma}_k = \lambda_k . h\Sigma_k + (1 - \lambda_k) . h_0\Sigma_0 \tag{4.19}$$

where h and h_0 are parameters that can be tuned in order to make the function rougher or smoother and .

Ge and Sealfon (2012) claimed that their technique is not only faster than state of art techniques, such as Misty Mountain, FLOCK, flowMeans, flowMerge and FLAME, but can also be applied to high dimensioned data and can identify irregularly shaped clusters. However, the user-defined parameters that make density function rougher or smoother can lead to different results for different parameter values. Moreover, this method still relies on a normal distribution that may not

be suitable for FCM data, even when some models with single peaks in the two overall density functions are merged together.

4.4.7 FlowGrid

Ye and Ho, 2018 proposed FlowGrid, an automated gating framework. This framework is a combination of DBSCAN and a grid-based clustering algorithm that provides high accuracy and better time efficiency compared with flowPeaks, FlowSOM, and FLOCK. DBSCAN can detect outliers and identify arbitrarily shaped clusters. FlowGrid combined the benefits of DBSCAN and reduced computational time by using equal-sized grids like the FLOCK algorithm. Each dimension is partitioned into an equal sized bin, so the total number of bins for d-dimensional data is $(N_{bin})^d$, where N_{bin} is the number of bins for each dimension. All data points in the same bin are treated as a single point by using a representative point, which is an index or label for the bin; moreover, only non-empty bins are considered. This is why this framework is faster than previous ones. Every Bin_i is labelled with a row of d positive numbers. For example, if $C_i = (5,2,3)$ is a coordinate of Bin_i, it means that the dataset has three dimensions, and this bin is located in the fifth bin of dimension one, the second bin of dimension two and the third bin of dimension three. The distance between Bin_i and Bin_j is defined as:

$$D(Bin_i, Bin_j) = \sqrt{\sum_{h=1}^{d} (C_{ih} - C_{jh})^2} \tag{4.20}$$

Some key terms involved in this algorithm are: $Den_b(Bin_i)$, $Den_c(Bin_i)$, directly connected, core bin, border bin, and noise bin, which are each defined below.

- Bin_i is directly connected to Bin_j if $D(Bin_i, Bin_j) \leq \varepsilon$, where ε is a parameter defined by the user
- $Den_b(Bin_i)$ is the density of Bin_i, which is the number of data points in Bin_i
- $Den_c(Bin_i)$ is the collective density of Bin_i, which is the total number of data points in Bin_i, where Bin_i is directly connected to Bin_j. Thus

$$Den_c(Bin_i) = \sum Den_b(Bin_j) \mid Bin_j \in \text{directly connected bin of } Bin_i \tag{4.21}$$

- Bin_i is a core bin if

  - $Den_b(Bin_i)$ is greater than $MinDen_b$, which is defined by the user, or
  - $Den_b(Bin_i)$ is greater than ρ% of its directly connected bins, where ρ is parameter defined by the user, or
  - $Den_c(Bin_i)$ is greater than $MinDen_c$, which is defined by the user.
- Bin_i is a border bin if it is not a core bin but is directly connected to a core bin
- Bin_i is a noise bin if it is neither core nor border bin
- Bin_i and Bin_j are considered as the same cluster if
  - they are directly connected and at least one of them is a core bin or
  - they are not directly connected but they are connected by a sequence of core bins

The FlowGrid clustering process is demonstrated in Appendix A, with a relevant data sample created for this algorithm.

Although FlowGrid is faster than many automated gating algorithms, provides high accuracy, and can deal with noise, there are still some user-defined parameters that can significantly affect the clustering result. Moreover, FlowGrid is based on DBCSCAN so it is not suitable for datasets with different density distributions.

4.5 Discussion

Each of automated gating techniques described is based on different clustering methods and have different strengths and weaknesses. Six requirements of automated gating techniques can be used to measure their performance, are explained below.

4.5.1 Computation time

Process time for manual analysis of FCM data depends on the number of experimental units, which are the number of cells and parameters. For many clinicians, it does not take much time to produce the analysis for small experiment units. However, manual analysis can become exceptionally time-consuming when the number of experimental units is large. Many automated gating techniques aim to improve time performance, such as FLOCK and FlowGrid. These two techniques reduce the number of data points by drawing a grid on the space and in each grid, cells are identified as a representative of all data in that cell, but the results are not significantly different from other techniques when applied to the dataset used in their experiments. Time is still a problem for some techniques such as flowMerge, FLAME, and flowMeans. The computational time is therefore one measurement used in this research.

4.5.2 Identification of cell populations regardless of shape

The most common requirement for automated gating techniques is the ability to identify non-spherical shaped clusters, which are always found in FCM data. Some statistically-based and partition-based clustering techniques can identify only spherical shapes because they assume that the clusters are spherical, so they do not work accurately with complicated geometrical shaped clusters such as K-means, K-Medians, K-Mediods, and Fuzzy C-Means. The clustering techniques that can identify clusters regardless of shape are model-based (e.g. EM, GMM) and density-based (e.g. DBSCAN, OPTICS). To help identify non-spherical shaped clusters, this research focuses on density-based clustering.

4.5.3 Detection of outliers

Outliers and detection of rare populations are still the problems for the gating process. Some automated gating techniques attempt to increase the performance of outlier detection, such as FLAME and FlowGrid. FLAME was improved from flowClust by replacing the Gaussian distribution model with a *t*-distribution model, making FLAME more robust to outliers than the flowClust technique. FlowGrid improved on FLOCK by replacing DBSCAN, a density-based clustering technique, and has an ability to automatically detect outliers, instead of K-Means that is highly sensitive to outliers. The performance of outlier detection is a metric used for measuring automated gating technique performed by this research.

4.5.4 Identify cell population accurately

Precisely accurate identification is still a limitation of automated gating techniques. The results from automated gating are currently a guideline for clinicians to classify cell populations. Manual gating is highly subjective when clinicians attempt to analyse original flow cytometry data. It would be easier and more standardised than manual gating if they analysed the data with the guidelines of cell populations provided from an automated gating technique. This research generates data synthesis with characteristics often found in blood samples. The data synthesis is clustered using a new technique and its accuracy then evaluated. The new automated gating technique will then be applied to real flow cytometry data and its accuracy evaluated by clinicians.

4.5.5 User-predefined parameters

Table 4-1 shows that most techniques require some user-defined parameters. Gating results from FlowGrid are not significantly different from others, but it is relatively fast compared to all other techniques. However, the FlowGrid technique requires five user-defined parameters that can be divided into mandatory parameters (ε, bin size, bin density) and optional parameters (collective bin density, ρ%). Although FlowGrid can provide acceptable accuracy and is relatively fast, it still has a problem with the number of user-defined parameters. This research intends to improve the performance of the FlowGrid technique by reducing the number of user-defined parameters and runtime.

4.5.6 Identification of optimal number of clusters

Generally, clinicians do not know the number of cell populations in the flow cytometry data before they consider it. Many automated gating techniques therefore attempt to automatically identify the number of clusters. Some are able to detect the number of clusters automatically, such as FLOCK and FlowGrid. Some can determine the number of clusters by applying different clustering techniques, such as flowClust which applies BIC, FLAME which applies minimise SWR, and flowMeans which applies change point detection, to identify the number of clusters.

4.6 Conclusions

Machine learning techniques have been applied in recent years to flow cytometry data for the gating process, especially the unsupervised machine learning technique. The major objectives of automated gating are to reduce the high subjectivity of manual gating, to standardise this process more, to reduce the process time, and to increase the performance of FCM analysis. Many automated gating frameworks have been proposed, and they can support clinicians in clustering homogeneous cells. There are many benefits to using these frameworks, but there are still some limitations. Some of their strengths and weaknesses are compared in Table 4-1.

Table 4-1 shows that most frameworks can identify non-spherical shape clusters, except flowClust and FLAME. All frameworks use different methods to detect the number of clusters, but FLOCK and FlowGrid can identify them automatically. FLOCK, flowMeans, and flowPeaks are sensitive to noise, while flowClust, FLAME, and flowMerge use t-distribution to deal with it; only FlowGrid can identify noise automatically. According to reported experiments, FlowGrid is the fastest clustering, while flowMerge is the slowest. flowMeans does not need to define the value of parameters, but five

user-defined parameters are needed for FlowGrid. Only FLOCK and FlowGrid cannot identify the dataset with different densities of sub-populations. FLAME, flowMeans, and flowPeaks do not have to transform data, which can be the cause of information loss.

In summary, FlowGrid is somewhat better than the others, but it still has too many user-defined parameters and cannot cluster different density datasets. Therefore, this project will focus on improving this technique.

Table 4-1: The comparison of individual automated gating technique

Method	Clustering base	Non-spherical shape identification	K identification (method used)	Dealing with noise	Time efficiency*	User-defined parameters	Various density detection	Data transformation	Directly apply to high dimensional data (max dimension)	Reference
flowClust	Multivariate t-mixture model	Yes	Yes (BIC)	Using t-mixture model with heavier tail	NA	- λ (Box-cox parameter)	Yes	Box-cox transformation	Yes (NA)	Lo, Brinkman and Gottardo, 2008
FLAME	Multivariate mixture of skew and heavy-tailed distribution	Yes	Yes (minimize the SWR)	Using heavy-tailed distribution	4	- skewness	Yes	-	Yes (NA)	Pyne *et al.*, 2009
flowMerge	Multivariate t-mixture model	Yes	Yes (change point analysis of entropy)	Using heavy-tailed distribution	5	- λ (Box-cox parameter) - Merging threshold	Yes	Box-cox transformation	Yes (NA)	Finak et al., 2009
FLOCK	- Grid base - Density base - K-means	Yes	Yes (auto detection)	Sensitive to noise	2	- Bin size - Density threshold	No	z-score and min-max normalisation	Yes (19)	Qian *et al.*, 2010
flowMeans	K-means with merging clusters	Yes	Yes (change point detection algorithm)	Sensitive to noise	3	- K	Yes	-	Yes (NA)	Aghaeepour *et al.*, 2010
flowPeaks	- K-means - Gaussian mixture model	Yes	No (identify largest K first and combine clusters later)	Sensitive to noise	2	- Merging threshold	Yes	-	Yes (NA)	Sugár and Sealfon, 2010
FlowGrid	- Density base (DBSCAN) - Grid base	Yes	Yes (auto detection)	Auto detection	1	- ε - bin size - density of a bin - density of collective bins - ρ%	No	Rage data between 1 and bin size plus one (1 to bin size +1)	Yes (NA)	Ye and Ho, 2018

* FlowGrid is fastest (1), flowMerge is slowest (5)

Chapter 5 Results and Discussion

This chapter provides the results of applying the clustering methods to datasets generated with different characteristics. The clustering methods used to evaluate the datasets were K-means, DBSCAN, OPTICS, and FlowGrid. These techniques were applied to identifying clusters in the datasets so that their performance when dealing with different types of datasets could be compared. The experiment was conducted on a computer with these specifications:

- Intel Core i7-6700 CPU @ 3.40GHz
- RAM 16.0 GB
- Windows 10 Enterprise, 64-bit Operating System

Three experiments were conducted, each with a different type of dataset.

5.1 Reference Datasets

Patterns or clusters of sample datasets obtained from different donors might exhibit different characteristics, even when they are conducted with the same flow cytometry experimental design. According to Chapter 2, a real data sample distribution of individual markers usually follows a normal distribution or Gaussian distribution with different parameters. Therefore, cluster shapes can be symmetric or asymmetric depending on the donors and markers used. The clusters might be circle-shaped with different radii or cigar-shaped with different width, height, and angle, as can be seen in Figure 5-1. For this reason, datasets used in these baseline experiments are circles with different radii (Mouse dataset), cigar shapes (Barcode dataset), and asymmetric shapes that mimic a real sample (Imitative dataset).

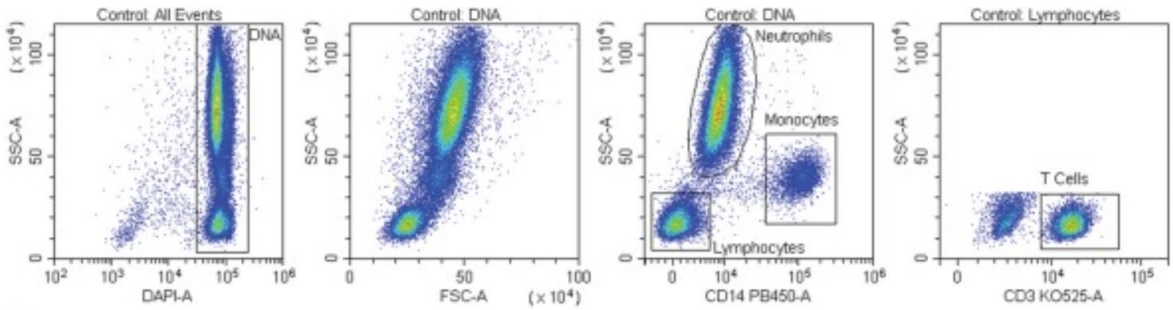


Figure 5-1 : Shape of clusters in the FCM dataset provided in Brittain, G.C. and Gulnik, S. (2017)

The dataset used in the experiments are mouse datasets, barcode datasets, and mimic dataset. The details for each datasets are described in following sections.

5.1.1 Mouse datasets

As discussed earlier, shapes of clusters in FCM data can be symmetric or asymmetric. Mouse dataset is symmetric and has three clusters. The shapes are circles with different radii that are often found in real sample, as can be seen in Figure 1-1, 1-2, and 5-1. The mouse dataset is that used to test how well the clustering techniques identify the three clusters. It is called the mouse dataset as the cluster shapes look like a mouse face with two ears. The data is generated within this experiment.

The mouse dataset is 2-dimensional, and was generated using the function `runif()` in RStudio 3.5.2. The structure of the dataset used in this experiment consists of three circular-shaped clusters for each dataset, as shown in Figure 5.2 This shows 377, 94, and 94 data points that belong to Cluster 1, Cluster 2, and Cluster 3 respectively.

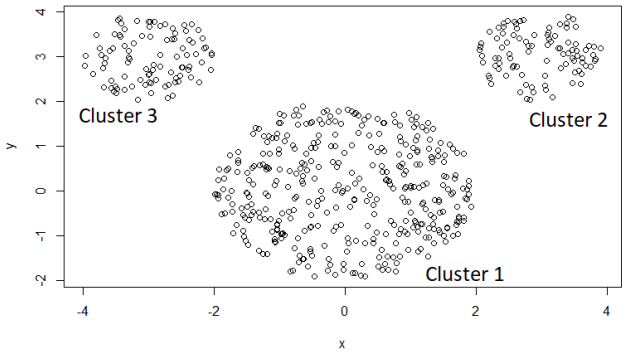


Figure 5-2 : The structure of the mouse dataset

The mouse dataset was divided into six levels of closeness within each cluster. For each level, three datasets were randomised. The criterion for randomisation for each level is shown in Table 5-1.

Table 5-1: Centre and radius of each cluster for each level of closeness

Closeness level	Cluster 1		Cluster 2		Cluster 3	
	centre	*radius*	*centre*	*radius*	*centre*	*radius*
Level1	x = 0.0 y = 0.0	2.0	x = 3.0 y = 3.0	1.0	x = –3.0 y = 3.0	1.0
Level2	x = 0.0 y = 0.0	2.0	x = 2.8 y = 2.8	1.0	x = –2.8 y = 2.8	1.0
Level3	x = 0.0 y = 0.0	2.0	x = 2.7 y = 2.7	1.0	x = –2.7 y = 2.7	1.0
Level4	x = 0.0 y = 0.0	2.0	x = 2.6 y = 2.6	1.0	x = –2.6 y = 2.6	1.0
Level5	x = 0.0 y = 0.0	2.0	x = 2.5 y = 2.5	1.0	x = –2.5 y = 2.5	1.0
Level6	x = 0.0 y = 0.0	2.0	x = 2.4 y = 2.4	1.0	x = –2.4 y = 2.4	1.0

The algorithm to randomly generate the points in the mouse dataset operates as follows.

Step 1: Iteratively create a random data point between –5.0 and 5.0, and assign it to Cluster 1 if the distance between the random data point and the centre of Cluster 1 is less than or equal to the radius of Cluster 1 (2.0). Continue until the number of data points assigned to Cluster 1 is equal to 377.

Step 2: Iteratively create a random a data point between –5.0 and 5.0, and assign it to Cluster 2 if the distance between the random data point and the centre of Cluster 2 is less than or equal to the radius of Cluster 2 (1.0). Continue until the number of data points assigned to Cluster 2 is equal to 94.

Step 3: Iteratively create a random a data point between –5.0 and 5.0, and assign it to Cluster 3 if the distance between the random data point and the centre of Cluster 3 is less than or equal to the radius of Cluster 3 (1.0). Continue until the number of data points assigned to Cluster 1 is equal to 94.

The centres and boundaries of each level of closeness for every cluster is shown in Figure 5-3.

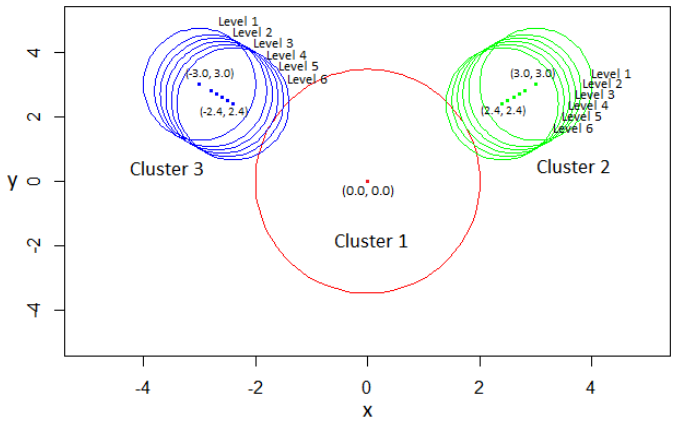
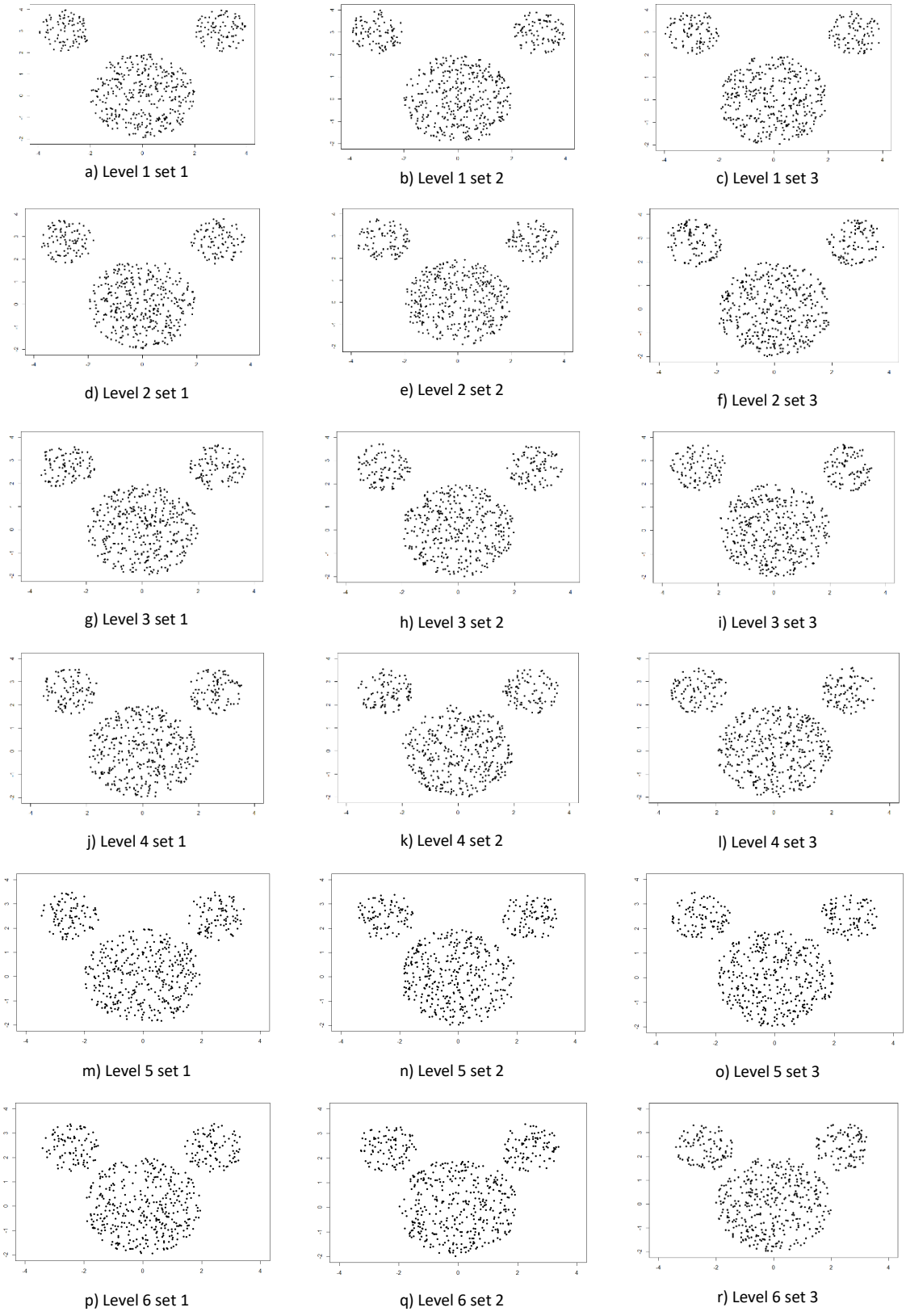


Figure 5-3 : Centres and boundaries of each level in the mouse dataset

All 18 mouse datasets for the experiment are shown in Figure 5-4. In this diagram, they are all coloured the same for convenience, but in subsequent figures, the points in each cluster are differently coloured, red if they belong to Cluster 1, green if they belong to Cluster 2, and blue for Cluster 3.



a) Level 1 set 1 b) Level 1 set 2 c) Level 1 set 3

d) Level 2 set 1 e) Level 2 set 2 f) Level 2 set 3

g) Level 3 set 1 h) Level 3 set 2 i) Level 3 set 3

j) Level 4 set 1 k) Level 4 set 2 l) Level 4 set 3

m) Level 5 set 1 n) Level 5 set 2 o) Level 5 set 3

p) Level 6 set 1 q) Level 6 set 2 r) Level 6 set 3

Figure 5-4 : Scatter plot of the 18 datasets

5.1.2 Barcode dataset

Another common cluster shape that often found in FCM data is cigar shape. Cigar shape is an oval that is very wide and small in height, as can be seen in Figure 2-9 and Figure 5-1. The barcode dataset were 2-dimensional, and were randomised with the function `runif()` in RStudio 3.5.2. The datasets used were called 'barcode datasets', consisting of three lines of clusters (barcode) for each dataset, as shown in Figure 5-5. Each dataset was divided into six levels of closeness between each cluster.

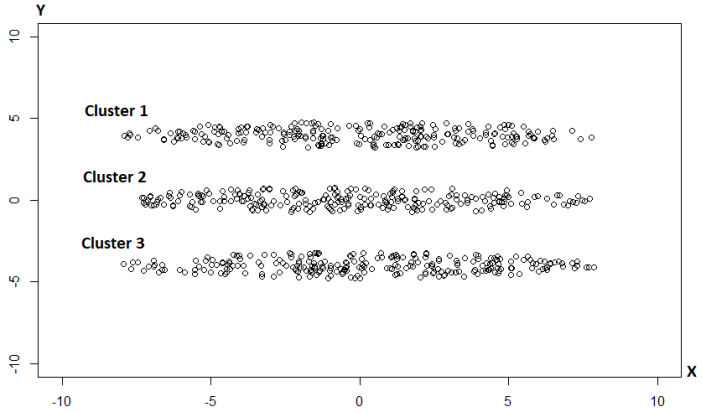


Figure 5-5 : The structure of the barcode dataset

The boundaries of Cluster 1 and Cluster 3 were moved closer at each level, as can be seen in Figure 5-6. Each level consisted of three datasets, so there were 18 barcode datasets used in the experiments. All data points were randomised between min = –10 and max = 10 by function `runif()` in RStudio version 3.5.2. The number of data points in each dataset was 723 (241 data points per cluster). Each data point had to fall within the area of three different ellipse functions, given by the ellipse equation, shown in Equation 5-1.

$$\frac{x^2}{h^2} + \frac{y^2}{k^2} \leq 1 \tag{5-1}$$

where

x, *y* are the coordinates of any point falling in the ellipse area

h, *k* are the radiuses on the *x* and *y* axes respectively

Three datasets were randomised for each closeness level, resulting in 18 datasets used for the experiments, example scatter plots of which are shown in Figure 5-7.

Based on the data in Table 5-2, the boundaries of each cluster are shown in Figure 5-6.

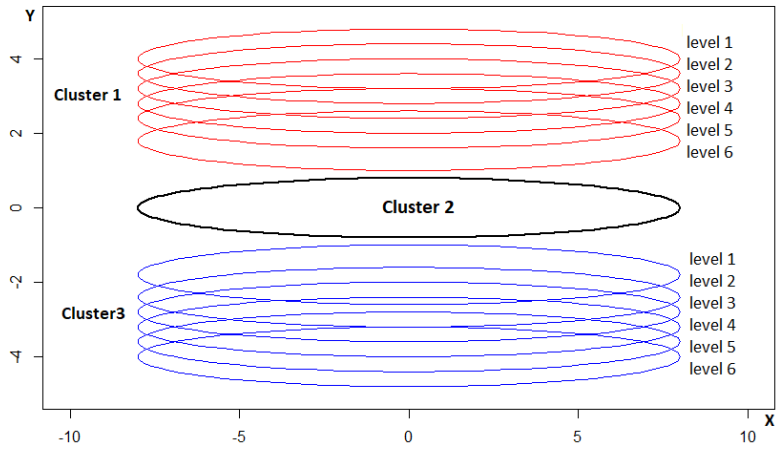
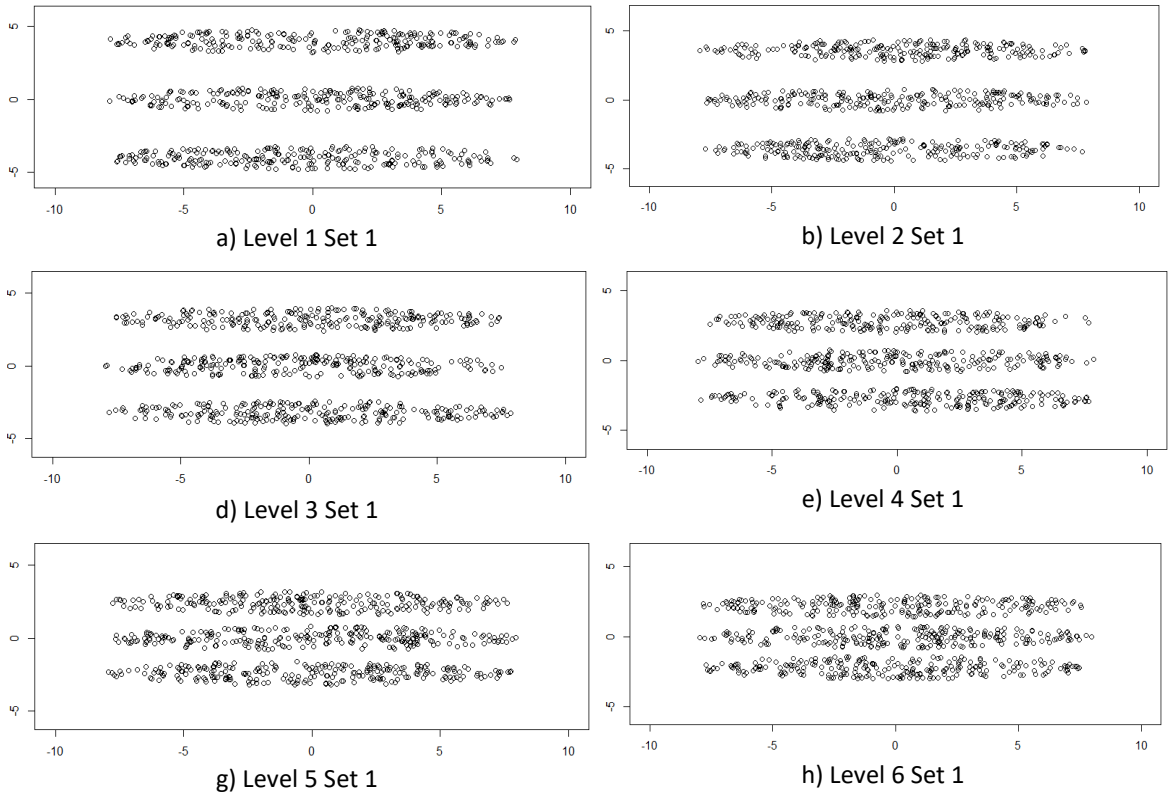


Figure 5-6 : Boundaries of each level for every cluster in the barcode dataset



a) Level 1 Set 1 b) Level 2 Set 1

d) Level 3 Set 1 e) Level 4 Set 1

g) Level 5 Set 1 h) Level 6 Set 1

Figure 5-7 : Scatter plots of barcode datasets

Table 5-2: Details of each closeness level of ellipse functions

Closeness level	Cluster1			Cluster 2			Cluster 3		
	centre	*h*	*k*	*centre*	*h*	*k*	*centre*	*h*	*k*
Level 1	x = 0.0 y = 4.0	8.0	0.8	x = 0.0 y = 0.0	8.0	0.8	x = 0.0 y = –4.0	8.0	0.8
Level 2	x = 0.0 y = 3.6	8.0	0.8	x = 0.0 y = 0.0	8.0	0.8	x = 0.0 y = –3.6	8.0	0.8
Level 3	x = 0.0 y = 3.2	8.0	0.8	x = 0.0 y = 0.0	8.0	0.8	x = 0.0 y = –3.2	8.0	0.8
Level 4	x = 0.0 y = 2.8	8.0	0.8	x = 0.0 y = 0.0	8.0	0.8	x = 0.0 y = –2.8	8.0	0.8
Level 5	x = 0.0 y = 2.4	8.0	0.8	x = 0.0 y = 0.0	8.0	0.8	x = 0.0 y = –2.4	8.0	0.8
Level 6	x = 0.0 y = 1.8	8.0	0.8	x = 0.0 y = 0.0	8.0	0.8	x = 0.0 y = –1.8	8.0	0.8

5.1.3 Imitative dataset

The imitative datasets used in the experiments were 2-dimensional datasets generated by the function `rmvnorm (n, mean, sigma)` in RStudio 3.5.2. This function randomly generates data using a multivariate normal distribution, which is often found on FCM data, as shown in Figure 5-1. For this function, three arguments are required: the number of data points (*n*), an average of the data (*mean*), and a covariance matrix (*sigma*). The imitative datasets used consisted of three clusters for each dataset. The number of data points in Clusters 1, 2 and 3 were 5 000, 2 500, and 2 500 respectively. They were generated with four different argument sets, as shown in Table 5-3, and generated three times for each set of arguments, so 12 datasets were used in the experiments. The datasets were divided into four levels of overlapping between the clusters; examples of these imitative datasets can be seen in Figure 5-8.

Table 5-3: Values of arguments for each level of closeness

Cluster	**Sigma (Covariance matrix)**	**N**	**Means (Centres)**			
			Level 1	**Level 2**	**Level 3**	**Level 4**
1	$\begin{vmatrix} 6 & 15 \\ 15 & 120 \end{vmatrix}$	5,000	x = 5, y = 35	x = 3, y =30	x = 1, y = 25	x = –1, y = 20
2	$\begin{vmatrix} 2 & 0.3 \\ 0.3 & 5 \end{vmatrix}$	2,500	x = –5, y = –10	x = –5, y = –10	x = –5, y = –10	x = –5, y = –10
3	$\begin{vmatrix} 3 & 2 \\ 2 & 10 \end{vmatrix}$	2,500	x = 16, y = 1	x = 12, y = 1	x = 8, y = 1	x = 4, y = 1

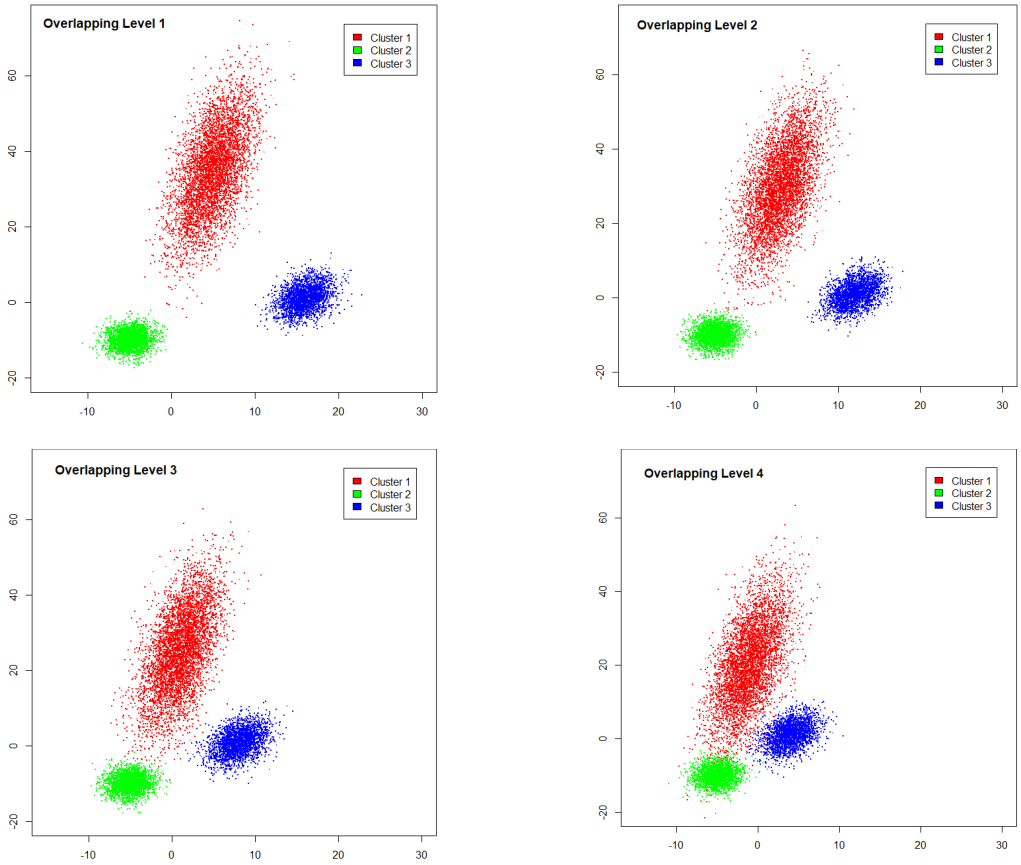


Figure 5-8 : The structure of imitative datasets with different overlapping

5.2 Experimentation

The experiments in this Chapter were divided regarding to the clustering techniques used. The techniques used in the experiments are K-means, DBSCAN, OPTICS, and FlowGrid. Each techniques were applied to mouse dataset, barcode dataset, and imitative dataset, so there are 3 sub-experiments for each clustering techniques.

5.2.1 Experiment 1: K-means clustering technique

As mentioned above, K-means were applied to 3 dataset so there are 3 sub-experiments in Experiment 1, consisting of Experiment 1-1 (applying K-means to mouse dataset), Experiment 1-2 (applying K-means to barcode dataset) and Experiment 1-3 (applying K-means to imitative dataset). The objective of this experiment is to investigate how close the clusters are to each other, when the radius or width of each cluster is different.

5.2.1.1 Experiment 1-1: Applying K-means to the Mouse datasets

All 18 mouse datasets described above were clustered by the function `kmeans()` in RStudio version 3.5.2, shown in the commands below; the results are shown in Table 5-2.

```
set.seed(11)  # to ensure that the initial centres are constant
res<-kmeans(data,centers = 3)  # the number of clusters = 3
```

The accuracy of each cluster is defined as:

$$accuracy = 100 \times \frac{n_0}{n_1} \tag{5-2}$$

where

n_0 is the number of correctly clustered data points in a cluster

n_1 is the number of all data points in a cluster

The overall accuracy of each dataset is defined as:

$$overall\ accuracy = 100 \times \frac{N_0}{N_1} \tag{5-3}$$

where

N_0 is the number of correctly clustered data points in the dataset

N_1 is the number of all data points in the dataset

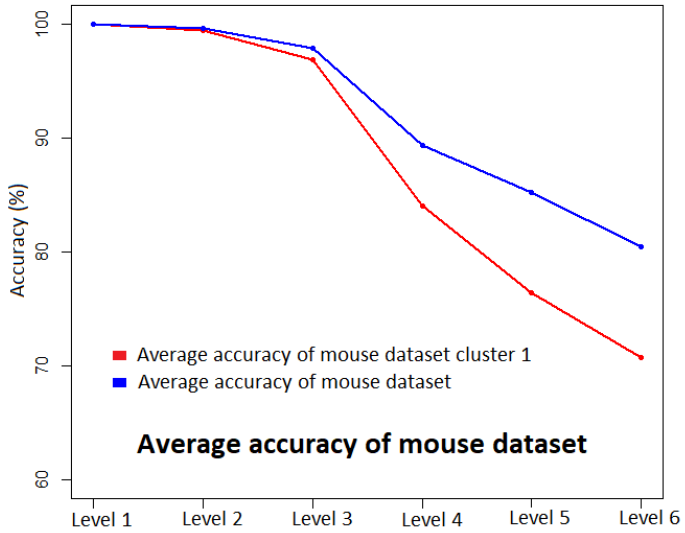
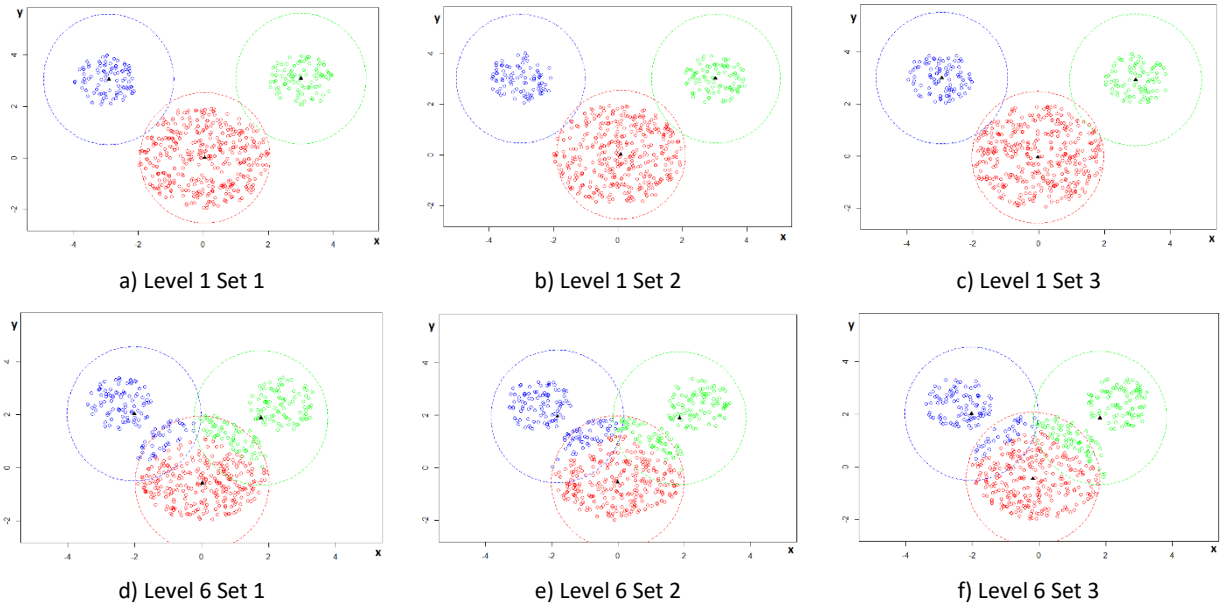


Figure 5-9 : Average accuracy of K-means applying to the Mouse datasets

Table 5-4: Result of the Experiment 1-1: applying K-means to the Mouse datasets

Dataset	Final centres						Runtime (ms.)	Accuracy (%)			
	Cluster 1		Cluster 2		Cluster 3						
	x	y	x	y	x	y		Cluster 1	Cluster 2	Cluster 3	Overall
L1 Set1	0.05	0.01	3.00	3.08	−2.90	3.04	2.00	100.00	100.00	100.00	100.00
L1 Set2	0.10	0.01	3.04	3.00	−3.01	2.97	2.01	100.00	100.00	100.00	100.00
L1 Set3	−0.02	−0.01	2.95	2.92	−2.94	2.99	2.01	100.00	100.00	100.00	100.00
Average							**2.01**	**100.00**	**100.00**	**100.00**	**100.00**
L2 Set1	−0.04	−0.03	2.79	2.84	−2.81	2.82	2.50	100.00	100.00	100.00	100.00
L2 Set2	0.05	0.10	2.84	2.77	−2.65	2.74	2.51	99.20	100.00	100.00	99.47
L2 Set3	−0.03	−0.08	2.72	2.76	−2.75	2.87	2.51	98.94	100.00	100.00	99.29
Average							**2.51**	**99.38**	**100.00**	**100.00**	**99.59**
L3 Set1	0.02	0.05	2.61	2.62	−2.67	2.72	2.51	98.67	100.00	100.00	99.12
L3 Set2	0.06	0.01	2.70	2.69	−2.50	2.41	2.51	96.29	100.00	100.00	97.52
L3 Set3	−0.07	−0.10	2.47	2.51	−2.67	2.62	3.01	95.49	100.00	100.00	96.99
Average							**2.67**	**96.82**	**100.00**	**100.00**	**97.88**
L4 Set1	0.12	−0.17	2.42	2.48	−2.16	2.22	3.01	87.53	100.00	100.00	91.68
L4 Set2	0.08	−0.23	2.32	2.39	−2.38	2.37	2.99	92.57	100.00	100.00	95.04
L4 Set3	−0.12	−0.37	1.79	1.87	−2.35	2.34	3.01	71.88	100.00	100.00	81.24
Average							**3.00**	**83.99**	**100.00**	**100.00**	**89.32**
L5 Set1	−0.04	−0.32	2.01	12.00	−2.23	2.20	2.52	79.31	100.00	100.00	89.19
L5 Set2	−0.17	−0.38	1.97	2.01	−2.07	2.06	2.00	74.27	100.00	100.00	82.83
L5 Set3	0.13	−0.46	2.23	2.26	−1.77	1.84	2.00	75.60	100.00	100.00	83.72
Average							**2.17**	**76.39**	**100.00**	**100.00**	**85.25**
L6 Set1	0.02	−0.59	1.77	1.87	−2.01	2.02	3.01	71.09	100.00	100.00	80.71
L6 Set2	−0.01	−0.55	1.86	1.89	−1.83	1.95	2.50	68.44	100.00	100.00	78.94
L6 Set3	−0.18	−0.46	1.83	1.85	−2.03	2.01	2.51	72.68	100.00	100.00	81.77
Average							**2.67**	**70.74**	**100.00**	**100.00**	**80.47**
All Average							**2.51**	**87.89**	**100.00**	**100.00**	**95.96**



a) Level 1 Set 1 b) Level 1 Set 2 c) Level 1 Set 3

d) Level 6 Set 1 e) Level 6 Set 2 f) Level 6 Set 3

Figure 5-10 : Result of the Experiment 1-1: Applying K-means to the Mouse datasets

Table 5-4 shows the results of applying the K-means clustering algorithm to the mouse datasets with different closeness between each cluster. The final centres of each clusters were quite close to the initial centres given in Table 5-1 when the datasets were generated. The average runtime was approximately 2.5 ms, meaning that it did not take much time to determine these dataset

clusters. However, the time spent for this was dependent on the number of data points and dimensions of the dataset. Only datasets with the closeness Level 1 were clustered 100% correctly. As can be seen in Figure 5-10, as the clusters get closer, the average percentages of accuracy drop.

The K-means clustering technique is strongly based on distance; as a result, a data point is considered as a member of the cluster depending on the closeness distance between the cluster centre (average of the data points that belong to the cluster) and that data point, as shown in Figure 5-10. Therefore, K-means clustering technique is not suitable for datasets with non-spherical shapes that are often found on FCM data.

5.2.1.2 Experiment 1-2: Applying K-means to the Barcode datasets

All barcode datasets were clustered using the function `kmeans()` in RStudio version 3.5.2. The commands used are shown below, and the results are shown in Table 5-5.

```
set.seed(11)  # to ensure that the initial centres are constant
res<-kmeans(data,centers = 3)  # the number of cluster = 3
```

As in Experiment 1-1, the accuracy of each cluster is defined by Equation 5-2, and the overall accuracy of each dataset is defined by Equation 5-3.

The purpose of this experiment was to see the effect of changing closeness among each cluster on the accuracy of K-means when applied to the Barcode dataset. The final centres of each cluster depend on initial centroids, and are not consistent because the cluster shapes of these datasets are non-spherical, and the similarity measurement of the K-means algorithm is strongly based on the distance between data points (centroids and a data point), as can be seen in Table 5-5. The average accuracy was only 51.42% so the result of applying K-means to the barcode datasets was not acceptable – even for dataset Level 1, which clearly separates the clusters. As can be seen in Figure 5-11, when the boundaries between each cluster are closer, the average percentages of accuracy decline.

Table 5-5: Result of the Experiment 1-2: Applying K-means to the Barcode datasets

Dataset	Final centres						Runtime (ms.)	Accuracy (%)			
	Cluster 1		Cluster 2		Cluster 3						
	x	y	x	y	x	y		Cluster 1	Cluster 2	Cluster 3	Overall
L1 Set1	2.78	4.05	2.84	−1.97	−4.12	−0.27	15.04	62.66	56.02	46.89	55.19
L1 Set2	1.80	4.00	2.91	−2.01	−4.15	−0.57	2.01	71.37	44.81	56.43	57.54
L1 Set3	3.07	2.22	−4.00	0.28	3.00	−3.98	2.51	52.70	58.51	66.39	59.20
						Average	**6.52**	**62.24**	**53.11**	**56.57**	**57.31**
L2 Set1	−1.49	3.19	3.99	−0.22	−3.40	−2.15	3.51	64.32	71.37	44.40	60.03
L2 Set2	3.47	2.20	−4.29	0.31	1.55	−2.89	2.00	61.83	39.83	71.37	57.68
L2 Set3	2.85	2.34	−3.71	−0.14	2.65	−3.09	1.50	66.80	49.38	63.90	60.03
						Average	**2.34**	**64.32**	**53.53**	**59.89**	**59.25**
L3 Set1	2.00	2.38	−3.79	−0.32	3.40	−2.31	2.01	67.63	53.94	48.55	56.71
L3 Set2	2.14	3.01	−4.08	−0.03	2.96	−1.96	2.02	49.79	69.29	66.39	61.83
L3 Set3	3.17	1.93	−3.52	0.37	2.21	−2.80	1.98	56.02	47.72	69.71	57.81
						Average	**2.00**	**57.81**	**56.98**	**61.55**	**58.78**
L4 Set1	−2.05	2.31	−3.70	−1.48	3.60	−0.60	2.01	70.12	56.85	39.00	55.33
L4 Set2	0.40	2.06	4.16	−0.89	−4.08	−0.70	2.01	34.85	59.34	52.70	48.96
L4 Set3	1.82	2.10	−4.08	−0.28	3.54	−1.81	2.01	42.74	67.63	56.43	55.60
						Average	**2.01**	**49.24**	**61.27**	**49.38**	**53.30**
L5 Set1	−0.11	0.36	4.54	−0.17	−4.78	−0.19	2.00	43.57	34.44	36.10	38.04
L5 Set2	2.37	1.76	−3.78	−0.06	3.80	−1.63	2.51	57.68	47.72	55.60	53.67
L5 Set3	−4.90	0.10	0.09	0.21	4.81	−0.42	2.01	37.34	41.49	34.85	37.90
						Average	**2.17**	**46.20**	**41.22**	**42.28**	**43.20**
L6 Set1	−4.75	0.10	4.88	0.14	0.12	−0.21	2.50	32.78	45.64	29.88	36.10
L6 Set2	4.54	0.16	−0.09	−0.03	−4.65	−0.17	2.51	35.27	41.08	32.37	36.24
L6 Set3	5.01	0.16	0.38	−0.07	−4.71	−0.03	2.01	45.64	34.02	33.61	37.76
						Average	**2.34**	**37.90**	**40.25**	**31.95**	**36.70**
						All Average	**2.90**	**52.95**	**51.06**	**50.25**	**51.42**

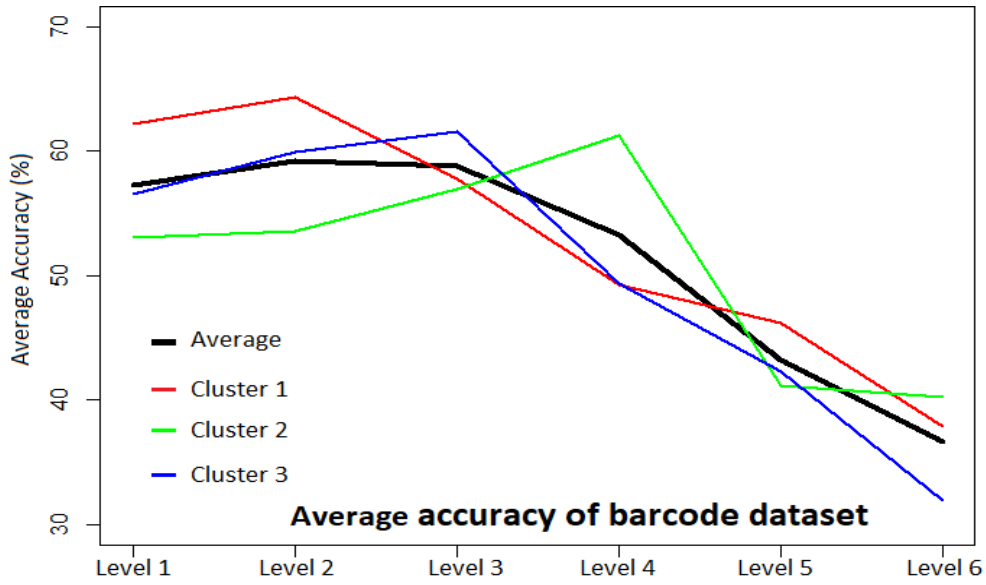


Figure 5-11 : Decreasing accuracy as the boundaries of the clusters get closer

Table 5-6: Result of the Experiment 1-3: Applying K-means to the Imitative datasets

Dataset		Accuracy (%)				Runtime (ms)
		Cluster 1	Cluster 2	Cluster 3	Overall	
Level 1	Set 1	50.44	100.00	0.00	50.22	2.01
	Set 2	52.10	100.00	0.00	51.05	2.01
	Set 3	94.08	100.00	100.00	97.04	1.01
Average		65.54	100.00	33.33	66.10	1.67
Level 2	Set 1	53.60	100.00	0.24	51.86	2.01
	Set 2	50.84	100.00	0.00	50.42	2.01
	Set 3	89.88	100.00	100.00	94.94	2.01
Average		64.77	100.00	33.41	65.74	2.01
Level 3	Set 1	80.96	100.00	99.64	90.39	0.99
	Set 2	52.00	100.00	0.00	51.00	2.01
	Set 3	48.70	100.00	0.00	49.35	3.01
Average		60.55	100.00	33.21	63.58	2.00
Level 4	Set 1	66.82	100.00	96.44	88.52	2.01
	Set 2	67.58	100.00	96.12	82.82	3.01
	Set 3	67.50	100.00	96.36	82.84	2.01
Average		67.30	100.00	96.31	84.73	2.34
All Average		**64.54**	**100.00**	**49.07**	**70.04**	**2.01**

5.2.1.3 Experiment: 1-3: Applying K-means to the Imitative datasets

All imitative datasets were clustered using function `kmeans()` in RStudio version 3.5.2; the following commands were used, and the results are shown in Table 5-6.

```
set.seed(10)  # to ensure that the initial centres are constant
res<-kmeans(data, centers = 3)  # the number of cluster = 3
```

The accuracy of each cluster was defined by Equation 5-1, and the overall accuracy of each dataset was defined by Equation 5-3.

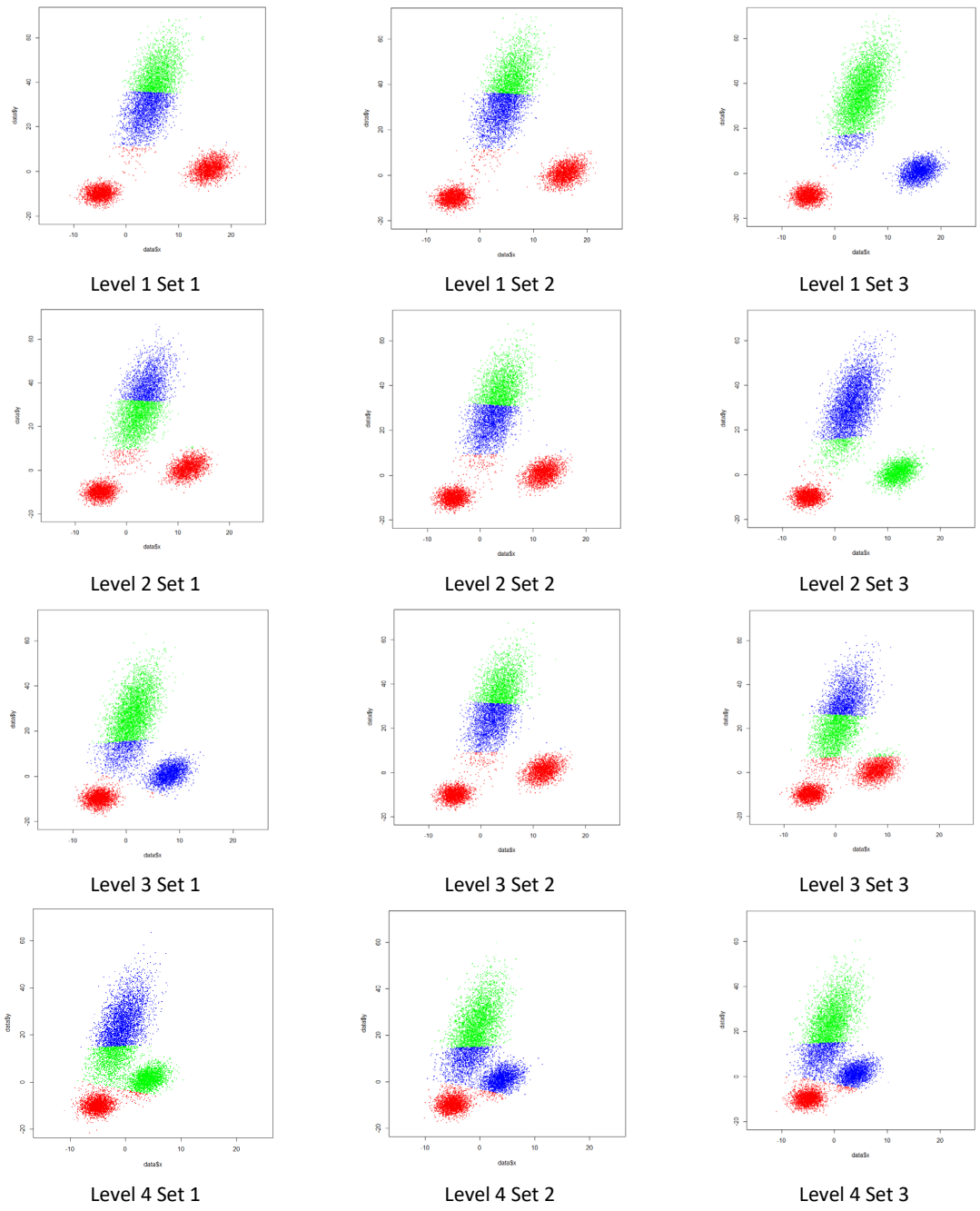


Figure 5-12 : Result of the Experiment 1-3: Applying K-means to the Imitative datasets

Table 5-6 shows the results of applying the K-means clustering algorithm to the imitative datasets with different closeness levels between each cluster. The results are not precise because the shape of these dataset clusters was non-spherical, and the clusters depended on initial centroids. Although the clustering results of imitative dataset Level 4 are quite consistent, the results could not be accepted, as can be seen in Figure 5-12.

5.2.2 Experiment 2: DBSCAN clustering technique

DBSCAN is a density-based clustering technique, in other word DBSCAN identifies the object belongs to a cluster if it is close to many points from that cluster. Therefore, DBSCAN has an ability to discover arbitrary shaped clusters and automatically detect noise. In Experiment 2, DBSCAN was applied to 3 dataset so there are 3 sub-experiments, consisting of Experiment 2-1 (applying DBSCAN to mouse dataset), Experiment 2-2 (applying DBSCAN to barcode dataset) and Experiment 2-3 (applying DBSCAN to imitative dataset). The objective of this experiment is to investigate how close the clusters are to each other, when the radius or width of each cluster is different.

5.2.2.1 Experiment: 2-1: Applying DBSCAN to the Mouse datasets

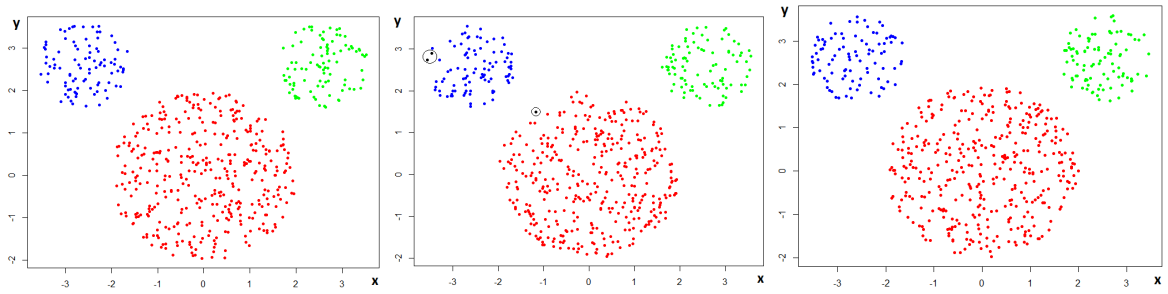
All the same mouse datasets were clustered using the DBSCAN clustering technique, with ε = 0.5 and MinPts = 10. The clustering function `dbscan()` was provided in RStudio 3.5.2, as can be seen in the R commands below:

```
res<-fpc::dbscan(data,eps = 0.5, MinPts = 10) # run the dbscan clustering with eps = 0.5 and MinPts = 10
```

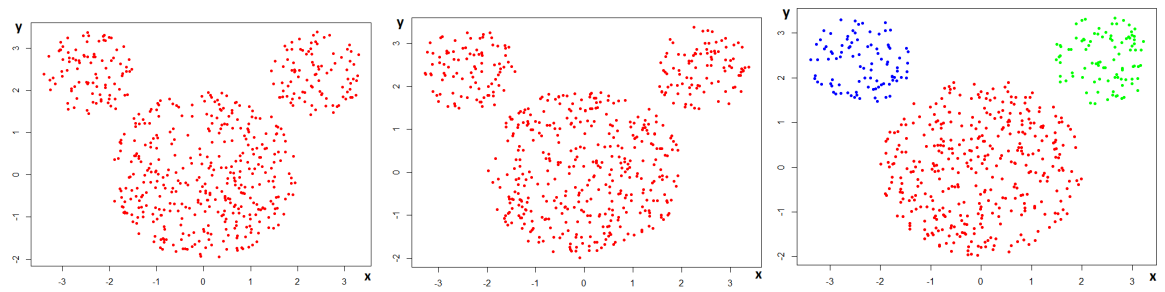
Table 5-7 shows that DBSCAN can separate data points into each cluster more accurately than the K-means clustering technique; the one exception is mouse datasets with closeness Level 6, which cannot be clustered by DBSCAN when ε = 0.5 and MinPts = 10. Figure 5-7 shows the clustering of three data sets. In Level 4 Set 2, two black circles are seen, which indicates that these points are incorrectly clustered. All the data points in the dataset Level 6 Set 1 and Level 6 Set 2 could not be separated by DBSCAN (all points red), while the dataset Level 6 Set 3 was clustered correctly. However, the result of clustering would have been better if the value parameters were better optimised. In terms of runtime, K-means was about 20 times faster than DBSCAN in this case.

Table 5-7: Result of the Experiment 2-1: Applying DBSCAN to the Mouse datasets

Dataset	Runtime (ms.)	Accuracy (%)			
		Cluster 1	Cluster 2	Cluster 3	Overall
L1 Set1	67.68	100.00	100.00	100.00	100.00
L1 Set2	47.16	100.00	100.00	100.00	100.00
L1 Set3	38.10	100.00	100.00	100.00	100.00
Average	**50.98**	**100.00**	**100.00**	**100.00**	**100.00**
L2 Set1	46.12	100.00	100.00	100.00	100.00
L2 Set2	52.64	100.00	100.00	100.00	100.00
L2 Set3	57.15	100.00	100.00	100.00	100.00
Average	**51.97**	**100.00**	**100.00**	**100.00**	**100.00**
L3 Set1	39.61	100.00	100.00	100.00	100.00
L3 Set2	68.68	100.00	100.00	100.00	100.00
L3 Set3	43.12	100.00	100.00	100.00	100.00
Average	**50.47**	**100.00**	**100.00**	**100.00**	**100.00**
L4 Set1	44.13	100.00	100.00	100.00	100.00
L4 Set2	39.11	99.73	100.00	97.87	99.87
L4 Set3	44.12	100.00	100.00	100.00	100.00
Average	**42.51**	**99.91**	**100.00**	**99.29**	**99.47**
L5 Set1	67.18	100.00	100.00	100.00	100.00
L5 Set2	46.62	100.00	100.00	100.00	100.00
L5 Set3	47.15	100.00	100.00	100.00	100.00
Average	**53.65**	**100.00**	**100.00**	**100.00**	**100.00**
L6 Set1	35.60	0.00	0.00	0.00	0.00
L6 Set2	33.59	0.00	0.00	0.00	0.00
L6 Set3	92.24	100.00	100.00	100.00	100.00
Average	**53.81**	**33.33**	**33.33**	**33.33**	**33.33**
All Average	**50.57**	**88.87**	**88.83**	**88.77**	**88.80**



a) Level 4 Set 1 b) Level 4 Set 2 c) Level 4 Set 3



d) Level 6 Set 1 e) Level 6 Set 2 f) Level 6 Set 3

Figure 5-13 : Result of the Experiment 2-1: Applying DBSCAN to the Mouse datasets

5.2.2.2 Experiment 2-2: Applying DBSCAN to the Barcode datasets

All barcode datasets were clustered using the DBSCAN clustering technique with ε = 0.8 and MinPts = 10. The clustering function was provided in RStudio 3.5.2, using the following R commands:

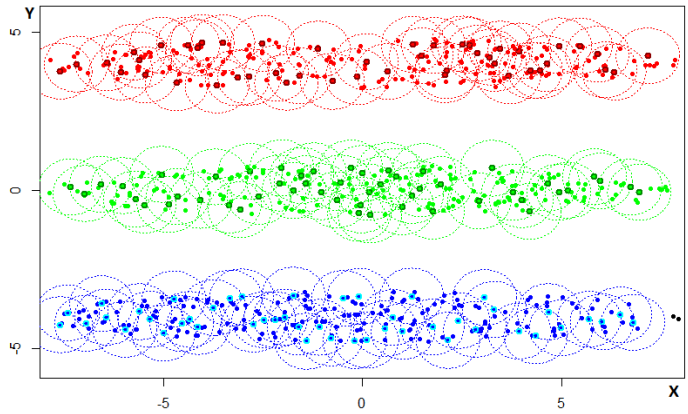
```
res<-fpc::dbscan(data,eps = 0.8, MinPts = 10)  # run the dbscan clustering with eps = 0.8 and MinPts = 10
```

Table 5-8: Result of Experiment 2-2: applying DBSCAN to the Barcode datasets

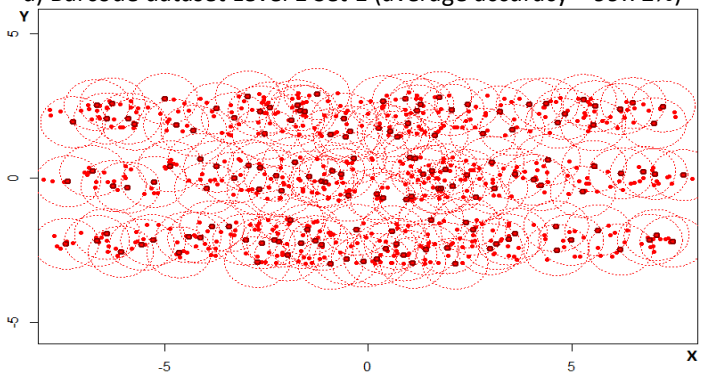
Dataset	Runtime (ms.)	Accuracy (%)			
		Cluster 1	Cluster 2	Cluster 3	Overall
Level 1 Set1	48.5593	100.00	100.00	99.17	99.72
Level 1 Set2	50.5967	100.00	100.00	99.59	99.86
Level 1 Set3	51.8590	100.00	100.00	97.93	99.31
Average	**50.3383**	**100.00**	**100.00**	**98.90**	**99.63**
Level 2 Set1	49.6761	100.00	99.59	99.59	99.72
Level 2 Set2	49.8343	98.34	100.00	99.59	99.31
Level 2 Set3	49.0620	100.00	100.00	100.00	100.00
Average	**49.5241**	**99.45**	**99.86**	**99.73**	**99.68**
Level 3 Set1	50.5319	100.00	94.61	100.00	98.20
Level 3 Set2	52.4777	99.59	100.00	100.00	99.86
Level 3 Set3	53.1869	99.59	99.59	98.76	99.31
Average	**52.0665**	**99.73**	**98.07**	**99.59**	**99.12**
Level 4 Set1	51.4660	99.17	99.59	100.00	99.59
Level 4 Set2	51.7221	99.59	100.00	100.00	99.86
Level 4 Set3	50.6158	100.00	100.00	99.17	99.72
Average	**51.2680**	**99.59**	**99.86**	**99.72**	**99.72**
Level 5 Set1	50.0096	100.00	100.00	97.51	99.17
Level 5 Set2	50.1500	100.00	100.00	100.00	100.00
Level 5 Set3	49.0778	99.59	100.00	100.00	99.86
Average	**49.7458**	**99.86**	**100.00**	**99.17**	**99.68**
Level 6 Set1	50.1289	0.00	0.00	0.00	0.00
Level 6 Set2	51.0061	100.00	15.35	99.59	71.65
Level 6 Set3	49.4346	0.00	0.00	0.00	0.00
Average	**50.1899**	**33.33**	**5.12**	**33.20**	**23.89**
All Average	**50.5219**	**88.66**	**88.82**	**88.38**	**86.95**

Table 5-8 shows that barcode datasets for Levels 1-5 can be properly clustered by DBSCAN with ε = 0.8 and MinPts = 10, but those for Level 6 could not. However, the result of DBSCAN is highly dependent on the value of ε; therefore, the accuracy would be better when ε is properly defined. For the barcode dataset Level 6, if the ε value is defined as smaller, it would give a higher accuracy. For example, from Figure 5-14b, the data points could not be separated into three clusters with ε = 0.8, but if the ε value was a bit smaller, the overlapping of individual core point boundaries would be reduced. Therefore, some data points in each cluster could be separated. However, if ε is too small, the number of clusters might increase and lead to less accuracy. Hence, the ε value should

be optimal in order to provide the best clustering result. The accuracy of DBSCAN was much better than K-means, but worse in terms of time taken.



a) Barcode dataset Level 1 Set 1 (average accuracy = 99.72%)



b) Barcode dataset Level 6 Set 1 (average accuracy = 0.00%)

Figure 5-14 : Result of Experiment 2-2: applying DBSCAN to the Barcode dataset

5.2.2.3 Experiment 2-3: Applying DBSCAN to the Imitative dataset

All imitative datasets clustered using DBSCAN with ε = 0.8 and MinPts = 10. The clustering function was provided in RStudio 3.5.2, using the following R commands:

```
res<-fpc::dbscan(data, eps = 0.8, MinPts = 10)  # run the dbscan clustering with eps = 0.8 and MinPts = 10
```

As shown in Table 5-9, imitative dataset Levels 1-3 were clustered by DBSCAN with an accuracy of around 95%, while the results of imitative dataset Level 4 are not acceptable. DBSCAN was around 3,356 times slower than K-means when applied to the imitative datasets. Although DBSCAN is much slower than K-means when applied to the Imitative datasets, DBSCAN is able to identify clusters more accurate when the overlapping levels were increased, as shown in Figure 5-15.

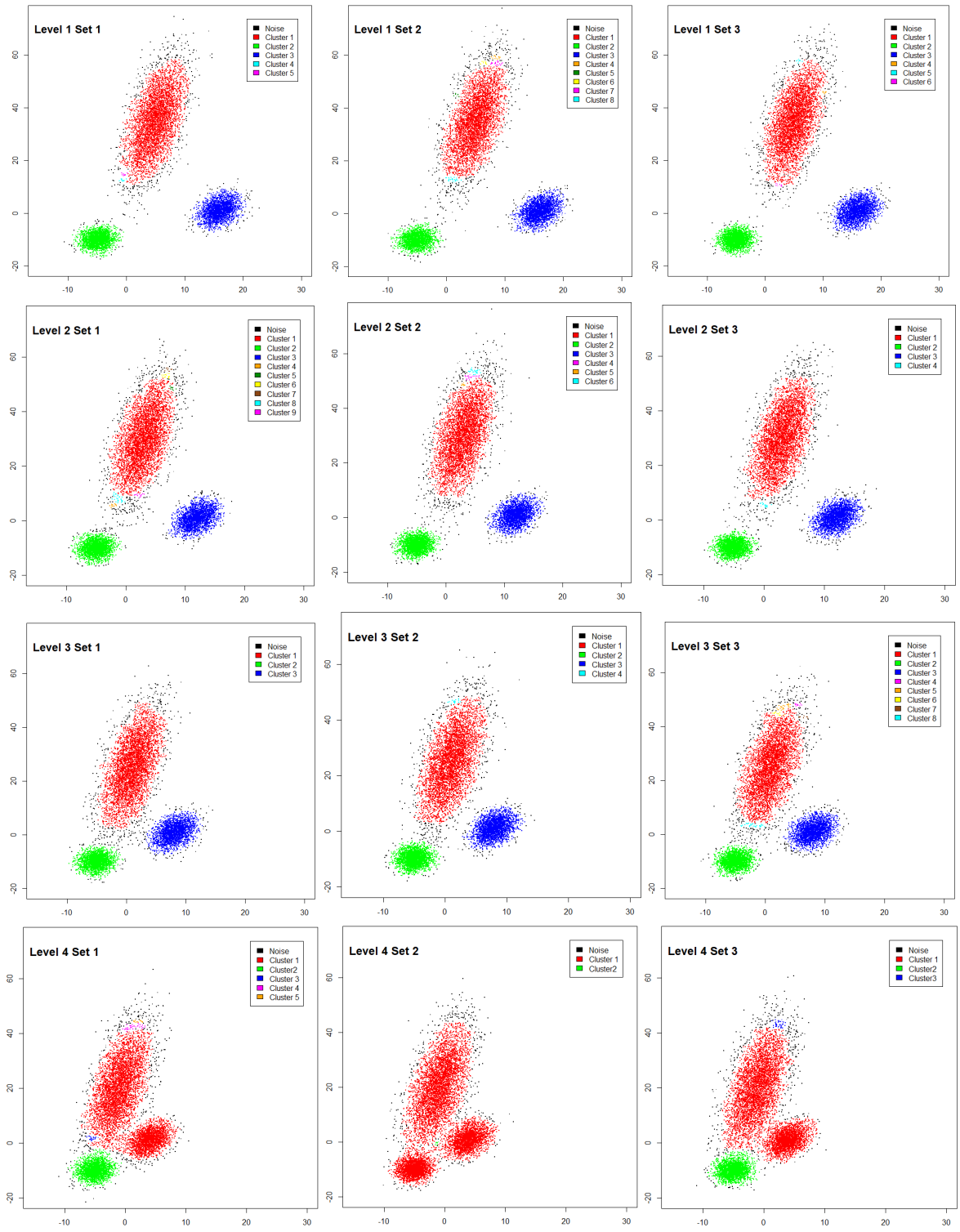


Figure 5-15 : Result of Experiment 2-3: applying DBSCAN to the imitative datasets

Table 5-9: Result of Experiment 2-3: applying DBSCAN to the Imitative datasets

Dataset		Accuracy (%)				Runtime (ms.)
		Cluster 1	Cluster 2	Cluster 3	Overall	
Level 1	Set 1	92.52	98.76	97.28	95.27	7,550.21
	Set 2	90.80	98.44	97.60	94.41	7,380.52
	Set 3	92.16	99.04	97.76	95.28	7,144.38
Average		**91.83**	**98.75**	**97.55**	**94.99**	**7,358.37**
Level 2	Set 1	90.56	98.56	98.04	94.43	7,370.12
	Set 2	91.16	98.76	97.60	94.67	6,512.26
	Set 3	92.14	98.80	96.96	95.01	6,688.77
Average		**91.29**	**98.71**	**97.53**	**94.70**	**6,857.05**
Level 3	Set 1	92.86	98.92	97.12	95.44	6,486.17
	Set 2	91.88	98.88	97.36	95.00	5,849.46
	Set 3	90.16	98.88	96.68	93.97	6,758.89
Average		**91.63**	**98.89**	**97.05**	**94.80**	**6,364.84**
Level 4	Set 1	91.20	98.96	0.04	70.35	6,522.29
	Set 2	94.56	0.00	0.00	47.28	6,058.06
	Set 3	91.52	98.80	0.00	70.46	6,422.07
Average		**92.43**	**65.92**	**0.01**	**62.70**	**6,334.14**
All Average		**91.79**	**90.57**	**73.04**	**86.80**	**6,728.60**

5.2.3 Experiment 3: OPTICS clustering technique

OPTICS is a density-based clustering technique as DBSCAN but OPTICS can identify the optimal epsilon. OPTICS has an ability to discover arbitrary shaped clusters and automatically detect noise. In Experiment 3, OPTICS was applied to 3 datasets so there are 3 sub-experiments, consisting of Experiment 3-1 (applying OPTICS to mouse dataset), Experiment 3-2 (applying OPTICS to barcode dataset) and Experiment 3-3 (applying OPTICS to imitative dataset). The objective of this experiment is to investigate how close the clusters are to each other, when the radius or width of each cluster is different.

5.2.3.1 Experiment 3-1: Applying OPTICS to the Mouse datasets

Experiment 2-1 found three datasets (Level 4 Set 2, Level 6 Set 1, and Level 6 Set 2) which could not be clustered by DBSCAN with ε = 0.5 and MinPts = 10. In this experiment, the OPTICS clustering technique was applied to these datasets in order to identify the optimal ε for each dataset. The OPTICS clustering function `optics()` was provided in RStudio 3.5.2, as can be seen in the following R command.

```
res<-dbscan::optics(data,minPts = 10)
```

OPTICS clustering was applied to the mouse dataset. The optimal values of parameter ε (providing 100% clustering accuracy for each mouse dataset) are shown in Table 5-10 below. A selection of plots of accuracy against parameter ε are given in Figure 5-8.

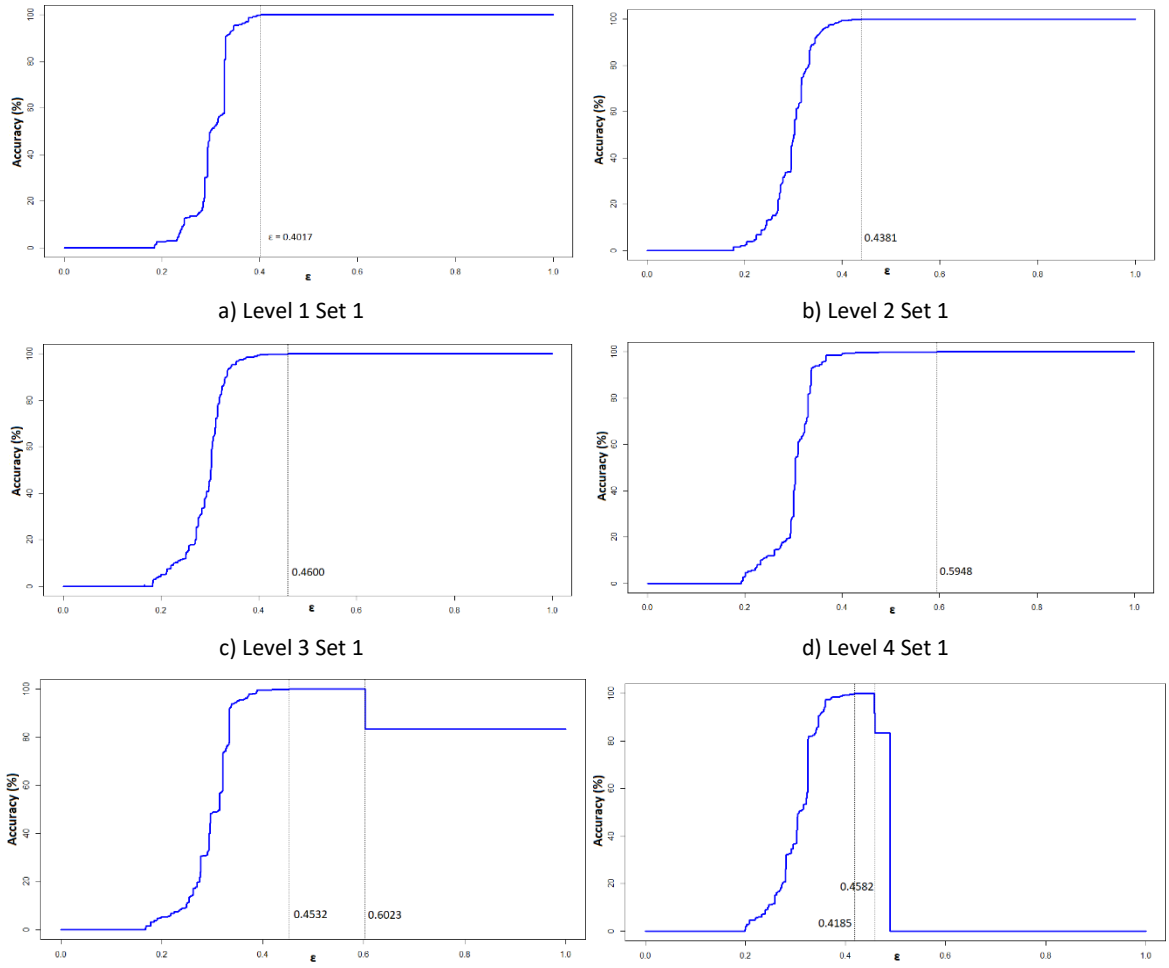


Figure 5-16 : Relationships between accuracy and ε for a selection of mouse datasets

In summary, the optimal ε for each dataset differed, even between datasets with similar patterns. The range of optimal ε values depended on how close the clusters were to each other. The value of the optimal ε is infinity if each cluster is clearly different, as can be seen from Level 1 to Level 4 in Table 5-10 and Figure 5-16, and the example result of mouse dataset Level 1 Set 1 is shown in Figure 5-17. For dataset Levels 5 and 6, the boundary of Cluster 1 is very close to the boundaries of Cluster 2 and Cluster 3, so the range of optimal values is limited, as can be seen in Table 5-10 and Figure 5-

16, and the example result of mouse dataset Level 6 Set 1 in Figure 5-18. The average range for dataset Levels 5 and 6 are 0.1142 and 0.0269 respectively. The range of ε would be smaller if the boundaries of the clusters were closer. In terms of runtime, OPTICS is about 3 times faster than DBSCAN.

Table 5-10: Optimal ε value given by OPTICS for each mouse dataset

Dataset	Optimal epsilon values (ε)			Accuracy (%)	** Runtime (ms)*		
	Min	Max	Range		Min	Max	Average
L1 Set1	0.4017	Infinity	Infinity	100.00	13.01	356.44	16.03
L1 Set2	0.4606	Infinity	Infinity	100.00	13.03	187.02	16.79
L1 Set3	0.4346	Infinity	Infinity	100.00	13.03	218.58	16.66
L2 Set1	0.4381	Infinity	Infinity	100.00	13.04	196.52	16.54
L2 Set2	0.4928	Infinity	Infinity	100.00	13.03	189.02	16.43
L2 Set3	0.5331	Infinity	Infinity	100.00	13.03	197.02	16.62
L3 Set1	0.4600	Infinity	Infinity	100.00	13.02	192.51	16.81
L3 Set2	0.4384	Infinity	Infinity	100.00	13.06	206.07	17.46
L3 Set3	0.4421	Infinity	Infinity	100.00	13.03	207.05	17.40
L4 Set1	0.5948	Infinity	Infinity	100.00	13.53	232.12	17.46
L4 Set2	0.5122	Infinity	Infinity	100.00	13.53	199.56	17.43
L4 Set3	0.4476	Infinity	Infinity	100.00	13.03	199.02	17.15
L5 Set1	0.4532	0.6023	0.1491	100.00	13.03	199.53	17.24
L5 Set2	0.5847	0.6066	0.0219	100.00	13.03	207.05	17.53
L5 Set3	0.4015	0.5730	0.1715	100.00	13.03	193.51	17.33
L6 Set1	0.4185	0.4582	0.0397	100.00	13.03	205.54	17.55
L6 Set2	0.4302	0.4347	0.0045	98.94	13.06	220.08	17.76
L6 Set3	0.4734	0.5099	0.0365	100.00	13.03	247.15	17.45
			Average	**99.94**	**13.09**	**241.10**	**17.09**

** Runtime of OPTICS for each dataset with ε = 0.0001 to 1.0000 incrementally increasing by 0.0001*

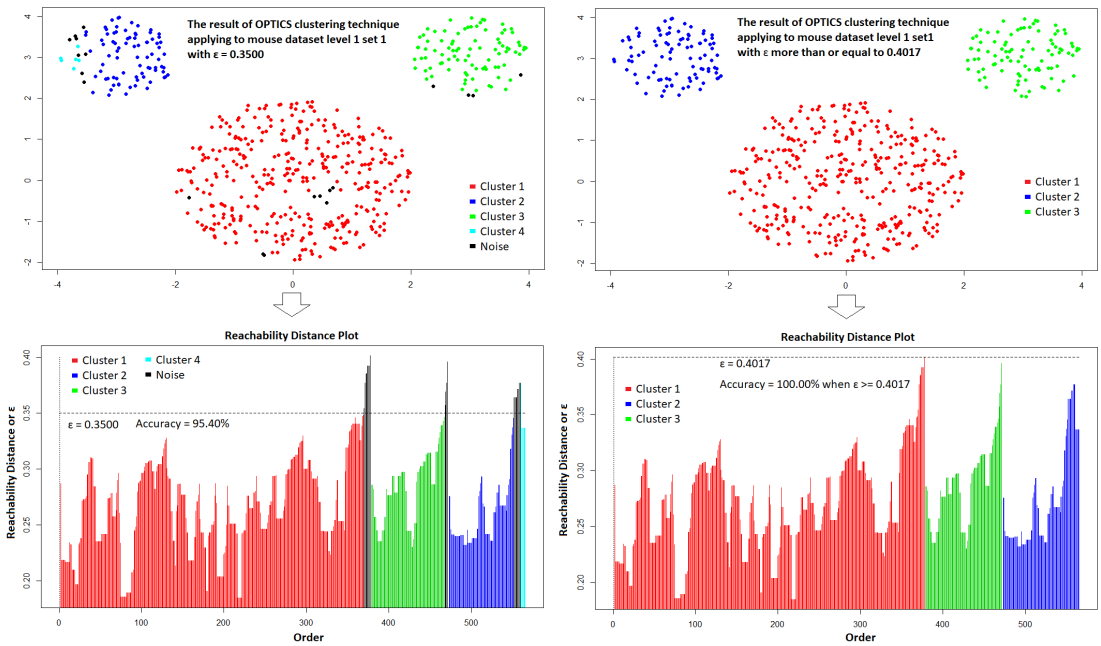


Figure 5-17 : Applying OPTICS to mouse dataset Level 1 Set 1 with ε = 0.3500 and 0.4017

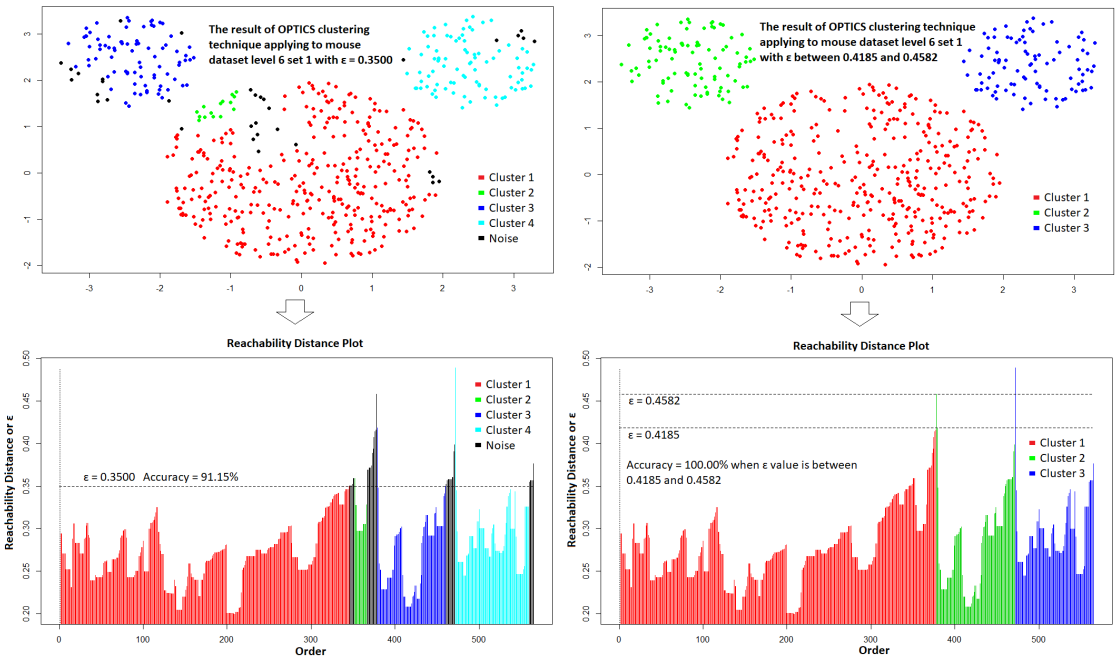


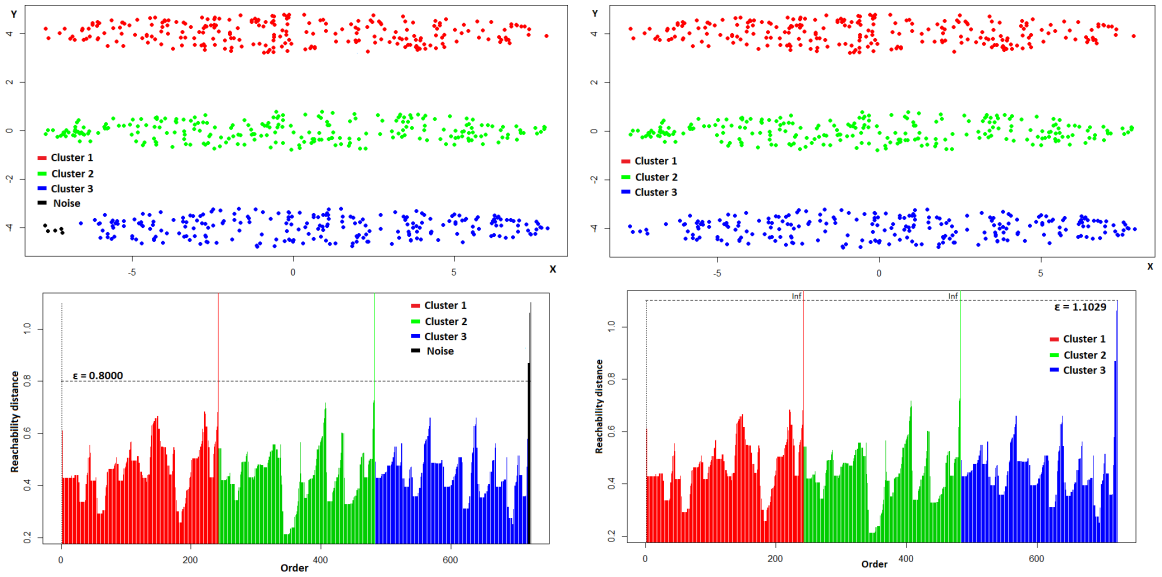
Figure 5-18 : Applying OPTICS to mouse dataset Level 6 Set 1, with ε = 0.3500, and ε between 0.4185 and 0.4582

5.2.3.2 Experiment 3-2: Applying OPTICS to the Barcode datasets

The objective of this experiment was to consider the optimal *ε* for each barcode dataset by using the OPTICS clustering technique. All barcode datasets were clustered by using the function `dbscan::optics` in RStudio 3.5.2 with MinPts = 10, as in Experiment 3-1, as can be seen in the following R commands:

```
res<-dbscan::optics(data,minPts = 10)
```

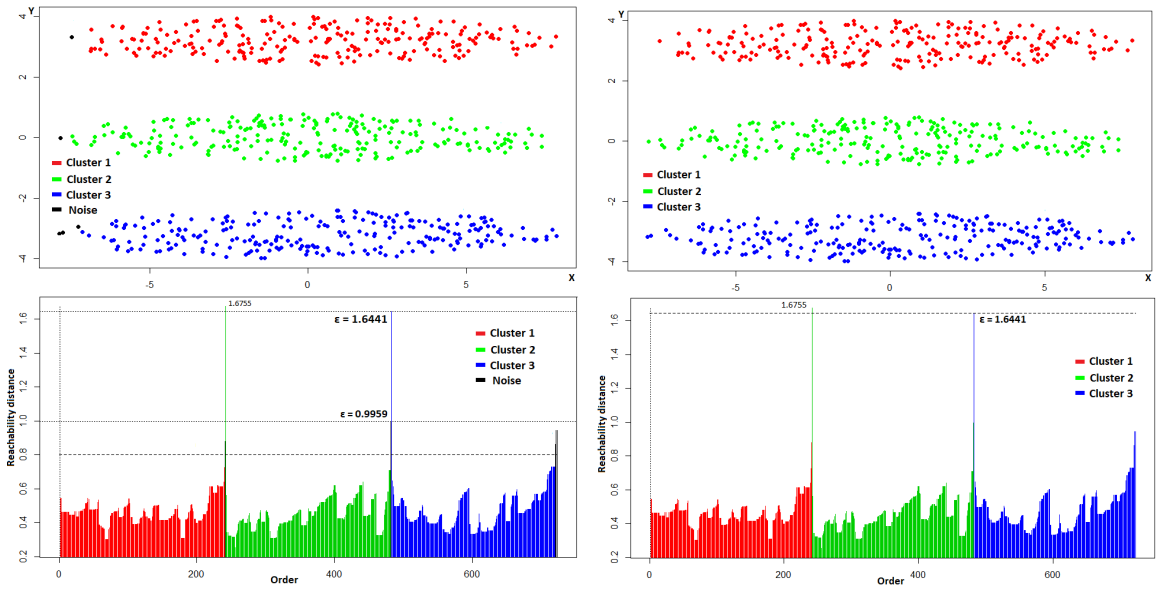
Figure 5-19 to Figure 5-21 show the results of using the OPTICS clustering technique for given values of *ε*. These plot reachability distance against ordering seeds and show the final clusters. The optimal values of parameter *ε* each barcode dataset are given in Table 5-11. Figure 5-22 shows examples of the relationship between *ε* and percentage accuracy for each barcode dataset.



a) epsilon = 0.8000, accuracy = 99.31%

b) epsilon >= 1.1029, accuracy = 100.00%

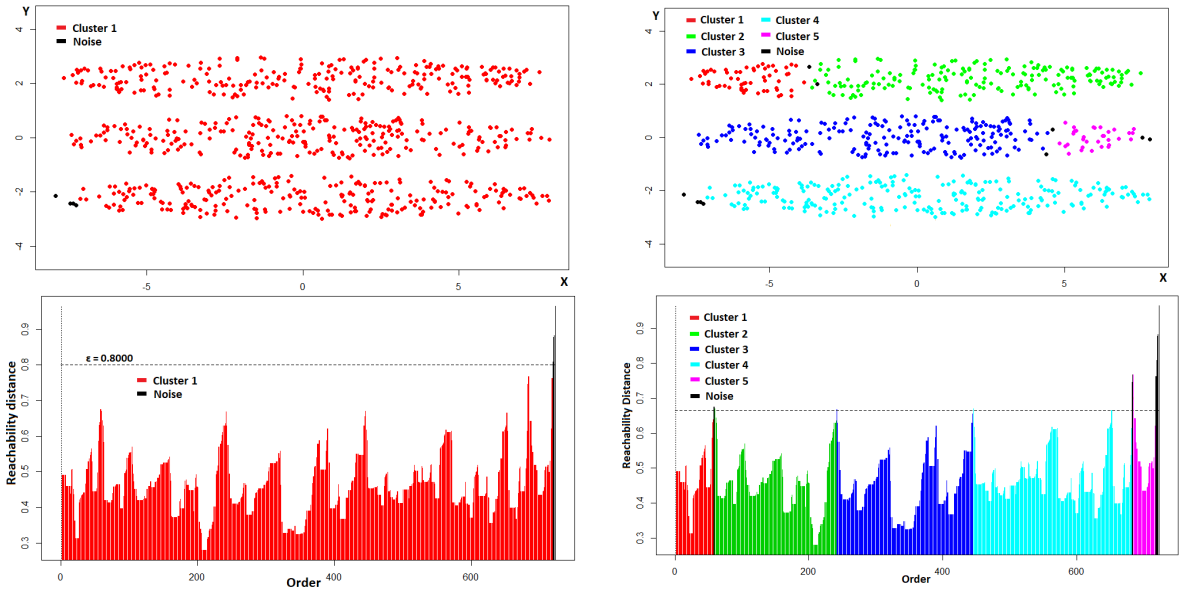
Figure 5-19 : Result of applying OPTICS to barcode dataset Level 1 Set 3



a) epsilon = 0.8000, accuracy = 99.31%

b) 0.9959 <= epsilon <= 1.6441 accuracy = 100.00%

Figure 5-20 : Result of applying OPTICS to barcode dataset Level 3 Set 3



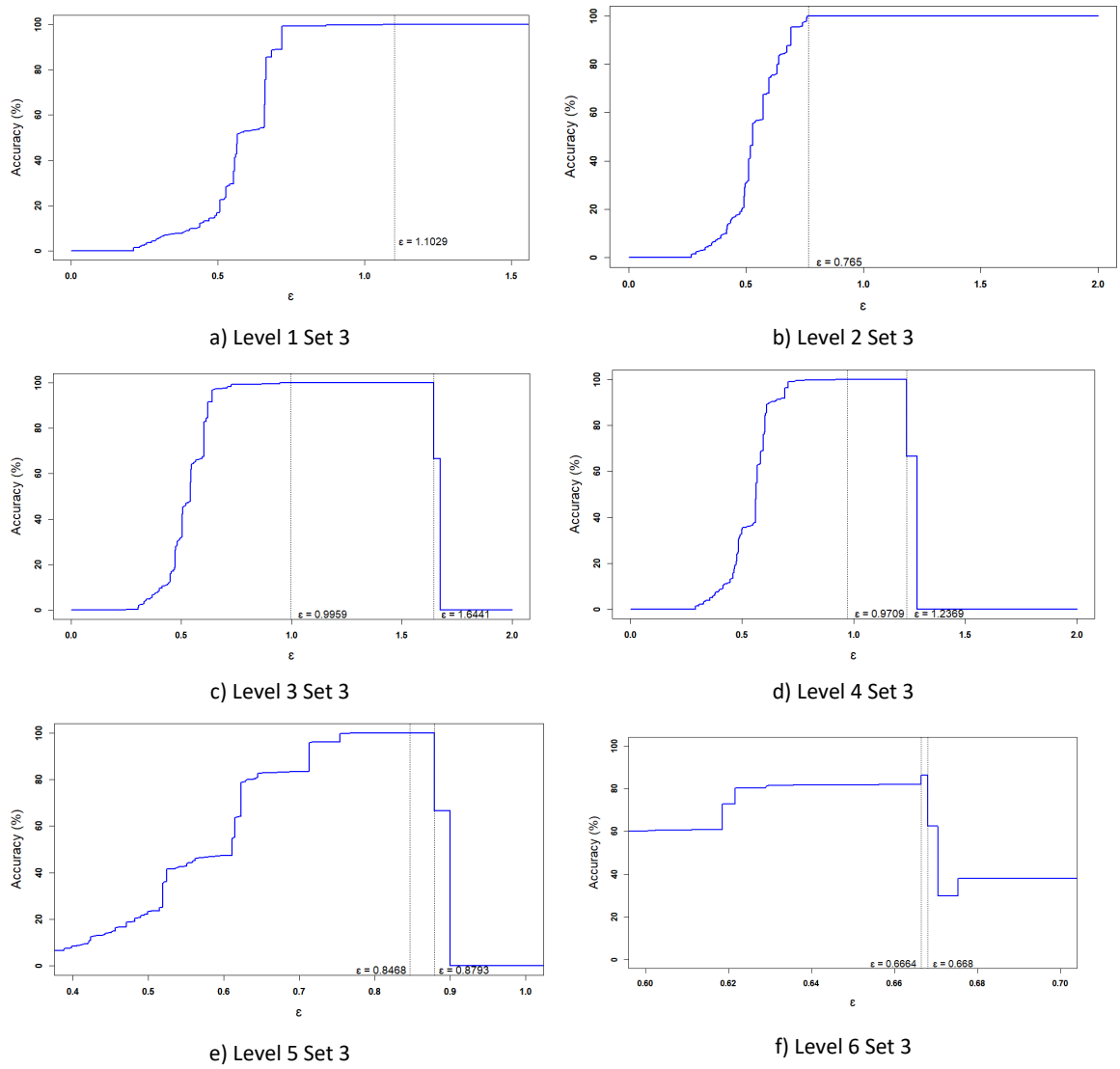
a) epsilon = 0.8000, accuracy = 0.00%

b) 0.6664 <= epsilon <= 0.6680, accuracy = 86.17%

Figure 5-21 : Result of applying OPTICS to barcode dataset Level 6 Set 3

Table 5-11: Result of Experiment 3-2: applying OPTICS to the barcode datasets

Dataset	Optimal epsilon values (ε)			Accuracy (%)	Runtime (ms)
	Min	Max	Range		
Level 1 Set 1	1.4338	Infinity	Infinity	100.00	20.45
Level 1 Set 2	0.8517	Infinity	Infinity	100.00	20.18
Level 1 Set 3	1.1029	Infinity	Infinity	100.00	20.07
			Average	**100.00**	**20.23**
Level 2 Set 1	0.9620	Infinity	Infinity	100.00	19.88
Level 2 Set 2	0.9227	Infinity	Infinity	100.00	19.51
Level 2 Set 3	0.7675	Infinity	Infinity	100.00	20.06
			Average	**100.00**	**19.82**
Level 3 Set 1	0.9104	Infinity	Infinity	100.00	20.13
Level 3 Set 2	0.8139	Infinity	Infinity	100.00	20.08
Level 3 Set 3	0.9959	1.6441	0.6482	100.00	19.93
			Average	**100.00**	**20.04**
Level 4 Set 1	1.1815	1.2903	0.1088	100.00	20.74
Level 4 Set 2	0.8383	1.3111	0.4728	100.00	19.90
Level 4 Set 3	0.9709	1.2369	0.2660	100.00	20.12
			Average	**100.00**	**20.25**
Level 5 Set 1	0.8058	0.8709	0.0651	100.00	20.36
Level 5 Set 2	0.8147	0.9040	0.0893	100.00	20.22
Level 5 Set 3	0.8468	0.8793	0.0325	100.00	20.01
			Average	**100.00**	**20.19**
Level 6 Set 1	0.7072	0.7218	0.0146	91.29	19.99
Level 6 Set 2	0.7721	0.7837	0.0113	94.47	21.21
Level 6 Set 3	0.6664	0.6680	0.0016	86.17	20.95
			Average	**90.64**	**20.72**
All Average				**98.44**	**20.21**



a) Level 1 Set 3

b) Level 2 Set 3

c) Level 3 Set 3

d) Level 4 Set 3

e) Level 5 Set 3

f) Level 6 Set 3

Figure 5-22 : The relationship between accuracy and ε for each Level of dataset Set 3 when applying OPTICS to the Barcode datasets

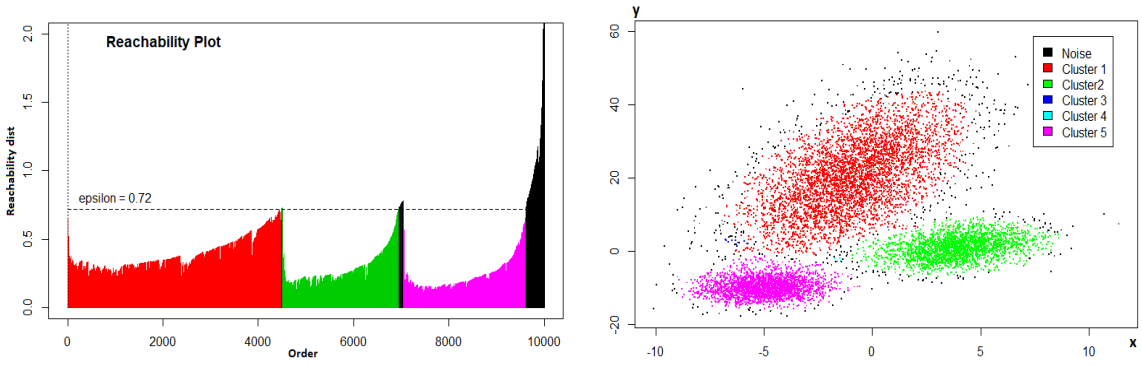
5.2.3.3 Experiment 3-3: Applying OPTICS to the Imitative datasets

The objective of this experiment is to consider the optimal *ε* for each imitative dataset by using the OPTICS clustering technique. All imitative datasets were clustered using function `dbscan::optics` in RStudio 3.5.2, with MinPts = 10, as can be seen in the following R command:

```
res<-dbscan::optics(data,minPts = 10)
```

Figure 5-23 shows the results of applying the OPTICS clustering technique to the imitative dataset with overlapping Level 4 Set 2, which could not be clustered in Experiment 2-3 (applying DBSCAN to the Imitative datasets). The optimal values of parameter *ε* (reachability distance) for imitative Level 4 Set 2 was 0.72, which can provide 93.61% accuracy. Table 5-12 shows the optimal values of parameter *ε*, the percentage of accuracy for each optimal *ε* value, and the average runtimes. Figure

5-24 shows the relationship between ε and the percentage of accuracy for each imitative dataset level for Set 2.



a) Reachability distance plot against order b) Scatter plot

Figure 5-23 : Result of applying OPTICS to imitative dataset Level 4 Set 2

Table 5-12: Result of applying OPTICS to imitative dataset

Dataset		Optimal ε	Accuracy (%)				Runtime (ms)
			Cluster 1	Cluster 2	Cluster 3	Overall	
Level 1	Set 1	2.95 – 3.79	99.90	100.00	100.00	99.95	1,794.20
	Set 2	2.74 – 2.75	99.76	100.00	99.96	99.87	1,941.30
	Set 3	3.85 – 3.95	99.96	100.00	100.00	99.98	2,047.39
Average			**99.87**	**100.00**	**99.99**	**99.93**	**1,927.63**
Level 2	Set 1	2.42 – 2.45	99.36	100.00	100.00	99.68	1,635.91
	Set 2	2.51 – 2.68	99.50	100.00	99.96	99.74	2,990.24
	Set 3	2.23 – 2.24	99.72	99.96	99.92	99.83	1,376.76
Average			**99.53**	**99.99**	**99.96**	**99.75**	**2,000.97**
Level 3	Set 1	1.29	97.02	99.88	99.24	98.29	1,741.37
	Set 2	1.27 – 1.28	97.72	99.84	99.48	98.69	1,932.01
	Set 3	1.37	97.76	99.92	99.88	98.83	1,625.87
Average			**97.50**	**99.88**	**99.53**	**98.60**	**1,766.42**
Level 4	Set 1	0.46	85.90	98.08	95.28	91.29	2,497.95
	Set 2	0.72	89.74	98.60	96.36	93.61	1,884.06
	Set 3	0.66	85.38	97.96	96.48	91.30	2,066.40
Average			**87.01**	**98.21**	**96.04**	**92.07**	**2,149.47**
Overall Average			**95.98**	**99.52**	**98.88**	**97.59**	**1,961.12**

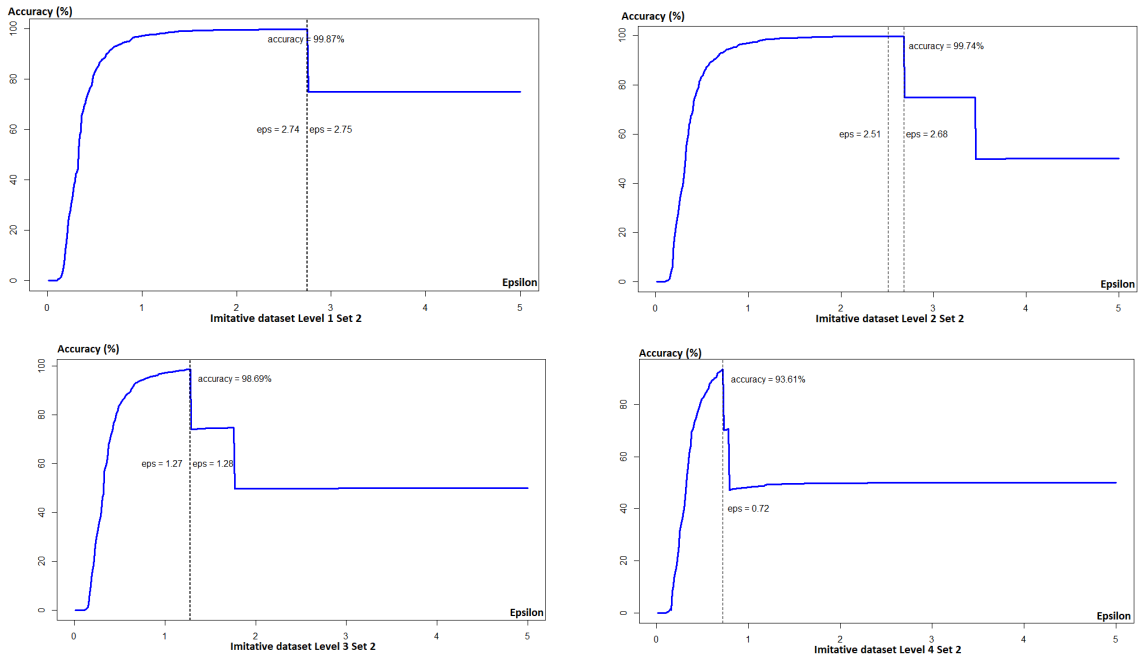


Figure 5-24 : Relation between ε value and accuracy for all Levels of imitative dataset Set 2 when applying OPTICS to the Imitative datasets

The values and ranges of optimal ε decreased as the closeness increased. An average of the accuracy values was around 97.59%, which is higher than K-Means and DBSCAN. DBSCAN may provide as high an accuracy as OPTICS if the optimal value of ε is given, but the optimal ε for the DBSCAN technique is difficult to determine. OPTICS was around 3.5 times faster than DBSCAN when applied to the imitative datasets, which have 10,000 data points in each set.

5.2.4 Experiment 4: FlowGrid clustering technique

FlowGrid is a new clustering technique, and is applied to FCM automated gating. It is a combination of DBSCAN and the FLOCK algorithm. FlowGrid was implemented by Ye and Ho (2019). It has 4 user-defined parameters, MinDenB, MinDenC, bin_size, and ε. In Experiment 4, FlowGrid was applied to 3 datasets so there are 3 sub-experiments, consisting of Experiment 4-1 (applying FlowGrid to mouse dataset), Experiment 4-2 (applying FlowGrid to barcode dataset) and Experiment 4-3 (applying FlowGrid to imitative dataset). The objective of this experiment is to investigate how close the clusters are to each other, when the radius or width of each cluster is different.

5.2.4.1 Experiment 4-1: Applying FlowGrid to the Mouse datasets

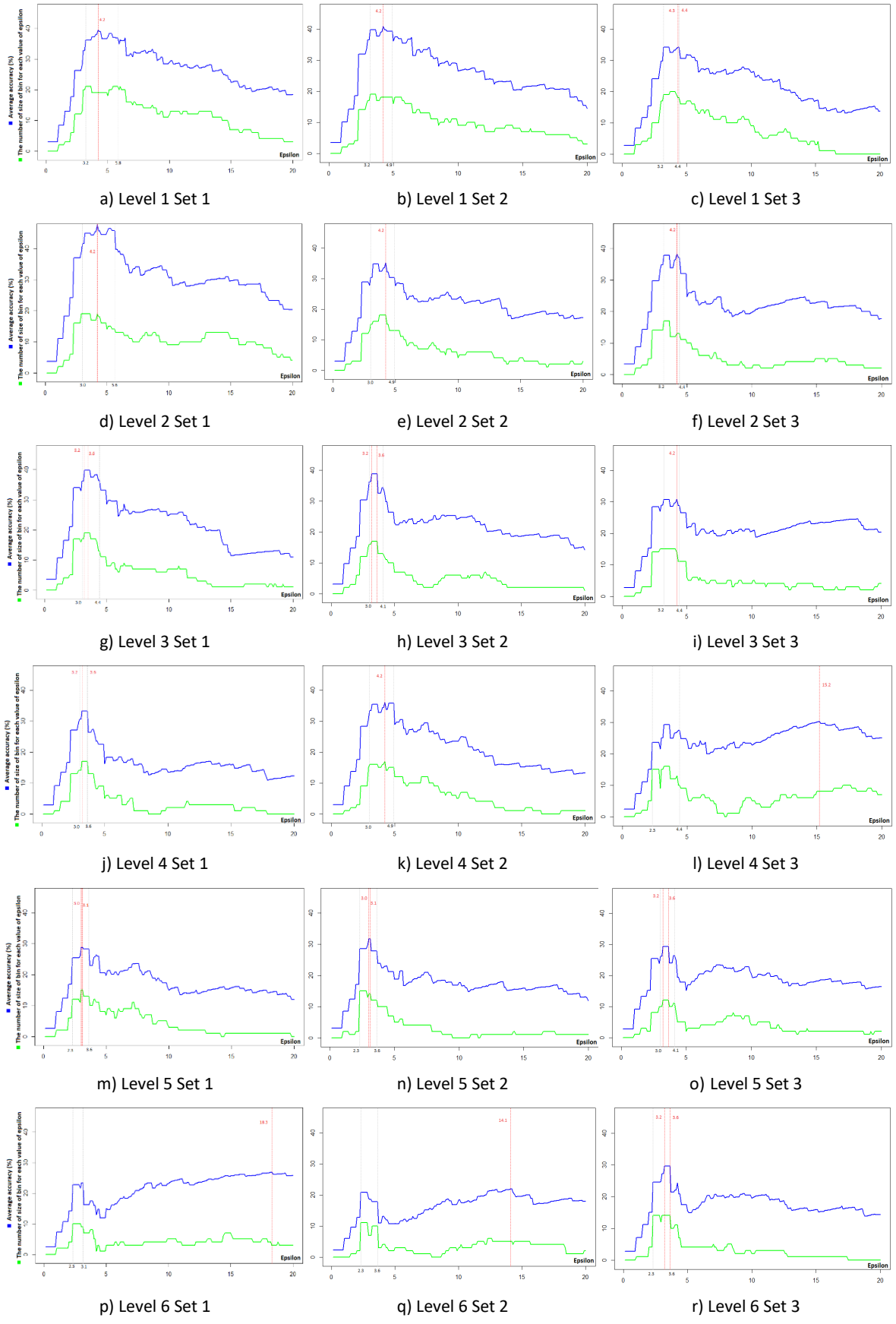
The objective of this experiment is to investigate the performance of this technique and the influence of two parameters (the size of bin and ε) by applying them to each mouse dataset used

in the previous experiments. For this experiment, MinDenB was defined as 3 and MinDenC was defined as 40, which are the default values in the program. The other parameters, bin_size and ε, are more varied; the value of ε ran from 0.1 to 20.0 in increments of 0.1, while the value of bin_size ran from 1 to 100 in increments of 1. Therefore, the datasets were clustered by FlowGrid 20,000 times each. Most of the mouse datasets were clustered by FlowGrid 100% correctly, except Level 5 Set 1 and Set 2, and Level 6 Set 1 and Set 3, whose highest accuracy values were 99.29%, 99.29%, 97.70% and 99.82% respectively. The average accuracy result for each value of ε with different sizes of bin, and the number of sizes of bin that provided accuracy of at least 80% for each value of ε, are shown in Figure 5-25.

These mouse datasets were randomised with the same density, but the distances between the centres of each cluster were different. Table 5-13 shows that the ranges of acceptable values of ε were not significantly different, indicating the density averages of each cluster are equal. However, when the distances between the centres of each cluster were closer, the lower and upper bounds of the range of acceptable ε slightly decreased, as can be seen in Figure 5-26. For Level 4 Set 3, Level 6 Set 1 and Level 6 Set 2, the ε value giving the highest accuracy did not fall within the acceptable range, which may be because the density of these three datasets was not smooth.

Table 5-13: Range of acceptable ε, and ε values giving the highest accuracy

Mouse datasets	Range of acceptable ε	ε value giving highest average accuracy
Level 1 Set 1	3.2-5.8	4.2
Level 1 Set 2	3.2-4.9	4.2
Level 1 Set 3	3.2-4.4	4.3-4.4
Level 2 Set 1	3.0-5.6	4.2
Level 2 Set 2	3.0-4.9	4.2
Level 2 Set 3	3.2-4.4	4.2
Level 3 Set 1	3.0-4.4	3.2-3.6
Level 3 Set 2	3.0-4.1	3.2-3.6
Level 3 Set 3	3.2-4.4	4.2
Level 4 Set 1	3.0-3.6	3.2-3.6
Level 4 Set 2	3.0-4.9	4.2
Level 4 Set 3	2.3-4.4	**15.2 out of range**
Level 5 Set 1	2.3-3.6	3.0-3.1
Level 5 Set 2	2.3-3.6	3.0-3.1
Level 5 Set 3	3.0-4.1	3.2-3.6
Level 6 Set 1	2.3-3.1	**18.3 out of range**
Level 6 Set 2	2.3-3.6	**14.1 out of range**
Level 6 Set 3	2.3-3.6	3.2-3.6



a) Level 1 Set 1 b) Level 1 Set 2 c) Level 1 Set 3

d) Level 2 Set 1 e) Level 2 Set 2 f) Level 2 Set 3

g) Level 3 Set 1 h) Level 3 Set 2 i) Level 3 Set 3

j) Level 4 Set 1 k) Level 4 Set 2 l) Level 4 Set 3

m) Level 5 Set 1 n) Level 5 Set 2 o) Level 5 Set 3

p) Level 6 Set 1 q) Level 6 Set 2 r) Level 6 Set 3

Figure 5-25 : Influence of Epsilon (ε) in the accuracy of FlowGrid clustering on the mouse datasets

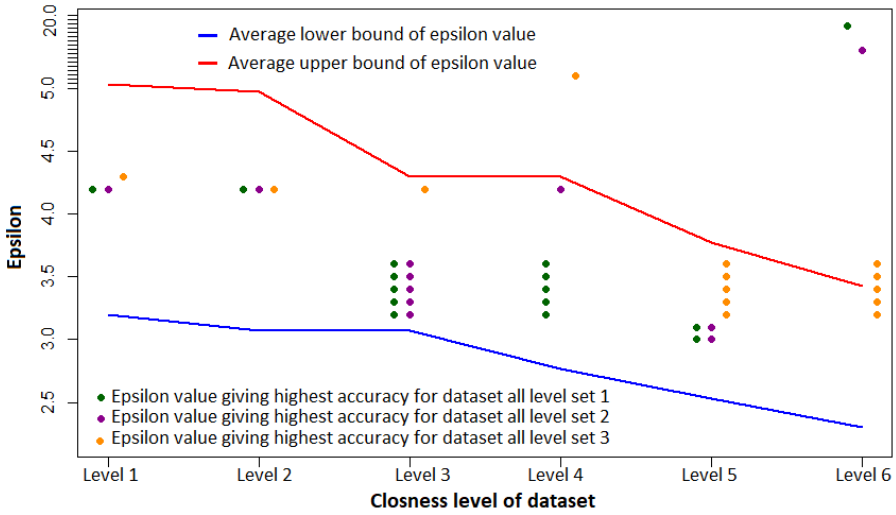


Figure 5-26 : Relationship between Epsilon value and closeness of each cluster (lines), and the Epsilon values giving the highest accuracy result for each dataset (dots)

5.2.4.2 Experiment 4-2: Applying FlowGrid to the Barcode datasets

The four user-defined parameters for FlowGrid are MinDenB, MinDenC, bin_size, and ε. For this experiment, MinDenB and MinDenC were defined as 3 and 40 respectively, (which is the default value of the program), while bin_size and ε were varied. The value of ε was run from 0.1 to 20.0 in increments of 0.1, and bin_size was run from 1 to 100 in increments of 1. Therefore, the datasets were clustered by FlowGrid 20 000 times each. Table 5-14 shows the performance of the FlowGrid technique applied to the barcode datasets. The barcode datasets Levels 1, 2 and 3 were clustered 100% correctly by FlowGrid, while the average accuracy for Levels 4, 5, and 6 were 99.91%, 98.94%, and 98.99% respectively. The percentage of accuracy dropped slightly when the boundaries were closer.

Figure 5-27 shows the influence of epsilon value on the clustering result. The blue line shows the relationship between the epsilon value and average accuracy; the green line shows the plotting of the number of sizes of bin which provide an accuracy of 80% or more. These plots show that the accuracy increases when epsilon increases, until the accuracy reaches a peak at an optimal epsilon, after which point accuracy gradually drops. FlowGrid's running time was slightly faster than OPTICS.

Table 5-14: Range of acceptable ε, and ε values giving highest accuracy for Barcode datasets

Barcode datasets	Range of acceptable ε	ε value giving highest average accuracy	Highest Accuracy (%)	Average Runtime (ms)
Level 1 Set 1	3.7-6.9	4.5-4.9	100.00	15.08
Level 1 Set 2	4.2-6.9	5.9	100.00	15.36
Level 1 Set 3	4.0-6.9	5.9	100.00	14.97
Average			**100.00**	**15.14**
Level 2 Set 1	3.7-5.9	4.5-4.9	100.00	15.36
Level 2 Set 2	3.7-5.3	4.5-4.9	100.00	15.24
Level 2 Set 3	3.2-5.3	4.5-4.9	100.00	14.66
Average			**100.00**	**15.09**
Level 3 Set 1	3.2-4.9	3.7-3.9	100.00	14.50
Level 3 Set 2	3.2-4.9	3.7-3.9	100.00	14.56
Level 3 Set 3	3.2-4.9	3.7-3.9	100.00	14.83
Average			**100.00**	**14.63**
Level 4 Set 1	3.2-4.9	3.7-3.9	99.72	15.23
Level 4 Set 2	3.2-4.1	3.7-3.9	100.00	15.30
Level 4 Set 3	2.9-4.1	3.7-3.9	100.00	14.90
Average			**99.91**	**15.15**
Level 5 Set 1	2.3-3.9	2.9	98.48	14.56
Level 5 Set 2	2.3-4.1	3.7-3.9	99.86	14.46
Level 5 Set 3	2.3-3.9	2.9	98.48	14.61
Average			**98.94**	**14.5458**
Level 6 Set 1	2.3-3.9	2.9	94.74	14.76
Level 6 Set 2	2.3-4.1	2.9	97.51	14.71
Level 6 Set 3	2.3-4.1	2.9	92.95	14.88
Average			**95.07**	**14.79**
All Average			**98.99**	**14.89**

5.2.4.3 Experiment 3-4: Applying FlowGrid to the Imitative datasets

Four user-defined parameters are required for FlowGrid, MinDenB, MinDenC, bin_size, and ε. MinDenB and MinDenC were defined as 3 and 40 respectively (which is the default value of the program), while bin_size and ε were varied; the value of ε was run from 0.1 to 16.0, in increments of 0.1, while bin_size was run from 1 to 100, in increments of 1. Therefore, the dataset were clustered by FlowGrid 16,000 times for each dataset. Table 5-15 shows the results of FlowGrid clustering applied to these imitative datasets with the optimal ε and size of bin. The average accuracy for imitative dataset closeness Levels 1-4 was 99.94%, 99.87%, 99.36%, and 97.40% respectively. The average accuracy decreased slightly as the clusters got closer. FlowGrid was around 10 and 3 times faster than DBSCAN and OPTICS, respectively.

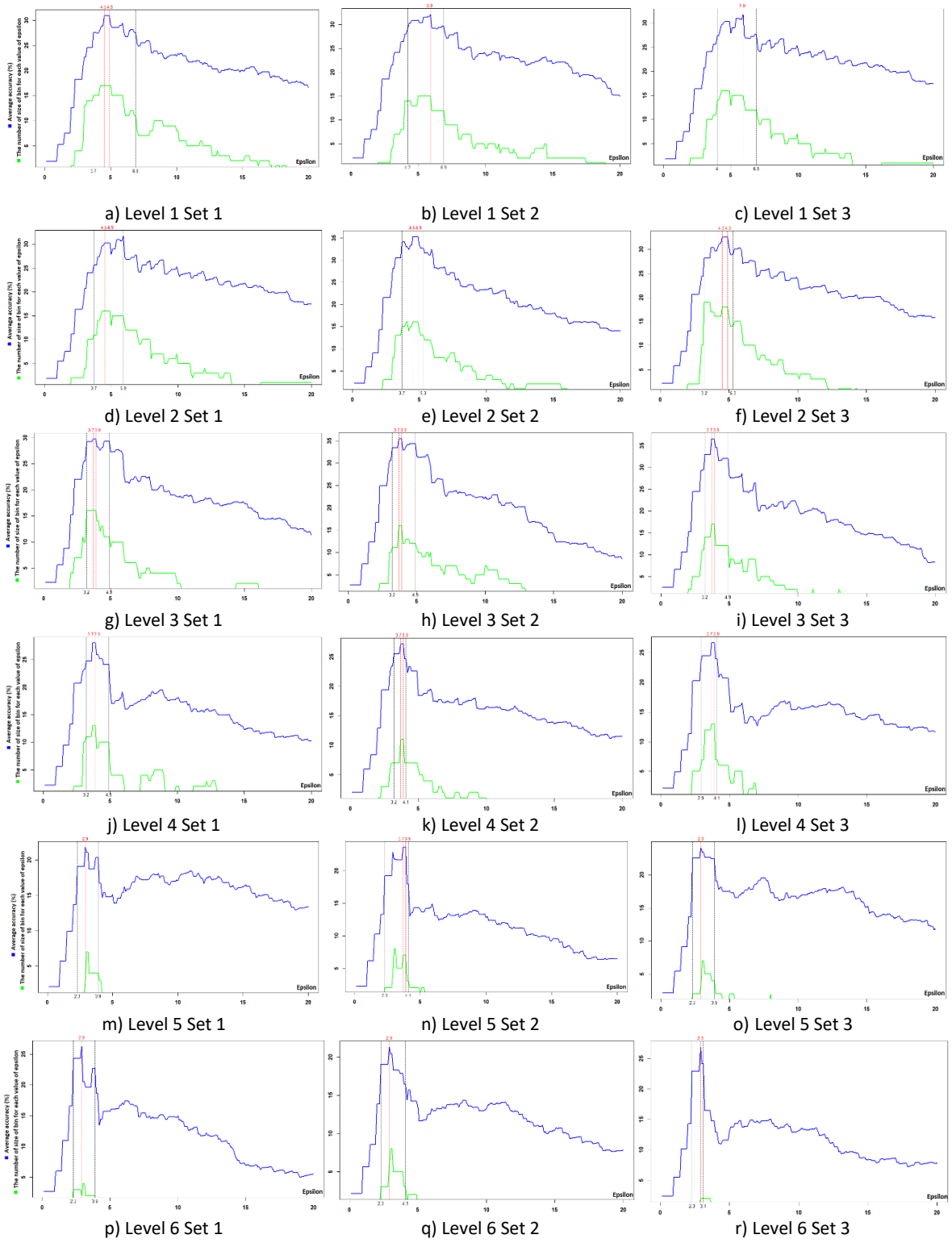


Figure 5-27 : Influence of Epsilon (ε) on the accuracy of FlowGrid clustering on the barcode datasets

Figure 5-28 shows the influence of the epsilon value on the accuracy of FlowGrid on every imitative dataset Level for Set 2. Although the range of ε values providing the highest average accuracy for each dataset was 4.0-4.5, the optimal value of ε was not in this range, as can be seen in Figure 5-28. By the nature of this dataset, it is difficult to identify an optimal ε value for the FlowGrid

technique. Therefore, it would be better if the FlowGrid technique could automatically define the optimal value of epsilon.

Table 5-15: The pair of ε and bin_size giving the highest accuracy when applying FlowGrid to the imitative datasets

Imitative datasets	Pairs of Epsilon and bin_size (ε, bin_size)	Accuracy (%)	Average Runtime (ms)
Level 1 Set 1	(14.0-14.1, 58), (15.0-15.5, 62), (15.9, 67), (16.0, 65), (16.0, 67)	99.93	678.89
Level 1 Set 2	(14.8-15.2, 53), (15.7-16.0, 76)	99.91	663.51
Level 1 Set 3	(14.8-15.0, 71), (15.0, 84)	99.97	663.57
	Average	**99.94**	**668.66**
Level 2 Set 1	(15.3-15.8, 84)	99.91	673.14
Level 2 Set 2	(15.7-16.0, 96)	99.83	659.01
Level 2 Set 3	(14.0, 79), (14.8, 80)	99.86	665.64
	Average	**99.87**	**665.93**
Level 3 Set 1	(11.8-12.0, 74)	99.44	667.70
Level 3 Set 2	(12.4-12.5, 70), (12.7-12.8, 70)	99.28	678.66
Level 3 Set 3	(15.2, 100)	99.37	680.78
	Average	**99.36**	**675.71**
Level 4 Set 1	(8.0, 85)	96.79	688.38
Level 4 Set 2	(9.1-9.2, 53)	97.86	696.63
Level 4 Set 3	(8.6, 65)	97.56	695.51
	Average	**97.40**	**693.51**
Overall Average		**99.14**	**675.95**

5.3 Discussion

According to the experiments above, four clustering techniques, includes K-means, DBSCAN, OPTICS, and FlowGrid, were applied to three different generated datasets, and includes the Mouse datasets, the Barcode datasets, and the Imitative datasets. Although K-means is the fastest and has the least number of required parameters, it is not able to identify non-spherical shapes of clusters (the Barcode and Imitative datasets) because K-means uses distances between objects and centroids to measure similarity. DBSCAN, OPTICS and FlowGrid has an ability to discover non-spherical shapes of clusters because these three techniques are density-based. The accuracy result of DBSCAN and OPTICS with optimal epsilon were not much different when applied to the Mouse, Barcode, and Imitative datasets. However, users need to manually define epsilon when they use DBSCAN that might not produce a good results, as can be seen in Table 5-16. Epsilon is a key user-defined parameter which has a great effect on clustering result, as can be seen in the experiments

above. FlowGrid is a clustering technique that developed for classifying clusters in flow cytometry data. FlowGrid is faster than DBSCAN and OPTICS, but the parameter epsilon still needs to be defined by users. Table 5-16 shows the comparisons of K-means, DBSCAN, OPTICS and FlowGrid applied to the Mouse, Barcode, and Imitative datasets.

5.4 Conclusion

The experiments applied four clustering techniques – K-means, DBSCAN, OPTICS, and FlowGrid – to three types of dataset – mouse datasets, barcode datasets, and imitative datasets – with different overlapping levels. This enabled an investigation of the influence of the epsilon parameter (for DBSCAN, OPTICS, and FlowGrid), the percentage of accuracy, and running time. The comparison of their accuracy and runtime is shown in Table 5-16 and Table 5-17

According to Table 5-16 and 5-17 K-means could not properly identify non-spherical shapes of clusters, because the K-means clustering technique is heavily based on distance. However, the technique is much faster than the other three techniques, which are based on density. For the mouse datasets, DBSCAN, OPTICS, and FlowGrid provided high average accuracy results of 99.99%, 99.93%, and 99.79% respectively. For barcode datasets, FlowGrid provided the best result, 98.99%, while the average accuracy results for OPTICS and DBSCAN were 98.44% and 98.21% respectively. For imitative datasets, FlowGrid provided the best result, 99.14%, while the average accuracy results for DBSCAN and OPTICS were 97.17% and 97.59% respectively. On this basis, FlowGrid can be considered most suitable for clustering FCM data. K-means runtime was faster than FlowGrid, OPTICS, and DBSCAN, but its lower accuracy when applied to non-spherical clusters was not considered acceptable. FlowGrid was around 10 and 3 times faster than DBSCAN and OPTICS, respectively. According to these experiments, FlowGrid provided the best results for accuracy and runtime, so it was chosen as a basis to further improve its performance.

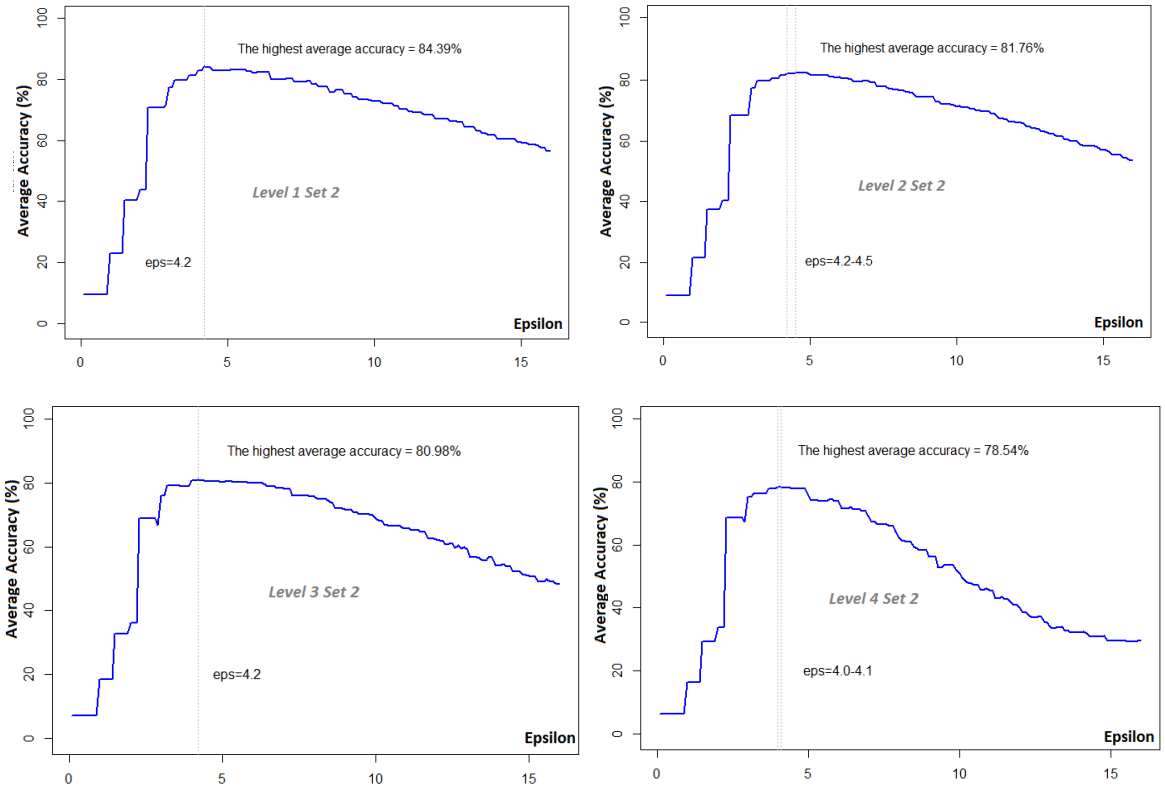


Figure 5-28 : Relationship between parameter ε and average accuracy when applying FlowGrid clustering to the imitative datasets Levels 1-4 Set 2

Table 5-16: Comparison of K-means, DBSCAN, OPTICS and FlowGrid clustering techniques

Technique	Average accuracy (%)			Average running time (ms.)			The highest overlapping level that provided acceptable clustering result			User-defined parameters	Identification of non-sherical shape of clusters
	Mouse	Barcode	Imitative	Mouse	Barcode	Imitative	Mouse	Barcode	Imitative		
K-means	95.96	51.42	70.04	2.51	2.9	2.01	6	-	-	- k	No
DBSCAN	88.80	86.95	86.80	50.57	50.52	6,728.60	5	5	3	- epsilon - MinPts	Yes
OPTICS	99.94	98.44	97.59	17.09	20.21	1,961.12	6	6	4	- epsilon (optimal) - MinPts	Yes
FlowGrid	100.00	98.99	99.14	12.64	14.89	6,75.95	6	6	4	- epsilon - bin size - density of bin - density of collective bin - %q (optional)	Yes

The experiments show that FlowGrid is better than the others, but it needs too many user-defined parameters that affect the clustering result. FlowGrid is combination of grid-based and DBSCAN clustering. DBSCAN is density-based technique and requires many parameters to be identified by the user, so FlowGrid was preferred while reducing the number of parameters. OPTICS was developed from DBSCAN and has the ability to detect the optimal value of epsilon (ε) automatically; moreover, it is faster than DBSCAN. Therefore, a novel clustering technique for FCM data could be a combination of grid-based and OPTICS, which is called "FLOPTICS", and will be explained in the next chapter.

Table 5-17: The four techniques compared when applied to the three types of dataset

Techniques	**Average Accuracy (%)**			**Runtime (ms)**		
	Dataset			**Dataset**		
	Mouse	**Barcode**	**Imitative**	**Mouse**	**Barcode**	**Imitative**
K-means	95.96	51.42	70.04	2.51	2.90	2.01
DBSCAN	*99.99	*98.21	*97.17	37.51	50.52	6,728.60
OPTICS	99.93	98.44	97.59	17.09	20.20	1,961.12
FlowGrid	99.79	98.99	99.14	12.28	14.89	675.95

* The results are different from Experiments 1-2 and 2-2 because this result was obtained from clustering the dataset with the optimal ε

Chapter 6 "FLOPTICS" – a novel automated gating technique

According to the experiments in Chapter 5, FlowGrid provides better result than other techniques so it was selected as the algorithm to improve. Although FlowGrid provides acceptable result, it still has a problem with too many user-defined parameters are required. FLOPTICS is a new automated gating technique that is improved from FlowGrid. FLOPTICS has an ability to automatically detect optimal epsilon, which is one of key user-defined parameter. The technique, experiment and results are explained in the following sections.

6.1 FLOPTICS technique

FLOPTICS is a novel automated gating technique that is a combination of grid-based and density-based clustering techniques, which is used for classification of homogeneous cells in flow cytometry (FCM). FLOPTICS is an improved algorithm based on the latest automated gating technique, called FlowGrid. FlowGrid was proposed by Ye and Ho (2018), they claimed that FlowGrid provides high accuracy and better time efficiency compared with some state of art techniques such as flowClust, flowMeans, flowMerge, flowPeaks, FlowSOM, and FLOCK.

FLOCK and FlowGrid reduced computational time by reducing the amount of FCM data, which in most cases is very dense. Although, the amount of data is reduced, the clustering result is not much different comparing to original data because FCM data density is always high. FLOCK and FlowGrid use a technique that partitions data into equal-sized of bin to reduce the number of data. Equal-sized of grids (bins), are created on a data space in each dimensions and every non-empty bins are treated as data points, it means that all data points in the same bin are grouped together and treated as a single point. However, this technique will produce satisfactory results only when data density is high enough. K-means is used in the last step of the FLOCK algorithm which causes FLOCK to have problems with noises-sensitivity. FlowGrid has handled the disadvantage of FLOCK algorithm by using DBSCAN instead of K-means. Although DBSCAN can automatically detecting noises or outliers, it still has the problem that too many user-defined parameters are needed. Therefore, FLOPTICS was developed to deal with too many parameters having to be defined by users.

The OPTICS clustering technique is used in FLOPTICS, as it not only reduces the number of user-defined parameters but also reducing the computational time, because time processing of OPTICS algorithms is slightly faster than DBSCAN. The main concepts of FLOPTICS can be divided into two parts; data partitioning and applying the OPTICS clustering technique.

6.1.1 Data partitioning

FCM data is extremely large amount of data with high density cause time consuming in clustering process. In order to reduce the processing time, the data size have to be reduced but this reduction must not significantly affect the clustering result. According to FCM data is typically extremely dense, many objects in dataset are so similar and almost at the same point. Therefore, one of these similar objects can be chosen as a representative to entering the clustering process without much impact on the clustering result. The data reduction process is the initial steps of FLOPTICIS that are explained below.

The initial steps are to normalize the data, followed by the creation of equal-sized grids (Bins) on the data space for each dimension. The number of bins is the main control for the user, defining the precision of the analysis: the more bins, the greater the precision in the result, but the slower the analysis. The primary function of this process is to reduce the amount of data being processed, normally significant and of high density in FCM. Clustering following data reduction does not significantly affect the outcome compared to using the entire data set, if the number of bins is not too few. The principle of this process is outlined in Figure 6. Each dimension is partitioned into an equal sized bin, so the total number of bins for d-dimensional data is $(N_{bin})^d$, where N_{bin} is the number of bins for each dimension. Every Bin_i is labelled with a row of d positive numbers. For example, if Bin_i is labelled as $\mathrm{C_l}(3, 2, 5)$ it means that Bin_i is located in bin 3, bin 2, and bin 5 for dimension 1, dimension 2, and dimension 3, respectively. Therefore, the distance between Bin_i and Bin_j is defined as:

$$D(Bin_i, Bin_j) = \sqrt{\sum_{h=1}^{d} (C_{ih} - C_{jh})^2} \tag{6.1}$$

From Figure 6-1, the distance between Bin_i to Bin_j and Bin_k are demonstrated as follows:

$$D(Bin_i, Bin_j) = \sqrt{(1-5)^2 + (7-6)^2} = 4.12$$
$$D(Bin_i, Bin_k) = \sqrt{(1-2)^2 + (7-1)^2} = 6.08$$

The objective of the data partitioning process is to reduce a large number of FCM data sample and runtime, without which the clustering process normally takes a long time. Although data partitioning might cause a data loss, FCM data normally has a very high density, therefore, the clustering result is not significant different compared between normal FCM data and FCM data after partitioning.

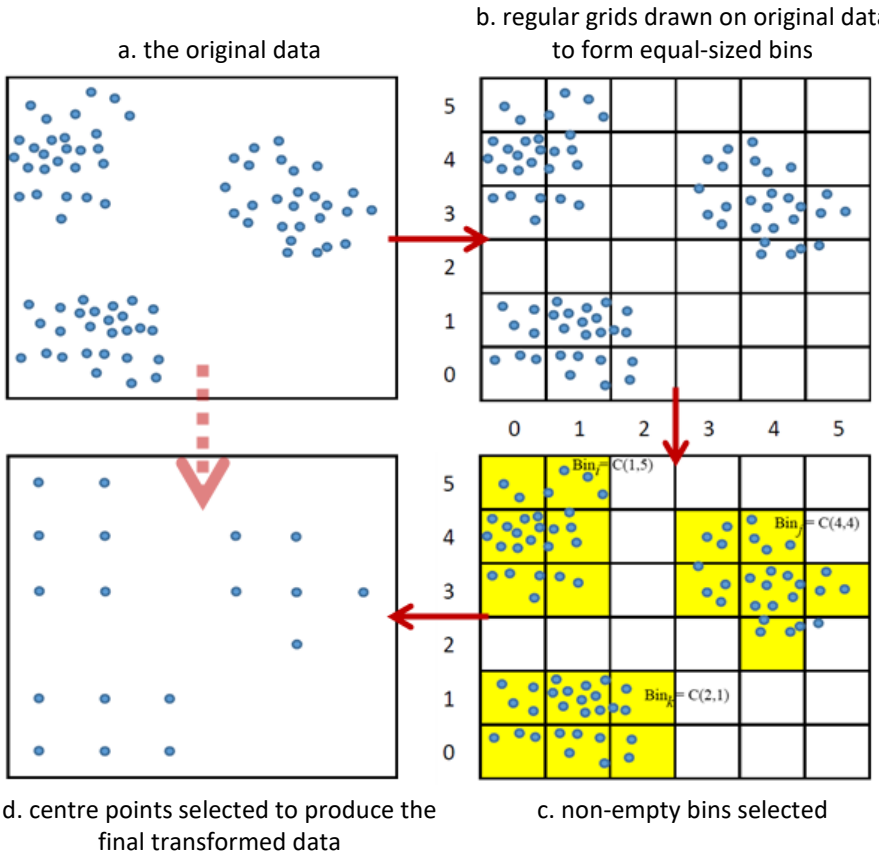


Figure 6-1 : Partitioning 2-dimensional data into equal-sized bins for 2D dataset

6.1.2 Applying OPTICS clustering technique

OPTICS is a density-based clustering technique that has an ability to automatically identify the optimal epsilon value and is faster than DBSCAN, as explained in Chapter 3. FLOPTICS is developed by applying OPTICS instead of DBSCAN that use in FlowGrid. After all data is partitioned into equal size of bins, each bin is treated as a data point. For the algorithm presented here, termed FLOPTICS, data is partitioned into equal-sized bins, and the data is clustered using the OPTICS algorithm (Ankerst et al., 1999). The radius distance (ε) has to be defined by the user, as in the DBSCAN (Ester et al., 1996) algorithm, but OPTICS can provide the optimal value of ε by showing the structure of data. Therefore, the number of user-defined parameters for FLOPTICS is fewer than FlowGrid, which is based on DBSCAN. The FLOPTICS algorithm can be summarized as follow:

Step 1: All data points are partitioned into equal sized bins for each dimension

Step 2: Only non-empty bins are processed

Step 3: Read an unprocessed bin (b) from the dataset obtained from Step 2

Step 2: If b is a core-bin, update core-distance of b

For each a ϵ $N\varepsilon(b)$,

update reachability of object a

Update the OrderSeeds list, which contains the objects ordered by reachability-distance (from smallest-to-largest)

Mark b as processed

Step 3: Read an unprocessed bin b from the OrderSeeds list if the list is not empty; otherwise, read the next unprocessed bin from the dataset

Step 4: Repeat Step 3 - Step 5 until the end of the dataset

The key terms involved in the algorithm are defined:

- *Nε(b)* is a set of neighbours of bin b for radius distance value *ε*, identified by the user
- Core-bin is the bin that its number of neighbour (regarding the radius *ε*) more than or equals to *MinPts*, which is identified by a user
- *Core-distance(b*) is the minimum distance that lead the number of neighbour of bin b reach *MinPts*
- Directly connected: Bin a is directly connected to Bin *b* if *Distance(a,b)≤ ε*
- Reachability-distance: If *b* and *o* are bins in the grid space, *ε* is the distance between two bins, *Nε(b)* is a set of neighbours of bin *b*, and *MinPts* is the minimum number of neighbours, then reachability-distance *ε*,*MinPts(o,b)* is equal to:
    - Infinity or undefined, if the number of members in $N\varepsilon(p)$ is less than $MinPts$
    - Otherwise the maximum of core-distance(b) or distance($o,\ b$)
- The OrderSeeds list is the list (queue) of bins in the grid space ordered by reachability-distance

6.2 Result

This section provides the experiment and result of applying traditional clustering method that includes K-means, DBSCAN and OPTICS to generated datasets with different overlapping. FlowGrid, which is a state-of-art automated gating technique, is also applied to these datasets in order to evaluate and compare the performance with the novel automated gating technique, FLOPTICS. The datasets were generated to mimic the distribution pattern of a real sample. The experiment was divided according to the number of dimensions of the data that consists of 2-dimensional and 3-dimentional data.

The experiment was conducted on a computer with specifications as follows:

- Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz
- RAM 16.0 GB
- Windows 10 Enterprise, 64 bit Operating System

6.2.1 Reference Data Sets

In Chapter 5, experiments with two-dimensional data were conducted to roughly evaluate the efficacy, strength, and weakness of each technique. However, flow cytometry data is typically in a multidimensional format, which can be more than two dimensions. The experiments in Chapter 6 were to demonstrate that FLOPTICS can be applied to multidimensional data. Therefore, three-dimensional data were applied for this experiment.

The pattern of clusters in real sample datasets obtaining from different patients will be have similar but different characteristics even from the same flow cytometry experimental design. A real data sample usually follows a normal or Gaussian distribution with different parameters, with the cluster shape symmetric or asymmetric depends on the patient and the markers used. Clusters can be circle shaped with different radius or cigar shaped with different width, height, and angle (Wiley, 2017)

The datasets used in this experiment were generated to mimic features of a real sample, while retaining control over different cluster and spacing parameters in such a way as to properly test the algorithms, as well as providing an absolute set of labels for classification. The datasets were generated using the function `numpy.random.multivariate(mean,cov,size)` in Python; this function randomly generates data from a multivariate normal distribution. This function, three arguments are required: an average of the data (mean), a covariance matrix (cov), and the number of data points (size).

For 2-dimensional dataset, each dataset consisted of the same three clusters for each dataset. The complete set of defining parameters are shown in Table 6-1, with the number of data points and covariance matrix for Clusters 1, 2 and 3 the same in each Dataset. The number of data points in Clusters 1, 2 and 3 were 5000, 2500 and 2500 respectively. There were four datasets in this experiment, each of which differed in the clarity of separation among clusters in the datasets. The four different Datasets are then defined by different means (values of x and y defining the centres of the clusters). The clusters in Dataset A are clearly separate and they are getting closer and closer in Dataset B, C, and D. The centres of Cluster 2 for each overlapping level are fixed at (-5, 10). The centres of Cluster 1 for Dataset A is (3, 35). The centres of Cluster 1 are then moved closer and closer to the centre of Cluster 2, decreasing x by 2 and decreasing y by 5 so the centres of Cluster 1

for Dataset B, C, and D are (1, 30), (-1, 25), and (-3, 20), respectively. The centres of Cluster 3 for Dataset A is (16, 1). The centres of Cluster 3 are then moved closer to the centre of Cluster 2, decreasing x by 4 so the centres of Cluster 3 for Dataset B, C, and D are (12, 1), (8, 1), and (4, 1), respectively. The resulting scatter plots of the four data sets are shown in Figure 6-2.

For 3-dimensional dataset, it uses the same principles as creating 2-dimensional dataset. The complete set of defining parameters are shown in Table 6-2, with the number of data points and covariance matrix for Clusters 1, 2, 3, and 4 the same in each Dataset. The number of data points was equal to 8,000 in each Dataset and a Dataset were separated into four clusters and each cluster has 2,000 data points. The four different Datasets are then defined by different means (values of x, y, and z defining the centres of the clusters). The complete set of defining parameters are shown in Table 6-2. The clusters in Dataset E are clearly separated. The clusters in datasets F, G, and H are getting closer and closer by 50%, 25% and 12.5%, respectively. The centres of Cluster 4 for all datasets are fixed at (0, 0, 0). The centres of Cluster 1, 2, and 3 for Dataset E are (20, 100, 60), (100, 60, 20), and (60, 20, 100). Dataset F, the centres are then moved 50% closer Cluster 4, so the centres of Cluster 1, 2, and 3 for Dataset F are (10, 50, 30), (50, 30, 10), and (30, 10, 50), respectively. Dataset G, the centres are moved 25% closer to Cluster 4, so the centres of Cluster 1, 2, and 3 for Dataset G are (7.5, 37.5, 22.5), (37.5, 22.5, 7.5), and (22.5, 7.5, 37.5), respectively. Dataset H, the centres are moved 12.5% closer to Cluster 4, so the centres of Cluster 1, 2, and 3 for Dataset H are (6.5625, 32.8125, 19.6875), (32.8125, 19.6875, 6.5625), and (19.6875, 6.5625, 32.8125), respectively. The resulting scatter plots of the four datasets are shown in Figure 6-3. The 3D-datasets are generated 3 times for each overlapping levels so the Datasets used in this experiment are E-1, E-2, E-3, F-1, F-2, F-3, G-1, G-2, G-3, H-1, H-2, and H-3.

The accuracy of each cluster is defined as

$$accuracy = 100 \times \frac{n_0}{n_1} \tag{6-2}$$

where

n_0 is the number of correct clustered data points in a cluster

n_1 is the number of all data points in a cluster

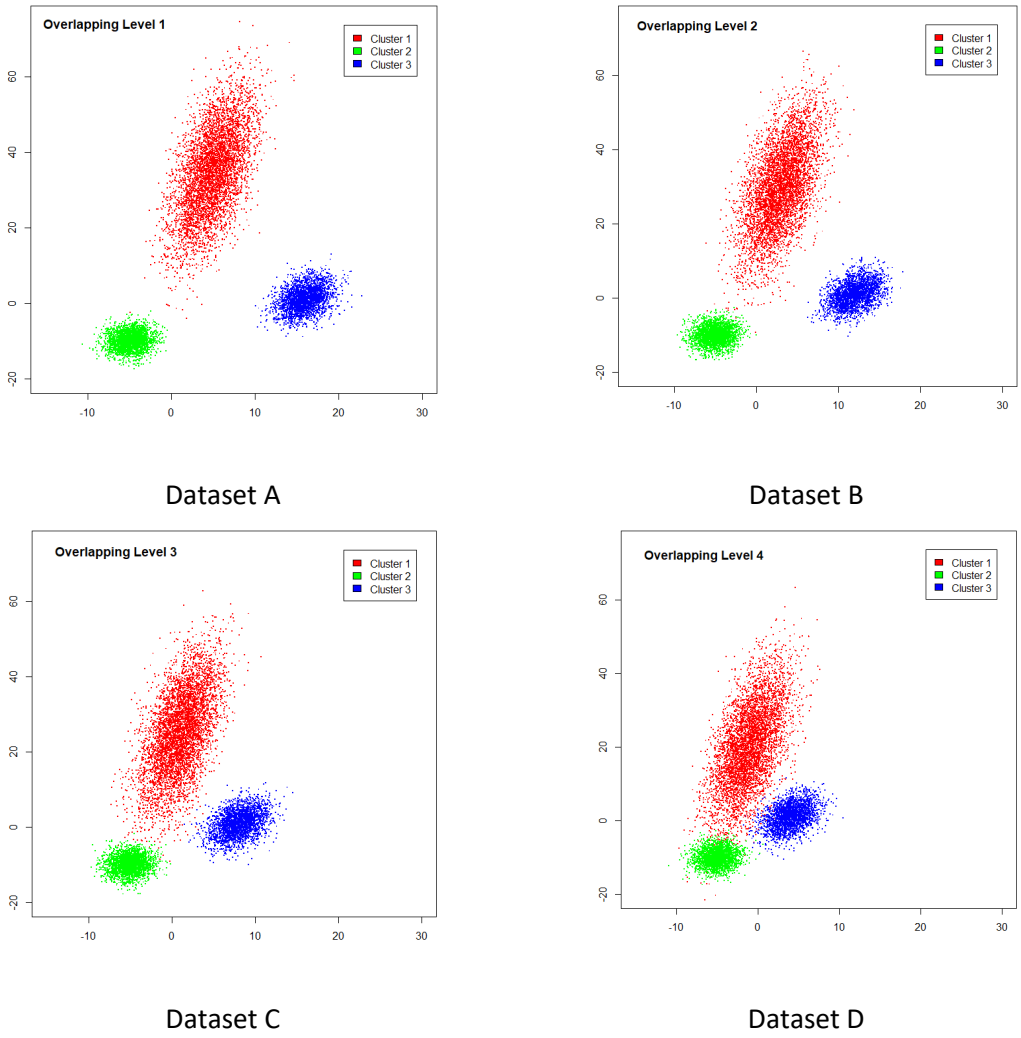
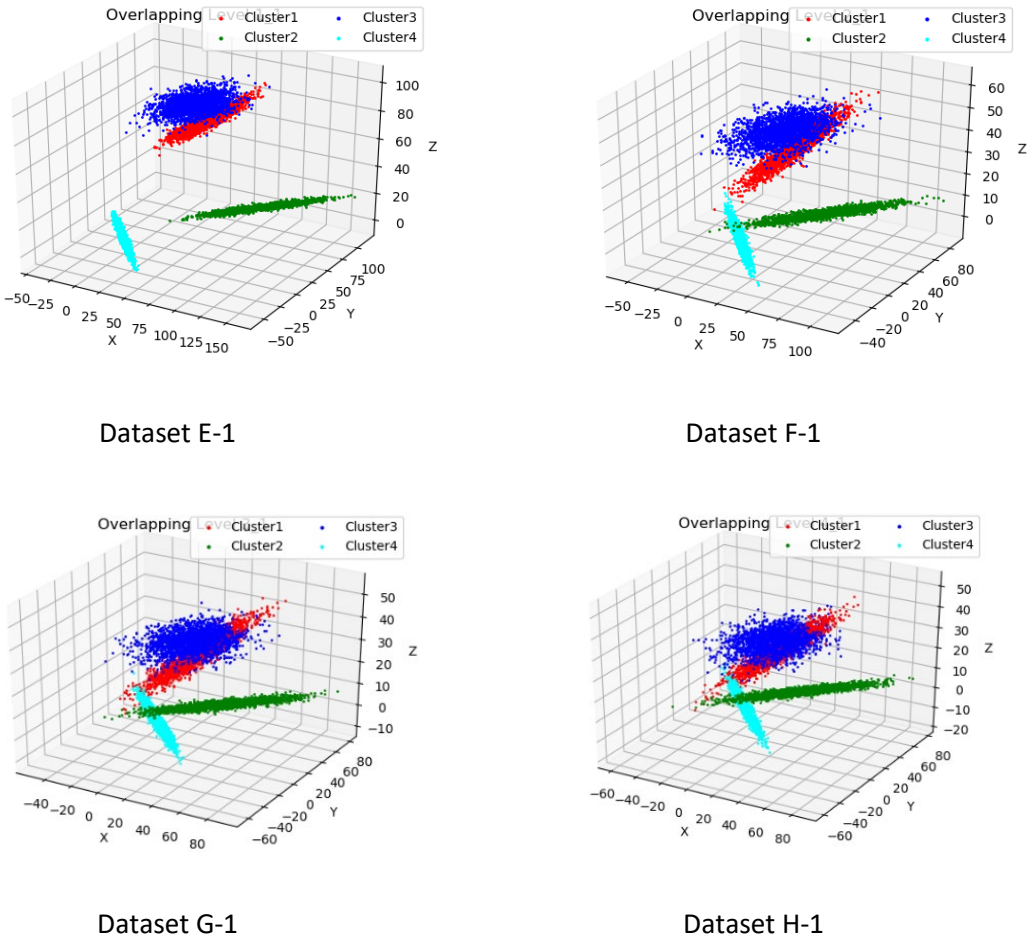


Figure 6-2 : The generated 2D-datasets with different overlapping

Table 6-1: The complete set of defining parameters for 2D-dataset generating

Cluster	Sigma	N	Means (Centres) (x,y)			
			Level 1 (Dataset A)	Level 2 (Dataset B)	Level 3 (Dataset C)	Level 4 (Dataset D)
1	$\begin{vmatrix} 6 & 15 \\ 15 & 120 \end{vmatrix}$	5,000	(5, 35)	(3, 30)	(1, 25)	(-1, 20)
2	$\begin{vmatrix} 2 & 0.3 \\ 0.3 & 5 \end{vmatrix}$	2,500	(-5, -10)	(-5, -10)	(-5,-10)	(-5, -10)
3	$\begin{vmatrix} 3 & 2 \\ 2 & 10 \end{vmatrix}$	2,500	(16, 1)	(12, 1)	(8, 1)	(4, 1)



Dataset E-1 Dataset F-1

Dataset G-1 Dataset H-1

Figure 6-3 : The generated 3D-datasets with different overlapping

6.2.2 2D-datasets experiment and results

The datasets were clustered using DBSCAN, OPTICS, FlowGrid, FLOPTICS, with user-defined parameters shown in Table 6-3. OPTICS and FLOPTICS can identify optimal epsilon by using reachability distance plotting, while DBSCAN and FlowGrid does not have this ability. In this experiment, the values of epsilon for DBSCAN and FlowGrid used the same values as OPTICS and FLOPTICS that are 0.8 and 6.0 respectively. These value of epsilon give the best result for both DBSCAN and FlowGrid. Generally, users have to continually configure the value of epsilon for DBSCAN and FlowGrid until the optimal one is obtained. Therefore, this may take a lot of time to adjust the epsilon. All techniques were implemented and run on RStudio 3.5.2, and the summary results are presented in Table 6-4.

The data were generated three times for each overlapping level which are four levels so there are 12 different datasets for the experiment. According to the result in Table 6-4, the average accuracy for each overlapping level of OPTICS, FlowGrid, and FLOPTICS is not much different, while, the

average accuracy of DBSCAN sharply drops when the dataset has high overlapping among between clusters. OPTICS provide highest accuracy in this table that is 97.59% because OPTICS has an ability to automatically decide the optimal value of the important parameter as ε.

Table 6-2: The complete set of defining parameters for 3D-dataset generating

Cluster	Sigma	N	Means (Centres) (x,y,z)			
			Level 1 (Dataset E)	**Level 2 (Dataset F)**	**Level 3 (Dataset G)**	**Level 4 (Dataset H)**
1	$\begin{vmatrix} 0 & 0 & 1 \\ 400 & 2 & 200 \\ 1 & 10 & 10 \end{vmatrix}$	2,000	(20, 100, 60)	(10, 50, 30)	(7.5, 37.5, 22.5)	(6.5625, 32.8125, 19.6875)
2	$\begin{vmatrix} -400 & -400 & 1 \\ 0 & 3 & 0 \\ 1 & 1 & 1 \end{vmatrix}$	2,000	(100, 60, 20)	(50, 30, 10)	(37.5, 22.5, 7.5)	(32.8125, 19.6875, 6.5625)
3	$\begin{vmatrix} 3 & 400 & 1 \\ 200 & 0 & 1 \\ 4 & 10 & 1 \end{vmatrix}$	2,000	(60, 20, 100)	(30, 10, 50)	(22.5, 7.5, 37.5)	(19.6875, 6.5625, 32.8125)
4	$\begin{vmatrix} 0 & 0 & 1 \\ -300 & 300 & 1 \\ 0 & 2 & 1 \end{vmatrix}$	2,000	(0, 0, 0)	(0, 0, 0)	(0, 0, 0)	(0, 0, 0)

DBSCAN and OPTICS are directly applied to original data so it provide high accuracy for overlapping dataset level 1, 2, and 3 but for level 4, only OPTICS can cluster data correctly. Although the accuracy of OPTICS is highest, it has the problem with time consuming. OPTICS is faster than DBSCAN around 3.5 times but it still much slower than FlowGrid and FLOPTICS around 2.7 and 7.4 times, respectively. As stated above, OPTICS is faster than DBSCAN so FLOPTICS is faster than FlowGrid because FLOPTICS technique is based on OPITCS, while, FlowGrid is based on DBSCAN. Moreover, OPTICS can automatically identify parameter ε so FLOPTICS also has this ability.

6.2.3 Discussion of 2-D results

As mentioned earlier, two main problems of manually automated gating process are that they are highly subjective and time consuming. Many modification and combination of clustering techniques have been applied for FCM data. FlowGrid is the latest approach and better than many state of the art automated gating techniques. Although FlowGride can provide properly gating results in term of accuracy and runtime, there are too many user-defined parameters for this technique. Therefore, FLOPTICS is proposed in order to deal with this problem by combination of grid-based and OPTICS, which is a density-based clustering technique. According to experiment result,

FLOPTICS is faster than FlowGrid and need less user-defined parameters. FlowGrid and FLOPTICS need to reduce amount of dataset by using equal size of bin before enter to the clustering process so the accuracy of these two techniques is slightly lower than OPTICS. The amount data is reduced for FlowGrid and FLOPTICS, but they provided a higher accuracy than DBSCAN even DBSCAN is applied to original data that is because FCM data is normally high density and high overlapping so if size of bin is properly defined, the accuracy can be higher than applying to the original data.

Table 6-3: The parameter values for each technique

DBSCAN	**OPTICS**	**FlowGrid**	**FLOPTICS**
ε = 0.8 MinPts = 10	ε = optimal MinPts = 10	ε = 6 MinDenB = 3 MinDenC = 40 Bin_size = 100	ε = optimal MinPts = 10 Bin_size = 100

Table 6-4: The results of applying the analysis methods techniques to the datasets

Techniques	**Average accuracy** (%)				**Overall average accuracy** (%)	**Average Runtime** (milli-second)
	Overlapping dataset Level					
	1	2	3	4		
DBSCAN	94.99	94.70	94.80	62.70	86.80	6,728.60
OPTICS	99.93	99.75	98.60	92.07	97.59	1,961.12
FlowGrid	96.00	95.75	95.01	93.59	95.09	723.80
FLOPTICS	99.87	99.49	97.60	90.45	96.85	265.88

Therefore, FLOPTICS is proposed in order to deal with this problem by combination of grid-based and OPTICS, which is a density-based clustering technique. According to experiment result, FLOPTICS is faster than FlowGrid and need less user-defined parameters. FlowGrid and FLOPTICS need to reduce amount of dataset by using equal size of bin before enter to the clustering process so the accuracy of these two techniques is slightly lower than OPTICS. The amount data is reduced for FlowGrid and FLOPTICS, but they provided a higher accuracy than DBSCAN even DBSCAN is applied to original data that is because FCM data is normally high density and high overlapping so if size of bin is properly defined, the accuracy can be higher than applying to the original data.

6.2.4 3D-datasets experiment and results

The 3D-datasets were clustered using K-means, DBSCAN, OPTICS, FlowGrid, and FLOPTICS, with user-defined parameters shown in Table 6-5. The experiment assumed that the users know that there are 4 clusters in the datasets so the parameter k for K-means technique is defined as 4. Therefore, the result of K-means in term of speed and accuracy when applying to the Dataset E and F is quite good that because the optimal k were given. However, in reality the user always does not know the actual number of clusters in the dataset. Likewise, DBSCAN does not has an ability to optimize epsilon value, as this is a user defined parameter. In order to find an optimal epsilon value, the OPTICS techniques was applied to the data set and used in the DBSCAN technique. For this experiment, the optimal epsilons were assigned to DBSCAN, resulting in very good results. However, if the user actually implements this technique, they will also have to spend a lot of time figuring out the optimal epsilon manually. The value of parameter Bin_size used in FLOPTICS and MinPts used in FLOPTICS and OPTICS are roughly estimated without any special methods, therefore, if these values are changed, the clustering result will change accordingly.

The metrics use in this experiment are running time (seconds), the number of noise, the number of false cluster and accuracy, as shows in Table 6-6. K-means does not has an ability to automatically detect noise and therefore is not shown in the table. In other words, the number of noise for K-means in every dataset equals to 0. The comparison result between FLOPTICS and FlowGrid were divided into two tables (Table 6-7 and Table 6-8) because the experiment was performed twice using different set of user-defined parameters. Both techniques assume that the value of Bin_size is equal to 120 and epsilon were detected by FLOPTICS and were assigned to FlowGrid. The results shown in Table 6-7 were given MinPts = 20 for FLOPTICS and MinDenC = 20 and MinDenB = 1 for FlowGrid. The results shown in Table 6-8 were given MinPts = 100 for FLOPTICS and MinDenC = 100 and MinDenB = 1 for FlowGrid.

All techniques were implemented by Python language on PyCharm 2021.2.2 and the comparison result are shown in Table 6-6. The matrices used in the 3D-experiment includes optimal epsilon, running time, the number of noises, the number of false clusters, and accuracy. FLOPTICS and OPTICS can identify the optimal epsilon by using reachability distance plotting, but the epsilon for DBSCAN must be defined by users. For this experiment, DBSCAN uses the epsilon obtained from OPTICS that provides the best result for DBSCAN. The optimal epsilon are shown in Table 6-6. According to Table 6-6, K-means seems to be the fastest technique comparing to others and it is a bit faster than DBSCAN, while, OPTICS is the slowest one. FLOPTICS is around 40% faster than OPTICS because the data entering to FLOPTICS is transformed first and results in a decrease in the amount of data.

The number of noise detection is one of the metrics used in this experiment. The datasets were generated without noise so it means that the technique that detect less number of noise is better than the technique that detect more noise. The number of noises detected by DBSCAN, OPTICS, and FLOPTICS in Dataset E, Dataset F, and Dataset G is not much different except for the Dataset H. For Dataset E and Dataset G, DBSCAN is slightly better than OPTICS and FLOPTICS. The average number of noise detected by DBSCAN in Dataset E is 1.33, while OPTICS and FLOPTICS are equal to 3.33. For Dataset G, the average number of noise detected by DBSCAN is 999.67, while OPTICS and FLOPTICS are 1,022.67 and 1,122.33, respectively. For Dataset F, FLOPTICS is lightly better than DBSCAN and OPTICS. The average number of noise detected by FLOPTICS is 352.33, while DBSCAN and OPTICS are 361.67 and 367.67, respectively. For Dataset H, which is the highest level of overlapping, FLOPTICS detected much less number of noise than those two, which is 1,149.00 in average, while DBSCAN and OPTICS are 1,727.00 and 1,768.33, respectively. Although DBSCAN is better than FLOPTICS and OPTICS when applying to Dataset E and Dataset G, DBSCAN is much worse when applying to Dataset H. According to the experiment, FLOPTICS provided the best result in term of noise detection, especially when the dataset has high overlapping among clusters.

False cluster is a matric use in the experiment. False cluster is identification of data to a cluster that is not the cluster that the data should belong to. K-means appears to be the worst if measured in terms of false cluster, while FLOPTICS provides the best result. The more overlapping level among clusters is the Datasets, the larger number of false clusters are identified by all techniques. However, the increasing of the number of false cluster detected for FLOPTICS was slower than other techniques as the overlapping level was increased. FLOPTICS, OPTICS, and DBSCAN gave the same the average number of false cluster that is 0.33 when they were applied to the Datasets E, which is minimal overlapping level. As the overlap among clusters increases in Datasets F, Datasets G, and Datasets H, the average amount of false clusters increases rapidly in DBSCAN that were 28.00, 243.00, and 304.00, respectively. The average amount of false clusters increases in OPTICS a bit slower than DBSCAN when it was applied to Datasets F, Datasets G, and Datasets H that were 27.67, 231.67, and 267.33, respectively. FLOPTICS gave the best result in term of the number of false cluster, that were 25.33, 77.67, and 131.67 when it was applied to the Datasets F, G, and H, respectively.

The accuracy of these 4 techniques is not much different when applied to the Datasets with clearly separated clusters that are Dataset E and F. DBSCAN gave the best average accuracy, which is 99.98%, when it was applied to the Datasets E, while FLOPTICS, OPTICS, and K-means gave 99.97%, 99.95%, and 99.78, respectively. However, when the clusters were moved closer and closer in the Datasets F, G, and H, FLOPTICS provided the best average accuracy, which are 95.51%, 85.00%, and 83.99%, respectively. K-means showed much reduction of accuracy when applied to Dataset F, G

and H that are 92.40%, 73.34%, and 68.75%, respectively. DBSCAN and OPTICS showed a slightly drop of average accuracy when the overlapping of clusters are increased. In term of accuracy, FLOPTICS showed the best result in this experiment.

Table 6-5: The parameter values for each technique

FLOPTICS	OPTICS	DBSCAN	K-means
ε = optimal Bin_size = 120 MinPts = 20	ε = optimal MinPts = 20	ε = optimal (getting from OPTICS)	K = 4

According to Table 6-7, it is clearly seen that FlowGrid is faster than FLOPTICS, but when it comes to accuracy, noise detection, and false cluster detection, then FLOPTICS yields more satisfactory results. Although, FlowGrid was faster than OPTICS in this experiment, FlowGrid did not include the time required to determine the optimal epsilon. The optimal epsilon obtained by OPTICS were assigned to FlowGrid and this process were calculated outside the experiment. In order to use FlowGrid, the user will actually spend a lot of time trying to investigate the optimal epsilon. According to Table 6-8, FlowGrid provides better results in terms of running time and noise detection, but for false cluster detection, FLOPTICS gave better result because the number of false cluster identifying by FLOPTICS is less than FlowGrid. In terms of accuracy, the result of these two techniques is not much different when applied to Dataset E and F but FlowGrid gave slightly better accuracy when applied to Dataset G and H.

Table 6-6: The comparison result of FLOPTICS, OPTICS, DBSCAN, and K-means applying to 3D-datasets

Dataset	Optimal Epsilon		Running time (seconds)				Number of noises			Number of false clusters				Accuracy (%)			
	FLOPTICS	OPTICS	FLOPTICS	OPTICS	DBSCAN	K-means	FLOPTICS	OPTICS	DBSCAN	FLOPTICS	OPTICS	DBSCAN	K-means	FLOPTICS	OPTICS	DBSCAN	K-means
E-1	17.4928	19.7352	3.6955	8.7076	0.2785	0.0377	2	2	1	1	1	1	15	99.99	99.96	99.98	99.81
E-2	18.1934	25.4144	3.6458	8.6268	0.2573	0.0533	3	3	0	0	0	0	25	99.95	99.96	100.00	99.74
E-3	18.6815	16.5166	3.6085	8.7285	0.2858	0.0377	5	5	3	0	0	0	18	99.98	99.94	99.96	99.78
Average	**18.1226**	**20.5554**	**3.6499**	**8.6867**	**0.2739**	**0.0429**	**3.33**	**3.33**	**1.33**	**0.33**	**0.33**	**0.33**	**18.00**	**99.97**	**99.95**	**99.98**	**99.78**
F-1	5.0990	5.2858	5.2750	8.6296	0.1356	0.0846	303	303	296	26	26	26	585	96.58	95.89	95.98	92.69
F-2	4.9999	4.8243	5.2458	8.6442	0.1627	0.0533	221	338	332	27	24	25	624	96.90	95.48	95.54	92.00
F-3	4.1231	4.2534	5.2410	8.7150	0.1316	0.0690	533	462	457	23	33	33	616	93.05	93.81	93.88	92.30
Average	**4.7407**	**4.7878**	**5.2539**	**8.6629**	**0.1433**	**0.0690**	**352.33**	**367.67**	**361.67**	**25.33**	**27.67**	**28.00**	**608.33**	**95.51**	**95.06**	**95.13**	**92.40**
G-1	3.1622	3.2692	5.5361	86905	0.1015	0.1159	1,464	949	931	93	260	277	2,200	80.54	84.89	84.90	72.50
G-2	3.7416	3.4583	5.2661	8.6295	0.1117	0.1380	721	917	892	79	108	116	2,166	90.00	87.19	87.40	72.93
G-3	3.1622	2.9967	5.4078	8.7751	0.1159	0.1314	1,182	1,202	1,176	61	327	336	2,034	84.46	80.89	81.10	74.58
Average	**3.3553**	**3.2414**	**5.4033**	**8.6984**	**0.1097**	**0.1284**	**1,122.33**	**1,022.67**	**999.67**	**77.67**	**231.67**	**243.00**	**2,133.33**	**85.00**	**84.32**	**84.47**	**73.34**
H-1	3.3166	2.5771	5.4519	8.6751	0.0938	0.1158	1,006	1,871	1,816	145	206	259	2,422	85.61	74.04	74.06	69.73
H-2	3.1622	2.7281	5.7147	8.6890	0.1002	0.0846	1,407	1,538	1,518	106	183	198	2,658	81.09	78.49	78.55	66.78
H-3	3.3166	2.5771	5.5684	8.8229	0.0848	0.1002	1,034	1,896	1,847	144	413	455	2,420	85.28	71.14	71.23	69.75
Average	**3.2651**	**2.6274**	**5.5783**	**8.7290**	**0.0929**	**0.1002**	**1,149.00**	**1,768.33**	**1,727.00**	**131.67**	**267.33**	**304.00**	**2,500.00**	**83.99**	**74.56**	**74.61**	**68.75**

Table 6-7: The comparison result of FLOPTICS and FlowGrid applying to 3D-datasets

Dataset	Optimal Epsilon	Running time (seconds)		Number of noises		Number of false clusters		Accuracy (%)	
		FLOPTICS	FlowGrid	FLOPTICS	FlowGrid	FLOPTICS	FlowGrid	FLOPTICS	FlowGrid
E-1	17.4928	3.6955	1.5568	2	6	1	0	99.99	99.93
E-2	18.1934	3.6458	1.9738	3	13	0	0	99.95	99.84
E-3	18.6815	3.6085	1.4350	5	10	0	0	99.98	99.88
Average	**18.1226**	**3.6499**	**1.6522**	**3.33**	**9.67**	**0.33**	**0.00**	**99.97**	**99.88**
F-1	5.0990	5.2750	0.5548	303	701	26	215	96.58	88.55
F-2	4.9999	5.2458	0.5929	221	624	27	339	96.90	87.96
F-3	4.1213	5.2410	0.5720	533	892	23	525	93.05	82.29
Average	**4.7407**	**5.2539**	**0.5732**	**352.33**	**739.00**	**25.33**	**359.67**	**95.51**	**86.27**
G-1	3.1622	5.5361	0.6641	1,464	1,854	93	376	80.54	67.63
G-2	3.7416	5.2661	0.4479	721	1,079	79	685	90.00	77.95
G-3	3.1622	5.4078	0.4480	1,182	1,626	61	731	84.46	70.54
Average	**3.3553**	**5.4033**	**0.5200**	**1,122.33**	**1,519.67**	**77.67**	**717.33**	**85.00**	**72.04**
H-1	3.3166	5.4519	0.4858	1,006	1,426	145	1,111	85.61	68.29
H-2	3.1622	5.7147	0.4542	1,407	1,745	106	867	81.09	67.35
H-3	3.3166	5.5684	0.4860	1,034	1,492	144	815	85.28	71.16
Average	**3.2651**	**5.5783**	**0.4753**	**1,149.00**	**1,554.33**	**131.67**	**931.00**	**83.99**	**68.93**

Value of parameters FLOPTICS, Bin_size = 120, MinPts = 20, epsilon = optimal epsilon**,** FlowGrid, Bin_size = 120, MinDenC = 20, MinDenB = 1, epsilon = optimal epsilon

Table 6-8: The comparison result of FLOPTICS and FlowGrid applying to 3D-datasets

Dataset	Optimal Epsilon	Running time (seconds)		Number of noises		Number of false clusters		Accuracy (%)	
		FLOPTICS	FlowGrid	FLOPTICS	FlowGrid	FLOPTICS	FlowGrid	FLOPTICS	FlowGrid
E-1	22.1359	3.6955	2.0279	2	3	1	0	99.96	99.96
E-2	20.1494	3.6458	1.8611	13	9	2	0	99.81	99.89
E-3	20.6397	3.6085	2.1275	10	9	3	0	99.84	99.89
Average	**20.9750**	**3.6499**	**2.0055**	**8.33**	**7.00**	**2.00**	**0.00**	**99.87**	**99.91**
F-1	8.0622	5.2750	1.0869	466	552	35	27	93.74	92.76
F-2	7.6811	5.2458	0.8074	455	502	32	29	93.91	93.36
F-3	7.3484	5.2410	1.0026	617	602	38	10	91.81	92.35
Average	**7.6972**	**5.2539**	**0.9656**	**512.67**	**552.00**	**35.00**	**22.00**	**93.15**	**92.82**
G-1	6.0827	5.5361	0.7486	1,472	1,418	136	70	79.90	81.40
G-2	6.1644	5.2661	0.8553	1,141	1,052	138	94	84.01	85.68
G-3	5.8309	5.4078	0.5543	1,379	1,202	133	88	81.10	83.88
Average	**6.0260**	**5.4033**	**0.7194**	**1,330.67**	**1,224.00**	**135.67**	**84.00**	**81.67**	**83.65**
H-1	5.3851	5.4519	0.5599	2,196	1,793	93	92	71.39	76.44
H-2	5.4772	5.7147	0.7863	2,081	1,181	129	242	72.38	74.34
H-3	5.3851	5.5684	0.6639	2,241	1,729	104	126	70.69	76.81
Average	**5.4158**	**5.5783**	**0.6700**	**2,172.67**	**1,567.67**	**108.67**	**153.33**	**71.49**	**75.86**

Value of parameters FLOPTICS, Bin_size = 120, MinPts = 100, epsilon = optimal epsilon**,** FlowGrid, Bin_size = 120, MinDenC = 100, MinDenB = 1, epsilon = optimal epsilon

6.3 Conclusion

Flow cytometry data is very powerful for diagnosing anomaly in blood sample but it is so large and complex that is difficult to manually analyse. Manual gating of FCM data is highly subjective and excessively time-consuming so the research community has made an effort to increase competence by developing several automated gating methods based on machine learning techniques. Many automated gating frameworks have been proposed, and they can support the clinicians in clustering homogeneous cells. There are many benefits from using these frameworks, but there are still some limitations as can be seen in Table 4.1. FLOPTICS is a novel automated gating technique that is combination between density-based and grid-based clustering technique. For 2D-datasets, the clustering result of FLOPTICS, DBSCAN, OPTICS, and FlowGrid is not much different but FLOPTICS is

faster than all techniques that have been tested. For 3D-datasets, FLOPTICS is the second slowest technique after OPTICS. FLOPTICS and FlowGrid gave a bit better accuracy than DBSCAN and OPTICS. However, the value of user-defined parameters greatly influence the outcome for FLOPTICS and FlowGrid. If the value of parameters are optimal, the result are satisfactory, if not then the result are not acceptable. For example, the optimal parameter for dataset E-1 is approximately 22.1359 that given accuracy of 99.99% and 99.93% for FLOPTICS and FlowGrid respectively, but if the epsilon is changed to 18 the accuracy may be reduced by 20-30%.

It may be concluded whether the results are satisfactory or not is depending on the value of user-defined parameters and the characteristic of data sample. Although, FLOPTICS is slower than FlowGrid in 3D-datasets but FLOPTICS can detect optimal epsilon, while FlowGrid cannot detect it.

Chapter 7 Conclusion

Flow cytometry (FCM) data is widely used for blood cell analysis. The first thing the clinician has to do with FCM data is to separate a huge number of cells into groups according to their characteristics. This step of the FCM data analysis is called manual gating. This process is time-consuming and highly subjective because the result depends on the clinician's experience of working with FCM data. Since manual gating requires grouping data together by manually drawing the boundaries around cells, the different groupings may result even from the same dataset. For this reason, many clustering techniques have been employed and can be applied to FCM data to reduce the subjectivity and time taken. These techniques are called automated gating. Although many automated-gating techniques have been developed, each has different shortcomings. While FlowGrid is an efficient automated-gating technique in terms of accuracy and time efficiency, it needs many user-defined parameters. This research aimed to improve the capabilities of FlowGrid by reducing the number of parameters and time taken.

This research presented a new technique, called FLOPTICS, used for flow cytometry data analysis and cell classification. FLOPTICS is a combination of density-based clustering and grid-based clustering. The main feature of FLOPTICS is that it requires fewer user-defined parameters than other techniques, and is faster than state-of-art techniques, when applied to 2-dimensional and 3-dimensional datasets. This new technique can therefore shorten the working time for clinicians, especially in regard to the selection of the epsilon (ε) parameter, which can be automatically detected by FLOPTICS.

7.1 Contribution

FLOPTICS is a novel automated gating application for flow cytometry data, which combines grid-based and density-based clustering. The contributions of this research are summarised below.

This research proposes a new algorithm for automatically identifying groups of cells, which is called FLOPTICS. FLOPTICS is an improvement on the latest automated gating technique, FlowGrid. FlowGrid provides great accuracy and a short processing time. However, FlowGrid needs too many user-defined parameters, which are ε (epsilon), bin size, density of bin, density of collective bin, and ρ%. These user-defined parameters greatly affect the result of clustering. Epsilon is the distance used to determine whether two data points will be counted as neighbours, so epsilon is one of the key parameters that affects the result .FlowGrid needs the value of epsilon to be provided by the user, while FLOPTICS has the ability to automatically identify epsilon. FLOPTICS is a modification of FlowGrid by applying the OPTICS algorithm to the clustering process. OPTICS itself is derived from

DBSCAN, which deals with the need for the user to provide the parameter epsilon. As shown in Chapter 5, FLOTPICS provides great results for clustering, in terms of accuracy and time processing, when applied to the synthesised dataset, which were emulated a real sample.

FLOPTICS is an alternative way to identify groups of cells automatically from flow cytometry data. Although various automated gating techniques have been developed, clinicians still classify cells from flow cytometry data manually, because the data to be analysed are health-related, so the used of these new techniques must be extremely prudent. These techniques are therefore still being tested and have not yet been put into practice.

7.2 Conclusion

The challenge of this research is that clinical agreement on the results from flow cytometry analysis is inconsistent. Gating is the first step of flow cytometry analysis, whose purpose is to group identical cells together. Manual gating is the process of drawing boundaries around groups of cells, and is highly subjective, and results in the groups being inconsistent. Therefore, the diagnosis can be reported differently and lead to different clinical decisions, depending on the clinician's experience of working with these data. For this reason, many machine learning techniques have been developed to deal with this subjectivity. Although many automated gating techniques have been developed, these techniques have not yet been used in practice because they have some problems, such as too many user-defined parameters and are time-consuming.

7.2.1 Blood analysis

Chapter 2 discussed blood analysis, which is one of the most important medical tools. There are many benefits of blood analysis: blood testing has the ability to distinguish illnesses with similar symptoms, it can be used for some kinds of treatment monitoring, and it can detect disease where patients infected with a disease have no symptoms. Apart from detection of disease in blood samples, blood cell count is a very important criterion because it can be used to define a donor's state of health. FCM is the technique used to detect the characteristics of cells in a blood sample, based on the concept of cell-scatter measurement and light emission after passing the sample through a laser beam. Flow cytometry has the ability to measure the characteristics of cells, such as the number of individual cells, size of cells, complexity, and granularity. Therefore, flow cytometry is a very powerful medical tool and is widely use in haematology and immunology such as leukaemia diagnosis, treatment monitoring, and allergic substance diagnosis. Although flow cytometry is very useful for blood analysis, the technician analysing this data must have advanced training and experience.

FCM data analysis does not give consistent results because the first step of FCM analysis, called gating, is the process of cell classification and is highly subjective. Gating is the process of grouping the cells that have similar characteristics or homogeneous cells into the same cluster. First of all, the technician has to select one or two features of cells that are represented in the histogram or scatterplot. Then they must start to create gates in order to separate individual types of cell by drawing a line on histogram or by drawing a polygon on scatterplot. Given the same dataset, even experts may analyse these differently because this type of analysis is based on experience, not on facts. In addition to subjectivity, the manual gating process has another problem in that it is also time-consuming, because many pairs of features must be considered, and the technician has to switch back and forth between many features.

7.2.2 Clustering techniques

A machine learning technique widely used for automated gating is the clustering technique. The many approaches to clustering can be divided into: partitioning methods, hierarchical clustering, density-based clustering, and model-based clustering. The partitioning method tries to construct partitions on the data space, so each object is in only one group, but the number of clusters needs to be set in advance. One of the most widely-known and used partitioning methods is K-means. The hierarchical clustering method can be divided into agglomerative clustering and divisive clustering. The agglomerative clustering assumes that initially each cluster has one object, in other words, each object is a cluster. Then the two nearest clusters are repeatedly combined, and the distance matrix updated, until it is found that the clusters are appropriate or only one cluster remains. The concept of divisive clustering works the opposite way to agglomerative clustering. The divisive clustering assumes that all objects are in the same cluster, then recursively splits it into smaller clusters until it provides appropriate clusters. The concept of density-based clustering is that sparser areas in the space are used for separating one cluster from another. The outstanding strength of the density-based clustering is its ability to identify arbitrary or non-spherical shapes and the number of clusters is not required to be specified beforehand. The main idea of model-based clustering is to discover a mathematical model that fits the data. The model-based clustering technique works well on the data that comply with the selected model. These traditional clustering techniques have different strengths and weaknesses, so some of them have been chosen for modification and development for flow cytometry automated gating.

7.2.3 Automated gating techniques

There is no one accepted technique for clustering, because each has different accuracy, different time taken, and too many user-defined parameters. FlowGrid is the latest automated-gating

technique proposed in 2018. It is a combination of DBSCAN, a density-based clustering technique, and a grid-based clustering technique. FlowGrid provides higher accuracy, and is faster, than many techniques such as flowPeaks, flowSOM, and FLOCK. Moreover, FlowGrid can automatically define the number of clusters and considers noise because it is based on DBSCAN, which is the technique that can detect the number of clusters and accounts for noise automatically. Although FlowGrid is an effective automated-gating technique, it still has too many user-defined parameters, including size of bin, minDenB, minDenC, and epsilon (ε), which has a great impact on the gating results. In order to deal with this problem, this research therefore investigated a solution to reduce the number of user-defined parameters while still providing accuracy close to the original method. The challenge of this research was the development of a new technique that support clinicians in shortening the processing time used for optimising parameters.

7.2.4 FLOPTICS

FLOPTICS is an improvement on FlowGrid with the expectation that it would perform faster and require fewer user-defined parameters. The research recognised that FlowGrid can increase its performance in terms of speed and the number of user-defined parameters because FlowGrid was developed from DBSCAN, which requires more user-defined parameters and is slower than OPTICS. Therefore, OPTICS has been used in FLOPTICS to improve the limitations of FlowGrid. The process of FLOPTICS can be divided into: data partitioning, and applying the OPTICS clustering technique. Data partitioning is the process of transforming the original data by creating a grid on the data space and only a non-empty grid will be considered as one data point. Users have to define the size of grid, which is called a bin for this technique, then each dimension is partitioned into equally sized bins. All data points in the same bin are treated as a single point by using a representative. Transformed-data obtained from the data partitioning process enters the OPTICS clustering process without configuration of an epsilon value.

A set of experiments was performed to compare the performance of FLOPTICS with FlowGrid. These two techniques were applied to the same dataset, which was generated to mimic features of a real sample. The dataset used in the experiment are divided into 2-dimensional and 3-dimensional datasets. Each 2D dataset consisted of the same three clusters for each dataset, while each 3D dataset consisted of the same four clusters. The experimental data were randomly generated from a multivariate normal distribution, which are often found in FCM data, and were divided into four levels of overlapping within each cluster. The clusters overlapping in level 1 are clearly separated, while the clusters overlapping in level 4 are very close to each other. The dataset was generated 3 times for each level, so there were 12 datasets used in the experiment. The experimental results of FLOPTICS were satisfactory when applied to the 2D generated dataset. Although the accuracy of

FLOPTICS and FlowGrid were not much different, FLOPTICS was faster than FlowGrid. Moreover, user did not need to define the value of epsilon (ε), which is required in FlowGrid. For the 3D dataset, FLOPTICS provided fewer false clusters, less noise, and better accuracy, but it was slower than FlowGrid. This experiment gave the optimal epsilon obtained from FLOPTICS to FlowGrid. Therefore, although FlowGrid was faster than the FLOPTICS, in reality FlowGrid requires the user to define the optimal epsilon themselves, which is quite time-consuming.

7.3 Discussion

Flow cytometry data normally has four main features: high density, normal distribution, non-spherical shapes of cluster, and a lot of noise in the sample. This is the reason why the automated-gating technique based on K-means, or using only distance between data points, is not suitable for analysing FCM data. DBSCAN and OPTICS are density-based techniques that work well on a high-density dataset and a dataset with non-spherical clusters. Moreover, these techniques are able to automatically detect noise in sample data. Therefore, many automated-gating frameworks had been proposed by modification of density-based clustering techniques. However, density-based clustering often has a problem with speed, which is the reason why the dataset has to be transformed before being entered into the clustering process in FlowGrid and FLOPTICS framework. OPTICS is the technique that provides the structure of each cluster in the dataset, and it can define optimal epsilon, which is an important parameter for density-based clustering and has a great impact on the result. FLOPTICS was developed based on OPTICS, so FLOPTICS can work without defining of epsilon value. However, both FlowGrid and OPTICS still have some limitations, since the partitioning process can cause some data loss, they are not appropriate for a low-density dataset, the size of bins still has to be identified by the user. This has great impact on the clustering result, and every non-empty bin is treated as a single data point without considering of the number of data points in the bins.

7.4 Future Works

According to the limitations of FLOPTICS stated previously, the performance of FLOPTICS can be improved. Bin size is another important user-defined parameter and also has great impact on the gating result. It would be better if there was a technique that could identify the optimal size of bin. A Change point detection algorithm is one method that is able to find points that are suddenly changing, in time series data. In other words, the change point detection technique is statistical analysis trying to determine when the data generating process has significantly changed. For example, it is used to identify number of clusters by plotting the values of dependent variable,

which is sum square error, according to the sequentially arranged values of independent variable, which is the number of clusters. Then a point that abruptly changes will be selected and identified as the optimal number of clusters. It may be possible that this method can therefore be used to find the optimal size of bin in the FLOPTICS framework. Another FLOPTICS process that could be improved is that of identification of core-bin. To identify the core-bin, only the number of neighbours is considered. The number of data points in a bin is normally different and, sometimes, there are huge differences, but every single bin is treated as one data point. Both FLOPTICS and FlowGrid consider core-bin by counting the number of neighbours. The bins will be considered as core-bins if those bins have a number of neighbours equal to or more than the criteria (MinPts), without consideration of the number of original data points in the bin being considered. What if bins can be identified as core-bin with consideration of, not only the number of neighbours, but also the number of data points?

Appendix A Clustering demonstration

A.1 DBSCAN clustering

The data sample and the scatter plot are represented in Figure B1. The parameters are defined as, Eps = 1.5, $MinPts$ = 3.

Point	x	y
a	1.0	2.0
b	1.5	3.0
c	3.5	5.0
d	3.5	3.0
e	4.0	5.0
f	1.0	1.0
g	1.5	5.0
h	3.0	4.0
i	2.0	5.5
j	2.0	1.5

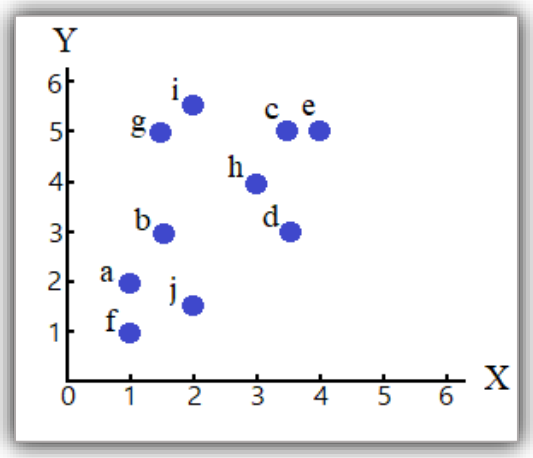


Figure A - 1: Example of data for DBSCAN

For step 1, label all data points as a core point, a border point, or a noise point. Create the following distance matrix.

From **a** to	
a =	**0.00**
b =	**1.12**
c =	3.91
d =	2.69
e =	4.24
f =	**1.00**
g =	3.04
h =	2.83
i =	3.64
j =	**1.12**

a is core point

From **b** to	
a =	**1.12**
b =	**0.00**
c =	2.83
d =	2.00
e =	3.20
f =	2.06
g =	2.00
h =	1.80
i =	2.55
j =	1.58

b is border point

From **c** to	
a =	3.91
b =	2.83
c =	**0.00**
d =	2.00
e =	**0.50**
f =	4.72
g =	2.00
h =	**1.12**
i =	1.58
j =	3.81

c is core point

From **d** to	
a =	2.69
b =	2.00
c =	2.00
d =	**0.00**
e =	2.06
f =	3.20
g =	2.83
h =	**1.12**
i =	2.92
j =	2.12

d is border point

From **e** to	
a =	4.24
b =	3.20
c =	**0.50**
d =	2.06
e =	**0.00**
f =	5.00
g =	2.50
h =	**1.41**
i =	2.06
j =	4.03

e is core point

From **f** to	
a =	**1.00**
b =	2.06
c =	4.72
d =	3.20
e =	5.00
f =	**0.00**
g =	4.03
h =	3.61
i =	4.61
j =	**1.12**

f is core point

From **g** to	
a =	3.04
b =	2.00
c =	2.00
d =	2.83
e =	2.50
f =	4.03
g =	**0.00**
h =	1.80
i =	**0.71**
j =	3.54

g is noise point

From **h** to	
a =	2.83
b =	1.80
c =	**1.12**
d =	**1.12**
e =	**1.41**
f =	3.61
g =	1.80
h =	**0.00**
i =	1.80
j =	2.69

h is core point

From **i** to	
a =	3.64
b =	2.55
c =	1.58
d =	2.92
e =	2.06
f =	4.61
g =	**0.71**
h =	1.80
i =	**0.00**
j =	4.00

i is noise point

From **j** to	
a =	**1.12**
b =	1.58
c =	3.81
d =	2.12
e =	4.03
f =	**1.12**
g =	3.54
h =	2.69
i =	4.00
j =	**0.00**

j is core point

- Point a, c, e, f, h, and j are core points, because these points have 3 ($MinPts$) or more neighbours.
- Points b and d are border points because these points share a neighbourhood with at least one core point whose distance is less than 1.5 (ε).
- Points g and j are noise points because these points do not share a neighbourhood with any core point whose distance is less than 1.5 (ε).

For step 2, treat each core point as the centre of group and the members of each group are their neighbours. Therefore, data members of each group are represented by the sets:

- Core point a, A = {b,f,j}
- Core point c, C = {e,h}
- Core point e, E = {c, h}
- Core point f, F = {a, j}
- Core point h, H = {c, d, e}
- Core point j, J = {a, f}

For step 3, merge each group together as a cluster if they have at least one overlap neighbour. Therefore,

- Cluster 1 = {a, b, f, j}
- Cluster 2 = {c, d, e, h}
- Noise = {g, i}

A.2 OPTICS clustering

The example below has 26-objects with two dimensions to test this algorithm and show how it works. Details of the data sample are given in Table A-1 and Figure A-2

Table A - 1: 26-objects of data sample with 2 dimensions

Object	*x*	*y*
a	1.0	3.5
b	4.5	6.5
c	1.5	7.5
d	6.5	4.0
e	7.5	3.0
f	2.0	4.0
g	9.0	8.0
h	1.5	2.5
i	2.0	7.0

Object	*x*	*y*
j	1.0	6.5
k	5.5	3.0
l	2.5	5.0
m	2.0	2.0
n	9.0	3.0
o	2.0	6.0
p	8.0	4.0
q	2.5	3.5
r	8.0	2.0

Object	*x*	*y*
s	1.0	5.0
t	1.5	6.5
u	1.5	3.5
v	6.5	2.0
w	9.0	1.5
x	1.5	6.0
y	6.5	3.0
z	2.0	3.0

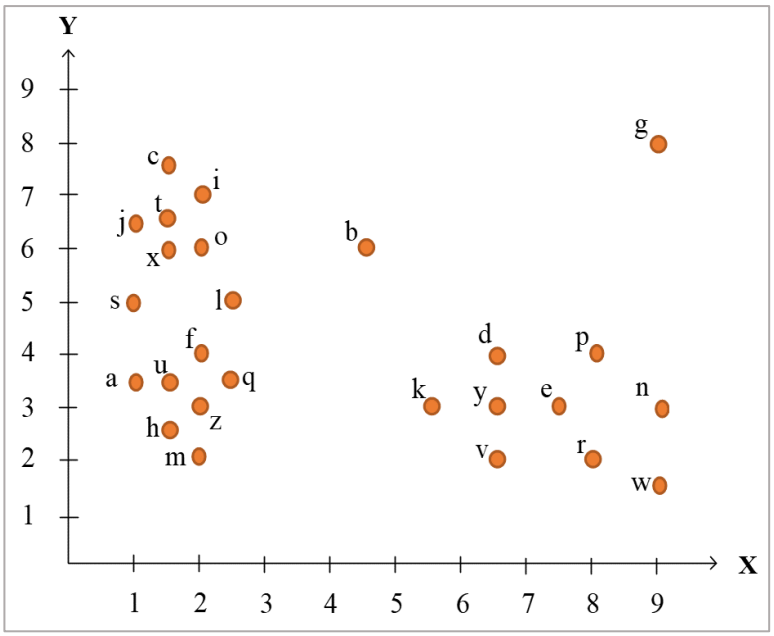


Figure A - 2 : Scatter plot of 26-objects data sample from Table A-1

Each iteration of running OPTICS with 26-object data sample is shown in Table A2.

- D is the Euclidian distance function, for example $D(a,obj)$ is the Euclidian distance between object a and object obj.
- N is the number of neighbours of an object
- *core* refers to core-point
- *c-dis* refers to core-distance of an object
- *r-dis* refers to reachability-distance of an object

The result of running OPTICS clustering algorithm and plotting reachability-distance for each object, with $\mathcal{E}$=1.50 and $MinPts$=4, is a clustering structure as in Figure A-3.

Table A - 2: OPTICS clustering demonstration

Iteration1		Iteration2		Iteration3		Iteration4		Iteration5		Iteration6		Iteration7		Iteration8		Iteration9		Iteration10		Iteration11		Iteration12		Iteration13	
a is processed		**u** is processed		**f** is processed		**z** is processed		**q** is processed		**h** is processed		**m** is processed		**l** is processed		**s** is processed		**o** is processed		**x** is processed		**i** is processed		**c** is processed	
obj	D(a,obj)	obj	D(u,obj)	obj	D(f,obj)	obj	D(z,obj)	obj	D(q,obj)	obj	D(h,obj)	obj	D(m,obj)	obj	D(l,obj)	obj	D(s,obj)	obj	D(o,obj)	obj	D(x,obj)	obj	D(i,obj)	obj	D(c,obj)
a	**0.00**	**a**	**0.50**	**a**	**1.12**	**a**	**1.12**	**a**	**1.50**	**a**	**1.12**	**a**	1.80	**a**	2.12	**a**	**1.50**	**a**	2.69	**a**	2.55	**a**	3.64	**a**	4.03
b	4.61	b	4.24	b	3.54	b	4.30	b	3.61	b	5.00	b	5.15	b	2.50	b	3.81	b	2.55	b	3.04	b	2.55	b	3.16
c	4.03	c	4.00	c	3.54	c	4.53	c	4.12	c	5.00	c	5.52	c	2.69	c	2.55	c	1.58	c	**1.50**	c	**0.71**	c	**0.00**
d	5.52	d	5.02	d	4.50	d	4.61	d	4.03	d	5.22	d	4.92	d	4.12	d	5.59	d	4.92	d	5.39	d	5.41	d	6.10
e	6.52	e	6.02	e	5.59	e	5.50	e	5.02	e	6.02	e	5.59	e	5.39	e	6.80	e	6.26	e	6.71	e	6.80	e	7.50
f	**1.12**	f	**0.71**	**f**	**0.00**	**f**	**1.00**	**f**	**0.71**	**f**	1.58	**f**	2.00	**f**	**1.12**	**f**	**1.41**	**f**	2.00	**f**	2.06	**f**	3.00	**f**	3.54
g	9.18	g	8.75	g	8.06	g	8.60	g	7.91	g	9.30	g	9.22	g	7.16	g	8.54	g	7.28	g	7.76	g	7.07	g	7.52
h	**1.12**	h	**1.00**	h	1.58	h	**0.71**	h	**1.41**	**h**	**0.00**	**h**	**0.71**	**h**	2.69	**h**	2.55	**h**	3.54	**h**	3.50	**h**	4.53	**h**	5.00
i	3.64	i	3.54	i	3.00	i	4.00	i	3.54	i	4.53	i	5.00	i	2.06	i	2.24	i	**1.00**	i	**1.12**	**i**	**0.00**	**i**	**0.71**
j	3.00	j	3.04	j	2.69	j	3.64	j	3.35	j	4.03	j	4.61	j	2.12	j	**1.50**	j	**1.12**	j	**0.71**	j	**1.12**	j	**1.12**
k	4.53	k	4.03	k	3.64	k	3.50	k	3.04	k	4.03	k	3.64	k	3.61	k	4.92	k	4.61	k	5.00	k	5.32	k	6.02
l	2.12	l	1.80	l	**1.12**	l	2.06	l	**1.50**	l	2.69	l	3.04	**l**	**0.00**	**l**	**1.50**	**l**	**1.12**	**l**	**1.41**	**l**	2.06	**l**	2.69
m	1.80	m	1.58	m	2.00	m	**1.00**	m	1.58	m	**0.71**	**m**	**0.00**	**m**	3.04	**m**	3.16	**m**	4.00	**m**	4.03	**m**	5.00	**m**	5.52
n	8.02	n	7.52	n	7.07	n	7.00	n	6.52	n	7.52	n	7.07	n	6.80	n	8.25	n	7.62	n	8.08	n	8.06	n	8.75
o	2.69	o	2.55	o	2.00	o	3.00	o	2.55	o	3.54	o	4.00	o	**1.12**	o	**1.41**	**o**	**0.00**	**o**	**0.50**	**o**	**1.00**	**o**	1.58
p	7.02	p	6.52	p	6.00	p	6.08	p	5.52	p	6.67	p	6.32	p	5.59	p	7.07	p	6.32	p	6.80	p	6.71	p	7.38
q	**1.50**	q	**1.00**	q	**0.71**	q	**0.71**	**q**	**0.00**	**q**	**1.41**	**q**	1.58	**q**	**1.50**	**q**	2.12	**q**	2.55	**q**	2.69	**q**	3.54	**q**	4.12
r	7.16	r	6.67	r	6.32	r	6.08	r	5.70	r	6.52	r	6.00	r	6.26	r	7.62	r	7.21	r	7.63	r	7.81	r	8.51
s	**1.50**	s	1.58	s	**1.41**	s	2.24	s	2.12	s	2.55	s	3.16	s	**1.50**	**s**	**0.00**	**s**	**1.41**	**s**	**1.12**	**s**	2.24	**s**	2.55
t	3.04	t	3.00	t	2.55	t	3.54	t	3.16	t	4.00	t	4.53	t	1.80	t	1.58	t	**0.71**	t	**0.50**	t	**0.71**	t	**1.00**
u	**0.50**	**u**	**0.00**	**u**	**0.71**	**u**	**0.71**	**u**	**1.00**	**u**	**1.00**	**u**	1.58	**u**	1.80	**u**	1.58	**u**	2.55	**u**	2.50	**u**	3.54	**u**	4.00
v	5.70	v	5.22	v	4.92	v	4.61	v	4.27	v	5.02	v	4.50	v	5.00	v	6.26	v	6.02	v	6.40	v	6.73	v	7.43
w	8.25	w	7.76	w	7.43	w	7.16	w	6.80	w	7.57	w	7.02	w	7.38	w	8.73	w	8.32	w	8.75	w	8.90	w	9.60
x	2.55	x	2.50	x	2.06	x	3.04	x	2.69	x	3.50	x	4.03	x	**1.41**	x	**1.12**	x	**0.50**	**x**	**0.00**	**x**	**1.12**	**x**	**1.50**
y	5.52	y	5.02	y	4.61	y	4.50	y	4.03	y	5.02	y	4.61	y	4.47	y	5.85	y	5.41	y	5.83	y	6.02	y	6.73
z	**1.12**	z	**0.71**	z	**1.00**	**z**	**0.00**	**z**	**0.71**	**z**	**0.71**	**z**	**1.00**	**z**	2.06	**z**	2.24	**z**	3.00	**z**	3.04	**z**	4.00	**z**	4.53
details of **a**		details of **u**		details of **f**		details of **z**		details of **q**		details of **h**		details of **m**		details of **l**		details of **s**		details of **o**		details of **x**		details of **i**		details of **c**	
N	7	N	6	N	7	N	7	N	7	N	6	N	3	N	6	N	7	N	7	N	8	N	6	N	5
core	Yes	core	Yes	core	Yes	core	Yes	core	Yes	core	Yes	core	No	core	Yes	core	Yes	core	Yes	core	Yes	core	Yes	core	Yes
c-dis	1.12	c-dis	0.71	c-dis	1.00	c-dis	0.71	c-dis	1.00	c-dis	1.00	c-dis	Inf	c-dis	1.41	c-dis	1.41	c-dis	1.00	c-dis	0.71	c-dis	1.00	c-dis	1.12
r-dis	Inf	r-dis	1.12	r-dis	0.71	r-dis	0.71	r-dis	0.71	r-dis	0.71	r-dis	0.71	r-dis	1.12	r-dis	1.41	r-dis	1.41	r-dis	1.00	r-dis	1.00	r-dis	1.00
OrderSeeds list		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**	
obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis
u	1.12	f	0.71	z	0.71	q	0.71	h	0.71	m	0.71	l	1.12	s	1.41	o	1.41	x	1.00	i	1.00	c	1.00	t	1.00
f	1.12	z	0.71	h	1.00	h	0.71	m	1.00	l	1.12	s	1.41	o	1.41	x	1.41	i	1.00	j	1.12	t	1.00	j	1.12
h	1.12	h	1.00	q	1.00	m	1.00	l	1.12	s	1.41			x	1.41	j	1.50	j	1.12	c	1.50	j	1.12		
z	1.12	q	1.00	l	1.12	l	1.12	s	1.41																
q	1.50	s	1.50	s	1.41	s	1.41																		
s	1.50																								

Iteration14		Iteration15		Iteration16		Iteration17		Iteration18		Iteration19		Iteration20		Iteration21		Iteration22		Iteration23		Iteration24		Iteration25		Iteration26	
t is processed		**j** is processed		**j** is processed		**d** is processed		**y** is processed		**e** is processed		**k** is processed		**v** is processed		**p** is processed		**r** is processed		**n** is processed		**w** is processed		**g** is processed	
obj	D(t,obj)	obj	D(j,obj)	obj	D(b,obj)	obj	D(d,obj)	obj	D(y,obj)	obj	D(e,obj)	obj	D(k,obj)	obj	D(v,obj)	obj	D(p,obj)	obj	D(r,obj)	obj	D(n,obj)	obj	D(w,obj)	obj	D(g,obj)
a	3.04	a	3.00	a	4.61	a	5.52	a	5.52	a	6.52	a	4.53	a	5.70	a	7.02	a	7.16	a	8.02	a	8.25	a	9.18
b	3.00	b	3.50	**b**	**0.00**	b	3.20	b	4.03	b	4.61	b	3.64	b	4.92	b	4.30	b	5.70	b	5.70	b	6.73	b	4.74
c	**1.00**	c	**1.12**	c	3.16	c	6.10	c	6.73	c	7.50	c	6.02	c	7.43	c	7.38	c	8.51	c	8.75	c	9.60	c	7.52
d	5.59	d	6.04	d	3.20	**d**	**0.00**	**d**	**1.00**	**d**	**1.41**	**d**	**1.41**	**d**	2.00	**d**	**1.50**	**d**	2.50	**d**	2.69	**d**	3.54	**d**	4.72
e	6.95	e	7.38	e	4.61	e	**1.41**	e	**1.00**	**e**	**0.00**	**e**	2.00	**e**	**1.41**	**e**	**1.12**	**e**	**1.12**	**e**	**1.50**	**e**	2.12	**e**	5.22
f	2.55	f	2.69	f	3.54	f	4.50	f	4.61	f	5.59	f	3.64	f	4.92	f	6.00	f	6.32	f	7.07	f	7.43	f	8.06
g	7.65	g	8.14	g	4.74	g	4.72	g	5.59	g	5.22	g	6.10	g	6.50	g	4.12	g	6.08	g	5.00	g	6.50	**g**	**0.00**
h	4.00	h	4.03	h	5.00	h	5.22	h	5.02	h	6.02	h	4.03	h	5.02	h	6.67	h	6.52	h	7.52	h	7.57	h	9.30
i	**0.71**	i	**1.12**	i	2.55	i	5.41	i	6.02	i	6.80	i	5.32	i	6.73	i	6.71	i	7.81	i	8.06	i	8.90	i	7.07
j	**0.50**	j	**0.00**	j	3.50	j	6.04	j	6.52	j	7.38	j	5.70	j	7.11	j	7.43	j	8.32	j	8.73	j	9.43	j	8.14
k	5.32	k	5.70	k	3.64	k	**1.41**	k	**1.00**	k	2.00	k	**0.00**	k	**1.41**	k	2.69	k	2.69	k	3.50	k	3.81	k	6.10
l	1.80	l	2.12	l	2.50	l	4.12	l	4.47	l	5.39	l	3.61	l	5.00	l	5.59	l	6.26	l	6.80	l	7.38	l	7.16
m	4.53	m	4.61	m	5.15	m	4.92	m	4.61	m	5.59	m	3.64	m	4.50	m	6.32	m	6.00	m	7.07	m	7.02	m	9.22
n	8.28	n	8.73	n	5.70	n	2.69	n	2.50	n	**1.50**	n	3.50	n	2.69	n	**1.41**	n	**1.41**	n	**0.00**	n	**1.50**	n	5.00
o	**0.71**	o	**1.12**	o	2.55	o	4.92	o	5.41	o	6.26	o	4.61	o	6.02	o	6.32	o	7.21	o	7.62	o	8.32	o	7.28
p	6.96	p	7.43	p	4.30	p	**1.50**	p	1.80	p	**1.12**	p	2.69	p	2.50	p	**0.00**	p	2.00	p	**1.41**	p	2.69	p	4.12
q	3.16	q	3.35	q	3.61	q	4.03	q	4.03	q	5.02	q	3.04	q	4.27	q	5.52	q	5.70	q	6.52	q	6.80	q	7.91
r	7.91	r	8.32	r	5.70	r	2.50	r	1.80	r	**1.12**	r	2.69	r	**1.50**	r	2.00	r	**0.00**	r	**1.41**	r	**1.12**	r	6.08
s	1.58	s	**1.50**	s	3.81	s	5.59	s	5.85	s	6.80	s	4.92	s	6.26	s	7.07	s	7.62	s	8.25	s	8.73	s	8.54
t	**0.00**	t	**0.50**	t	3.00	t	5.59	t	6.10	t	6.95	t	5.32	t	6.73	t	6.96	t	7.91	t	8.28	t	9.01	t	7.65
u	3.00	u	3.04	u	4.24	u	5.02	u	5.02	u	6.02	u	4.03	u	5.22	u	6.52	u	6.67	u	7.52	u	7.76	u	8.75
v	6.73	v	7.11	v	4.92	v	2.00	v	**1.00**	v	**1.41**	v	**1.41**	v	**0.00**	v	2.50	v	**1.50**	v	2.69	v	2.55	v	6.50
w	9.01	w	9.43	w	6.73	w	3.54	w	2.92	w	2.12	w	3.81	w	2.55	w	2.69	w	**1.12**	w	**1.50**	w	**0.00**	w	6.50
x	**0.50**	x	**0.71**	x	3.04	x	5.39	x	5.83	x	6.71	x	5.00	x	6.40	x	6.80	x	7.63	x	8.08	x	8.75	x	7.76
y	6.10	y	6.52	y	4.03	y	**1.00**	y	**0.00**	y	**1.00**	y	**1.00**	y	**1.00**	y	1.80	y	1.80	y	2.50	y	2.92	y	5.59
z	3.54	z	3.64	z	4.30	z	4.61	z	4.50	z	5.50	z	3.50	z	4.61	z	6.08	z	6.08	z	7.00	z	7.16	z	8.60
details of **t**		details of **j**		details of **b**		details of **d**		details of **y**		details of **e**		details of **k**		details of **v**		details of **p**		details of **r**		details of **n**		details of **w**		details of **g**	
N	6	N	7	N	1	N	5	N	5	N	7	N	4	N	5	N	4	N	5	N	5	N	3	N	1
core	Yes	core	Yes	core	No	core	Yes	core	Yes	core	Yes	core	Yes	core	Yes	core	Yes	core	Yes	core	Yes	core	No	core	No
c-dis	0.71	c-dis	1.12	c-dis	Inf	c-dis	1.41	c-dis	1.00	c-dis	1.12	c-dis	1.41	c-dis	1.41	c-dis	1.50	c-dis	1.41	c-dis	1.50	c-dis	Inf	c-dis	Inf
r-dis	1.00	r-dis	0.71	r-dis	Inf	r-dis	Inf	r-dis	1.41	r-dis	1.00	r-dis	1.00	r-dis	1.00	r-dis	1.12	r-dis	1.12	r-dis	1.41	r-dis	1.41	r-dis	Inf
OrderSeeds list		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**		**OrderSeeds list**	
obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis	obj	r-dis
j	0.71	empty		empty		y	1.41	e	1.00	k	1.00	v	1.00	p	1.12	r	1.12	n	1.41	w	1.41	empty		empty	
						e	1.41	k	1.00	v	1.00	p	1.12	r	1.12	n	1.50	w	1.41						
						k	1.41	v	1.00	p	1.12	r	1.12												
						p	1.50	p	1.50	r	1.12														

As can be seen in Figure A3, objects belonging to a cluster have a low reachability-distance to their closest core-point. A valley corresponds to a cluster, giving cluster A that contains two sub-clusters (cluster A1 and cluster A2), and cluster B. For identification of the density threshold, if ε' is defined lower than 1.50, such as 1.25, the result would be consisting of three clusters with no sub-clusters and the number of noise objects will increase. Therefore, the deeper of the valley, the denser the cluster.

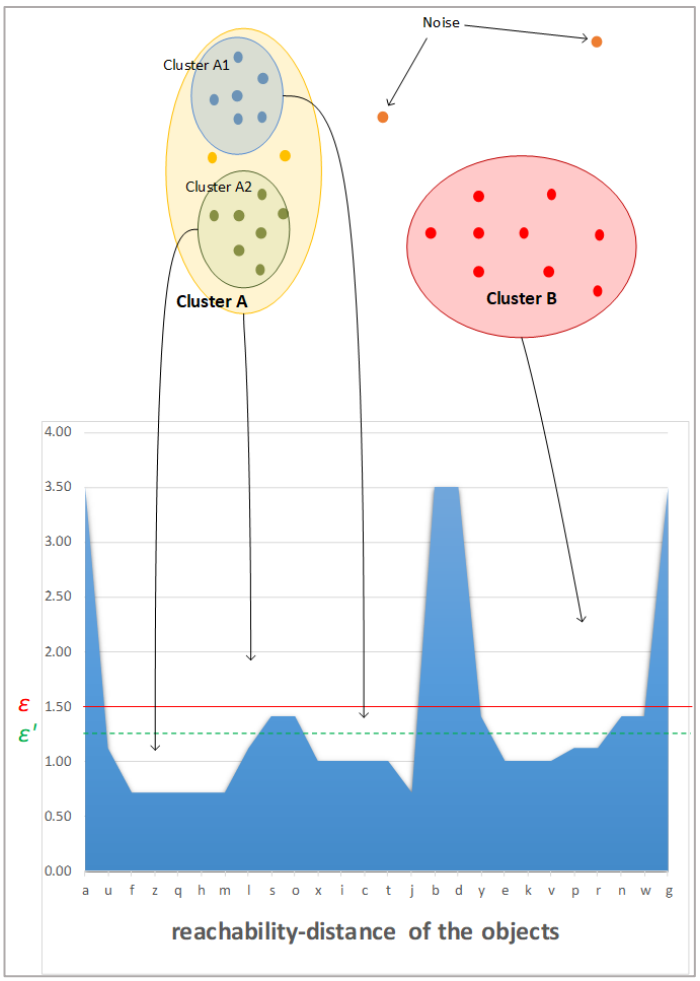


Figure A - 3 : Plot of reachability-distance, for each object, from Table A-2

A.3 FlowGrid automated gating

The example given in Figure A4 uses FlowGrid with 2-dimensional dataset, whose features are explained below.

- Figure A4-i, the scatter plot of the 2D data example
- Figure A4-ii, N_{bin} is defined as 15
- Figure A4-iii, identify core bin by consideration of parameter $MinDen_b$
    - Suppose that $MinDen_b$ is defined as 4. The cells with density larger than 4 are then considered as core bins.

- The example of finding directly connected bin, suppose that ε is defined as $\sqrt{2}$, so *Bin1*, which is labelled by *C(3,10*), *(it is located at the third bin of dimension one (x-axis) and the tenth bin of dimension two (y-axis))* is directly connected to *Bin2*, which is labelled by *C(4,9)*, because $\sqrt{(3-4)^2+(10-9)^2} \leq \sqrt{2}$. *Bin2* is not directly connected to $Bin3$ because $\sqrt{(4-5)^2+(9-7)^2} > \sqrt{2}$
- All of the red coloured bins are considered as core bins.

- Figure A4-iv, identify core bins by consideration of parameter ρ
  - Suppose that ρ is equal to 80. If $Den_b(Bin_i)$ larger than 80% of the number of all its directly connected bins, Bin_i is identified as a core bin. Figure B5(ii) shows all directly connected to *Bin4*, and the number of data points in *Bin4* is more than the density of bin number 1, 2, 3, 4, 6, 7, and 8 (seven bins). 80% of 8 (the number of all directly connected bins of *Bin4*) is 6.4, so *Bin4* is identified as a core bin because 7 is larger than 6.4.
  - All of the blue coloured bins are considered core bins.
- Figure A4-v, identify core bin by consideration of parameter $MinDen_c$
  - For this example, $MinDen_c$= 30 so Bin_i is a core bin, if summation of data points in all of its directly connected bins is more than 30. Figure A5(iii) shows that *Bin5* is core bin because $Den_c(Bin5)$ = 35, which is more than 30.
  - All of the green coloured bins are considered as core bins.
- Figure A4-vi, all of the yellow coloured bins are considered as border bins because their density is not more than 4 and are directly connected by at least one core bin.
- Figure A4-vii, all of the grey coloured bins are considered as noise bins because they are neither core bin nor border bin.

Appendix A Clustering demonstration

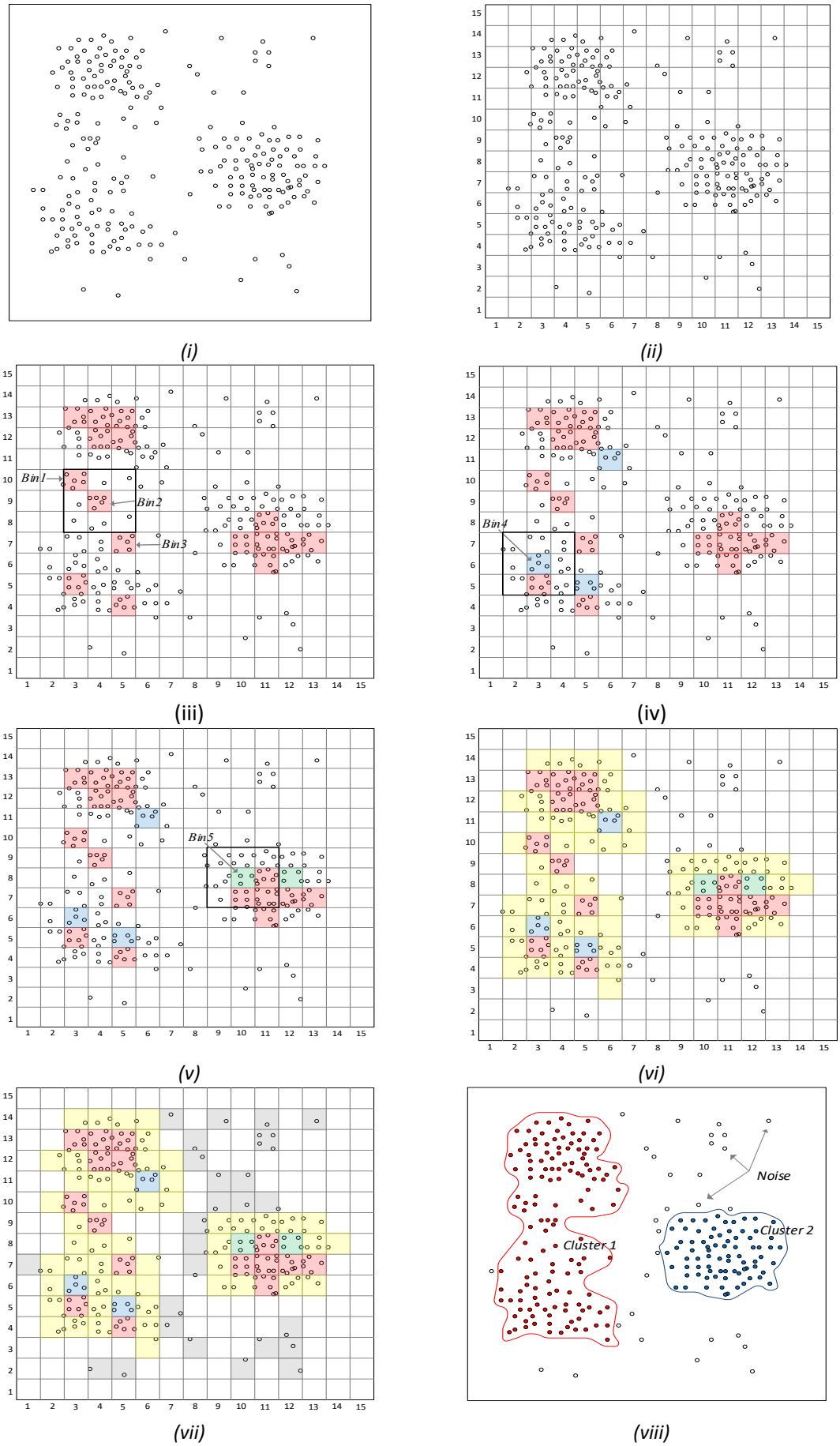


Figure A - 4 : Example of FlowGrid applied to 2-dimensional data

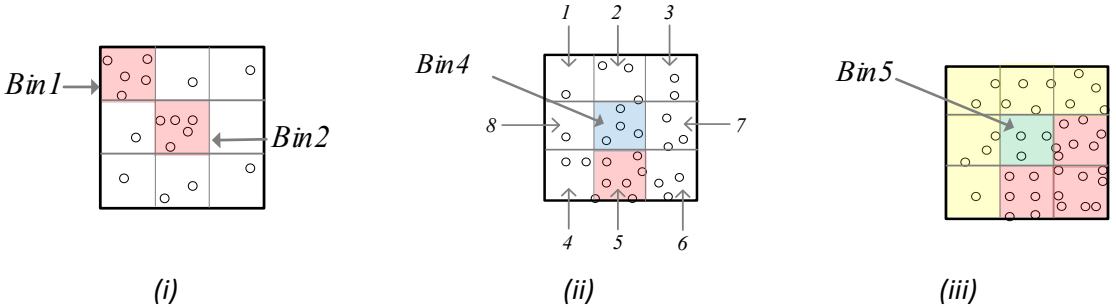
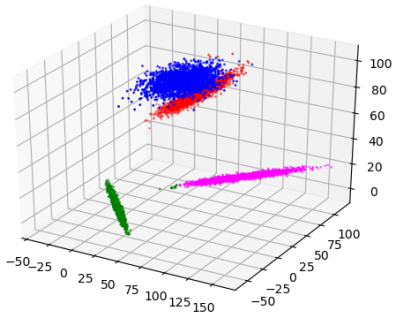


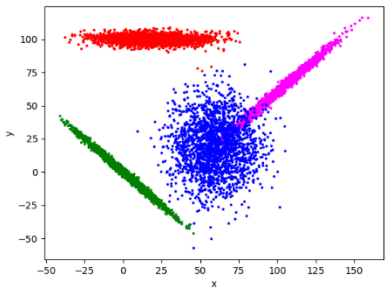
Figure A - 5 : Consideration core bin for the previous example in Figure A-4

Appendix B Figures show the result of 3D experiment

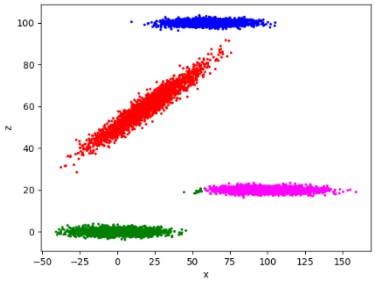
B.1 The result of K-means



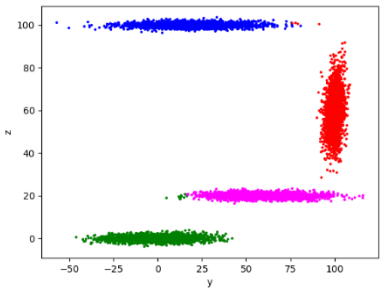
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

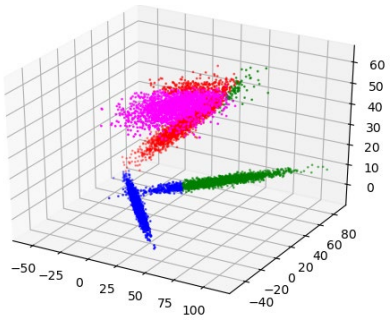


(iii) The coordinates (x,z) of a point in 3D

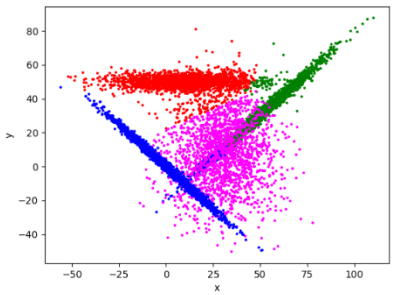


(iv) The coordinates (y,z) of a point in 2D

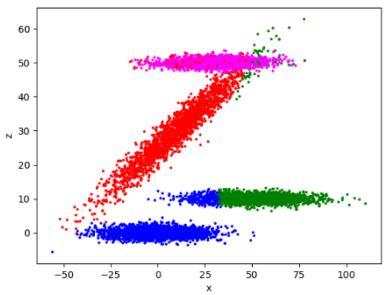
Figure B - 1: Applying K-means to Dataset E-1



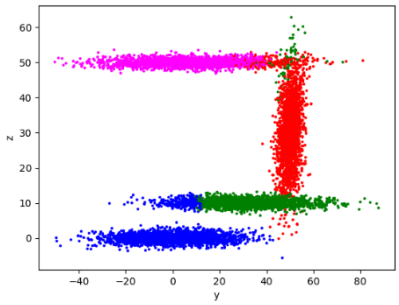
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



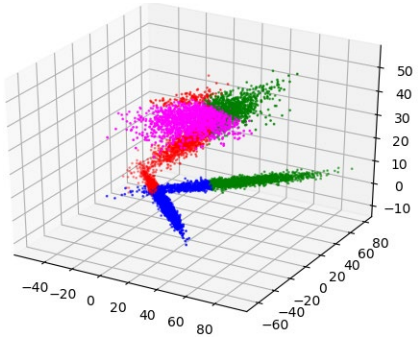
(iii) The coordinates (x,z) of a point in 3D



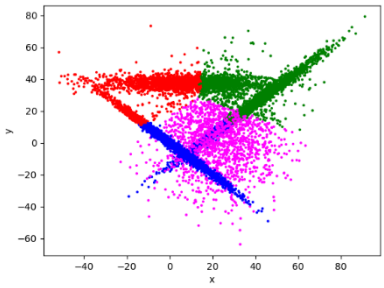
(iv) The coordinates (y,z) of a point in 2D

Figure B - 2: Applying K-means to Dataset F-1

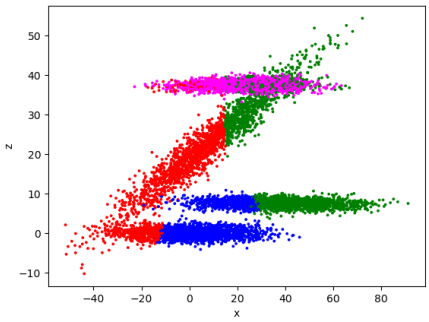
Appendix B Figures show the result of 3D experiment



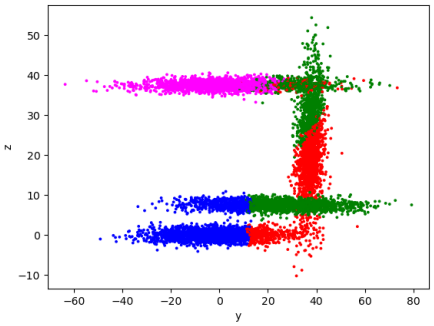
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

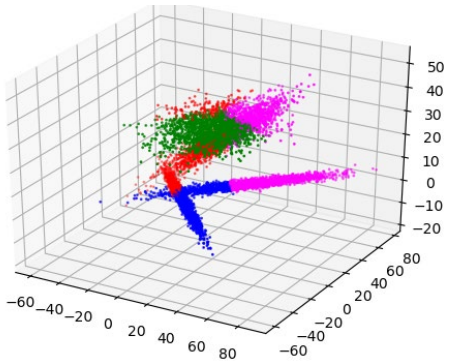


(iii) The coordinates (x,z) of a point in 3D

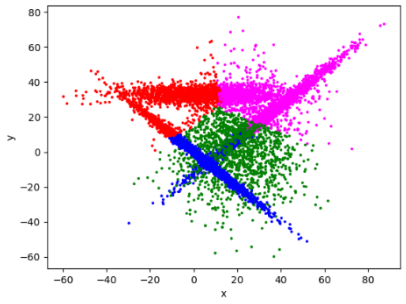


(iv) The coordinates (y,z) of a point in 2D

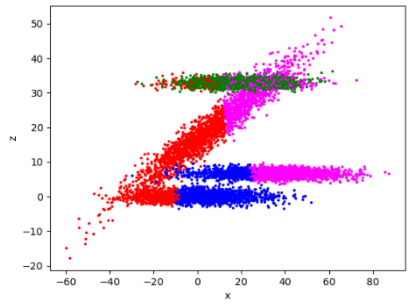
Figure B - 3: Applying K-means to Dataset G-1



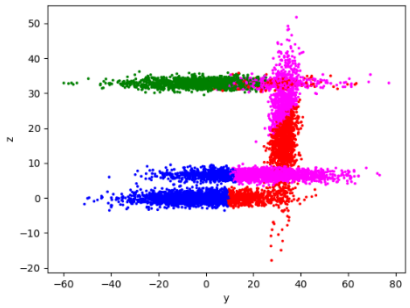
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



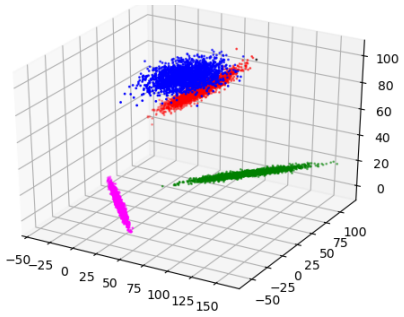
(iii) The coordinates (x,z) of a point in 3D



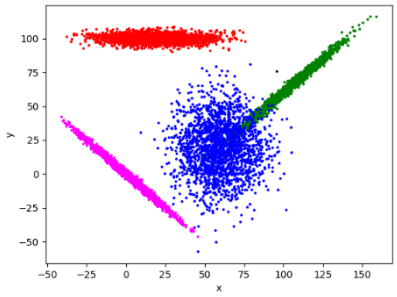
(iv) The coordinates (y,z) of a point in 2D

Figure B - 4: Applying K-means to Dataset H-1

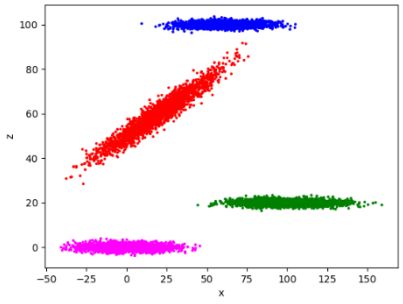
B.2 The result of DBSCAN



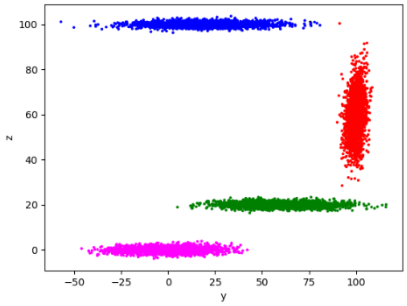
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

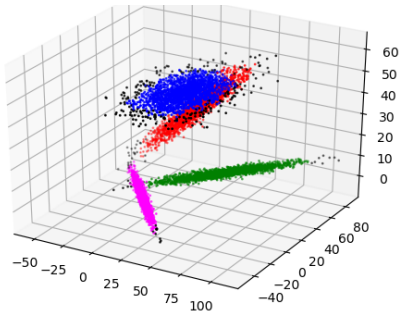


(iii) The coordinates (x,z) of a point in 3D

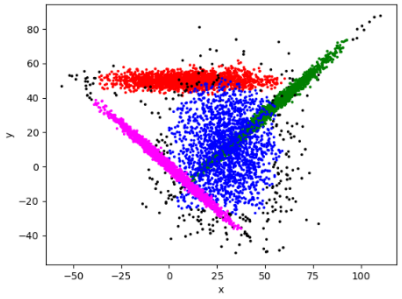


(iv) The coordinates (y,z) of a point in 2D

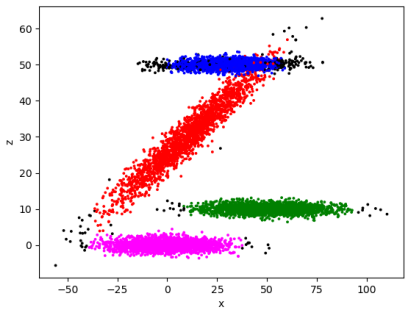
Figure B - 5: Applying DBSCAN to Dataset E-1



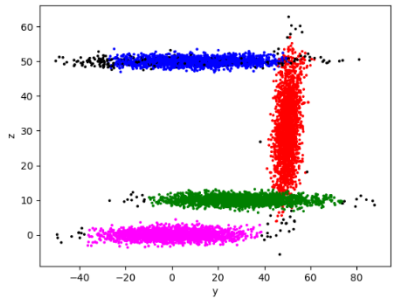
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



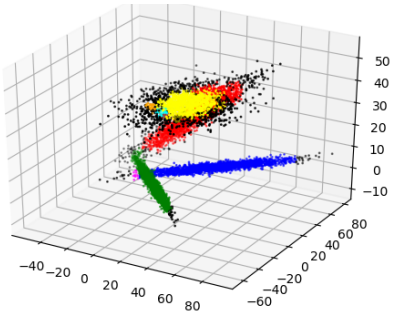
(iii) The coordinates (x,z) of a point in 3D



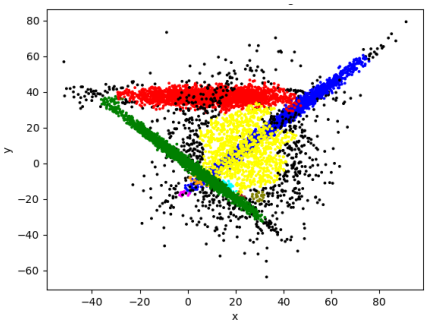
(iv) The coordinates (y,z) of a point in 2D

Figure B - 6: Applying DBSCAN to Dataset F-1

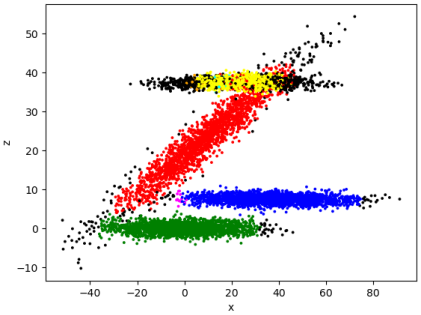
Appendix B Figures show the result of 3D experiment



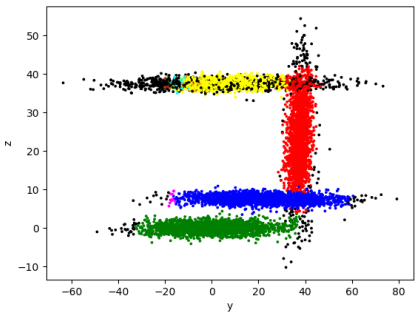
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

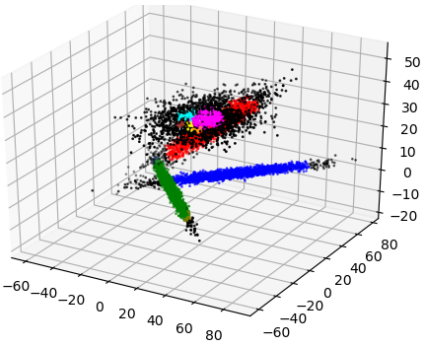


(iii) The coordinates (x,z) of a point in 3D

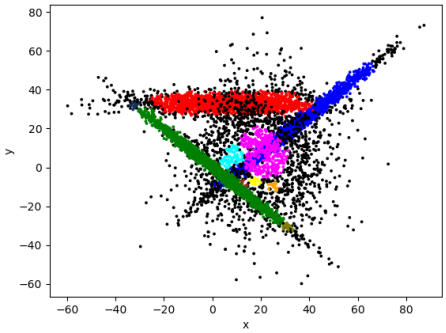


(iv) The coordinates (y,z) of a point in 2D

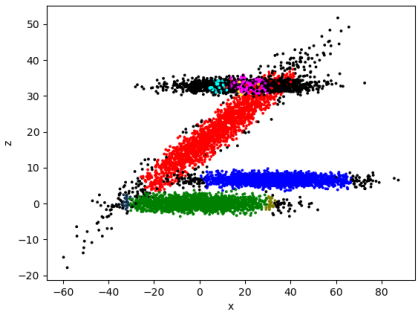
Figure B - 7: Applying DBSCAN to Dataset G-1



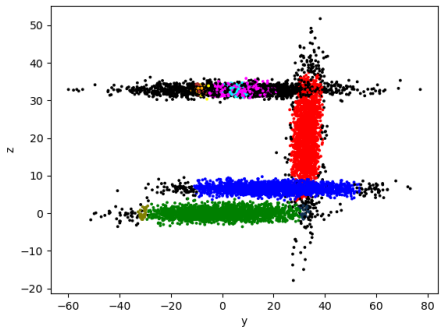
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



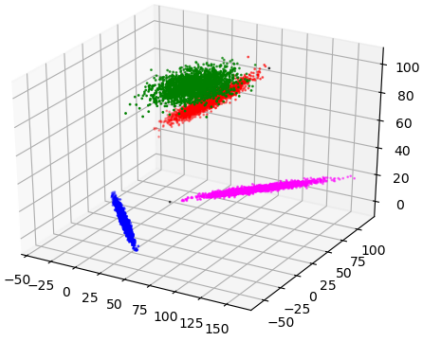
(iii) The coordinates (x,z) of a point in 3D



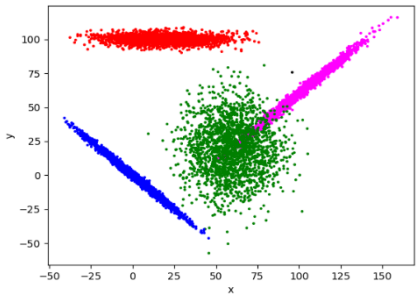
(iv) The coordinates (y,z) of a point in 2D

Figure B - 8: Applying DBSCAN to Dataset H-1

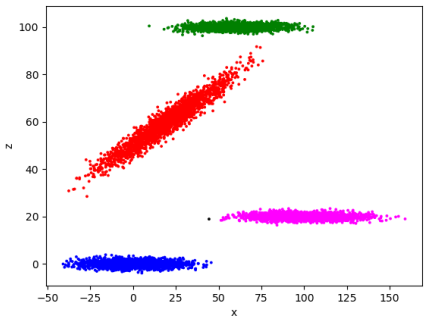
B.3 The result of OPTICS



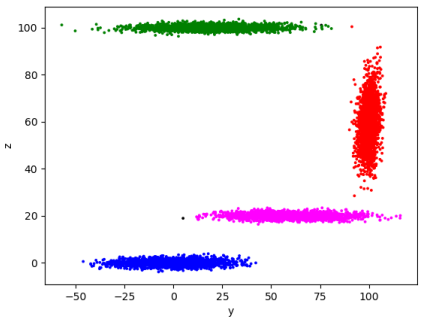
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

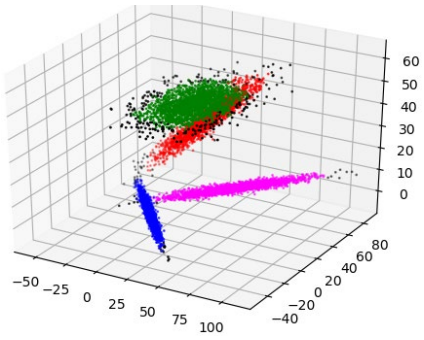


(iii) The coordinates (x,z) of a point in 3D

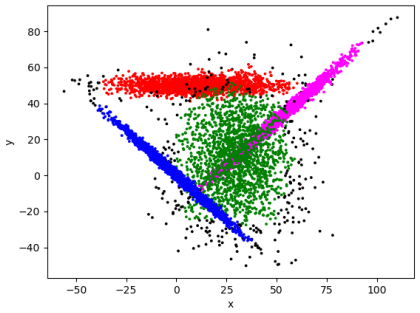


(iv) The coordinates (y,z) of a point in 2D

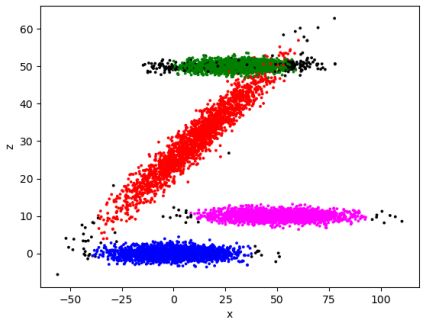
Figure B - 9: Applying OPTICS to Dataset E-1



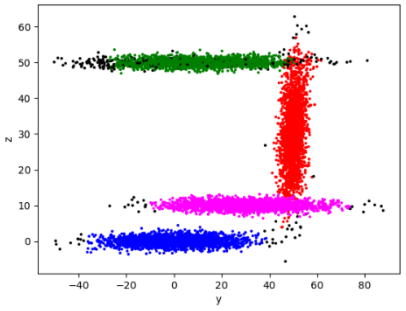
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



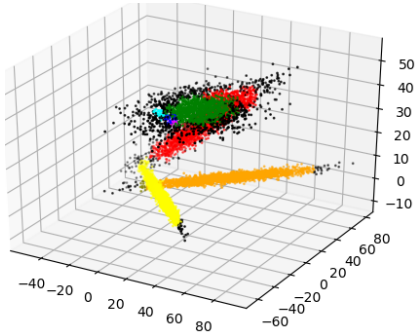
(iii) The coordinates (x,z) of a point in 3D



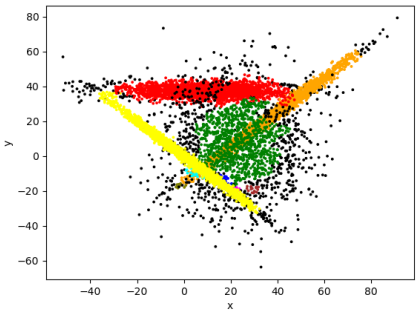
(iv) The coordinates (y,z) of a point in 2D

Figure B - 10: Applying OPTICS to Dataset F-1

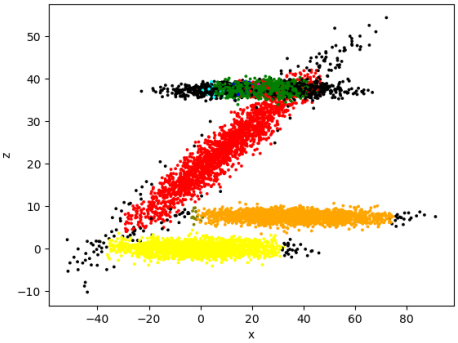
Appendix B Figures show the result of 3D experiment



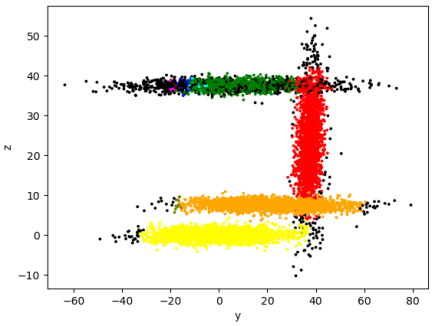
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

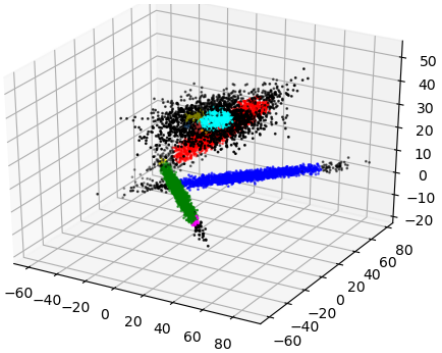


(iii) The coordinates (x,z) of a point in 3D

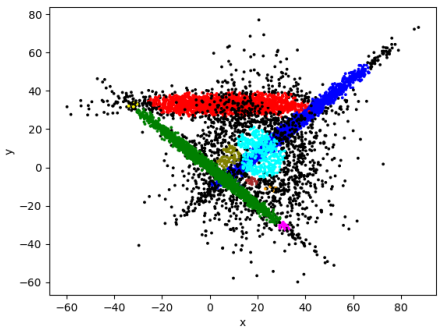


(iv) The coordinates (y,z) of a point in 2D

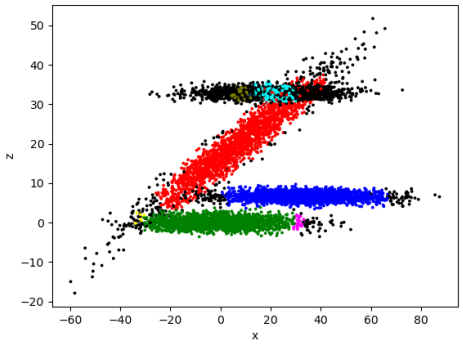
Figure B - 11: Applying OPTICS to Dataset G-1



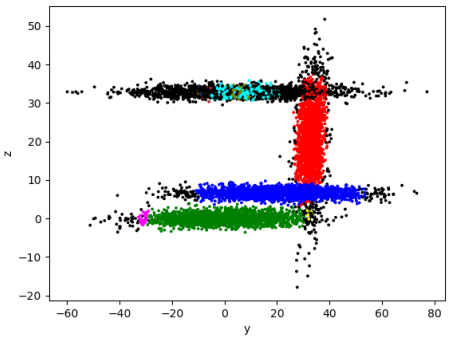
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



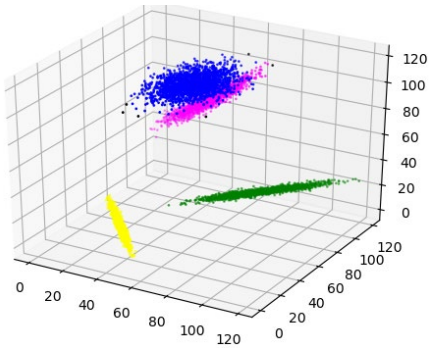
(iii) The coordinates (x,z) of a point in 3D



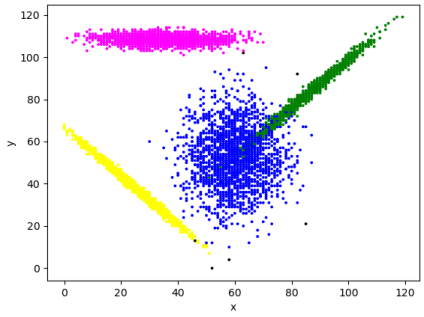
(iv) The coordinates (y,z) of a point in 2D

Figure B - 12: Applying OPTICS to Dataset H-1

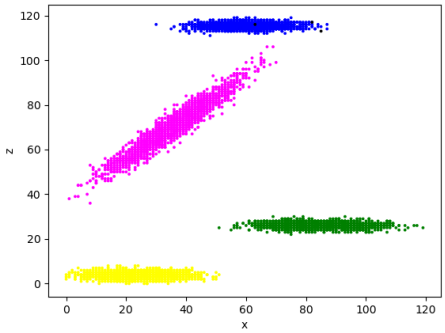
B.4 The result of FlowGrid (MinDenC = 20)



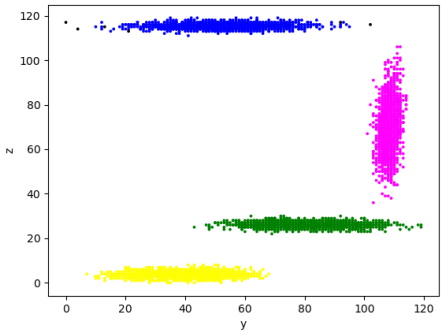
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

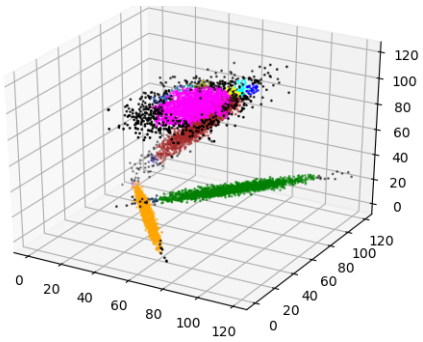


(iii) The coordinates (x,z) of a point in 3D

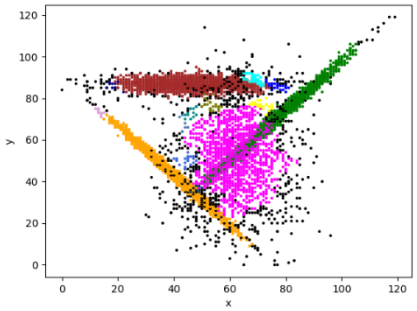


(iv) The coordinates (y,z) of a point in 2D

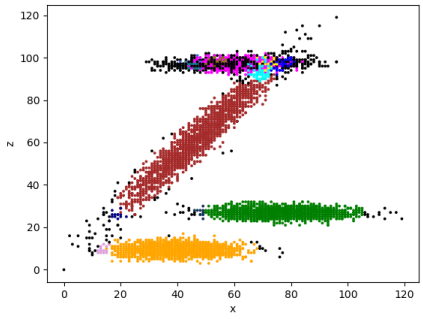
Figure B - 13: Applying FlowGrid to Dataset E-1 (MinDenC = 20)



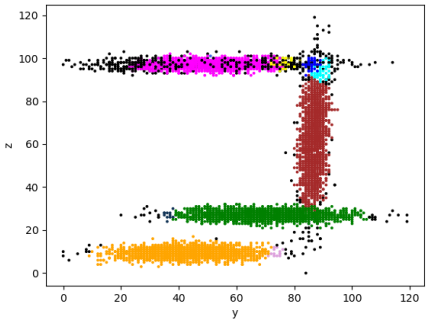
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



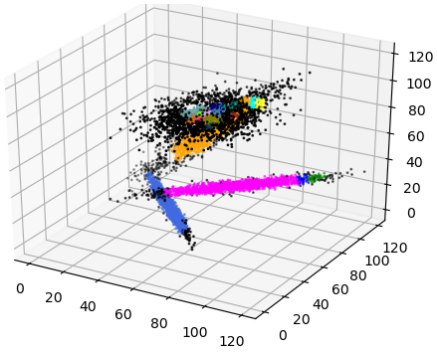
(iii) The coordinates (x,z) of a point in 3D



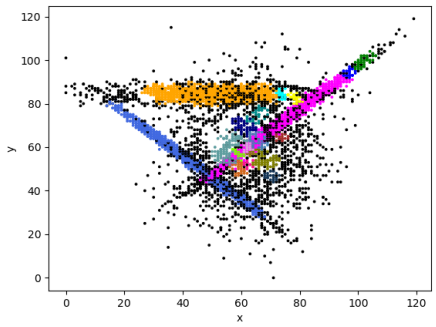
(iv) The coordinates (y,z) of a point in 2D

Figure B - 14: Applying FlowGrid to Dataset F-1 (MinDenC = 20)

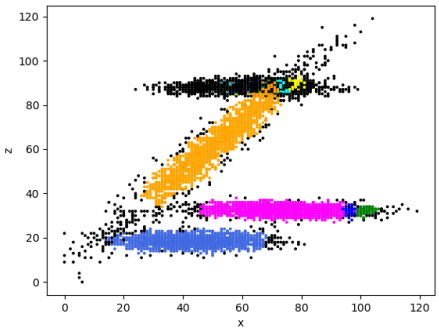
Appendix B Figures show the result of 3D experiment



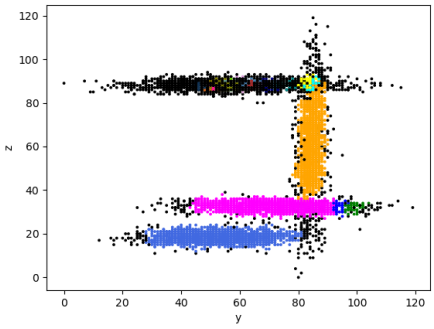
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

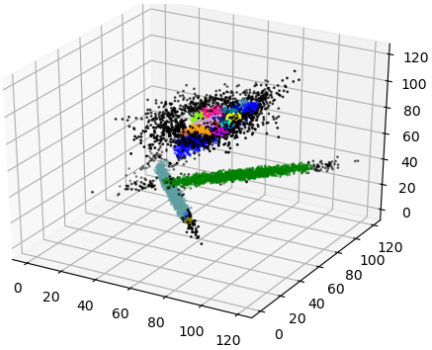


(iii) The coordinates (x,z) of a point in 3D

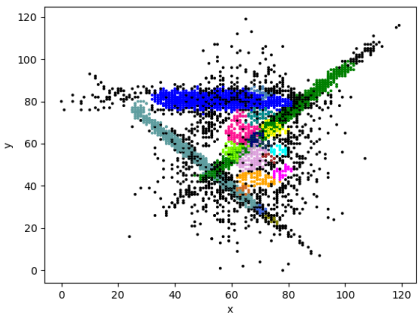


(iv) The coordinates (y,z) of a point in 2D

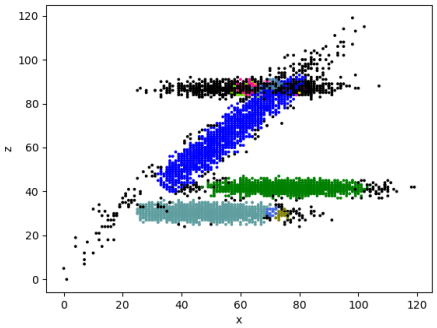
Figure B - 15: Applying FlowGrid to Dataset G-1 (MinDenC = 20)



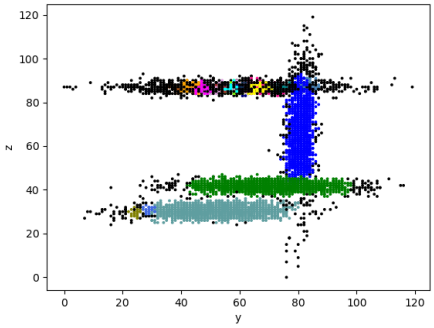
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



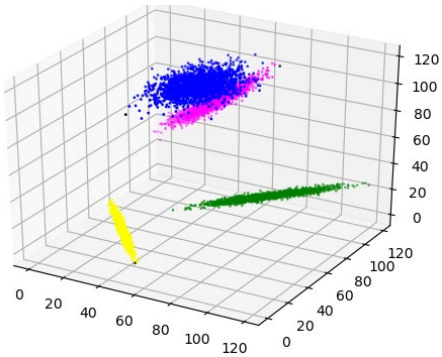
(iii) The coordinates (x,z) of a point in 3D



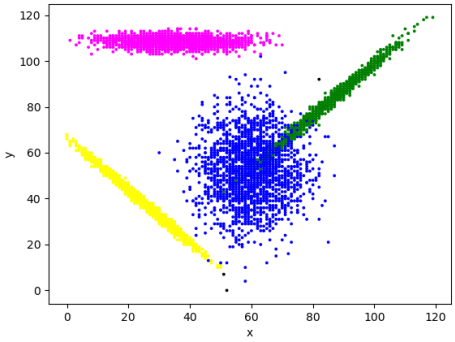
(iv) The coordinates (y,z) of a point in 2D

Figure B - 16: Applying FlowGrid to Dataset H-1 (MinDenC = 20)

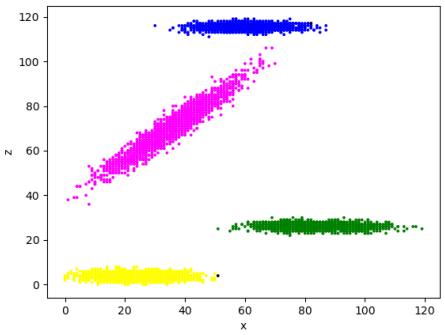
B.5 The result of FlowGrid (MinDenC = 100)



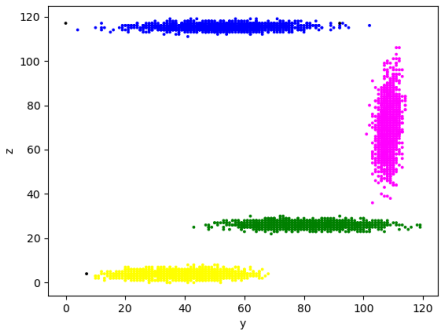
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

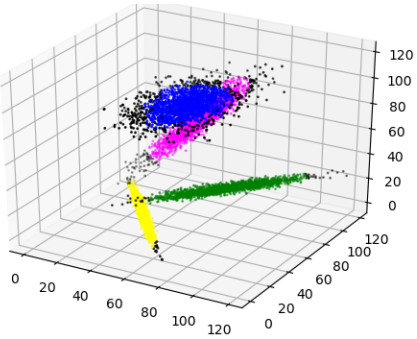


(iii) The coordinates (x,z) of a point in 3D

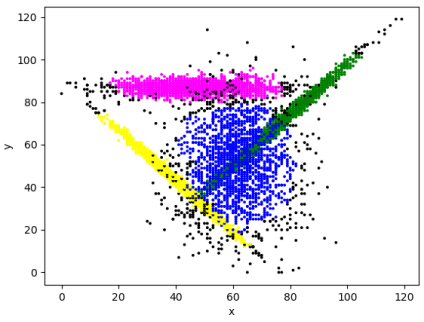


(iv) The coordinates (y,z) of a point in 2D

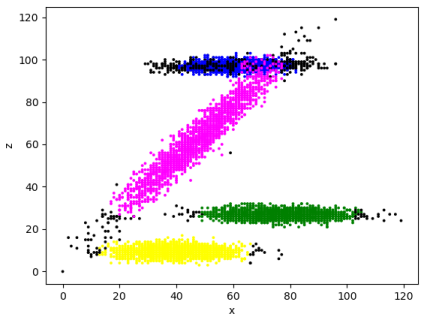
Figure B - 17: Applying FlowGrid to Dataset E-1 (MinDenC = 100)



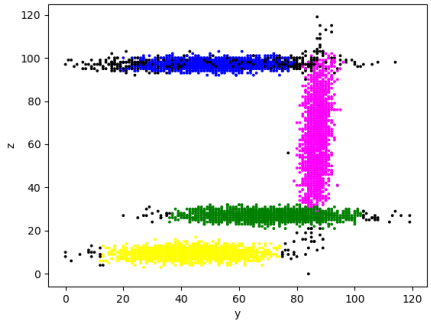
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



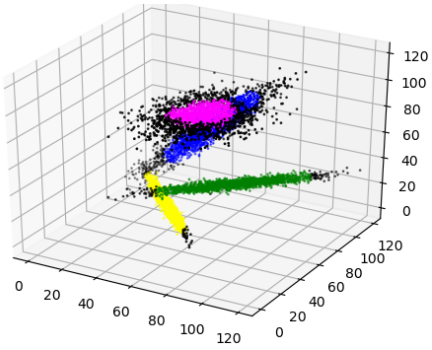
(iii) The coordinates (x,z) of a point in 3D



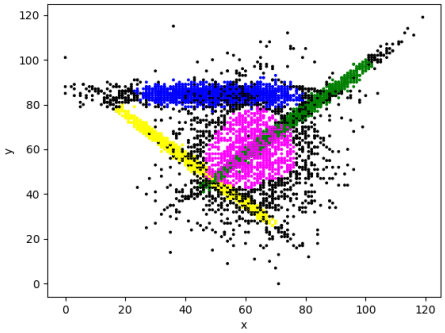
(iv) The coordinates (y,z) of a point in 2D

Figure B - 18: Applying FlowGrid to Dataset F-1 (MinDenC = 100)

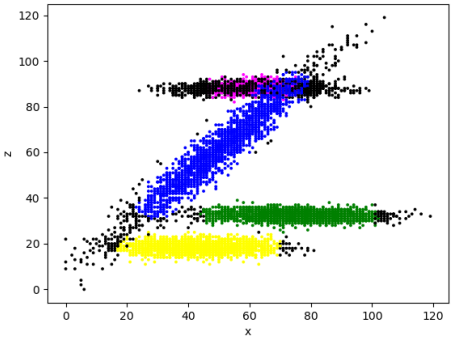
Appendix B Figures show the result of 3D experiment



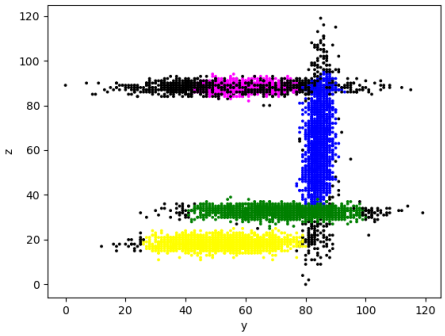
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

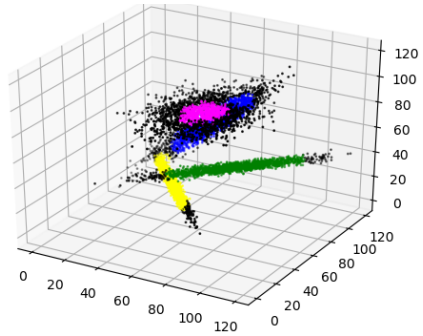


(iii) The coordinates (x,z) of a point in 3D

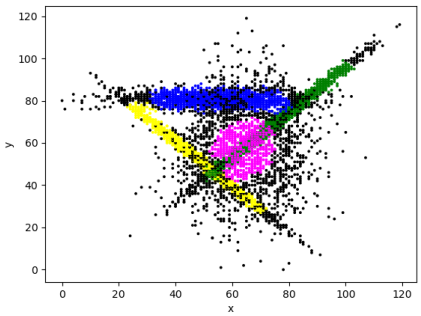


(iv) The coordinates (y,z) of a point in 2D

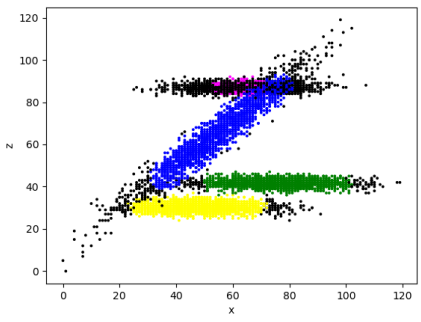
Figure B - 19: Applying FlowGrid to Dataset G-1 (MinDenC = 100)



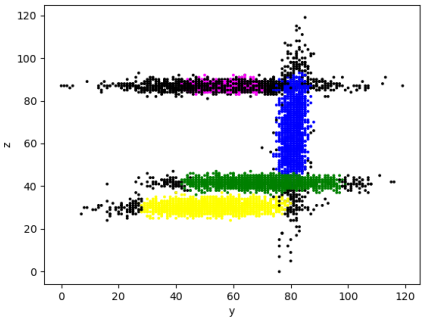
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



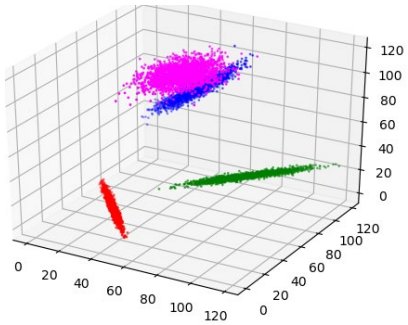
(iii) The coordinates (x,z) of a point in 3D



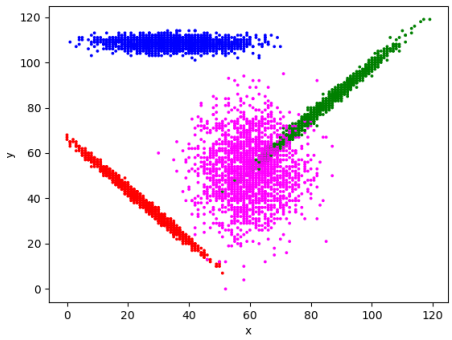
(iv) The coordinates (y,z) of a point in 2D

Figure B - 20: Applying FlowGrid to Dataset H-1 (MinDenC = 100)

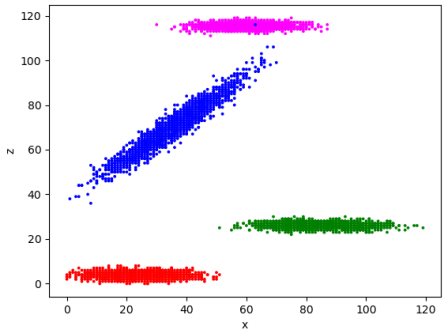
B.6 The result of FLOPTICS (MinPts = 20)



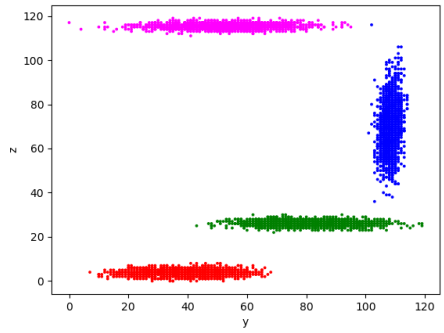
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

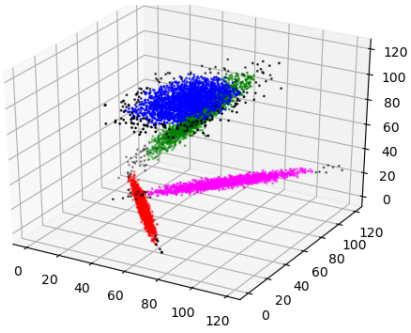


(iii) The coordinates (x,z) of a point in 3D

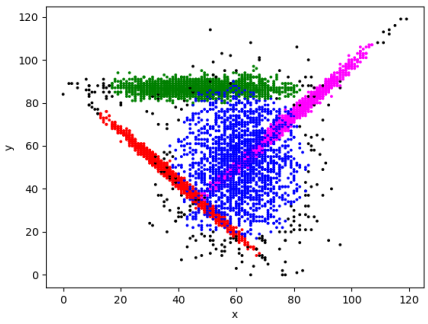


(iv) The coordinates (y,z) of a point in 2D

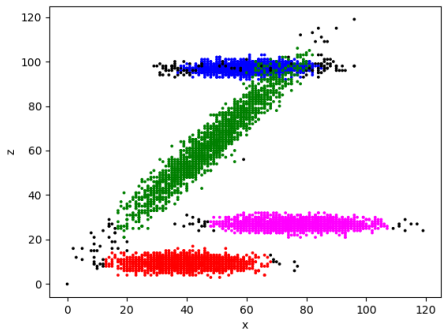
Figure B - 21: Applying FLOPTICS to Dataset E-1 (MinPts = 20)



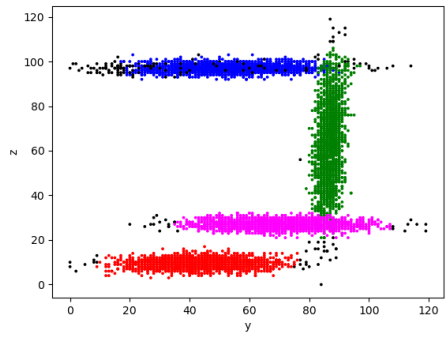
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



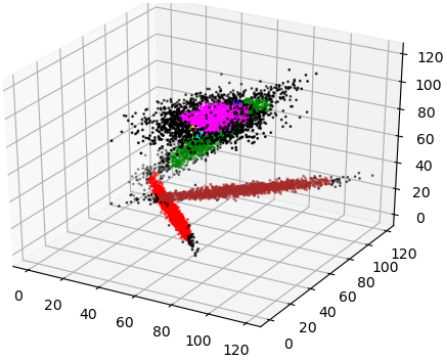
(iii) The coordinates (x,z) of a point in 3D



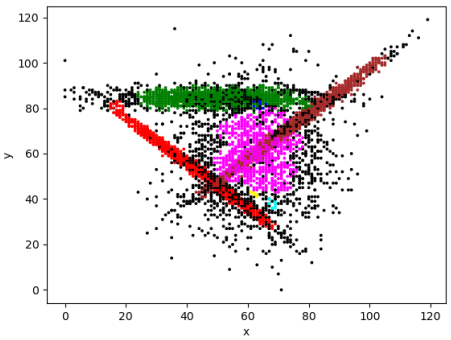
(iv) The coordinates (y,z) of a point in 2D

Figure B - 22: Applying FLOPTICS to Dataset F-1 (MinDenC = 20)

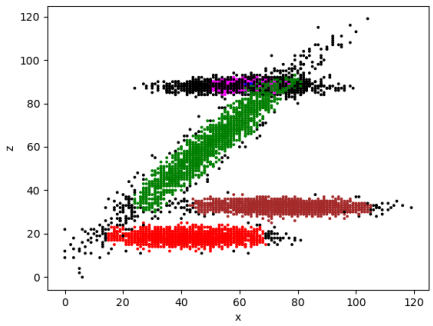
Appendix B Figures show the result of 3D experiment



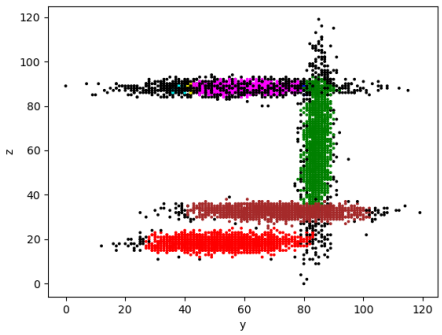
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

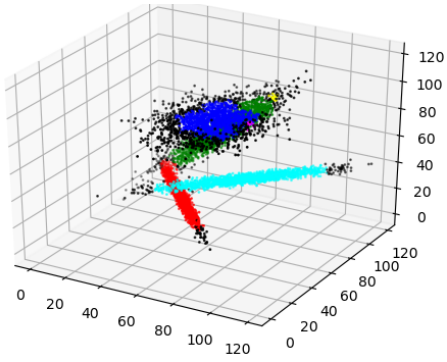


(iii) The coordinates (x,z) of a point in 3D

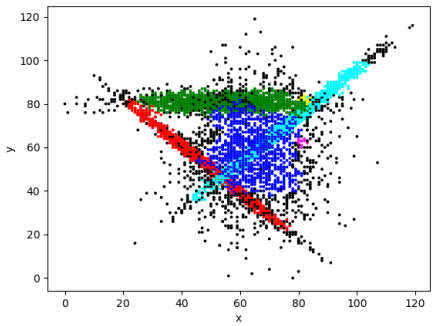


(iv) The coordinates (y,z) of a point in 2D

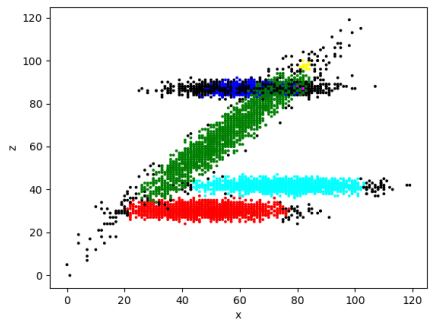
Figure B - 23: Applying FLOPTICS to Dataset G-1 (MinDenC = 20)



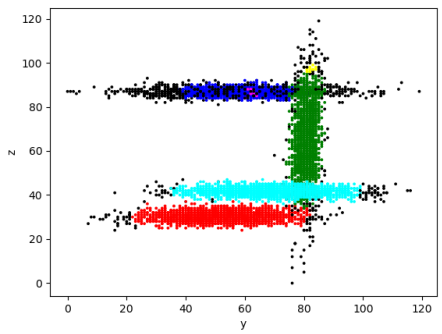
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



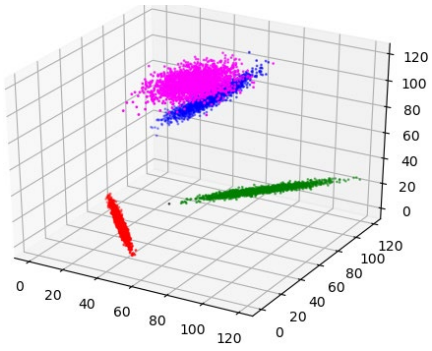
(iii) The coordinates (x,z) of a point in 3D



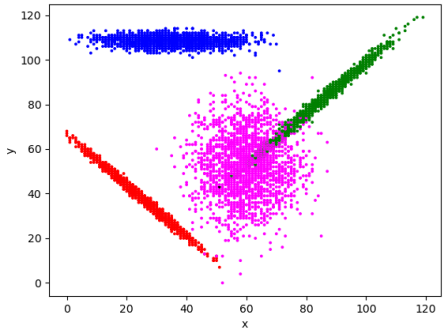
(iv) The coordinates (y,z) of a point in 2D

Figure B - 24: Applying FLOPTICS to Dataset H-1 (MinDenC = 20)

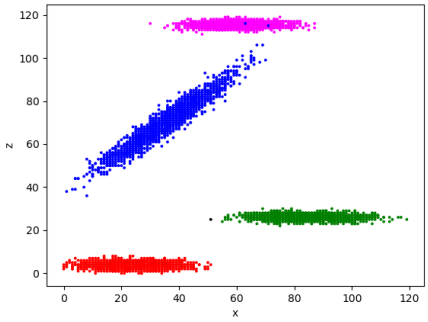
B.7 The result of FLOPTICS (MinPts = 100)



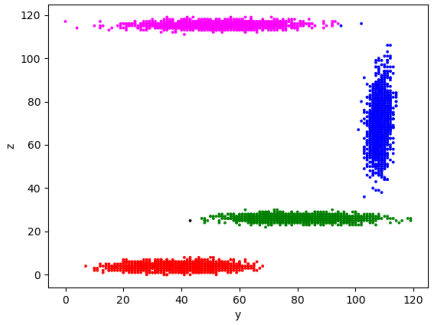
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

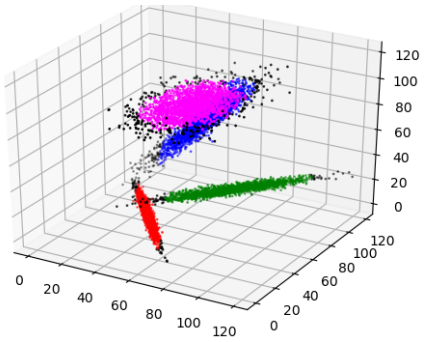


(iii) The coordinates (x,z) of a point in 3D

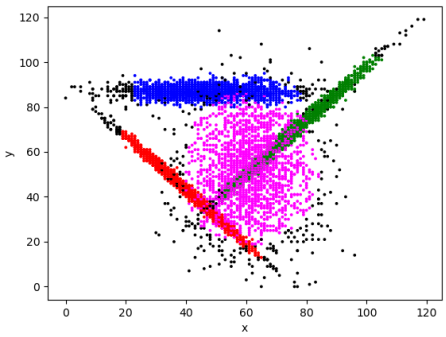


(iv) The coordinates (y,z) of a point in 2D

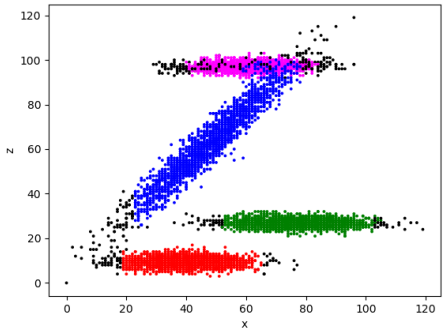
Figure B - 25: Applying FLOPTICS to Dataset E-1 (MinPts = 100)



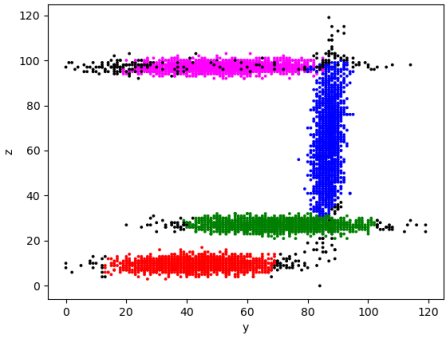
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



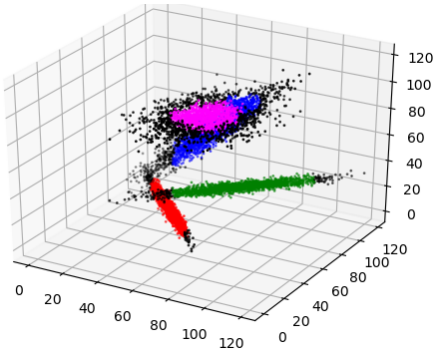
(iii) The coordinates (x,z) of a point in 3D



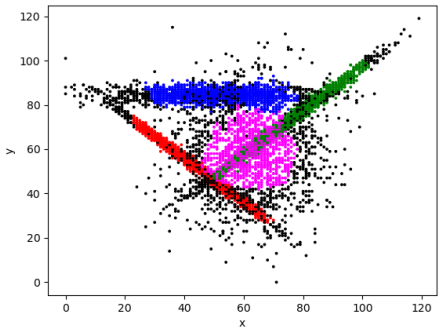
(iv) The coordinates (y,z) of a point in 2D

Figure B - 26: Applying FLOPTICS to Dataset F-1 (MinDenC = 100)

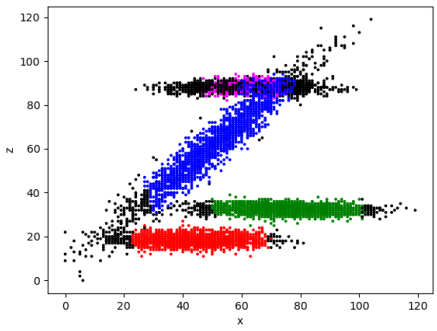
Appendix B Figures show the result of 3D experiment



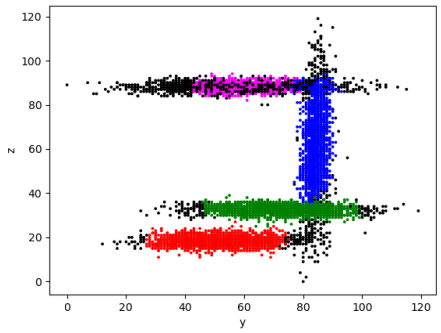
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D

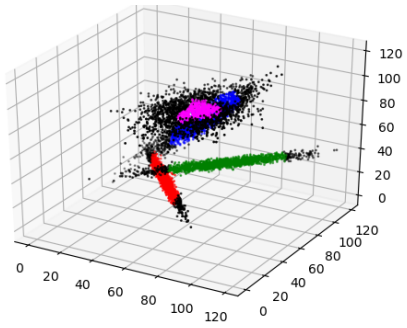


(iii) The coordinates (x,z) of a point in 3D

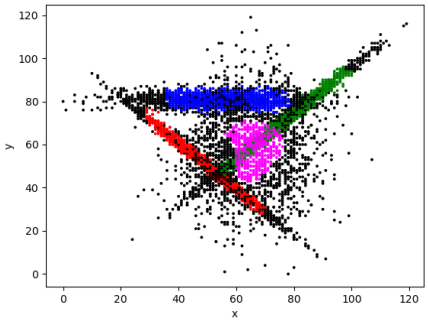


(iv) The coordinates (y,z) of a point in 2D

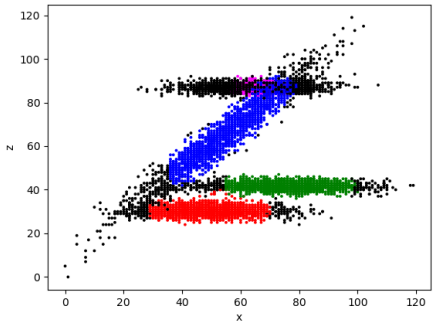
Figure B - 27: Applying FLOPTICS to Dataset G-1 (MinDenC = 100)



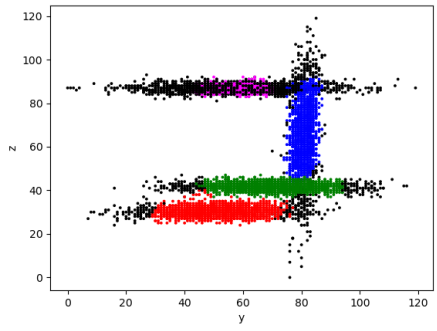
(i) The coordinates (x,y,z) of a point in 3D



(ii) The coordinates (x,y) of a point in 2D



(iii) The coordinates (x,z) of a point in 3D



(iv) The coordinates (y,z) of a point in 2D

Figure B - 28: Applying FLOPTICS to Dataset H-1 (MinDenC = 100)

References

abcam (2019) *Human CD antigen chart*, *Abcam plc.* Available at: https://www.abcam.com/primary-antibodies/human-cd-antigen-guide (Accessed: 23 May 2019).

Aghaeepour, N. *et al.* (2010) 'Rapid Cell Population Identification in Flow Cytometry Data', *International Society for Advancement of Cytometry*, 79(1), pp. 6–13.

Akaike, H. (1974) 'A New Look at the Statistical Model Identification', *IEEE Transactions on Automatic Control*, AC-19(6), pp. 716–723.

Albert, B. *et al.* (2002) *Molecular Biology of the Cell*. 4th edn. Garland Science. Available at: https://www.ncbi.nlm.nih.gov/books/NBK26919/table/A4143/ (Accessed: 1 April 2019).

Alexander, T. S. (2016) 'Human Immunodeficiency Virus Diagnostic Testing: 30 Years of Evolution', *Clinical and Vaccine Immunology*, 23(4), pp. 249–253.

Ankerst, M. *et al.* (1999) 'OPTICS: Ordering Points To Identify the Clustering Structure', in *Proc. ACM SIGMOD'99 Int. Conf. on Management of Data*. Philadelphia.

ASH (2009) *Blood Basics*, *American Society of Hematology*. Available at: https://www.hematology.org/Patients/Basics/ (Accessed: 1 April 2019).

Bashashati, A. and Brinkman, R. R. (2009) 'A Survey of Flow Cytometry Data Analysis Methods', *Advances in Bioinformatics*, 2009, pp. 1–19.

Baudry, J. *et al.* (2008) 'Combining Mixture Components for Clustering', *Journal of Computational and Graphical Statistics*, 19(2), pp. 332–353.

Bernábe-Loranca, B. *et al.* (2014) 'Extensions to K-medoids with balance restrictions over the cardinality of the partitions', *Journal of Applied Research and Technology*. Elsevier, 12(3), pp. 396–408.

Biernacki, C., Celeux, G. and Govaert, G. (2010) 'Exact and Monte Carlo calculations of integrated likelihoods for the latent class model', *Journal of Statistical Planning and Inference*. Elsevier, 140(11), pp. 2991–3002.

Bio-Rad (2018a) *Flow Cytometry - User Manual*, *Bioinformatics and Biomedical Engineering, 2008. ICBBE 2008. The 2nd International Conference on*.

Bio-Rad (2018b) *Introduction to Flow Cytometry Basics Guide | Bio-Rad*. Available at: https://www.bio-rad-antibodies.com/introduction-to-flow-cytometry.html (Accessed: 28

March 2019).

Breitkreutz, D. and Casey, K. (2008) *Clusterers: a comparison of partitioning and density-based algorithms and a discussion of optimisations*, *Work*. Available at: http://eprints.jcu.edu.au /11999/.

Bullinaria, J. A. (2004) *Self Organizing Maps: Fundamentals, Introduction to Neural Networks : Lecture 16*.

Caarls, W. *et al.* (2010) 'Characterization of coupled ground state and excited state equilibria by fluorescence spectral deconvolution', *Journal of Fluorescence*, 20(1), pp. 181–190.

Cancer Research (2018) *Cancer statistics for the UK*, *Cancer Research UK*. Available at: http://www.cancerresearchuk.org/health-professional/cancer-statistics (Accessed: 24 August 2021).

Carpenter, G. A. and Grossberg, S. (2002) 'The Handbook of Brain Theory and Neural Networks - Adaptive Resonance Theory', (617), pp. 1–12.

Dalatu, P. I. (2016) 'Time Complexity of K-Means and K-Medians Clustering Algorithms in Outliers Detection', *Global Journal of Pure and Applied Mathematics*, 12(5), pp. 4405–4418.

Day, W. H. E. and Edelsbrunner, H. (1984) 'Efficient algorithms for agglomerative hierarchical clustering methods', *Journal of Classification*, 1(1), pp. 7–24.

Dean L. (2005) *Blood Groups and Red Cell Antigens [Internet]*, *National Center for Biotechnology Information (US)*. Available at: https://www.ncbi.nlm.nih.gov/books/NBK2263/ (Accessed: 15 March 2019).

Dickinson, B. (2002) *Introduction to Flow Cytometry : A Learning Guide*. Becton, Dickinson and Company, Franklin Lakes.

Diez-Silva, M. *et al.* (2010) 'Shape and biomechanics characteristics of human red blood cells in health and disease', *MRS Bulletin*, 35(5), pp. 382–388.

Duong, T. *et al.* (2008) 'Feature significance for multivariate kernel density estimation', *Computational Statistics and Data Analysis*, 52(9), pp. 4225–4242.

Ester, M. *et al.* (1996) 'A Density-Based Algotithm for Discovering Clusters in Large Spatial Database with Noise', *Comprehensive Chemometrics*, 2, pp. 635–654.

Farley, A., Hendry, C. and McLafferty, E. (2012) 'Blood components', *Nursing Standard*, 27(13), pp.

35–42.

Finak, G. *et al.* (2009) 'Merging Mixture Components for Cell Population Identification in Flow Cytometry', *Advances in Bioinformatics*, 2009, pp. 1–12.

Fleisher, T. A. and Oliveira, J. B. (2019) *Flow Cytometry*. Fifth Edition, *Clinical Immunology: Principles and Practice*. Fifth Edition. Elsevier Ltd.

Ge, Y. and Sealfon, S. C. (2012) 'Flowpeaks: A fast unsupervised clustering for flow cytometry data via K-means and density peak finding', *Bioinformatics*, 28(15), pp. 2052–2058.

GEORGE, E. L. and PANOS, A. (2005) 'Does a high WBC count signal infection?', *Nursing2023*, 35(1). Available at: https://journals.lww.com/nursing/Fulltext/2005/01000/Does_a_high_WBC_count_signal_infection_.14.aspx.

Groeneveld-Krentz, S. *et al.* (2016) 'The Role of Machine Learning in Medical Data Analysis. A Case Study: Flow Cytometry', (January 2016), pp. 303–310.

Hedley, B. D. and Keeney, M. (2013) 'Technical issues: Flow cytometry and rare event analysis', *International Journal of Laboratory Hematology*, 35(3), pp. 344–350.

Jahan-Tigh, R. R. *et al.* (2012) 'Flow Cytometry', *J. Invest. Dermatol.* Nature Publishing Group, 132(10), p. e1.

Jain, A. K., Murty, M. N. and Flynn, P. . (1999) 'Data clustering: a review', *ACM Computing Surveys*, 31(3), pp. 264–323.

Kabat, G. C. *et al.* (2017) 'White Blood Cell Count and Total and Cause-Specific Mortality in the Women's Health Initiative', *American Journal of Epidemiology*, 186(1), pp. 63–72.

Kalina, T. *et al.* (2019) 'CD maps—dynamic profiling of CD1–CD100 surface expression on human leukocyte and lymphocyte subsets', *Frontiers in Immunology*, 10(OCT).

King, W., Toler, K. and Woodell-May, J. (2018) 'Role of White Blood Cells in Blood- and Bone Marrow-Based Autologous Therapies', *BioMed Research International*. Hindawi, 2018.

Kuhn, V. *et al.* (2017) 'Red Blood Cell Function and Dysfunction: Redox Regulation, Nitric Oxide Metabolism, Anemia', *Antioxidants and Redox Signaling*, 26(13), pp. 718–742.

Lee, H. C. *et al.* (2017) 'Automated cell type discovery and classification through knowledge transfer', *Bioinformatics*, 33(11), pp. 1689–1695.

Lo, K., Brinkman, R. R. and Gottardo, R. (2008) 'Automated gating of flow cytometry data via robust

model-based clustering', *Cytometry Part A*, 73(4), pp. 321–332.

McKinnon, K. M. (2018) 'Flow Cytometry: An Overview.', *Current protocols in immunology*. United States, 120, pp. 5.1.1-5.1.11.

McLachlan, G. J. (1999) 'Mahalanobis Distance', *Resonance*, pp. 20–26.

Müller, S. and Nebe-Von-Caron, G. (2010) 'Functional single-cell analyses: Flow cytometry and cell sorting of microbial populations and communities', *FEMS Microbiology Reviews*, 34(4), pp. 554–587.

Nagel, A. *et al.* (2014) 'CD3-positive B cells: A storage-dependent phenomenon', *PLoS ONE*, 9(10), pp. 1–10.

Pyne, S. *et al.* (2009) 'Automated high-dimensional flow cytometric data analysis', *Proceedings of the National Academy of Sciences*, 106(21), pp. 8519–8524.

Qian, Y. *et al.* (2010) 'Elucidation of seventeen human peripheral blood B-cell subsets and quantification of the tetanus response using a density-based method for the automated identification of cell populations in multidimensional flow cytometry data', *Cytometry Part B: Clinical Cytometry*, 78B(S1), pp. S69–S82.

Rahim, A. *et al.* (2018) 'High throughput automated analysis of big flow cytometry data', 135, pp. 164–176.

Razzak, R. A. *et al.* (2018) 'High-normal blood glucose levels may be associated with decreased spatial perception in young healthy adults', *PLoS ONE*, 13(6), pp. 1–12.

Robinson, S. *et al.* (2018) 'The administration of blood components: a British Society for Haematology Guideline', *Transfusion Medicine*, 28(1), pp. 3–21.

Roettger, R. *et al.* (2009) 'Combining Mixture Components for Clustering for Clustering', *Cytometry Part A*. 4th edn. IEEE, 6(1), pp. 1–12.

Rokach, L. and Maimon, O. (2005) 'Clustering Methods, Book Section', *Data Mining and Knowledge Discovery Handbook*, pp. 321–352.

Rota, P. *et al.* (2016) 'The Role of Machine Learning in Medical Data Analysis. A Case Study: Flow Cytometry', 3(Visigrapp), pp. 303–310.

Rowley, T. (2019) 'Flow Cytometry - A Survey and the Basics', *Materials and Methods*, 2019. Available at: http://www.labome.com/method/Flow-Cytometry-ASurvey-and-the-Basics.htm

(Accessed: 21 Dec 2019).

Rye, C. *et al.* (2017) *Biology*. e-book. Rice University, Texas. Available at: https://cnx.org/contents/GFy_h8cu@10.4:_XipwKIy@4/Components-of-the-Blood.

Sautbine, N., Voslar, A. and Bancud, E. (2011) 'Effects of the white blood cell count on labeling efficiency using indium-111', *Journal of Nuclear Medicine*, 52(supplement 1), pp. 2412 LP – 2412. Available at: http://jnm.snmjournals.org/content/52/supplement_1/2412.abstract.

Schwarz, G. (1978) 'Estimating the dimension of a model', *The Annals of Statistics*, 6(2), pp. 461–464.

Shahrabi, S. *et al.* (2020) 'CD markers polymorphisms as prognostic biomarkers in hematological malignancies', *Oncology Reviews*, 14(2), pp. 98–107.

Shaked, M. *et al.* (2019) 'Perspective of healthy asymptomatic patients requesting general blood tests from their physicians: a qualitative study', *BMC Family Practice*. BMC Family Practice, 20(1), pp. 1–8.

Shapiro, H. M. (2013) *PRACTICAL FLOW*. 4th edn. John Wiley & Sons Inc.

Sugár, I. P. and Sealfon, S. C. (2010) 'Misty Mountain clustering: Application to fast unsupervised flow cytometry gating', *BMC Bioinformatics*, 11(Icl).

Sun, H. *et al.* (2017) 'Clinical significance of routine blood test-associated inflammatory index in breast cancer patients', *Medical Science Monitor*, 23, pp. 5090–5095.

Tassinari, M. *et al.* (2018) 'Rapid diagnosis of bloodstream infections in the critically ill: Evaluation of the broad-range PCR/ESI-MS technology', *PLoS ONE*, 13(5), pp. 1–12.

Vembadi, A., Menachery, A. and Qasaimeh, M. A. (2019) 'Cell Cytometry: Review and Perspective on Biotechnological Advances', *Frontiers in Bioengineering and Biotechnology*, 7(JUN).

Verschoor, C. P. *et al.* (2015) 'An introduction to automated flow cytometry gating tools and their implementation', *Frontiers in Immunology*, 6(JUL), pp. 1–9.

Wang, W., Yang, J. and Muntz, R. (1997) 'STING : A Statistical Information Grid Approach to Spatial Data Mining', *Proceedings of the 23rd VLDB conference*, pp. 1–10.

Weber, L. M. and Robinson, M. D. (2016) 'Comparison of clustering methods for high-dimensional single-cell flow and mass cytometry data', *Cytometry Part A*, 89(12), pp. 1084–1096.

Whelan, C., Harrell, G. and Wang, J. (2015) 'Understanding the K-Medians Problem', in. Int'l Conf.

Scientific Computing, pp. 219–222.

WHO (2018) *The top 10 causes of death*. Available at: https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death (Accessed: 27 March 2019).

Wood, J. C. S. (2021) 'How well can your flow cytometer detect photons?', *Cytometry Part A*, 99(7), pp. 664–667.

World Health Organization, R. O. for S.-E. A. (2009) 'Laboratory guidelines for enumerating CD4 T lymphocytes in the context of HIV/AIDS', *WHO Regional Office for South-East Asia*, pp. 1–86.

Yadav, J. and Kumar, D. (2014) 'Subspace Clustering using CLIQUE : An Exploratory Study', *International Journal of Advanced Research in Computer Engineering & Technology*, 3(2), pp. 372–378.

Yang, M. S., Lai, C. Y. and Lin, C. Y. (2012) 'A robust em clustering algorithm for Gaussian mixture models', *Pattern Recognition*. Elsevier, 45(11), pp. 3950–3961.

Ye, X. and Ho, J. W. K. (2018) 'Ultrafast clustering of single-cell flow cytometry data using FlowGrid', *BMC Systems Biology*, 13.