ITERATIVE ESTIMATION OF ROBUST GAUSSIAN MIXTURE MODELS IN HETEROGENEOUS DATA SETS

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Caner Mercan

July, 2014

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Selim Aksoy (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Çiğdem Gündüz Demir

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Sinan Gezici

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

ABSTRACT

ITERATIVE ESTIMATION OF ROBUST GAUSSIAN MIXTURE MODELS IN HETEROGENEOUS DATA SETS

Caner Mercan
M.S. in Computer Engineering
Supervisor: Assoc. Prof. Dr. Selim Aksoy
July, 2014

Density estimation is the process of estimating the parameters of a probability density function from data. The Gaussian mixture model (GMM) is one of the most preferred density families. We study the estimation of a Gaussian mixture from a heterogeneous data set that is defined as the set of points that contains interesting points that are sampled from a mixture of Gaussians as well as non-Gaussian distributed uninteresting ones. The traditional GMM estimation techniques such as the Expectation-Maximization algorithm cannot effectively model the interesting points in a heterogeneous data set due to their sensitivity to the uninteresting points as outliers. Another potential problem is that the true number of components should often be known a priori for a good estimation. We propose a GMM estimation algorithm that iteratively estimates the number of interesting points, the number of Gaussians in the mixture, and the actual mixture parameters while being robust to the presence of uninteresting points in heterogeneous data. The procedure is designed so that one Gaussian component is estimated using a robust formulation at each iteration. The number of interesting points that belong to this component is also estimated using a multi-resolution search procedure among a set of candidates. If a hypothesis on the Gaussianity of these points is accepted, the estimated Gaussian is kept as a component in the mixture, the associated points are removed from the data set, and the iterations continue with the remaining points. Otherwise, the estimation process is terminated and the remaining points are labeled as uninteresting. Thus, the stopping criterion helps to identify the true number of components without any additional information. Comparative experiments on synthetic and real-world data sets show that our algorithm can identify the true number of components and can produce a better density estimate in terms of log-likelihood compared to two other algorithms.

Keywords: Gaussian mixture model, robust Gaussian estimation, iterative Gaussian mixture estimation, identifying number of mixture components.

ÖZET

GAUSS KARIŞIM MODELLERİNİN TÜRDEŞ OLMAYAN VERİ ÖBEKLERİNDE YİNELEMELİ KESTİRİMİ

Caner Mercan
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Assoc. Prof. Dr. Selim Aksoy
Temmuz, 2014

Verilerden olasılık yoğunluk fonksiyonu parametrelerini kestirme işlemine yoğunluk kestirimi denir. Gauss karışım modeli (GKM) en çok tercih edilen yoğunluk ailelerinden biridir. Gauss karışımlarından örneklenmiş ilginç noktalar ile Gauss dağılımına sahip olmayan ilginç olmayan noktalardan oluşan veri öbeklerinden Gauss karışım kestiriminin nasıl yapıldığını inceliyoruz. Expectation-Maximization algoritması gibi geleneksel GKM kestirim teknikleri, ilginç olmayan noktalara olan hassaslığından dolayı, türdeş olmayan verilerdeki ilginç noktaları etkin olarak modelleyememektedir. Bir diğer olası sorun ise, iyi bir kestirim yapılabilmesi için öncesinde gerçek bileşen sayısının genellikle bilinmesinin gerektiğidir. Biz, ilginç olmayan noktaların varlığına karşı sağlam, ilginç noktaların sayısını, karışımdaki Gauss sayısını ve gerçek karışım parametrelerini yinelemeli kestiren bir GKM kestirim algoritması tasarlıyoruz. Yöntemimiz, her yinelemede bir Gauss bileşenini sağlam bir formulasyonla kestirir. Bu bileşene ait ilginç nokta sayısı, çoklu-çözünürlük arama yöntemi kullanılarak adaylar arasından kestirilir. Eğer bu noktaların Gauss dağılımından gelme hipotezi kabul edilirse, kestirilen Gauss, karışım bileşeni olarak tutulur, ilgili noktalar veri öbeğinden çıkarılır ve yinelemeler kalan noktalarla devam eder. Aksi durumda, kestirim süreci durdurulur ve kalan noktalar ilginç olmayan şeklinde etiketlenir. Bu şekilde, durdurma kriteri herhangi ek bir bilgi olmadan gerçek bileşen sayısını belirler. Sentetik ve gerçek-dünya veri öbekleri üzerinden yapılan karşılaştırmalı deneyler gösteriyor ki, algoritmamız gerçek bileşen sayısını belirleyebiliyor ve diğer iki algoritmaya göre daha iyi yoğunluk kestirimi yapabiliyor.

Anahtar sözcükler: Gauss karışım modeli, sağlam Gauss kestirimi, yinelemeli Gauss karışım kestirimi, karışım bileşen sayısının belirlenmesi.

Acknowledgement

This thesis is the culmination of my interactions with many people. Their knowledge, guidance and companionship made this thesis what it is. I would like to acknowledge these remarkable individuals.

First and foremost, my biggest debt of gratitude goes to my supervisor Selim Aksoy. I would like to thank him for his invaluable vision, encouragement and motivation. This thesis would not be possible without his guidance.

Special thanks to Çiğdem Gündüz Demir and Sinan Gezici for kindly accepting to be in my committee. I owe them my appreciation for their support and helpful suggestions.

I would like to thank my mother, Kimya Mercan, for her unrequited & perpetual love, my father, Cengiz Mercan, for his unwavering trust and my sister, Pınar Mercan, for her sincere support. I am tremendously grateful for all the selflessness and the sacrifices you have made on my behalf.

I consider myself to be one of the luckiest people to have friends like Esra, Anıl, Aslı, Taylan, Seçkin, Pelin and Özüm. Never once have they failed to welcome me with open, loving arms. I thank them for our long-lasting friendship and for always being there when I need them.

Bilkent would not mean anything without my friends Alican, İnci, Elif, Nermin, İlker, Anıl, Berk, Seher, Gökhan, Eren and Acar. I am deeply grateful for their companionship and for making Bilkent a lively place with full of joyful memories.

Last but not least, I would like to express my gratitude for TUBITAK for supporting me with BIDEB scholarship for two years during my studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Gaussian Mixture Model

Density estimation is the construction of an estimate of a probability density function from the observed data points [1]. In order to overcome the limitations of simple probability distributions, we can take their linear combinations and obtain mixture models [2, 3]. Mixture models have established an important place in density estimation problems and statistical analysis of data [4]. Gaussian mixture model (GMM) [4, 5, 6, 7, 8] is one of the most widely used among the various types of mixture models due to its advantages over other mixtures. We can sum up some of them as follows. It is one of the most statistically mature methods. Presence of well-studied inference techniques allows us to exploit its strengths more easily. Estimating its parameters is rather fast compared to other types of mixtures. Due to their high flexibility, they are commonly used in fields like image processing [9, 10, 11, 12, 13, 14, 15], computer vision [16, 17, 18, 19] and pattern recognition [4, 7, 8].

Besides density estimation, GMMs are also frequently used for clustering [20, 21, 22, 23]. Clusters are chosen according to the component that maximizes the posterior probability. GMM clustering is often associated with the k-means clustering algorithm [24]. However, unlike k-means, clustering via GMMs

is often considered as a soft clustering method due to the fact that the posterior probabilities for each data point indicate that each data point has some probability of belonging to each cluster. In this work, we are more interested in using GMMs for density estimation rather than as a means of clustering.

In a more formal and mathematical perspective, we define Gaussian densities and GMMs as follows. A vector-valued random variable $\mathbf{x} \in \mathbb{R}^d$ follows Gaussian distribution $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{S}^d_{++}$ if its probability density function is given as

$$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right).$$

Moreover, a linear superposition of K Gaussian densities results in a K component mixture of Gaussian distributions as

$$p(\mathbf{x}) = \sum_{k=1}^{K} \alpha_k p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

Each density in the mixture $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ has its own parameters, mean vector $\boldsymbol{\mu}_k$, covariance matrix $\boldsymbol{\Sigma}_k$, and the mixing coefficients α_k satisfy the following constraints

$$0 \leq \alpha_k \leq 1,$$
$$\sum_{k=1}^{K} \alpha_k = 1.$$

1.2 Expectation-Maximization

There is one big fundamental issue that prevents a simple estimation of parameters in GMM. Even though all data points are known to come from GMM, which data point comes from which component in the mixture is not known. It can be handled by introducing latent variables which naturally motivate the use of EM in finding the maximum likelihood estimates of the GMM parameters. The expectation-maximization (EM) algorithm and its variants [25, 26, 27] are the most widely used algorithms for finding the maximum likelihood estimates of

model parameters in the presence of latent variables or missing data. The fundamental idea behind the EM algorithm is to use an upper bound function on the negative log-likelihoods of the observed variables by introducing distributions over the hidden variables. This bound is a function of the negative log-likelihoods of the joint distributions of both the hidden and the observed variables and the introduced distributions over the hidden variables. The EM algorithm consists of two steps. The first step is called the Expectation Step (E-Step). In this step, the bound function is minimized over the introduced distributions over the hidden variables while holding the parameters found in the previous iteration fixed. The second step is called the Maximization Step (M-Step). In this step, the bound function is minimized over the parameters while holding the introduced distributions found in the E-step fixed. The algorithm employs an alternating optimization between these two steps. This procedure goes on until a fixed point of the algorithm corresponding to a local optimum is reached. The EM algorithm is guaranteed to monotonically decrease the negative log-likelihood and to converge to a local minimum [25, 26].

Notwithstanding to its popularity, finding the maximum likelihood estimates of parameters in GMM via EM algorithm has various shortcomings. First, when there is not sufficient number of data points for a component in the mixture, estimating a covariance matrix becomes problematic, e.g., when a component collapses onto a single data point, singularities in likelihood arise. As a result, the EM algorithm diverges and yields solutions with infinite likelihood. However, this problem is rather easy to solve as one can regularize the covariance matrix artificially beforehand. Second, the EM algorithm does not guarantee a global optimum solution as it can get stuck at a local optimum. This problem is frequently seen in real-world problems and it mostly arises due to the poorly initialized values of the parameters. One can initialize the algorithm several times with different initial values to prevent local optimum solutions. However, as the space dimension and the number of data points start to increase, finding the global optimum solution pretty much becomes searching for a needle in a haystack. Another well-known and preferred method is to run k-means beforehand to find suitable initialization values for the mean vectors but the k-means

algorithm is also sensitive to the initialization. Third, the true number of components in the mixture should be known for a good estimation of the parameters. We will talk about the advancements as well as the shortcomings of other works in this field later on. Lastly, density estimation cannot be done properly in the presence of unwanted/uninteresting points in the data. EM in its nature, is very sensitive to any outlying points and it fails to capture the model that includes only the wanted/interesting points. This problem will also be discussed in detail later on.

1.3 Heterogeneous Data

We define heterogeneous data as the combination of interesting and uninteresting data points. Other than being non-Gaussian distributed, we do not make a particular assumption about the distribution of uninteresting points. On the other hand, interesting points are assumed to be sampled from a mixture of Gaussians. Data sets with such structures are the Achilles' heel of the classical GMM estimation methods. They expose one of the previously mentioned weaknesses of the traditional methods; sensitivity to any kind of uninteresting points.

As a matter of fact, our primary aim is to estimate a robust GMM in a heterogeneous data set where we can model only the interesting points while being robust to the presence of uninteresting ones. Moreover, our secondary aim is to make the estimation process without the knowledge of the true number of components and to identify it once the estimation is done.

1.4 Related Work

As previously mentioned, the classical Maximum Likelihood Estimator (MLE) is used to estimate the parameters in a mixture of distributions via EM. The traditional approach cannot handle the presence of any uninteresting points; thus, it fails greatly in finding the maximum likelihood estimates of the parameters of

a GMM from which the interesting points are sampled in a heterogeneous data set.

The problem of modeling heterogeneous data sets has been addressed by many. There have been mainly two approaches that are used to cope with the presence of uninteresting points in heterogeneous data sets. The first one is to fit an additional mixture component or a mixture distribution to the uninteresting points. In [20], the uninteresting points are assumed to be drawn from a uniform distribution while [28] fits a Poisson distribution to the uninteresting points. Likewise, a mixture of t-distributions is used to model the uninteresting points in [4, 29]. These methods rely heavily on the assumption of the underlying distribution of the uninteresting points. Hence, they are rendered useless once the assumption does not hold and the points do not follow the assumed distribution. The second way is to model only the interesting points in heterogeneous data sets to overcome the shortcoming of the classical approach. Robust variations of MLE have been proposed in order to reduce the problem of sensitivity against the uninteresting points of the classic approach. In [30], robust fitting of mixtures is conducted with trimming via the Weighted Trimmed Likelihood Estimator (WTLE) [31]. The main idea of this approach is partitioning n data points into m-subsets randomly, and out of all $\binom{n}{m}$ combinations, choosing the subset with the MLE fit for which the negative log-likelihood is minimal. However, this approach is heuristic and it is not practical for any dataset with a reasonable size. Another variation of TLE, called FAST-TLE [32], has been proposed to speed up the TLE process. In FAST-TLE, a large m is chosen and the algorithm consists of two steps; trial and refinement. For each trial step, the EM algorithm is run. Also, the number of interesting points is assumed to be approximately known. Thus, operations are performed for data points which are believed to be interesting.

There have been other works that deal with other weaknesses of the traditional GMM estimation. The initialization problem leads to convergence problems and produces local optimum solutions. This problem is rather well known and it is widely researched [33, 34]. The problem that the dependence to the true number of components for a good estimation is addressed in [35]. [36] handles the problem by penalizing over-complex models and [37] searches through minimum possible

component size to the maximum to determine the true number of components for some criterion. [38] presents a method for both robustness to initialization and finding the true number of components. Additionally, Dirichlet Processes (DP) [39, 40] are frequently incorporated for mixture model learning where the component size is not fixed, but is instead inferred from data [41, 42, 43]. A hierarchical DP [44] reuses a common set of components in order to model related data sets. However, in none of these works, robustness against uninteresting points are taken into account. Thus, they fail greatly even in the presence of small amounts of uninteresting points.

We can summarize our proposed method as follows. At each iteration of our algorithm, Gaussian models that are robust to the presence of uninteresting points are estimated from multiple candidate sets of points with different sizes. Then, among these models, the best model that corresponds to a subset of the interesting points is found. After that, a stopping criterion based on a Gaussianity test is applied on these points. If the hypothesis is accepted, the respective points are removed from the data set and stored along with the model parameters. On the other hand, the algorithm stops estimating a new robust Gaussian model if the hypothesis is rejected. Linear superposition of the estimated robust Gaussians corresponds to the Gaussian mixture that the interesting points are sampled from. Note that, not only does our approach not need to know the true number of components prior to the estimation process but also the stopping criterion helps to identify the true number of components without additional information as well.

1.5 Our Contributions

The classical maximum likelihood estimation of Gaussian mixtures by EM is plagued by many problems most of which were mentioned in Section 1.2. We propose a new approach that solves two of its major problems; the sensitivity to the presence of any kind of uninteresting points in the data set and the dependence to the prior knowledge of the true number of components.

The first problem arises when data consist of a heterogeneous set as described in Section 1.3 where only the interesting points come from a Gaussian mixture while the uninteresting ones are non-Gaussian distributed. Traditional estimation methods tend to estimate a model from all points in the data set. Thus, they fail greatly when data are heterogeneous. Since our algorithm is robust against uninteresting points in the data set, it is able to estimate a model from the interesting points where it can differentiate and disregard uninteresting ones.

The second problem is one of the most important shortcomings of classical MLE. Our algorithm not only does not depend on the true number of components for a good estimation but also can identify the true number of components, as well. We can summarize how it handles the problem and identifies the optimal number of components as follows. We estimate a robust Gaussian in each iteration of the algorithm. Then, we use a stopping criterion that employs a Gaussianity test on a subset of points selected on a candidate for a new component in the estimated model. If Gaussianity hypothesis is rejected, it means that the selected points and the remaining points in the data set are from the uninteresting set. Hence, the number of robust Gaussian models estimated up to that iteration corresponds to the number of components in the mixture.

1.6 Organization of the Thesis

In Chapter 2, we describe the problem definition and the general outline of the algorithm.

In Chapter 3, we present the details of parameter estimation of robust Gaussian model for a fixed number of points. After that, we describe how the optimal number is found using a procedure that we call multi-resolution search. Lastly, we propose a refinement for the final number of data points that belong to the estimated robust Gaussian.

In Chapter 4, we give details about the iterations for estimating multiple components and the stopping criterion. Finally, we describe the process of estimating

the mixing coefficients of robust Gaussian mixture model from the estimated robust Gaussians and their respective data points.

In Chapter 5, we describe the synthetic data generation process and the performance evaluation criteria. Finally, we present comparative experiments over synthetic and real-world data sets and discuss the results.

In Chapter 6, we summarize the thesis and present the advantages and disadvantages of the approach. We conclude with our plans for future work.

Chapter 2

Iterative Learning of Gaussian Mixture Models

2.1 Problem Definition

We consider the scenario where a data set $\mathcal{X}$ in $\mathbb{R}^d$ is composed of a mixture of interesting, $\widetilde{\mathcal{X}}$, and uninteresting, $\widehat{\mathcal{X}}$, points where $\widetilde{N} = |\widetilde{\mathcal{X}}|$ is the number of interesting points and $\widehat{N} = |\widehat{\mathcal{X}}|$ is the number of uninteresting points among $N = |\mathcal{X}| = \widetilde{N} + \widehat{N}$. We assume that the interesting points in $\widetilde{\mathcal{X}}$ come from a GMM

$$p(\mathbf{x}|\boldsymbol{\Theta}) = \sum_{k=1}^{K} \alpha_k \, p_k(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.1}$$

that is fully defined by the set of parameters $\boldsymbol{\Theta} = \{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$ where K is the number of components, $\boldsymbol{\mu}_k \in \mathbb{R}^d$ and $\boldsymbol{\Sigma}_k \in \mathbb{S}_{++}^d$ denote the mean vector and the covariance matrix of the k'th Gaussian component, respectively, and the mixing probabilities $\alpha_k \in [0, 1]$ are constrained to sum up to 1, i.e., $\sum_{k=1}^{K} \alpha_k = 1$. However, we do not make an explicit assumption about the distribution of the uninteresting points in $\widehat{\mathcal{X}}$.

We assume that, in the above scenario, the number of interesting points, $\widetilde{N}$, and the number of components in the mixture, K, are not known a priori. Our

goal is to estimate $\widetilde{N}$, K, and $\mathbf{\Theta}$ from the heterogeneous data $\mathcal{X}$.

We will provide the key steps of our algorithm on a sample illustrative data set given in Figure 2.1.

2.2 Algorithm Overview

The proposed algorithm aims to iteratively estimate the Gaussian components that model the interesting data points while being robust to the presence of uninteresting ones. The estimation procedure is designed so that one Gaussian component is removed at each iteration, and the estimation process is terminated when no such component can be found.

The input to the algorithm is the heterogeneous data set $\mathcal{X}$ that consists of N data points. Since we do not know the number of interesting data points in $\mathcal{X}$, nor do we know the number of Gaussian components that they belong to, we estimate both of them iteratively as follows. Let $N^{(k)}$ denote the size of the input data set and $\widetilde{N}^{(k)}$ denote the unknown number of interesting points in iteration k, where $N^{(0)} = N$ is the size of the initial data set $\mathcal{X}$ and $\widetilde{N}^{(0)} = 0$. In each iteration, we aim to identify $\widetilde{N}^{(k)}$ points that can be robustly modeled as belonging to a single Gaussian using the algorithm proposed in Chapter 3. If a hypothesis on the Gaussianity of these points is accepted, the estimated Gaussian is kept as a component in the final mixture, the $\widetilde{N}^{(k)}$ points are removed from the data set, and the iterations continue with $N^{(k+1)} = N^{(k)} - \widetilde{N}^{(k)}$ points. Otherwise, the iterations stop, and the remaining $N^{(k)}$ points are labeled as uninteresting as described in Chapter 4. Details of each of these steps are described in the following chapters. The general outline of the algorithm is given in Algorithm 1.

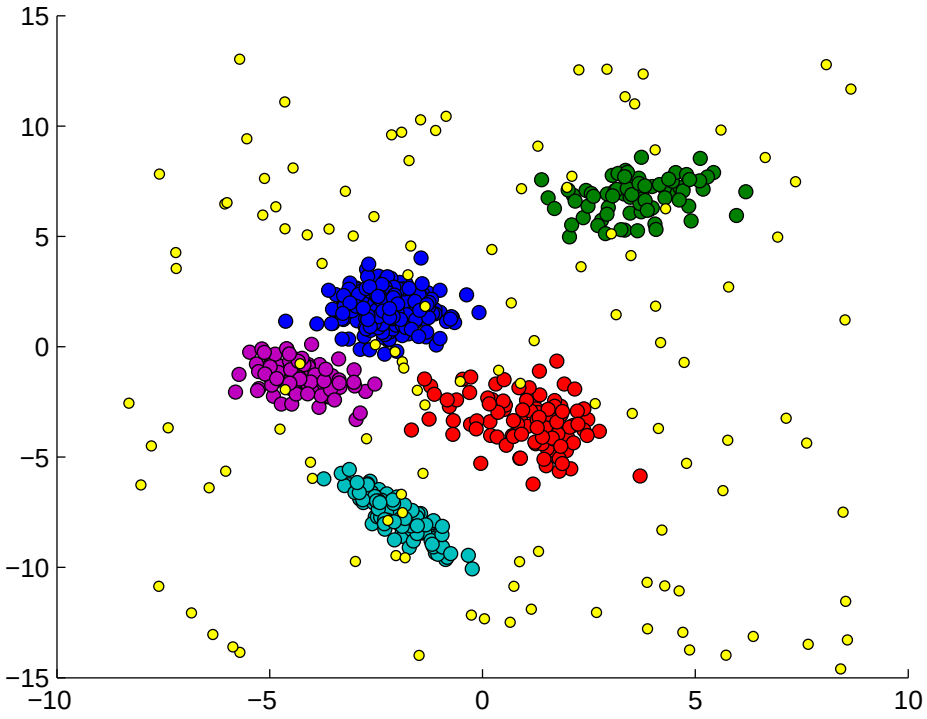


Figure 2.1: Illustrative data set that consists of 720 points. 600 of these points form the interesting set $\widetilde{\mathcal{X}}$ and are generated from a 5-component, 2-dimensional Gaussian mixture with 2-separated, 15-eccentric components (see Section 5.2.1 for details on data generation). The remaining 120 points form the uninteresting set $\widehat{\mathcal{X}}$ and are drawn from a uniform distribution.

Algorithm 1 Iterative Robust Gaussian Mixture Model Learning

INPUT: data set $\mathcal{X}$, regularization parameter γ

OUTPUT: $\{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

```
k ← 1
X^(k) ← X
while true do
  estimate robust Gaussians with γ from multiple candidates of Ñ^(k)
  find Ñ^(k) by multi-resolution search
  refine Ñ^(k) to include data points in the tails of the Gaussian
  if Gaussianity hypothesis is rejected on X̃^(k) then
    k ← k − 1
    break
  end if
  add N(μ_k, Σ_k) to mixture
  remove X̃^(k) from X^(k)
  k ← k + 1
end while
K ← k
α_j ← Ñ^(j)/Ñ , j = 1, ..., K
```

Chapter 3

Robust Gaussian Model

In this chapter, the goal is to model the interesting subset of the input heterogeneous data set, that consists of both interesting and uninteresting points, using a Gaussian distribution. First, we show how the parameters of this Gaussian can be estimated for a fixed number of interesting points (Section 3.1). Then, we show how the number of interesting points can be estimated using a multi-resolution search procedure (Section 3.2).

3.1 Robust Estimation for a Fixed $\widetilde{N}$

In information theory, the relative entropy or the Kullback-Leibler (KL) divergence [7] between two probability distributions $\widetilde{p}(\mathbf{x})$ and $p(\mathbf{x})$ can be used for model selection, description, or approximation. It can be interpreted as the additional amount of information required to specify the value of $\mathbf{x}$ as a result of using p instead of the true distribution $\widetilde{p}$, and is computed as

$$KL(\widetilde{p}\|p) = \int \widetilde{p}(\mathbf{x}) \log \frac{\widetilde{p}(\mathbf{x})}{p(\mathbf{x})} d\mathbf{x}. \tag{3.1}$$

In the problem studied in this thesis, the interesting points that we would like to model are typically observed as part of a larger set of observations where

the rest of the points have an unknown distribution. For a fixed number, $\widetilde{N}$, of points that are embedded in a larger set of size N, the empirical distribution [45] can be defined as

$$\widetilde{p}(\mathbf{x}) = \frac{1}{\widetilde{N}} \sum_{i=1}^{N} z_i \delta(\mathbf{x} - \mathbf{x}_i) \tag{3.2}$$

where δ is the Dirac delta function and $z_i \in \{0, 1\}, i = 1, \ldots, N$, are the binary indicator variables that identify the points of interest, satisfying the constraint $\sum_{i=1}^{N} z_i = \widetilde{N}$. $\widetilde{p}(\mathbf{x})$ in (3.2) assigns an equal probability of $1/\widetilde{N}$ to the $\widetilde{N}$ points of interest whose corresponding binary indicator variables z_i are 1, and a probability of 0 is assigned to the remaining points. Thus, we do not make any explicit assumption about the distribution of the remaining $N - \widetilde{N}$ points.

The estimation of the parameters of the Gaussian model $p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ that best approximates the empirical distribution $\widetilde{p}(\mathbf{x})$ can be obtained by minimizing the KL divergence

$$\begin{aligned}
&KL\big(\widetilde{p}(\mathbf{x}) \| p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\big) \\
&\quad = \int \widetilde{p}(\mathbf{x}) \log \widetilde{p}(\mathbf{x}) d\mathbf{x} - \int \widetilde{p}(\mathbf{x}) \log p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})) d\mathbf{x} \\
&\quad = \int \frac{1}{\widetilde{N}} \sum_{i=1}^{N} z_i \delta(\mathbf{x} - \mathbf{x}_i) \log \widetilde{p}(\mathbf{x}) d\mathbf{x} - \int \frac{1}{\widetilde{N}} \sum_{i=1}^{N} z_i \delta(\mathbf{x} - \mathbf{x}_i) \log p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{x} \\
&\quad = \frac{1}{\widetilde{N}} \sum_{i=1}^{N} z_i \log \widetilde{p}(\mathbf{x}_i) - \frac{1}{\widetilde{N}} \sum_{i=1}^{N} z_i \log p(\mathbf{x}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\
&\quad = -\log \widetilde{N} - \frac{1}{\widetilde{N}} \sum_{i=1}^{N} z_i \log p(\mathbf{x}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma})
\end{aligned} \tag{3.3}$$

as

$$\begin{aligned}
&\text{minimize } -\frac{1}{\widetilde{N}} \sum_{i=1}^{N} z_i \log p(\mathbf{x}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad \text{over } \mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma} \\
&\text{subject to } z_i \in \{0, 1\},\ i = 1, \ldots, N, \\
&\qquad \sum_{i=1}^{N} z_i = \widetilde{N}.
\end{aligned} \tag{3.4}$$

However, the KL divergence tends to favor models with small volume that collapse on individual data points. Consequently, with no constraints on the number

of Gaussians, the optimal solution is to overfit to the data by placing an individual Gaussian on top of each individual data point. Therefore, we introduce a regularization term that uses the volume of the Gaussian density that is expressed as the log-determinant of its covariance matrix to favor Gaussians with larger volume as

$$KL_{reg}\big(\widetilde{p}(\mathbf{x})\|p(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma})\big) = -\log\widetilde{N} - \frac{1}{\widetilde{N}}\sum_{i=1}^{N} z_i \log p(\mathbf{x}_i|\boldsymbol{\mu},\boldsymbol{\Sigma}) + \gamma(-\log\det\boldsymbol{\Sigma}) \tag{3.5}$$

where γ is the regularization parameter that will be selected empirically in Chapter 5. The comparison of KL divergence that favors models with small volume and regularized KL divergence that favors models with larger volume with respect to increasing $\widetilde{N}$ is shown for the illustrative data set in Figure 3.1.

For a fixed $\widetilde{N}$, the estimation of the robust Gaussian model can be formulated as the minimization of the regularized KL divergence as

$$\begin{aligned} &\text{minimize } -\frac{1}{\widetilde{N}}\sum_{i=1}^{N} z_i \log p(\mathbf{x}_i|\boldsymbol{\mu},\boldsymbol{\Sigma}) + \gamma(-\log\det\boldsymbol{\Sigma}) \quad \text{over } \mathbf{z},\boldsymbol{\mu},\boldsymbol{\Sigma} \\ &\text{subject to } z_i \in \{0,1\},\ i=1,\ldots,N, \\ &\qquad\qquad \sum_{i=1}^{N} z_i = \widetilde{N} \end{aligned} \tag{3.6}$$

where $\mathbf{z} = (z_1,\ldots,z_N)$ is the vector of indicator variables. This problem can be solved by introducing a binary relaxation to the indicator variables $z_i \in \{0,1\},\ i=1,\ldots,N$ as $0 \le z_i \le 1$ where an optimal solution still consists of binary values.

The solution to (3.6) can be obtained via alternating optimization where the objective function is minimized over $\mathbf{z}$ for fixed $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, and over $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ for fixed $\mathbf{z}$ iteratively. For fixed $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, the optimization problem reduces to a linear program in $\mathbf{z}$. Minimization of a linear objective over a unit box with a total sum constraint has the following solution:

$$z_i^{(t+1)} = \begin{cases} 1, & \mathbf{x}_i \text{ is among } \widetilde{N} \text{ data points with largest } p(\mathbf{x}_i|\boldsymbol{\mu}^{(t)},\boldsymbol{\Sigma}^{(t)}), \\ 0, & \text{otherwise.} \end{cases} \tag{3.7}$$

For fixed $\mathbf{z}$, the update equations for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ can be derived as

$$\boldsymbol{\mu}^{(t+1)} = \frac{\sum_{i=1}^{N} z_i^{(t+1)} \mathbf{x}_i}{\sum_{i=1}^{N} z_j^{(t+1)}} \tag{3.8}$$

$$\boldsymbol{\Sigma}^{(t+1)} = \frac{1}{(1-2\gamma)} \frac{\sum_{i=1}^{N} z_i^{(t+1)} \left(\mathbf{x}_i - \boldsymbol{\mu}^{(t+1)}\right)\left(\mathbf{x}_i - \boldsymbol{\mu}^{(t+1)}\right)^T}{\sum_{i=1}^{N} z_j^{(t+1)}} \tag{3.9}$$

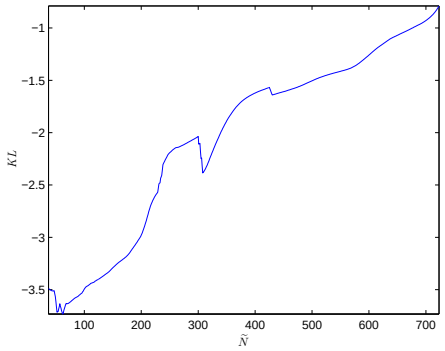
where t indicates the iteration number.

3.2 Multi-Resolution Search for Finding $\widetilde{N}$

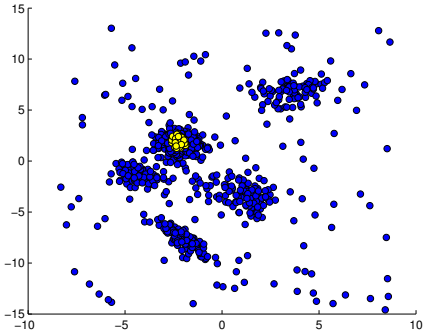
The procedure described in the previous section finds the optimal Gaussian parameters and the interesting points that belong to this Gaussian for a fixed $\widetilde{N}$ that is assumed to be known. The procedure can be repeated for different $\widetilde{N}$ values where each value results in a different KL_{reg} score according to (3.5) as shown in Figure 3.1. Given the scores for a range of $\widetilde{N}$ values, $\widetilde{N} = N_0, \ldots, N$ where N_0 is the smallest acceptable set size for interesting points, the next step aims to find the $\widetilde{N}$ for which the minimum KL_{reg} is attained.

Normally the $\widetilde{N}$ of interest coincides with the global maximum of $\big(KL_{reg}(N_0), \ldots, KL_{reg}(N)\big)$ that often corresponds to a large and dense Gaussian component in the data. However, in practice, when multiple Gaussian components that are not sufficiently dense (high within-component variance) are located close to each other (low between-component variance), especially combined with a clutter of uninteresting points, the KL_{reg} score for the combination of these components may be lower than the scores for the individual components. Consequently, a local minimum that corresponds to a smaller $\widetilde{N}$ may be preferred. Therefore, we use the following multi-resolution search procedure to evaluate multiple local minima of $\big(KL_{reg}(N_0), \ldots, KL_{reg}(N)\big)$ as candidates for the final number of interesting points.

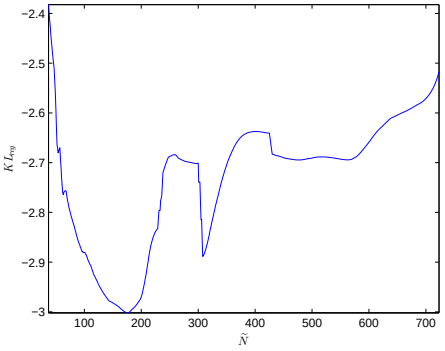
The proposed algorithm searches for local minima by dividing the $N_0, \ldots, N$ range into different number of bins in each resolution. In particular, the resolution r divides $N_0, \ldots, N$ into r equally sized bins, and records r candidates for minima



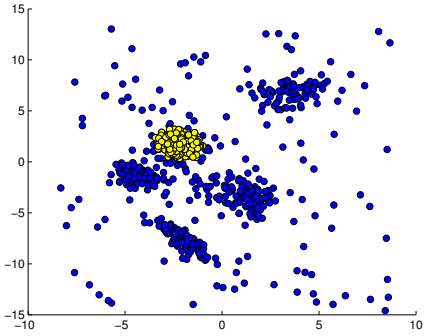
(a) KL values with respect to increasing $\widetilde{N}$. Minimum KL is achieved at $\widetilde{N} = 62$.



(b) $\widetilde{N} = 62$ points with $z_i = 1$, $i = 1, \ldots, N$ are marked in yellow.



(c) $KL_{reg}(\gamma = 0.25)$ values with respect to increasing $\widetilde{N}$. Minimum KL_{reg} is achieved at $\widetilde{N} = 177$.



(d) $\widetilde{N} = 177$ points with $z_i = 1$, $i = 1, \ldots, N$ are marked in yellow.

Figure 3.1: The comparison of KL and $KL_{reg}(\gamma = 0.25)$ values as solutions to (3.4) and (3.6), respectively, for different values of $\widetilde{N}$ for the illustrative data set in Figure 2.1. (a) KL is minimized for very small $\widetilde{N}$ which corresponds to a model with small volume. (b) Set of points with $z_i = 1$, $i = 1, \ldots, N$ after KL minimization is denoted in yellow. (c) KL_{reg} favors larger $\widetilde{N}$ corresponding to a model with larger volume. (d) Set of points with $z_i = 1$, $i = 1, \ldots, N$ after KL_{reg} minimization is denoted in yellow.

where each candidate corresponds to the global minimum in one of the bins. This procedure continues for a given number of resolutions, and all candidate minima receive votes from all resolutions. The global minimum in $N_0, \ldots, N$ will receive the highest number of votes. We also expect that few strong local minima that correspond to significant changes in the Gaussian structure will receive relatively higher number of votes compared to many insignificant minima that correspond to small changes in the assignment of points to the Gaussian component. Note that, a simple threshold on the number of votes could prove useful in identifying the strong minima. However, it gives birth to a new parameter to adjust, as well. We want to find those points using a non-parametric, statistical method. Thus, the next step aims to identify these strong minima with the help of a statistical outcome over modified Z-score [46] as

$$Z_{mod}(n) = \frac{0.6745\,(\,v(n) - M\,)}{MAD} \tag{3.10}$$

where $v(n)$ is the number of votes of a candidate, M is the median of the votes and MAD denotes the median absolute deviation of the votes of all candidate points. It is stated that points with modified Z-scores with an absolute value greater than 3.5 can be considered as potential outliers [46]. In our case, we have a small subset of points with high number of votes and a quite larger subset of points with few number of votes. Obviously, the points in the small subset can be regarded as outliers. Hence, we compute Z_{mod} of each candidate point and take the ones whose Z_{mod} is greater than 3.5 which, in our case, correspond to strong minima. As previously mentioned, these points correspond to significant changes in the Gaussian structure and we are looking for the smallest set that causes this phenomenon. Hence, $\widetilde{N}$ is chosen as the smallest of the points with strong minima.

The first 10 resolutions of multi-resolution search with their respective candidate points are demonstrated for the illustrative data set in Figure 3.2. Additionally, the modified Z-scores of the candidate points for the illustrative data set are presented in Figure 3.3 and the KL_{reg} scores overlaid with the modified Z-scores of the candidate points are given in Figure 3.4. Finally, the models corresponding to the strong local minima points are drawn on the illustrative data set in Figure 3.5.

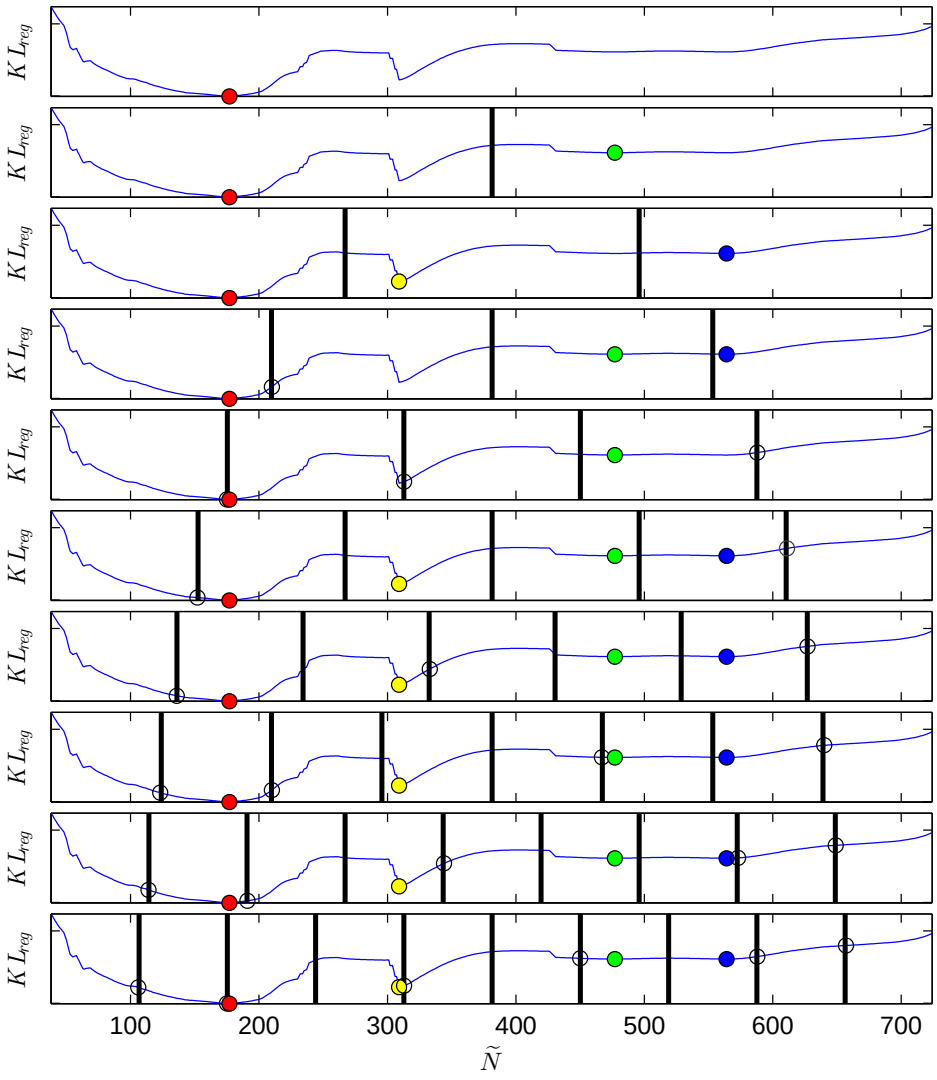


Figure 3.2: The first 10 resolutions of multi-resolution search procedure for the illustrative data set. Black straight lines show the separation of bins at each resolution. Strong minima points are marked in red, yellow, green and blue. The remaining candidate points are shown as empty black circles.

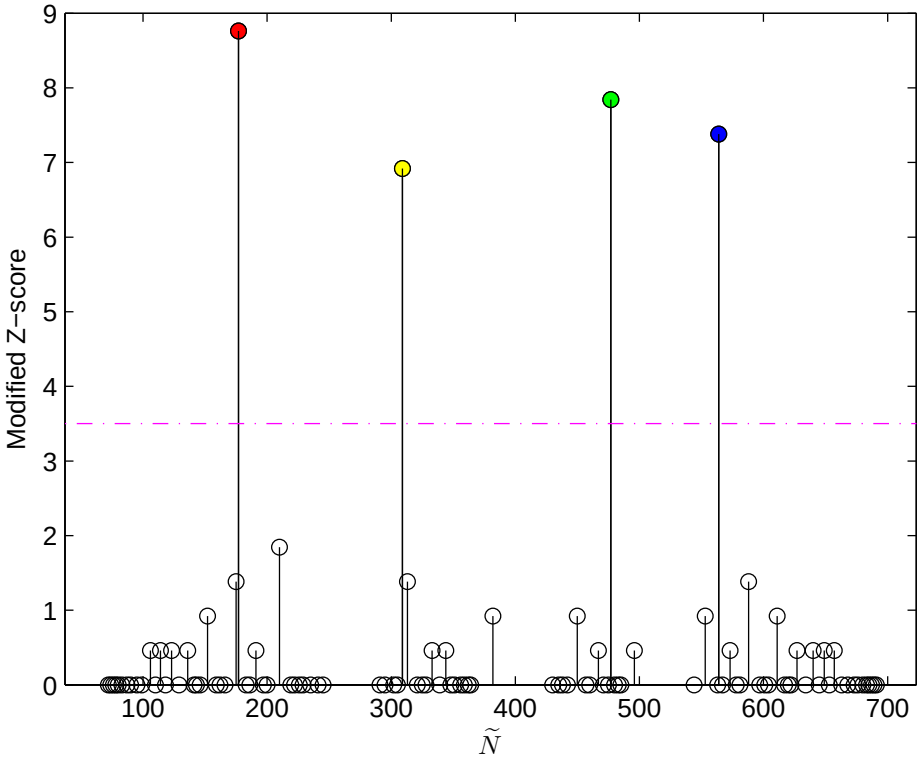


Figure 3.3: The modified Z-scores of each candidate points. The threshold value 3.5 is denoted by the magenta line. Candidates with a modified Z-score over the threshold value marked in red, yellow, green and blue correspond to strong local minima.

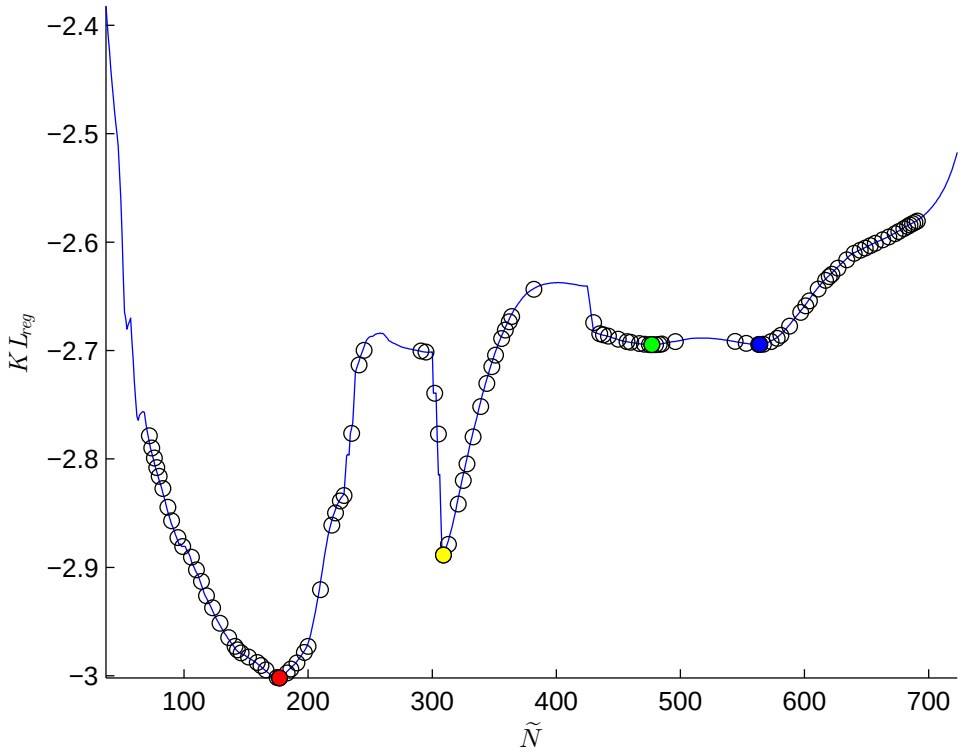


Figure 3.4: KL_{reg} scores are overlaid with modified Z-scores. Candidate points are marked as empty black circles. Strong local minima detected by multi-resolution search are denoted by red, yellow, green and blue circles. Among those local minima $(177, 309, 477, 564)$, the smallest one, 177 is chosen as $\widetilde{N}$.

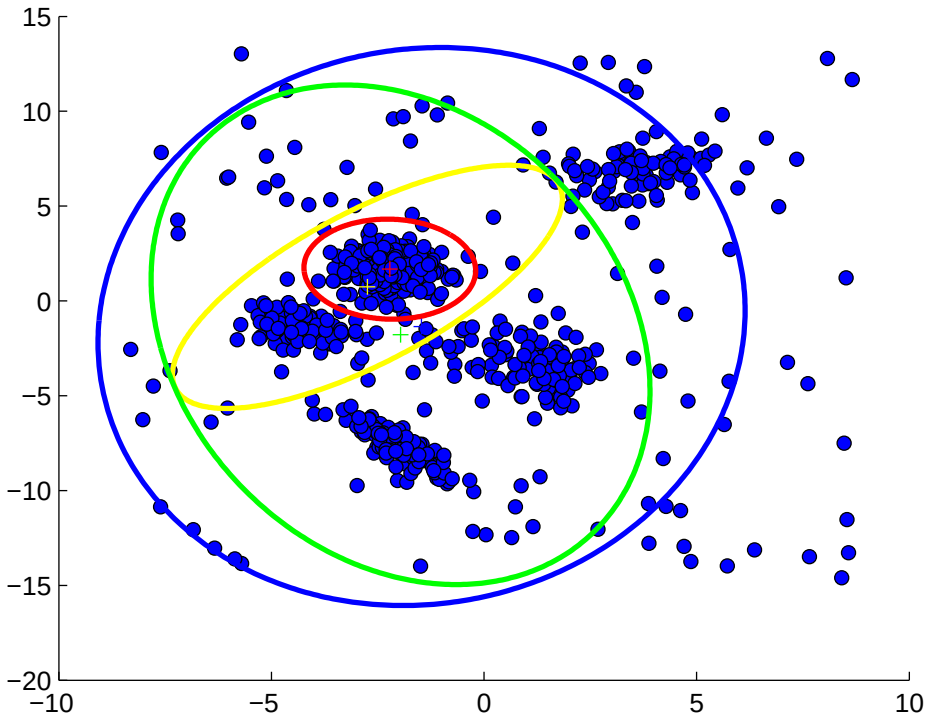


Figure 3.5: The models corresponding to strong local minima as shown in Figure 3.4 are drawn in red, yellow, green and blue ellipses. The model with the smallest $\widetilde{N}$ (177) is chosen as the best model, marked as the red ellipse.

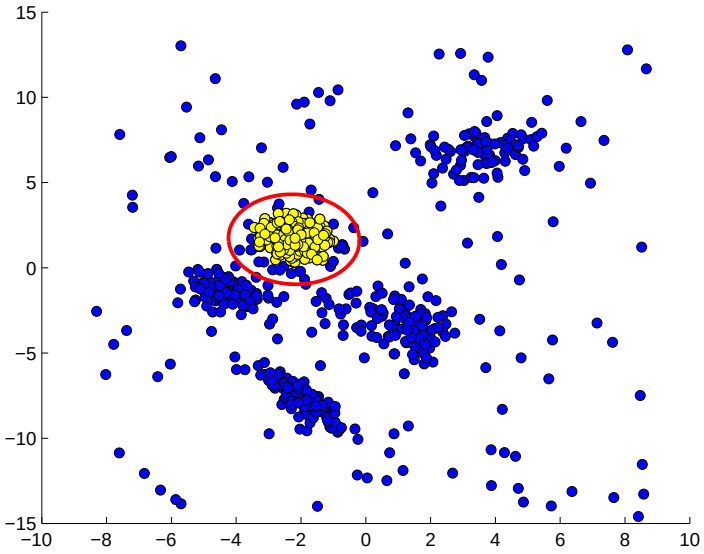
3.3 Refining $\widetilde{N}$

After $\widetilde{N}$ is selected and the corresponding Gaussian is estimated as in (3.6), the final step is to identify the data points that belong to this Gaussian. We use the extreme value theory [47] as

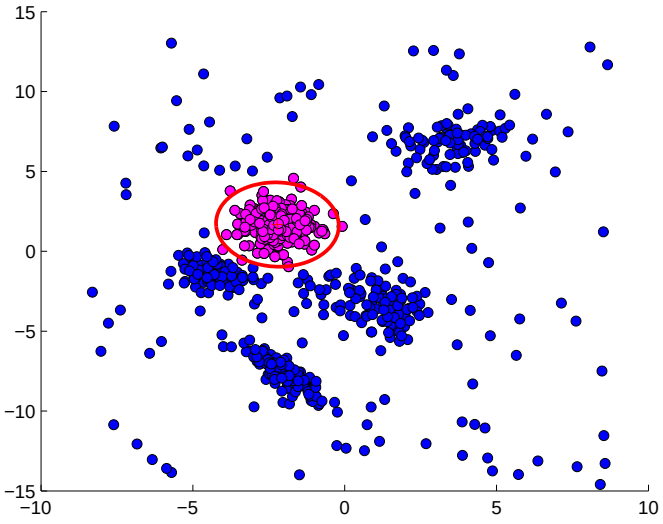
$$P_{extreme}(\mathbf{x}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \exp(-\exp(-\sqrt{(\mathbf{x}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu})})) \tag{3.11}$$

to quantify the association of the point $\mathbf{x}_i$ to the Gaussian with parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, where smaller values of $P_{extreme}$ indicate that $\mathbf{x}_i$ is closer to the mean and larger values imply that it is closer to the tail. Even though $\widetilde{N}$ is a good estimate of the number of interesting points (with $z_i = 1$) that belong to the Gaussian, we use a threshold on (3.11) to recover from small errors in the estimation of $\widetilde{N}$ using the KL_{reg} score under the presence of strong clutter from non-Gaussian distributed uninteresting points.

The interesting points as $\widetilde{N}$ data points with largest log-likelihood values and as data points whose $P_{extreme}$ values (3.11) smaller than a fixed threshold are shown on the illustrative data set in Figure 3.6. While the former includes the points around the mean, the latter also contains the points that lie in the tails of the Gaussian.



(a) The $\widetilde{N}$ points with largest log-likelihood values are marked in yellow



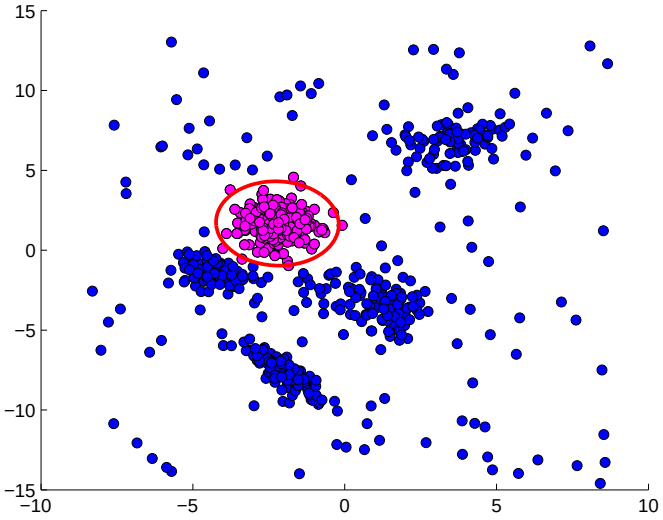
(b) The points selected by EVT with a threshold of 0.94 are marked in magenta

Figure 3.6: EVT can handle the data points that lie in the tails of the distribution as given in (b) as opposed to (a). Both Gaussians have the same parameters and are drawn at three standard deviations.

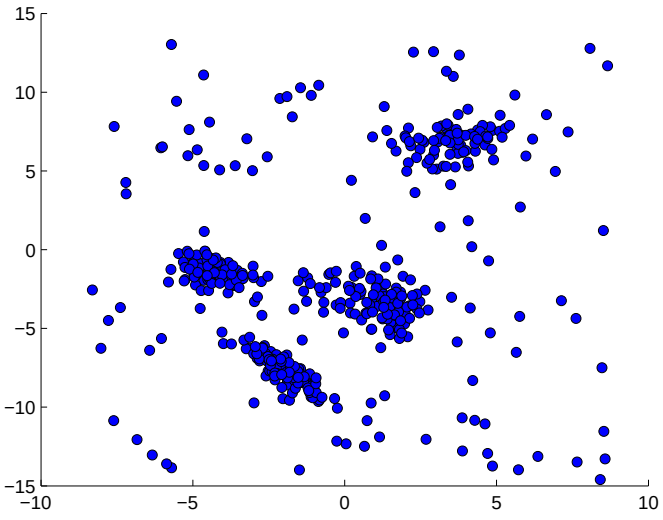
Chapter 4

Robust Gaussian Mixture Model

Previously, we estimated a robust Gaussian model and obtained the data points that belong to that density. We now go one step further and extend this framework to a mixture of robust Gaussian models. The robust Gaussian model estimated in the previous chapter corresponds to one of the components in the mixture. In order to estimate each component as a single robust Gaussian, we need to remove its respective data points after its estimation. The removal process on the illustrative data set is shown in Figure 4.1. We estimate a new robust Gaussian from the remaining data set with the same procedure until there are no components left. However, we do not have the information of the number of components in the mixture. Therefore, we apply a stopping criterion to determine if all the components in the mixture are discovered. The stopping criterion automatically halts the algorithm and also helps to identify the number of components in the mixture model. The stopping criterion will be given in Section 4.1 and we will present the way we incorporate the stopping criterion into finding the optimal number of components in Section 4.2. Lastly, we describe the calculation process of the mixing coefficients of the robust Gaussian mixture model in Section 4.3.



(a) The points marked in magenta are chosen by EVT as belonging to the Gaussian shown as the red ellipse. Gaussian is drawn at three standard deviations.



(b) The remaining data set with the removal of points shown in Figure 4.1(a) marked in magenta.

Figure 4.1: Data removal process after the estimation of a robust Gaussian model.

4.1 Stopping Criterion

One of the two main contributions of this work is the identification of the true number of components. Obviously, we do not know how many robust Gaussian models that we should estimate or when to stop the algorithm as we do not know the number of components in the mixture model. We propose a simple yet powerful method to handle this problem by exploiting the fact that the interesting points in the data set come from a mixture of Gaussians. The stopping criterion is applied as follows. After estimating a robust Gaussian model, we find the points that belong to this model. Then, we apply a Gaussianity test on those data points to determine whether they actually come from a Gaussian distribution. This method is simple as it does not introduce extra parameters(we fix the significance level as its preferred default value; 0.05) and it is powerful because of the assumption we make regarding the interesting points.

There are many different procedures that have been proposed for testing multivariate Gaussianity. Unfortunately, conclusions vary from one test to another about the Gaussianity of a data set by different procedures [48]. We chose the Royston's test [49] which is the multivariate extension of the well-known Shapiro-Wilk test [50] for the stopping criterion. Shapiro-Wilk test is considered to be a very reliable way of determining the Gaussianity of a set of points and it has the highest power among other Gaussianity tests [51]. Royston took the same idea and extended its framework to make it available for multivariate sets. The idea behind Royston's test can be summarized as follows. Let W_j denote the value of the Shapiro-Wilk test statistic for the jth variable in a p-variate distribution. Royston's test is defined as

$$R_j = \left\{ \Phi^{-1} \left[\frac{1}{2} \Phi \left\{ - \left((1 - W_j)^{\lambda} - \mu \right) / \sigma \right\} \right] \right\}^2 \tag{4.1}$$

where λ, μ and σ are calculated from polynomial approximations given in [50] and $\Phi(\cdot)$ is the standard normal cumulative density function(cdf). If the set of points comes from a multivarite Gaussian, $H = \xi \sum R_j / p$ is approximately $\chi^2_{\hat{\xi}}$ distributed where

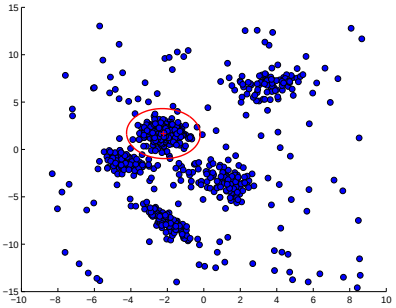
$$\hat{\xi} = p/[1 + (p-1)\bar{c}] \tag{4.2}$$

and $\bar{c}$ is an estimate of the average correlation among the R_j's. [49]. The $\chi^2_{\hat{\xi}}$ distribution is used to obtain the critical or P-value for the test.

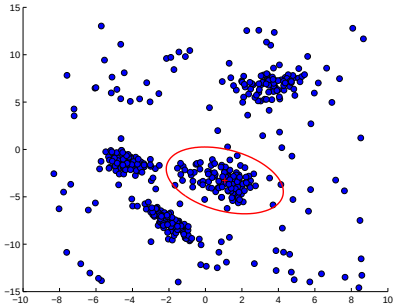
According to our previous assumption, interesting points come from a mixture of Gaussians and uninteresting ones are non-Gaussian distributed. If the hypothesis on the Gaussianity of the set of points is rejected, it means that those points are uninteresting. But more importantly, it also means that there are no interesting points left in the data set. Hence, once the Gaussianity hypothesis is rejected, the algorithm stops searching for a new robust Gaussian model. On the other hand, if the hypothesis is accepted, it means that we have discovered a new component in the mixture. Thus, we remove those points from the data set, then store them along with the parameters of the robust Gaussian model. After that, the algorithm goes on to estimate another robust Gaussian model until the stopping criterion is met. Estimated robust Gaussians from the illustrative data set until the stopping criterion is met are presented in Figure 4.2.

4.2 Optimal Number of Components

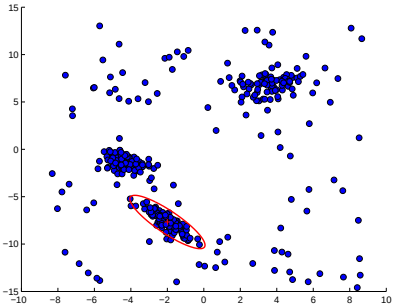
If the Gaussianity hypothesis on a set of selected points is accepted, we discover one component in the mixture that the interesting points are generated from. As we estimate a robust Gaussian at each iteration, we also identify a component in the process as well. Once the stopping criterion is met, we conclude that both the selected points and the points left in the data set are uninteresting. In other words, we have already discovered each component in the mixture, hence there are no interesting points left in the data set. This gives birth to a natural conclusion as well. Once the stopping criterion is met, the number of iterations up to that point actually corresponds to the number of components in the mixture.



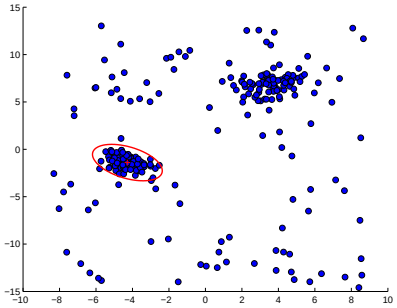
(a) The first estimated robust Gaussian. Hypothesis accepted with P-value: 0.4810.



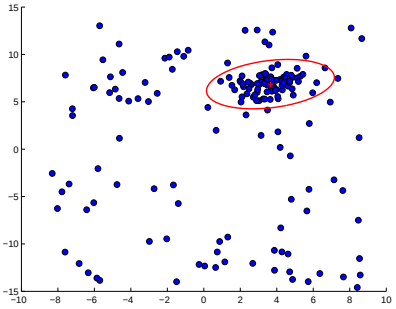
(b) The second estimated robust Gaussian. Hypothesis accepted with P-value: 0.5295.



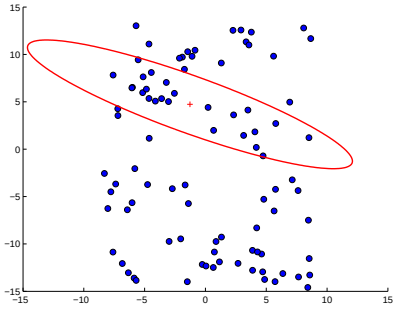
(c) The third estimated robust Gaussian. Hypothesis accepted with P-value: 0.5636.



(d) The fourth estimated robust Gaussian. Hypothesis accepted with P-value: 0.3251.



(e) The fifth estimated robust Gaussian. Hypothesis accepted with P-value: 0.4457.



(f) The sixth estimated robust Gaussian. Hypothesis rejected with P-value: 0.0423.

Figure 4.2: Estimated robust Gaussians in each iteration are shown on the illustrative data set through (a), (b), (c), (d), (e), (f) in red ellipses. Stopping criterion is met on the sixth estimated Gaussian where Royston's test rejected the hypothesis.

4.3 Mixing Coefficients

The algorithm terminates when the stopping criterion is met. After that, we have the information of the following; the number of robust Gaussians K, robust Gaussian models with parameters $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$ and data points that belong to each robust model with sizes $\widetilde{N}^{(1)}, \ldots, \widetilde{N}^{(K)}$ with $\sum_k \widetilde{N}^{(k)} = \widetilde{N}$.

We can think of each estimated robust Gaussian as one component in the mixture. Now that we have the model parameters and the information of which point comes from which model, we can easily incorporate these knowledge into computing the mixing coefficients of the robust Gaussian mixture model as

$$\alpha_k = \frac{\widetilde{N}^{(k)}}{\widetilde{N}}. \tag{4.3}$$

With the calculation of mixing coefficients, we now complete the estimation of robust Gaussian mixture model which is fully defined by the set of parameters $\boldsymbol{\Theta} = \{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

Finally, the resulting robust GMM estimated from the illustrative data set can be seen in Figure 4.3.

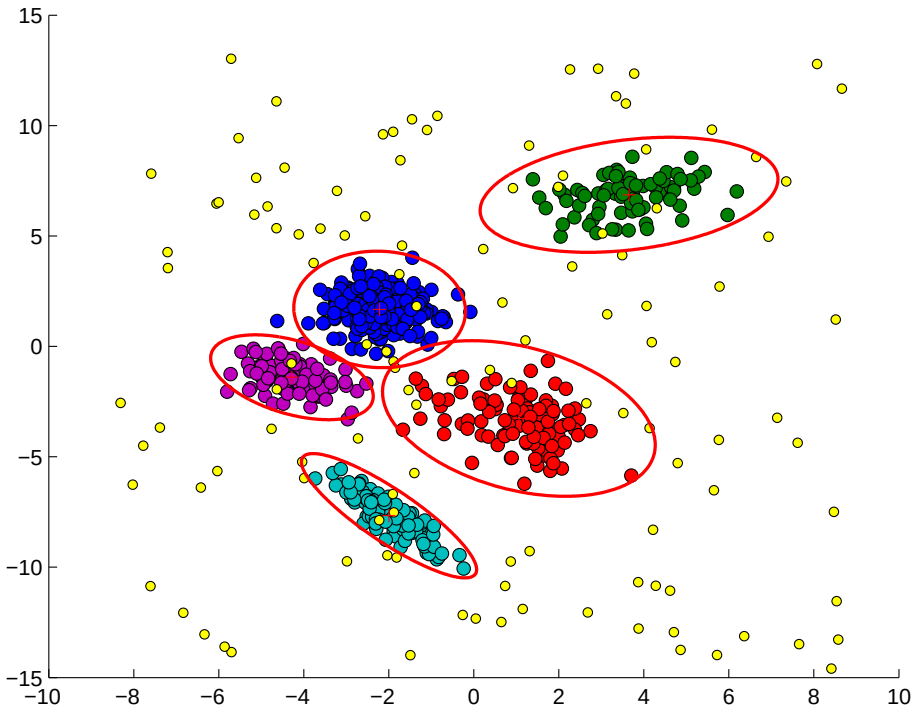


Figure 4.3: The illustrative data set is overlaid with the estimated robust GMM as red ellipses drawn at three standard deviations.

Chapter 5

Experiments

5.1 Experimental Setup

We did experiments on both synthetic and real-world data sets. For synthetic data sets, we define generating model of a data set as the model that the interesting points are sampled from. Unfortunately, we do not know the generating model parameters of the real-world data sets that are used in the experiments. Thus, we define the generating model of a real-world data set as the model estimated from the data set with the help of available class information of each data point.

The performance criterion is the difference in log-likelihood values between the generating model and the model obtained from competing algorithms. We compare our results with the traditional EM algorithm and the Efficient Greedy Learning of Gaussian Mixtures [37] algorithm, which will be referred to as GLM from now on. As a result, an algorithm performs better if its difference in log-likelihood is closer to zero.

5.2 Experiments on Synthetic Data Sets

5.2.1 Synthetic Data Set Generation

In the first step of experiments, we generate synthetic data sets with various configurations which are comprised of training and test sets. Training sets are composed of interesting as well as uninteresting points. On the other hand, test sets are composed of only interesting points which are sampled from the mixture model that is used to generate the interesting points of the corresponding training set. Hence, a test set and the interesting points in the corresponding training set come from the same mixture distribution. The test set and the interesting points in the training set are generated with the following settings. We generated $\widetilde{N} \in \{400, 800, 1200\}$ interesting points in $\mathbb{R}^d$, $d \in \{2, 3, 5\}$. These points are sampled from Gaussian mixtures with $K \in \{4, 6, 8\}$ components. We controlled the separation of components by adjusting the c-separation value introduced by [52]. Any two Gaussian distributions $p(\mathbf{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $p(\mathbf{x}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ are c-separated if

$$||\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2||_2 \leq c\sqrt{d\ max\{\lambda_{max}(\boldsymbol{\Sigma}_1), \lambda_{max}(\boldsymbol{\Sigma}_2)\}}$$

We used $c \in \{4.0, 6.0, 8.0\}$ with eccentricity values $e \in [15, 175]$ which is the ratio of the largest singular value of the covariance matrix to the smallest one. c-separation value around 8.0 results in well separated components, while a value around 4.0 yields components that are very close to one another and are fairly difficult to differentiate even by visual inspection. The overall data set settings are given in Table 5.1.

On the other hand, uninteresting points are sampled from a Uniform distribution, $Uniform[0, 250]^d$. We generate 10 sets of uninteresting points with various sizes for each interesting set. Number of points in uninteresting sets is proportional to the number of points in the corresponding interesting set; smallest uninteresting set size is 5% of the interesting set and the largest is half the size of the interesting set. Overall, we generate training sets by combining each interesting set with 10 different uninteresting sets, resulting in a total of 180 training sets. Also, we have 18 test sets that are generated from the mixture

Table 5.1: Synthetic data set settings

Setting#	d	$\widetilde{N}$	K	c	e
1	2	400	4	8	15
2	2	400	6	8	15
3	2	400	8	8	15
4	3	400	4	8	15
5	3	400	6	8	15
6	5	400	4	8	15
7	2	600	6	6	25
8	2	600	8	6	25
9	3	600	4	6	25
10	3	600	6	6	25
11	5	600	4	6	75
12	5	600	6	6	75
13	2	800	8	4	35
14	3	800	6	4	35
15	3	800	8	4	35
16	5	800	4	4	125
17	5	800	6	4	125
18	5	800	8	4	175

models from which we sampled the interesting points in the training sets. We kept the size of the test sets the same with the training sets, as well. E.g. for setting #1 in Table 5.1, we have 10 training sets on which we run an algorithm and obtain 10 different models. Then, we test each model on the same test set so that we can determine the impact of increasing number of uninteresting points to the algorithm. Also note that, for each training and test setting, we generate 50 randomly initialized data sets in order to reduce the effect of randomness. All in all, we make experiments on 9000 distinct training sets and 900 distinct test sets.

5.2.2 Best γ Selection

There are two criteria for choosing the best γ value. The first criterion is the performance in finding the true number of components and the second criterion is the performance in minimizing the log-likelihood difference. Based on these

criteria, we will select a γ value that gives the best overall performance on all settings in Table 5.1 and then select best γ values for different types of data sets, as well.

5.2.2.1 Finding True Number of Components

As previously indicated, we have two contributions to the classical MLE of GMMs via EM. One of them is that we do not require the true number of components to be known a priori in the estimation of parameters in RGMM. Once we obtain a robust Gaussian, we check whether its respective data points come from a Gaussian distribution or not via a multivariate Gaussianity test. If the hypothesis is rejected, then the algorithm stops searching for new robust Gausssians. We conclude that removed points are uninteresting and that we have found each component in the mixture once the stopping criterion is met.

As one of the performance criteria, we count the total number of cases where the number of components in RGMM correctly matches the number of components in the generating model and where they fail to agree. The failures can be divided into three categories. The first category is when the algorithm cannot reach the number of components in the generating model and estimates a model with fewer number of components. The second category is when the algorithm exceeds the number of components in the generating model and estimates a model with higher number of components. Lastly, the third category simply suggests that the algorithm gives up after discovering one component by concluding that the discovered set does not come from a Gaussian. This case can be seen when more than one component in the mixture merge into a larger component with a non-Gaussian geometry or when large amounts of uninteresting points change the structure of interesting points. After we count the occurrences of each of these cases, we divide them by the total number of cases in order to obtain the corresponding percentages.

Our algorithm requires only one parameter, γ, which controls the volume of the robust Gaussian model. We will present our results regarding the search

for the true number of components for γ values between 0.15 and 0.40. We conduct the experiments as follows. The number of components in a training set and in its corresponding test set is the same as the interesting points in the training set and the points in the test set come from the same Gaussian mixture. Thus, we simply check whether our algorithm's output of mixture has the same number of components as the generating model of each training set. As previously mentioned, we have a total of 9000 training sets. We will inspect the results in four-folds; over 2-dimensional training sets (3000 sets and models), over 3-dimensional training sets (3000 sets and models), over 5-dimensional training sets (3000 sets and models) and finally over all training sets (9000 sets and models).

For each γ value from 0.15 to 0.40, we present the results for 2-dimensional sets, 3-dimensional sets, 5-dimensional sets and all settings combined through Figures 5.1, 5.2, 5.3, 5.4, respectively. The green bars correspond to the number of cases agreeing in the number of components. In order to make its change with the increasing γ values seen easier, we overlay the figures with a white line. Hence, the length of a green bar equals to the value of the white square cursor, for a specific γ value. The yellow bars denote the case where the RGMM has more components than the generating model whereas the blue bars denote the opposite. Lastly, the black bars correspond to the cases where the algorithm quits after the first iteration due to the reasons indicated above.

The results for 2-dimensional sets are given in Figure 5.1. We can see that as the γ value increases, the number of RGMMs with both fewer and larger number of components decrease. However, the number of unsuccessful models goes beyond 10% of all sets as the γ value goes above 0.30 and a γ value around 0.40 yields unsuccessful models in most of the cases. The number of RGMMs with correct number of components peaks at $\gamma = 0.28$. For this value, 65.80% of all RGMMs estimated from 2-dimensional training sets have the same number of components with the generating model. The percentage of estimated RGMMs with too many components is 5.13% whereas 19.80% of them have fewer number of components than the generating model. Additionally, in 9.27% of the sets, the algorithm cannot estimate a model properly. Note that, even though RGMMs

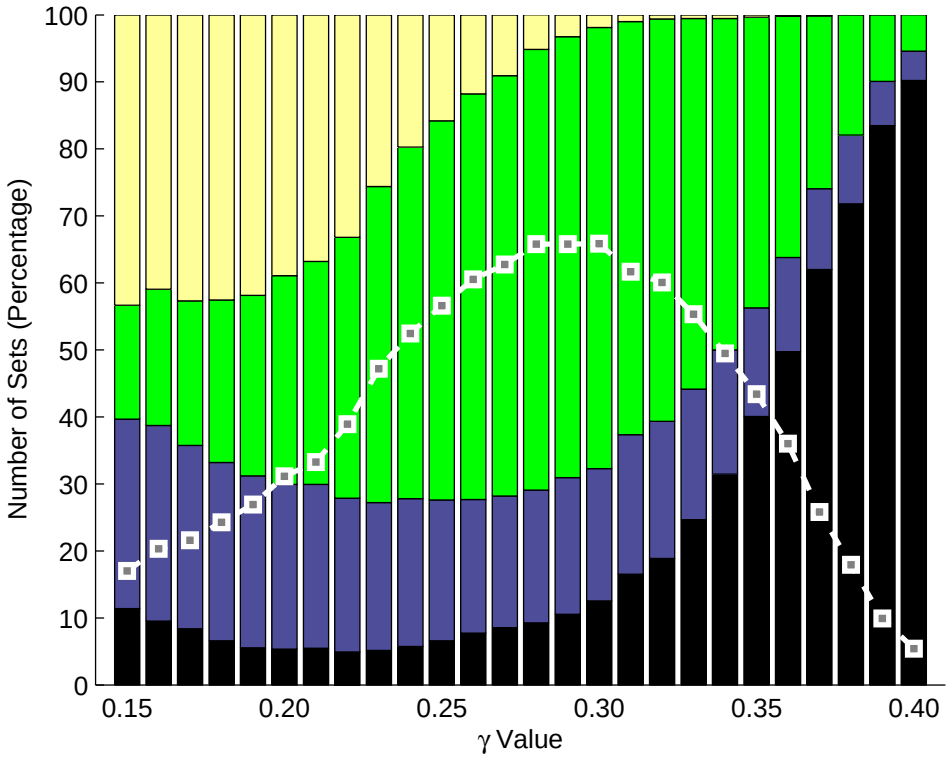


Figure 5.1: Percentage of the number of matches in the number of components between the generating models and the RGMMs estimated from the 2-dimensional training sets. The green bars denote the correct matches. The yellow bars correspond to the cases where RGMM estimates more components than the generating model while the blue bars denote the opposite. The black bars correspond to the unsuccessful models. The number of true matches with respect to different γ values are also shown with the white line.

with $\gamma = 0.30$ also produce comparatively good results, the number of unsuccessful models is higher. While models with $\gamma = 0.28$ cannot deal with approximately 9.27% of the sets, models with $\gamma = 0.30$ fails at more than 12.50%. The best setting among the 2-dimensional sets is the first setting in Table 5.1 with 82.60% of the estimated models from a total of 500 sets match the number of components in the generating model and only 3.40% of the estimates are unsuccessful models. This result is expected as this setting has the highest c-separation with the fewest number of components and the lowest eccentricity among the settings with 2-dimensional sets.

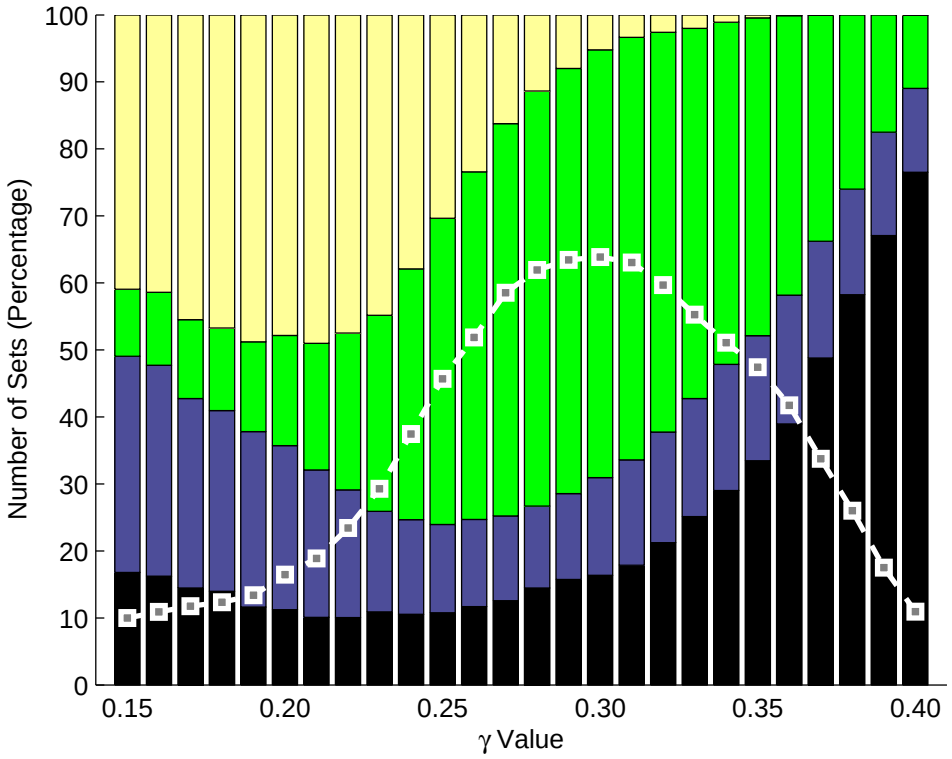


Figure 5.2: Percentage of the number of matches in the number of components between the generating models and the RGMMs estimated from the 3-dimensional training sets. The green bars denote the correct matches. The yellow bars correspond to the cases where RGMM estimates more components than the generating model while the blue bars denote the opposite. The black bars correspond to the unsuccessful models. The number of true matches with respect to different γ values are also shown with the white line.

The results for 3-dimensional sets are given in Figure 5.2. We can see that as the γ value increases, the number of RGMMs with fewer components starts to decrease to as low as 12% and then stabilizes at around 17%. On the other hand, the number of RGMMs with too many components starts to decrease from $\gamma = 0.21$ and onward. Almost in the same fashion as the results of 2-dimensional sets, the number of unsuccessful models first decreases and then goes beyond 10% of all sets as the γ value goes above 0.24 and a γ value around 0.40 yields unsuccessful models in 77% of all 3-dimensional training sets. The number of RGMMs with true number of components peaks at $\gamma = 0.30$. For this value, 63.87% of all RGMMs estimated from 3-dimensional training sets have the same

number of components with the generating model. The percentage of RGMMs with too many components is 5.20% whereas 14.56% of them have fewer number of components than the generating model. Additionally, in 16.37% of the sets, the algorithm cannot estimate a model properly. The best setting among the 3-dimensional sets is the fourth setting in Table 5.1 with 89.60% of the estimated models (from a total of 500 sets; 50 different seeds with each having 10 distinct sets of uninteresting points) match the number of components in the generating model and only 3.60% of the models are unsuccessful. As previously stated for 2-dimensional sets, this result is also expected as the fourth setting has the highest c-separation with the fewest number of components and the lowest eccentricity among the settings with 3-dimensional sets.

The results for 5-dimensional sets are given in Figure 5.3. The behavior of RGMMs with fewer and larger number of components resemble the results seen in 3-dimensional sets. We can see that as the γ value increases, the number of RGMMs with fewer components starts to decrease to as low as 12% and then stabilizes at around 19%. On the other hand, the number of RGMMs with too many components starts to decrease from $\gamma = 0.23$ and onward. Almost in the same fashion as the results of 2-dimensional and 3-dimensional sets, the number of unsuccessful models first decreases to as low as 10% of all sets for $\gamma = 0.24$, then starts to increase. A γ value around 0.40 yields unsuccessful models in 69% of 5-dimensional training sets. The number of RGMMs with true number of components peaks at $\gamma = 0.32$. For this value, 62.03% of RGMMs estimated from 5-dimensional training sets have the same number of components with the generating model. The percentage of RGMMs with too many components is 5.77% whereas 12.8% of them have fewer number of components than the generating model. Additionally, in 19.40% of the sets, the algorithm cannot estimate a model properly. The best setting among the 5-dimensional sets is the sixth setting in Table 5.1 with 85.60% of the models estimated from a total of 500 distinct training sets match the number of components in the generating model and only 3.00% of the models are unsuccessful. Similar to the previous cases, this result is expected as the sixth setting has the highest c-separation with less number of components and low eccentricity among the settings with 5-dimensional sets.

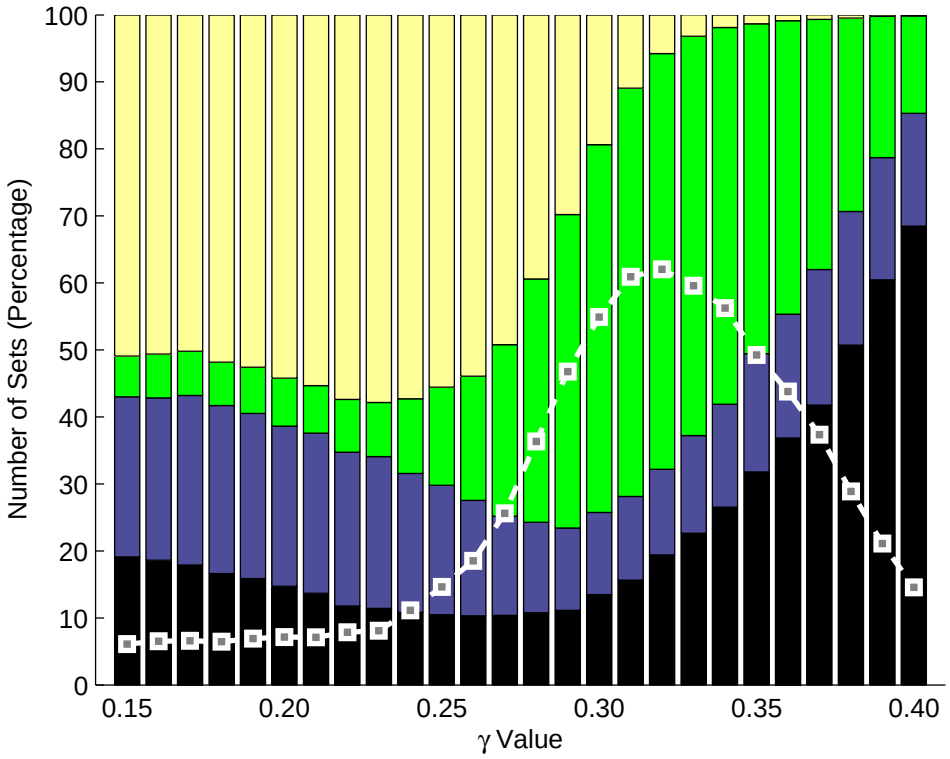


Figure 5.3: Percentage of the number of matches in the number of components between the generating models and the RGMMs estimated from the 5-dimensional training sets. The green bars denote the correct matches. The yellow bars correspond to the cases where RGMM estimates more components than the generating model while the blue bars denote the opposite. The black bars correspond to the unsuccessful models. The number of true matches with respect to different γ values are also shown with the white line.

The overall results for all settings of training sets combined are given in Figure 5.4. This result can be seen as the combination of the results of 2-dimensional, 3-dimensional and 5-dimensional sets. Thus, we see that the number of RGMMs with fewer number of components decreases to 15% for $\gamma = 0.30$, then stays in the $15 - 20\%$ interval for higher γ values. The number of RGMMs with higher number of components shows a subtle increase for γ from 0.15 to 0.20, then it also decreases at a rapid pace. On the contrary, the number of unsuccessful models decreases to as low as 9% of all training sets for $\gamma = 0.22$, only to increase thereafter; it hits 79% for $\gamma = 0.40$. The number of RGMMs with the true number of components peaks at $\gamma = 0.30$. For this value, 61.54% of all RGMMs estimated

from all training sets have the same number of components with the generating model. The percentage of RGMMs with too many components is 8.81% whereas 15.51% of them have fewer number of components than the generating model. Additionally, in 14.13% of the training sets, the algorithm cannot estimate a model properly. Note that, even though RGMMs with $\gamma = 0.31$ also produce comparatively good results, the number of unsuccessful models is higher in this case. While models with $\gamma = 0.30$ cannot deal with approximately 14.1% of the training sets, models with $\gamma = 0.31$ fails at 16.7% of them. Also note that, while 2-dimensional and 3-dimensional set results do not show a steep performance decrease at larger γ values which favors 5-dimensional sets, the 5-dimensional set results are too low at γ values where especially 2-dimensional set results were better. As a result, the general output is rather biased towards the results of 3-dimensional and mostly 5-dimensional sets.

5.2.2.2 Minimizing Log-likelihood Difference

Our second contribution is the robustness of our algorithm to the presence of uninteresting data points. Contrary to the classical MLE of GMMs via EM, our algorithm can find and model only the interesting points in heterogeneous data sets. In this part, we will show how our algorithm performs in the presence of increasing number of uninteresting points for various data settings. As indicated before, for each 50 randomly initialized data setting, we have 10 training sets that share the same interesting points but differ in the number of uninteresting points. Additionally, we have a test set which is drawn from the same model that is used to generate the interesting points of the training sets. Hence, we learn a model for each training set and then apply it on the same test set to explore the impact of the increasing number of uninteresting points. Our performance criterion is the log-likelihood value of the estimated model computed on the test set. We then take its difference from the log-likelihood of the generating model computed on the test set and do comparisons based on this value.

Our algorithm has only one free parameter which is the γ value ranging from

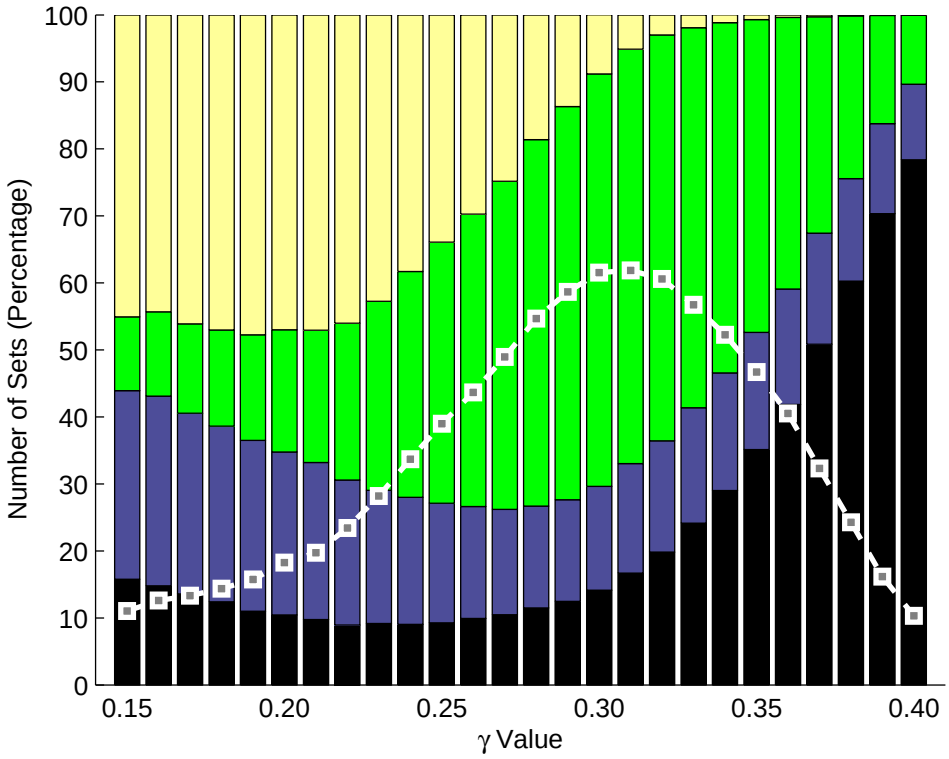


Figure 5.4: Percentage of the number of matches in the number of components between the generating models and the RGMMs estimated from all training sets. The green bars denote the correct matches. The yellow bars correspond to the cases where RGMM estimates more components than the generating model while the blue bars denote the opposite. The black bars correspond to the unsuccessful models. The number of true matches with respect to different γ values are also shown with the white line.

0.15 to 0.40. We present the density estimation results for each γ value. Previously, we presented the results by averaging over all sets of uninteresting points. However, this time we will also show how the algorithm performs as the number of uninteresting points increases since we are testing whether our algorithm is robust to the presence of increasing amounts of uninteresting points or not. Thus, in each case, the results will be presented for 10 levels of uninteresting points whose numbers range from 5% to 50% of the corresponding interesting points. Similar to the previous section, we will inspect the results in four-folds. First, we show the results over 2-dimensional sets (300 models estimated from 300 sets for each level of uninteresting points). Then, in the same fashion, we present the

results of 3-dimensional and 5-dimensional sets. Finally, we give the results for the case where all data settings are combined. Note that, the results from now on will only include the models that successfully identified the true number of components in our previous experiments. First, we present the results over the 2-dimensional sets. The performance with respect to the changing γ values and increasing number of uninteresting points can be seen in Figure 5.5. For small values of γ, the difference in log-likelihood, 3.398, is too high. As γ is increased up to a point, the algorithm gradually performs better. Once the γ value is set to anywhere between 0.26 and 0.29, the algorithm yields the smallest differences in log-likelihood values. For γ values greater than 0.30, the corresponding RGMMs starts to gradually decrease in performance, yielding higher differences in log-likelihood values. Aside from the very hectic models with very small γ values from 0.15 to 0.20, we can say that RGMM does not regress in performance in the presence of varying amounts of uninteresting points. As it can be seen from Figure 5.5, there is little to none increase in difference in log-likelihood values as the number of uninteresting points increase for a fixed γ value. For a more detailed view of both the impact of γ values and the increasing number of uninteresting points, please refer to Table 5.2. As a result, for 2-dimensional sets, we can conclude that RGMM with γ values set to 0.28 performs the best. This result actually coincides with the previous one where we concluded that RGMMs estimated with $\gamma = 0.28$ performed the best in terms of matching the true number of components in the generating model in 2-dimensional sets, as well.

Second, we share the results over the 3-dimensional sets. The results are more or less in agreement with the results of the 2-dimensional sets. There are higher difference in log-likelihood values for small γ values and the best results are achieved for γ values slightly higher than that of 2-dimensional sets. The estimation performance with respect to the changing γ values and the increasing number of uninteresting points can be seen in Figure 5.6. For small values of γ, the difference in log-likelihood, 5.986, is too high. As γ value is increased up to a point, the algorithm gradually performs better. Once the γ is set to anywhere between 0.27 and 0.30, the algorithm produces the smallest differences in log-likelihood values. For γ values greater than 0.31, corresponding RGMMs

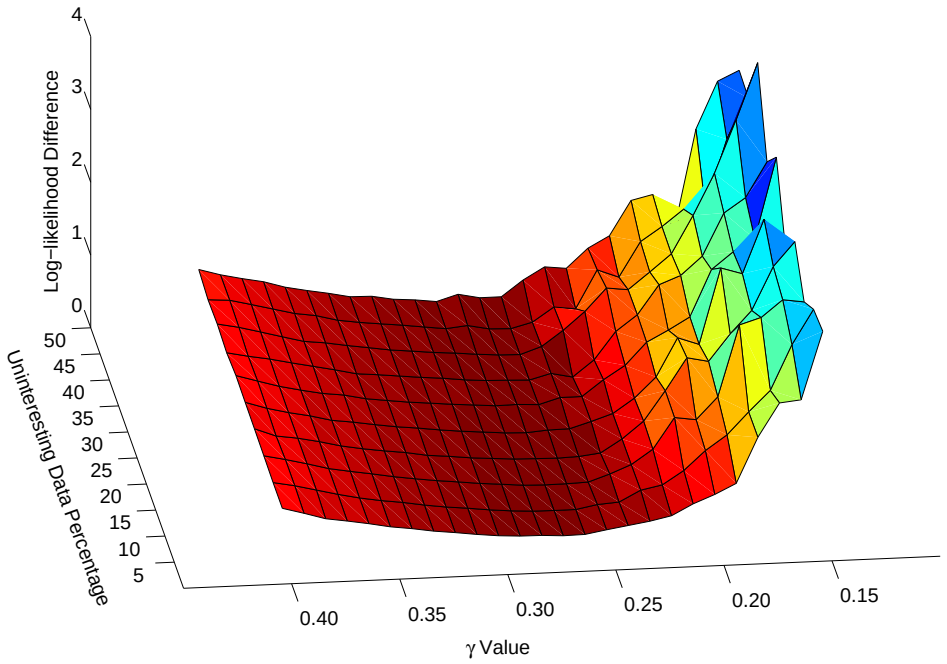


Figure 5.5: Differences in log-likelihood between the generating model and the RGMM estimated from 2-dimensional training sets with respect to various γ values and increasing number of uninteresting points.

Table 5.2: Differences in log-likelihood between the generating model and the RGMM estimated from 2-dimensional training sets with respect to various γ values and increasing number of uninteresting points.

γ Value \ Uninteresting Data Percentage	5	10	15	20	25	30	35	40	45	50
0.15	2.795	2.739	2.484	2.955	2.481	3.398	2.977	3.979	3.134	3.162
0.16	1.866	2.265	2.526	2.623	2.915	2.287	2.497	3.207	2.582	3.023
0.17	1.835	1.649	2.293	2.033	2.179	2.123	2.298	2.486	1.672	2.379
0.18	1.375	1.368	2.203	1.723	2.271	1.894	1.719	1.919	1.648	1.017
0.19	0.755	1.065	1.229	1.359	1.235	1.255	1.465	1.406	1.423	1.506
0.20	0.616	0.863	0.929	1.421	1.338	1.087	1.352	1.170	1.230	1.437
0.21	0.500	0.601	0.972	0.982	0.811	0.933	0.952	0.939	0.754	0.964
0.22	0.348	0.416	0.547	0.577	0.535	0.628	0.735	0.975	0.866	0.806
0.23	0.286	0.355	0.350	0.344	0.336	0.444	0.358	0.664	0.343	0.541
0.24	0.240	0.206	0.233	0.163	0.240	0.224	0.133	0.418	0.380	0.575
0.25	0.193	0.183	0.134	0.127	0.151	0.223	0.196	0.208	0.184	0.407
0.26	0.143	0.129	0.124	0.118	0.146	0.139	0.120	0.111	0.109	0.188
0.27	0.137	0.126	0.120	0.124	0.122	0.153	0.130	0.119	0.120	0.189
0.28	0.153	0.130	0.129	0.133	0.133	0.147	0.136	0.139	0.176	0.247
0.29	0.157	0.147	0.155	0.152	0.156	0.170	0.159	0.169	0.148	0.161
0.30	0.175	0.173	0.179	0.174	0.179	0.192	0.194	0.190	0.182	0.197
0.31	0.206	0.208	0.211	0.207	0.212	0.221	0.219	0.220	0.220	0.217
0.32	0.243	0.236	0.243	0.233	0.249	0.248	0.248	0.249	0.250	0.271
0.33	0.275	0.277	0.280	0.283	0.279	0.282	0.279	0.294	0.284	0.278
0.34	0.322	0.325	0.323	0.318	0.322	0.319	0.306	0.335	0.326	0.337
0.35	0.360	0.364	0.372	0.360	0.369	0.383	0.372	0.384	0.401	0.389
0.36	0.415	0.412	0.419	0.411	0.406	0.418	0.432	0.416	0.444	0.447
0.37	0.465	0.474	0.471	0.483	0.480	0.475	0.492	0.494	0.516	0.533
0.38	0.509	0.546	0.548	0.544	0.538	0.539	0.557	0.571	0.591	0.588
0.39	0.596	0.618	0.620	0.586	0.610	0.626	0.609	0.647	0.663	0.653
0.40	0.676	0.673	0.682	0.697	0.703	0.713	0.667	0.706	0.682	0.740

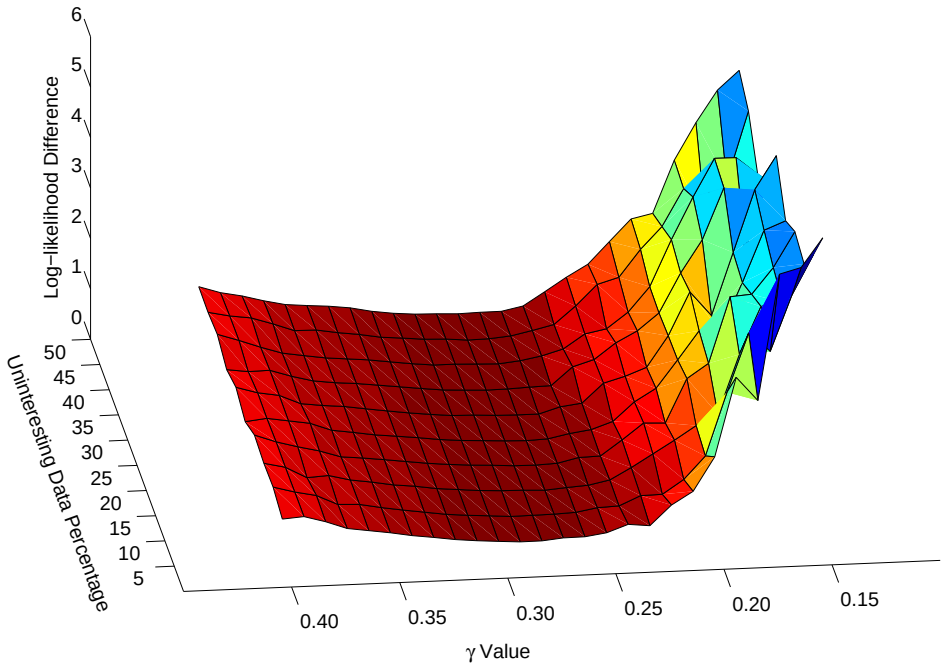


Figure 5.6: Differences in log-likelihood between the generating model and the RGMM estimated from 3-dimensional training sets with respect to various γ values and increasing number of uninteresting points.

Table 5.3: Differences in log-likelihood between the generating model and the RGMM estimated from 3-dimensional training sets with respect to various γ values and increasing number of uninteresting points.

γ Value	Uninteresting Data Percentage									
	5	10	15	20	25	30	35	40	45	50
0.15	5.986	5.154	4.467	4.609	4.325	5.116	4.136	3.744	4.498	4.808
0.16	5.411	4.367	3.723	3.913	4.155	4.456	2.764	4.092	3.143	4.437
0.17	5.301	3.269	3.659	3.402	3.134	3.229	4.013	4.098	2.407	3.807
0.18	2.812	3.647	2.727	3.370	2.172	1.998	2.999	3.348	3.015	3.079
0.19	3.287	2.493	1.564	2.066	2.214	2.485	2.359	2.521	2.281	2.042
0.20	1.730	1.246	1.349	1.784	1.652	1.631	1.584	1.606	1.846	1.948
0.21	1.064	0.798	1.079	0.919	1.107	1.024	1.246	1.119	1.310	1.518
0.22	0.809	0.522	0.825	0.937	0.856	0.864	1.026	0.806	1.188	1.088
0.23	0.415	0.514	0.570	0.616	0.486	0.743	0.713	0.721	0.775	0.845
0.24	0.461	0.328	0.340	0.317	0.341	0.366	0.587	0.441	0.427	0.575
0.25	0.311	0.310	0.254	0.305	0.281	0.311	0.231	0.320	0.325	0.310
0.26	0.248	0.235	0.227	0.208	0.193	0.185	0.218	0.244	0.236	0.222
0.27	0.243	0.217	0.190	0.189	0.175	0.187	0.201	0.199	0.197	0.222
0.28	0.203	0.210	0.209	0.192	0.190	0.192	0.196	0.212	0.189	0.208
0.29	0.202	0.207	0.217	0.222	0.211	0.219	0.212	0.227	0.225	0.226
0.30	0.229	0.235	0.237	0.241	0.232	0.225	0.234	0.228	0.240	0.238
0.31	0.260	0.262	0.275	0.273	0.267	0.264	0.280	0.272	0.280	0.273
0.32	0.298	0.302	0.311	0.312	0.320	0.313	0.332	0.324	0.312	0.329
0.33	0.360	0.359	0.375	0.367	0.370	0.371	0.382	0.383	0.387	0.413
0.34	0.411	0.422	0.420	0.420	0.413	0.430	0.429	0.421	0.441	0.465
0.35	0.485	0.486	0.494	0.472	0.478	0.492	0.466	0.457	0.509	0.496
0.36	0.547	0.566	0.553	0.510	0.539	0.530	0.520	0.521	0.515	0.534
0.37	0.607	0.583	0.659	0.645	0.693	0.584	0.628	0.588	0.666	0.625
0.38	0.757	0.781	0.632	0.747	0.760	0.764	0.764	0.722	0.801	0.737
0.39	0.878	0.864	0.857	0.862	0.826	0.835	0.755	0.875	0.858	0.820
0.40	0.845	0.980	0.955	1.014	0.778	0.960	0.811	1.012	0.949	0.956

start to gradually decrease in performance, yielding higher differences in log-likelihood values. Aside from the very hectic models with very small γ values from 0.15 to 0.22, we can say that our algorithm does not regress in performance in the presence of varying amounts of uninteresting points. As it can be seen from Figure 5.6, there is little to none increase in difference in log-likelihood values as the number of uninteresting points increase for a fixed γ value. For a more precise look at the differences in log-likelihood values for different γ values with the increasing number of uninteresting points, please refer to Table 5.3. As a result, for 3-dimensional sets, we can conclude that RGMMs with γ values around 0.29 and 0.30 perform the best. This result actually coincides with the previous case where we concluded that RGMMs estimated with $\gamma = 0.30$ performed the best in terms of matching the true number of components in the generating model in 3-dimensional sets, as well.

Third, we give the results over the 5-dimensional sets. The results are more or less in agreement with the results of the previous sets. There are larger differences in log-likelihood values for small γ and the best results are achieved for γ values slightly higher than that of the previous sets. We now go on to elaborate on detailed results. The performance with respect to the changing γ values and the increasing number of uninteresting points can be seen in Figure 5.7. For small values of γ, the difference in log-likelihood, 8.584, is too high. As the γ value is increased up to a point, the algorithm gradually performs better. Once the γ value is set to anywhere between 0.30 and 0.33, the algorithm yields the smallest differences in log-likelihood values. For γ values greater than 0.33, the corresponding RGMMs start to gradually decrease in performance, yielding higher differences in log-likelihood values. Aside from the very hectic models with very small γ values ranging from 0.15 to 0.26, we can say that RGMM does not regress in performance in the presence of different amounts of uninteresting points. As seen from Figure 5.7, there is little to none increase in differences in log-likelihood values when the number of uninteresting points increases for a fixed γ value. For a more precise evaluation of both the γ values and the impact of the increasing number of uninteresting points, please refer to Table 5.4. As a result, for the 5-dimensional sets, we can conclude that RGMMs with γ values around 0.31 and

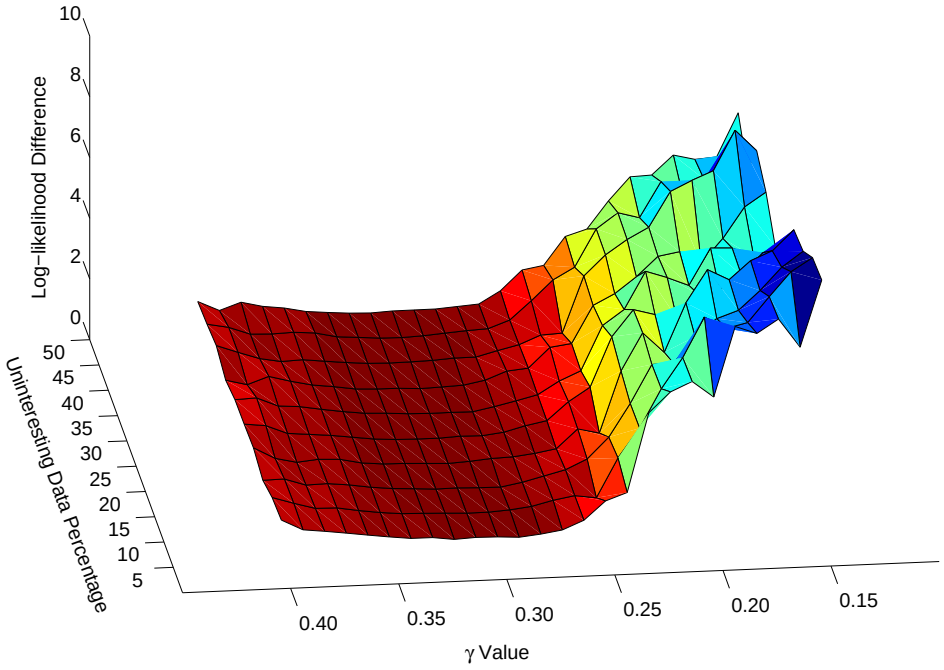


Figure 5.7: Differences in log-likelihood between the generating model and the RGMM estimated from 5-dimensional training sets with respect to various γ values and increasing number of uninteresting points.

Table 5.4: Differences in log-likelihood between the generating model and the RGMM estimated from 5-dimensional training sets with respect to various γ values and increasing number of uninteresting points.

γ Value \ Uninteresting Data Percentage	5	10	15	20	25	30	35	40	45	50
0.15	8.584	8.488	7.867	7.746	6.082	5.510	6.486	7.011	5.539	6.600
0.16	6.405	8.055	7.371	7.078	6.184	5.213	6.024	7.718	6.074	5.167
0.17	7.358	7.267	6.632	6.781	5.479	5.533	5.013	6.411	5.295	5.120
0.18	6.917	6.441	6.009	6.186	4.996	5.562	4.225	6.124	4.818	5.295
0.19	7.174	6.275	5.770	6.545	4.912	4.361	4.640	5.789	5.280	4.662
0.20	4.879	6.679	5.200	5.482	4.154	4.647	4.518	4.637	4.121	4.633
0.21	5.420	5.373	4.352	3.883	4.290	4.885	3.870	4.269	4.317	3.867
0.22	5.108	4.376	4.345	4.557	4.481	3.653	3.344	3.789	3.809	2.970
0.23	4.451	3.103	3.090	3.326	3.620	3.157	2.987	3.125	2.577	2.759
0.24	1.850	2.157	2.008	1.310	1.992	1.696	1.759	1.269	2.189	1.834
0.25	1.605	0.877	1.320	1.474	1.540	1.388	1.663	1.232	1.532	1.713
0.26	0.999	0.974	0.995	0.852	0.944	0.967	1.008	0.946	0.924	1.056
0.27	0.707	0.850	0.820	0.728	0.804	0.796	0.940	0.783	0.674	0.652
0.28	0.596	0.678	0.642	0.643	0.620	0.632	0.672	0.600	0.634	0.596
0.29	0.525	0.594	0.514	0.619	0.539	0.493	0.535	0.596	0.581	0.547
0.30	0.564	0.548	0.519	0.559	0.551	0.501	0.507	0.548	0.527	0.539
0.31	0.575	0.543	0.542	0.517	0.516	0.500	0.482	0.529	0.498	0.537
0.32	0.541	0.575	0.537	0.545	0.516	0.514	0.523	0.520	0.506	0.500
0.33	0.629	0.599	0.572	0.572	0.531	0.562	0.529	0.539	0.558	0.524
0.34	0.633	0.616	0.609	0.584	0.573	0.589	0.557	0.593	0.608	0.576
0.35	0.712	0.680	0.683	0.650	0.663	0.673	0.662	0.667	0.764	0.639
0.36	0.802	0.775	0.785	0.798	0.755	0.778	0.745	0.746	0.818	0.785
0.37	0.896	0.895	0.857	0.825	0.813	0.898	0.902	0.716	0.790	0.863
0.38	0.983	0.909	0.839	0.755	0.878	0.948	1.087	0.840	0.812	1.031
0.39	1.064	1.103	1.072	0.920	1.162	1.243	0.844	1.076	1.066	0.788
0.40	1.411	1.321	1.056	1.297	1.240	1.214	1.021	1.330	1.199	1.111

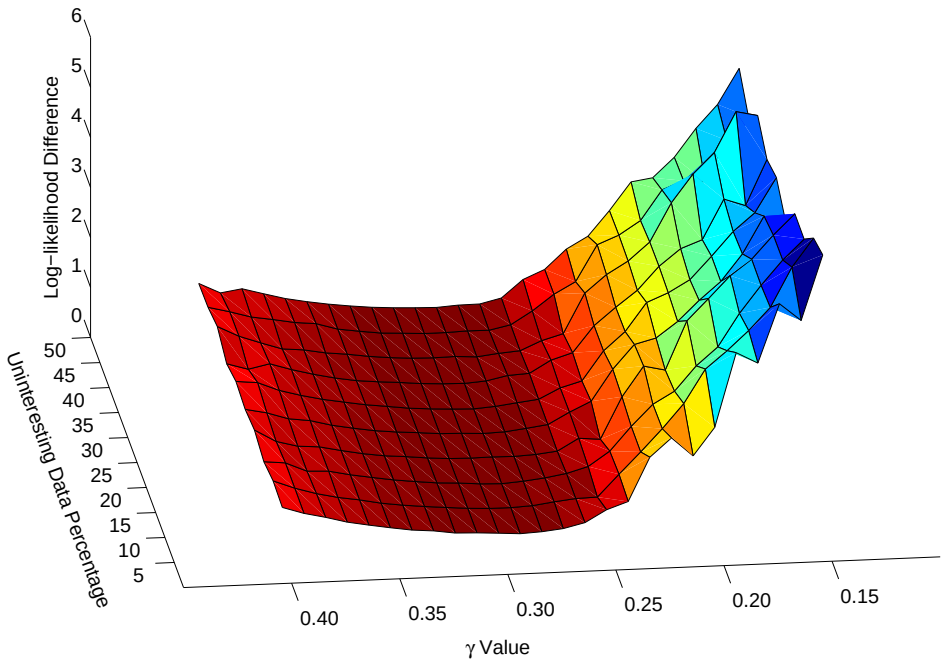


Figure 5.8: Differences in log-likelihood between the generating model and the RGMM estimated from all training sets with respect to various γ values and increasing number of uninteresting points.

0.32 performs the best. This result again coincides with our previous experiments where we concluded that a RGMM estimated with $\gamma = 0.32$ performed the best in terms of matching the true number of components in the generating model in 5-dimensional sets, as well.

Lastly, we give the results over all data settings combined. It can be seen as the combination of the results of the previously mentioned 2-dimensional, 3-dimensional and 5-dimensional cases. The performance with respect to the changing γ values and the increasing number of uninteresting points can be seen in Figure 5.8. For small γ values, the differences in log-likelihood are too high with the highest being 5.624. As the γ value is increased up to a certain point, the algorithm gradually performs better. Once γ value is set to anywhere between 0.28 and 0.32, the algorithm performs better with small differences in log-likelihood. For γ values more than 0.32, the corresponding RGMMs starts to gradually decrease in performance, yielding higher differences in log-likelihood values. Aside

Table 5.5: Differences in log-likelihood between the generating model and the RGMM estimated from all training sets with respect to various γ values and increasing number of uninteresting points.

γ Value	Uninteresting Data Percentage									
	5	10	15	20	25	30	35	40	45	50
0.15	5.624	5.460	4.939	4.948	4.296	4.674	4.533	4.912	4.391	4.857
0.16	4.330	4.896	4.540	4.538	4.418	3.985	3.629	5.005	3.933	4.153
0.17	4.683	4.108	4.051	4.072	3.681	3.628	3.775	4.209	3.125	3.690
0.18	3.512	3.664	3.507	3.759	3.146	3.152	2.981	3.797	3.160	3.130
0.19	3.739	3.277	2.854	3.323	2.787	2.701	2.821	3.239	2.995	2.737
0.20	2.263	2.929	2.493	2.896	2.382	2.455	2.485	2.471	2.399	2.673
0.21	1.710	2.257	2.004	1.928	2.069	2.281	2.023	2.109	2.127	2.116
0.22	2.088	1.771	1.762	2.024	1.957	1.715	1.702	1.857	1.954	1.621
0.23	1.717	1.324	1.337	1.429	1.481	1.348	1.353	1.504	1.232	1.382
0.24	0.850	0.897	0.860	0.596	0.857	0.762	0.827	0.709	0.999	0.995
0.25	0.703	0.457	0.570	0.636	0.657	0.641	0.696	0.587	0.681	0.810
0.26	0.464	0.446	0.449	0.393	0.428	0.430	0.449	0.403	0.423	0.455
0.27	0.363	0.398	0.377	0.347	0.367	0.379	0.424	0.367	0.330	0.354
0.28	0.317	0.339	0.327	0.323	0.314	0.324	0.335	0.317	0.333	0.351
0.29	0.295	0.316	0.295	0.331	0.302	0.294	0.302	0.340	0.318	0.320
0.30	0.322	0.319	0.312	0.325	0.321	0.306	0.311	0.330	0.316	0.332
0.31	0.347	0.338	0.343	0.332	0.332	0.328	0.333	0.340	0.339	0.350
0.32	0.361	0.371	0.364	0.363	0.368	0.358	0.375	0.371	0.362	0.379
0.33	0.421	0.412	0.409	0.408	0.400	0.412	0.404	0.412	0.417	0.421
0.34	0.455	0.454	0.458	0.441	0.443	0.462	0.446	0.473	0.475	0.475
0.35	0.519	0.510	0.525	0.502	0.511	0.533	0.516	0.518	0.603	0.542
0.36	0.588	0.595	0.595	0.583	0.587	0.607	0.610	0.595	0.644	0.624
0.37	0.667	0.661	0.688	0.674	0.699	0.703	0.720	0.633	0.720	0.735
0.38	0.780	0.756	0.703	0.704	0.753	0.819	0.852	0.773	0.788	0.851
0.39	0.877	0.880	0.887	0.817	0.889	0.993	0.762	0.922	0.912	0.787
0.40	1.011	1.045	0.929	1.064	0.975	1.034	0.888	1.133	1.018	0.991

from the very hectic models with very small γ values ranging from 0.15 to 0.25, we can say that the algorithm does not regress in performance in the presence of varying amounts of uninteresting points. As it can be seen from Figure 5.8, there is little to none increase in differences in log-likelihood as the number of uninteresting points increases for a fixed γ value. For a more precise evaluation of both the γ values and the impact of the increasing number of uninteresting points, please refer to Table 5.5. For all settings combined, we can state that RGMMs with γ values around 0.29 and 0.30 perform the best. This result also coincides with the previous experiments where we concluded that RGMMs estimated with $\gamma = 0.30$ performed the best in terms of matching the true number of components in the generating model in all data settings combined.

In conclusion, we can say that the results we obtained in this section are in line with the results found in our previous experiments. As a result, if γ value is chosen carefully, the estimated RGMM will not only correctly finds the true number of components of the generating model but also yield the best density estimate.

5.2.3 Comparison with Other Competing Algorithms

In this section, we conducted the experiments for the EM, GLM and RGMM algorithms over the synthetically generated data sets. Parameters in the traditional EM algorithm are initialized as follows. Provided that the true number of components K is given, mean vectors are determined by the k-means algorithm. Covariance matrices are computed from the corresponding Voronoi regions, and the mixing weights are initialized proportional to the points in each region. Even though GLM is posed as an algorithm that does not need the exact number of components, it still requires an upper limit for the number of components. In our experiments, we provided the true number of components K to the algorithm. In the proposed algorithm referred to as RGMM from now on, we initialize the mean vector randomly as the order of the discovered components does not matter. The covariance matrix is computed from N_0 points around the mean with the largest likelihood values.

Performance comparison with other competing algorithms will be done in two steps. The first step will include the log-likelihood differences between the generating model and the models learned from each competing algorithm. The smaller an algorithm outputs a log-likelihood difference, the better its performance is. Then, we will evaluate the impact of uninteresting set sizes in the second step. What we are evaluating here is whether the algorithm is affected by the increasing size of uninteresting points.

We will present our results in two-folds. First, we have chosen the γ parameter of our algorithm as 0.30 for each setting. Thus, we estimated a RGMM for each training set with the same γ value. The reason behind the selected value of the γ parameter stems from the fact that RGMMs estimated with $\gamma = 0.30$ have shown the best results in terms of both finding the true number of components of the generating model and the difference in log-likelihood values, as presented in our previous experiments. Second, we will split the training and test sets into three groups according to the dimension information at hand. For each group, we will provide the best γ value based on our previous experimental results. The γ parameters of RGMMs for 2-dimensional, 3-dimensional and 5-dimensional sets are set to 0.28, 0.30 and 0.32, respectively. In the most ideal case, we could instead partition data settings according to their structural information, i.e., whether they are compact or sparse/scattered. Then, we could set γ values with respect to this information. However, in almost all problems, this information is not readily available. It still can be extracted with certain preprocessing techniques but the process could be too taxing to make the effort worthwhile. Hence, we are using the readily available information which is the dimension of a set instead.

In the first part, we will set the γ parameter of RGMM to 0.30 for each data setting and compare our results with other competing algorithms. As previously stated, an algorithm performs better if its difference in log-likelihood is closer to zero. Moreover, we can also evaluate the stability of an algorithm, as we have 50 randomly initialized versions of each setting. We define the stability of an algorithm over a data setting as follows. If results of an algorithm across the randomly initialized 50 versions of a data setting are consistent, we deem the algorithm as stable in that particular data setting. Overall performance and

stability of each algorithm can be viewed on Figure 5.9.

By the overall performance in terms of the log-likelihood differences, we can deduce that the best performing algorithm is the RGMM out of the three. In almost all of the settings, the log-likelihood values obtained from the RGMM are closer to the log-likelihood directly obtained from the generating model. Hence, the log-likelihood differences of our algorithm are closer to zero. However, the performance of RGMM takes a hit when the components in the mixture are poorly separated and the data is sparse or scattered, especially in high dimensional settings. Hence, in very small number of cases such as the ones within the data setting 18, GLM is close to the performance of RGMM while both outperform the traditional EM algorithm. However, in most of the cases, RGMM yields better results than both EM and GLM, especially when the components in the mixture are separated well. The stability of the algorithms varies from one data setting to another. In most of the cases, EM is the worst of the three while GLM is better and the RGMM comes out on top with the best results.

The impact of the increasing number of uninteresting set size is given in Figure 5.10. Again, we can interpret the results in two ways; the overall performance decrease and the stability of the algorithms with the increasing number of uninteresting points. We can clearly see the impact of the increasing number of uninteresting points in EM and GLM as their performance gradually decrease while RGMM is not affected by the presence of uninteresting points. Moreover, the stability of EM starts to degrade as the uninteresting data percentage reaches 0.15. Likewise, the stability of GLM gradually decreases as the uninteresting set size increases. On the other hand, RGMM shows high stability again as the results across varying number of uninteresting points are largely the same.

In the second part, we partition the training and test sets into three groups based on the dimension information, as previously mentioned. The performance results over log-likelihood differences and the stability of each algorithm can be seen in Figure 5.11. The results of EM and GLM are the same as the previous case, but now, we see better performance in our algorithm and its stability is also improved in some of the cases such as in data settings 11 and 16. Furthermore,

the impact of increasing uninteresting set size and the stability across varying number of uninteresting points are still largely the same as the first part and can be seen in Figure 5.12.

Finally, we computed the mean values of the log-likelihood differences of EM, GLM and RGMM over randomly initialized versions of each data setting. The performance results of each algorithm for each data setting with respect to the increasing number of uninteresting points are given in Figure 5.13. The γ parameter of RGMM is set to 0.30 for each set. The results are in line with our previous observations. EM is the worst of the three algorithms while RGMM outperforms GLM by a high margin in well separated data settings and by a small margin in high dimensional data settings where the c-separation is the lowest and the eccentricity is the highest. Additionally, the number of components plays an important role since small c-separation values with high eccentricity cause points in the components to merge into larger components where the points no longer preserve the Gaussian geometry. Thus, in such configurations, the interesting points collapse on each other which prevents a healthy RGMM estimation. On the other hand, the competing algorithms do not get affected much because of the fact that the true number of components are given as input prior to the estimation processes.

5.3 Experiments on Real-World Data Sets

We use the Iris, Wine and Glass data sets from the UCI Machine Learning Repository [53] and the Old Faithful Geyser [54] data set to compare our results with the competing algorithms, EM and GLM.

The old Faithful Geyser data set consists of points that are measured from the eruption duration and the waiting time between eruptions of the Old Faithful geyser in the Yellowstone National Park, Wyoming, USA. Hence, it is a data set used for clustering in its core. There are mainly two groups of data present in the data set that can be seen at first glance. However, some of the points around the

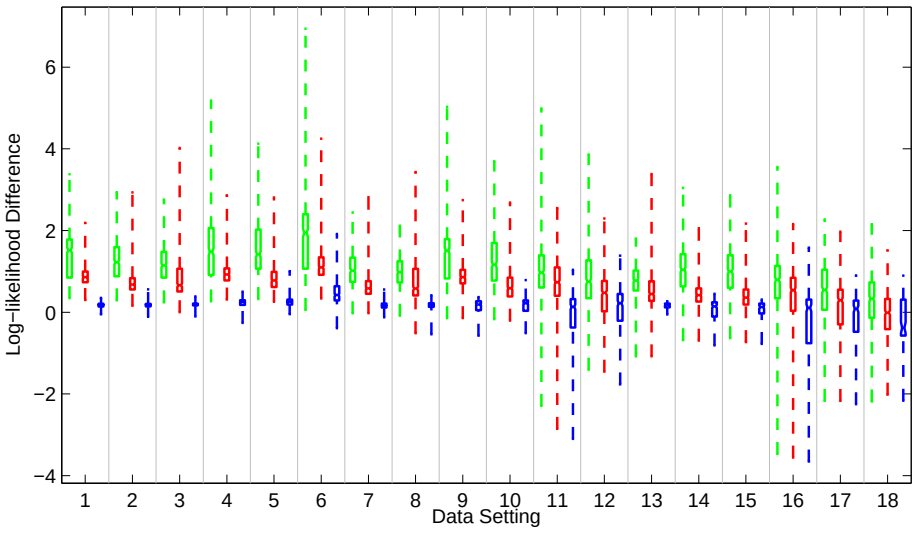


Figure 5.9: The difference in log-likelihood values for the synthetic data sets using the GMM parameters estimated via the EM (green), GLM (red) and RGMM (blue) with the γ parameter for RGMM set to 0.30 for each setting. The boxes show the lower quartile, median, and upper quartile of the log-likelihood differences. The whiskers drawn as dashed lines extend out to the extreme values.

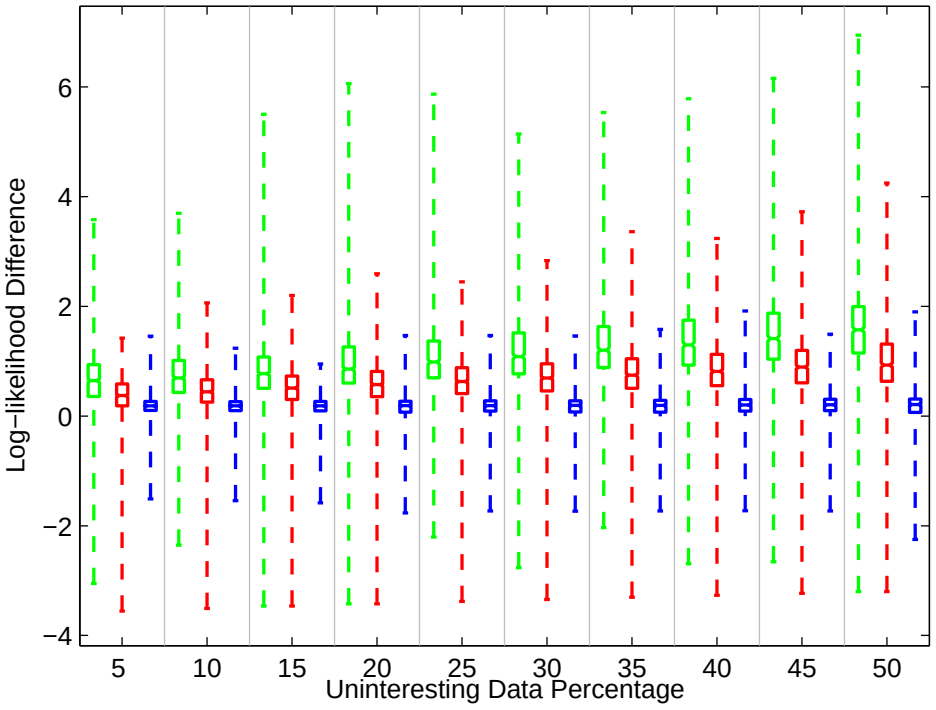


Figure 5.10: The difference in log-likelihood values in the presence of increasing percentage of uninteresting points using the GMM parameters estimated via the EM (green), GLM (red) and RGMM (blue) with the γ parameter for RGMM set to 0.30 for each setting. The boxes show the lower quartile, median, and upper quartile of the log-likelihood differences. The whiskers drawn as dashed lines extend out to the extreme values.

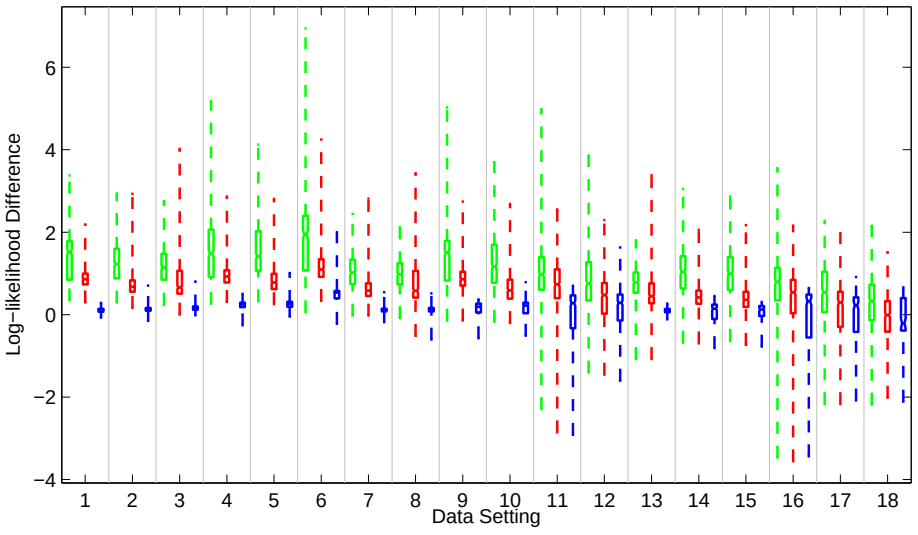


Figure 5.11: The difference in log-likelihood values for the synthetic data sets using the GMM parameters estimated via the EM (green), GLM (red) and RGMM (blue) with the γ parameter for RGMM set to 0.28, 0.30 and 0.32 for 2, 3 and 5-dimensional settings, respectively. The boxes show the lower quartile, median, and upper quartile of the log-likelihood differences. The whiskers drawn as dashed lines extend out to the extreme values.

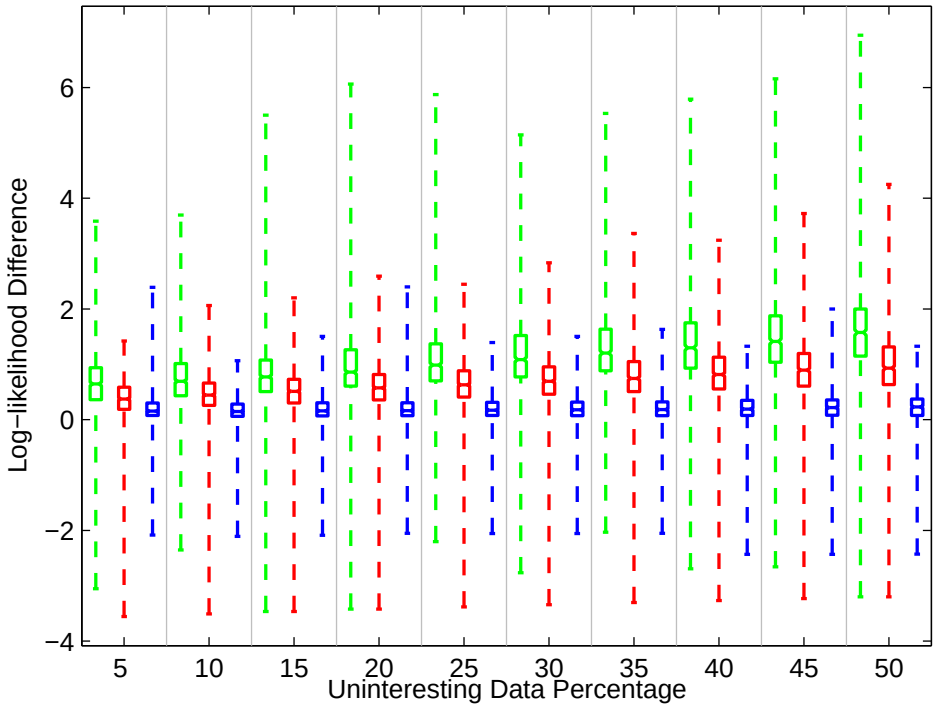
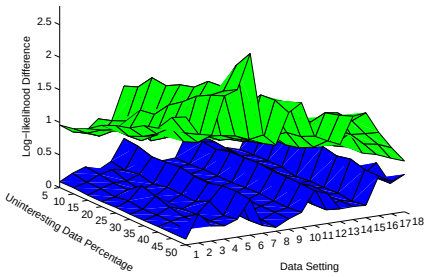
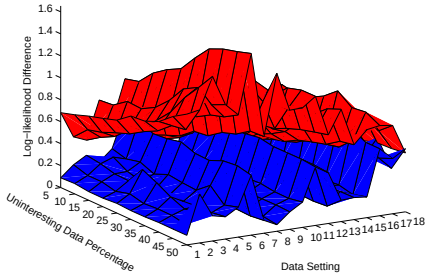


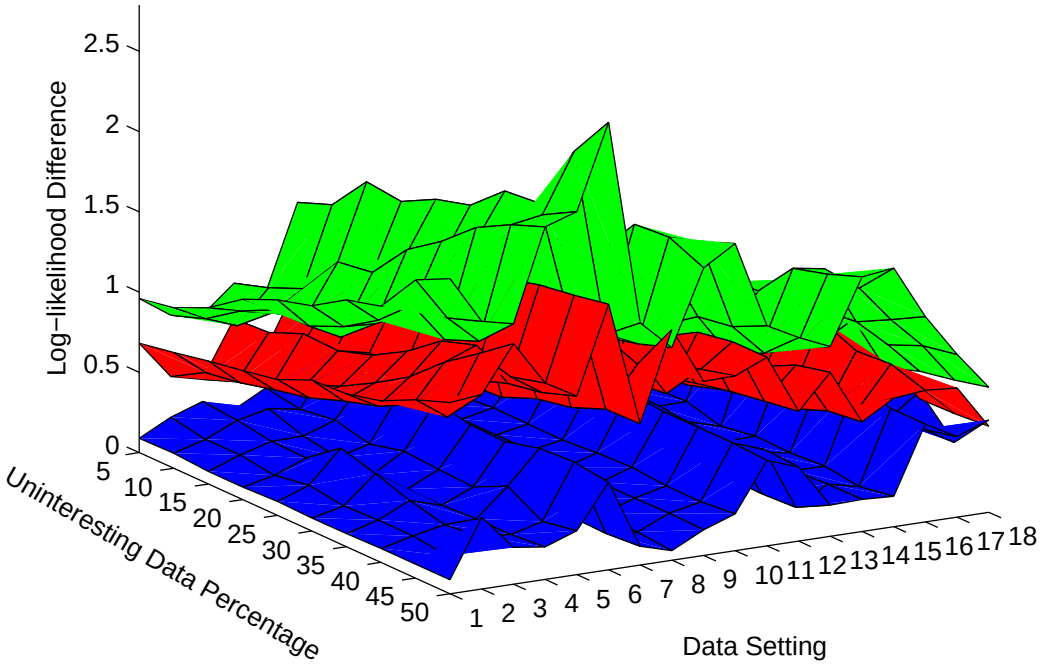
Figure 5.12: The difference in log-likelihood values in the presence of increasing percentage of uninteresting points using the GMM parameters estimated via the EM (green), GLM (red) and RGMM (blue) with the γ parameter for RGMM set to 0.28, 0.30 and 0.32 for 2, 3 and 5-dimensional settings, respectively. The boxes show the lower quartile, median, and upper quartile of the log-likelihood differences. The whiskers drawn as dashed lines extend out to the extreme values.



(a) EM (green) vs RGMM (blue)



(b) GLM (red) vs RGMM (blue)



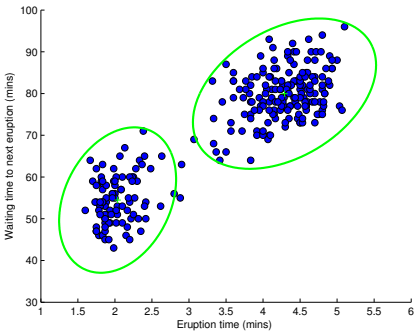
(c) EM (green) vs GLM (red) vs RGMM (blue)

Figure 5.13: Log-likelihood differences for each data setting with the increasing percentage of uninteresting points for EM (green), GLM (red) and RGMM (blue).

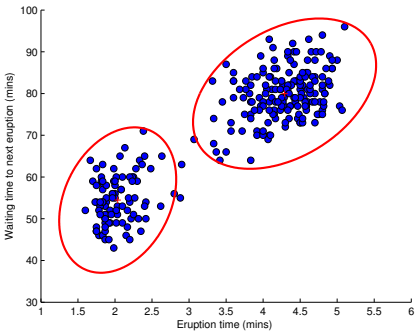
clusters are spread out. It is really difficult to assign those to one of the groups. Those points could correspond to the rare cases where the volcano's eruption is caused by unnatural events. As a result, those points can be regarded as outliers and in our terms, we can think of them as uninteresting points(or interesting points in another scenario). We run the algorithms on the data set and the performance of each method can be seen in Figure 5.14. Parameter initialization of the algorithms is done the same way as before. The γ parameter is set to 0.25 in RGMM algorithm as the groups in the data set are fairly dense.

Other than the Old Faithful Geyser data set, we synthetically add uninteresting points to each data set with the same way explained in the previous section. Hence, we now have 10 training sets for each of them. Also, we use the original data sets as the test sets in this case. Parameter initialization is also done the same way as before. As we do not know the true underlying distributions of the data sets, we estimate GMMs from the original data sets with the help of class information of their respective data points. We compute the average log-likelihood of this model. Then, we model each training set with EM, GLM and RGMM. After that, we compute the average log-likelihood values from each model on the corresponding test set. Finally, we compute the differences of the results of the algorithms from the estimated average log-likelihood value. The results on Iris, Glass and Wine data sets are given in Table 5.6, Table 5.7 and Table 5.8, respectively. The results for the Iris data set suggest that the EM algorithm is not suitable for density estimation in the presence of even small amounts of uninteresting points. GLM shows the best performance for small amounts of uninteresting points whereas RGMM shows comparatively good results. However, once the uninteresting data size is around 30% of the interesting set size, RGMM starts to outperform GLM. Glass data set results imply that the GLM algorithm cannot be used for estimation due to its very poor performance in all cases. EM shows the best performance for small amounts of uninteresting points but once the uninteresting set size goes above 15% of the interesting set size, RGMM gains the upper hand and beats its competitors by a high margin. Finally, the results of the Wine data set state that the best performing algorithm among the competing algorithms is RGMM in each level of uninteresting set size. Additionally,

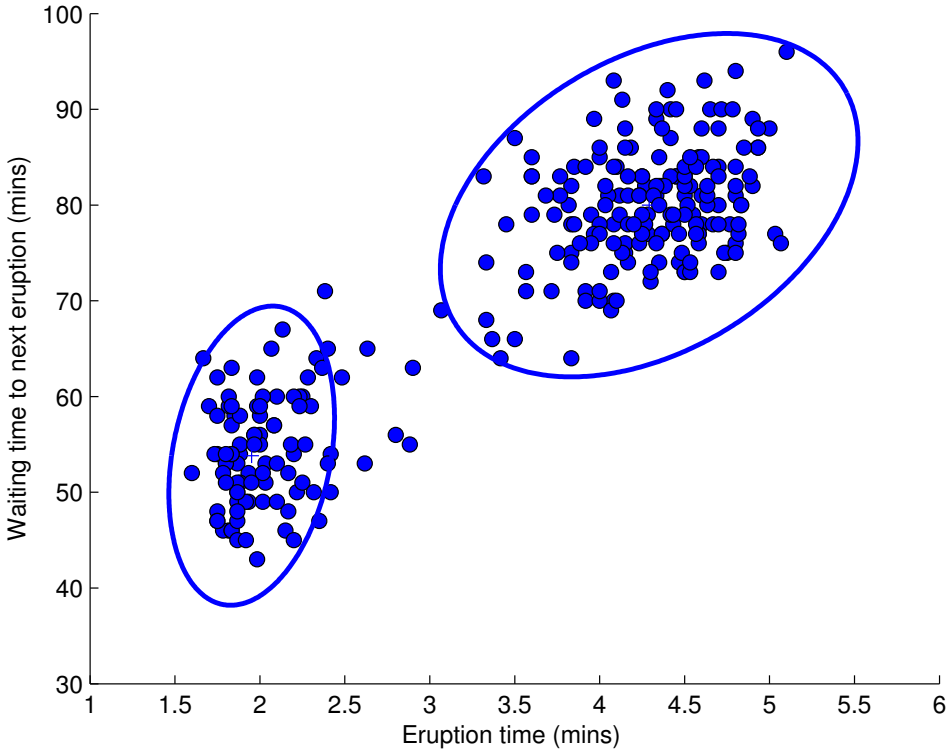
in all of the results of each three data sets, the performance of EM and GLM is severely affected by the increasing number of uninteresting points while the performance of RGMM is either decreased subtly or not affected by the increasing uninteresting set size at all.



(a) GMM estimated via EM



(b) GMM estimated with GLM



(c) Estimated RGMM(Our Algorithm)

Figure 5.14: Estimated GMMs with two-components on the Old Faithful Geyser data set with EM in (a), GLM in (b) and RGMM in (c).

Table 5.6: Log-likelihood differences of EM, GLM and RGMM with $\gamma = 0.26$ on Iris data set

	Uninteresting Data Percentage									
	5	10	15	20	25	30	35	40	45	50
RGMM	0.307	0.561	0.507	0.540	0.493	0.464	0.551	0.584	0.548	0.647
GLM	0.297	0.305	0.368	0.392	0.430	0.472	0.507	0.578	0.611	0.733
EM	1.548	1.486	1.030	1.288	1.600	1.516	1.686	1.642	1.779	1.711

Table 5.7: Log-likelihood differences of EM, GLM and RGMM with $\gamma = 0.31$ on Glass data set

	Uninteresting Data Percentage									
	5	10	15	20	25	30	35	40	45	50
RGMM	2.328	2.218	2.214	2.322	1.963	1.986	1.988	1.994	2.068	1.800
GLM	7.537	9.116	10.190	10.860	11.497	12.039	12.383	12.772	13.013	13.337
EM	0.876	1.363	2.003	3.534	4.670	5.186	5.889	6.560	6.997	7.140

Table 5.8: Log-likelihood differences of EM, GLM and RGMM with $\gamma = 0.36$ on Wine data set

	Uninteresting Data Percentage									
	5	10	15	20	25	30	35	40	45	50
RGMM	2.282	2.291	2.056	2.483	2.750	1.976	2.561	2.109	2.082	2.335
GLM	4.405	5.469	6.197	6.837	7.330	7.674	8.011	8.289	8.521	8.751
EM	3.167	4.281	5.034	5.675	6.170	6.554	6.845	7.182	7.435	7.739

Chapter 6

Conclusions and Future Work

In this thesis, we presented an iterative approach for learning Gaussian mixtures from heterogeneous data sets. The method can model only the interesting points while being robust to the presence of uninteresting ones. Contrary to the traditional approaches, it did not need the number of components as input for a good estimation. Moreover, it actually found the true number of components without additional information as well.

First, we presented the mathematical analysis of estimating robust Gaussians for a fixed number of interesting points using the properties of Kullback-Leibler divergence. Then, we introduced a procedure which we call multi-resolution search in order to estimate the actual number of interesting points. As the last step of robust Gaussian model estimation, we explored the points that lied in the tails of the estimated Gaussian and included those points as part of the set that belonged to the Gaussian.

The second part of the thesis consisted of the process of estimation of a mixture of Gaussians from the previously estimated robust Gaussian models and their respective data points. In order to estimate multiple Gaussian components from the data, we removed the set of points that belonged to a particular robust Gaussian after each iteration of the algorithm. Once the stopping criterion is met, the algorithm stopped searching for new components. Thus, it automatically

identified the true number of components as the number of robust Gaussians estimated up to that iteration. The last step included the computation of mixing coefficients in the mixture model which concluded the estimation process of robust Gaussian mixture.

Finally, we demonstrated the results of our method in identifying the true number of components and estimating a robust mixture model in the presence of uninteresting points once the regularization parameter was set properly. We compared our results with other methods and showed the superiority of our algorithm. Among the competing methods, our algorithm produced the smallest log-likelihood difference from the generating model which meant that it was the closest estimate to the generating model that the interesting points in heterogeneous data were sampled from.

The direction for future work could include improving the time complexity of the algorithm, extending its framework to new applications, and improving the capabilities of the proposed model such as making it applicable for other purposes than density estimation.

Bibliography

[1] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*, vol. 26. CRC press, 1986.

[2] G. J. McLachlan and D. Peel, "Robust cluster analysis via mixtures of multivariate t-distributions," in *Advances in Pattern Recognition*, pp. 658–666, Springer, 1998.

[3] D. Peel and G. J. McLachlan, "Robust mixture modelling using the t-distribution," *Statistics and Computing*, vol. 10, no. 4, pp. 339–348, 2000.

[4] G. McLachlan and D. Peel, *Finite Mixture Models*. John Wiley & Sons, 2004.

[5] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the em algorithm," *SIAM Review*, vol. 26, no. 2, pp. pp. 195–239, 1984.

[6] D. M. Titterington, A. F. M. Smith, and U. E. Makov, *Statistical Analysis of Finite Fixture Distributions*. Chichester: John Wiley & Sons, 1985.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[8] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley-Interscience, 2000.

[9] N. Friedman and S. Russell, "Image segmentation in video sequences: A probabilistic approach," in *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pp. 175–181, Morgan Kaufmann Publishers Inc., 1997.

[10] T. N. Pappas, "An adaptive clustering algorithm for image segmentation," *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 901–914, 1992.

[11] X. Yang and S. M. Krishnan, "Image segmentation using finite mixtures and spatial information," *Image and Vision Computing*, vol. 22, no. 9, pp. 735–745, 2004.

[12] M. Harville, "A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models," in *Computer Vision-ECCV*, pp. 543–560, Springer, 2002.

[13] J. Cheng, J. Yang, Y. Zhou, and Y. Cui, "Flexible background mixture models for foreground segmentation," *Image and Vision Computing*, vol. 24, no. 5, pp. 473–482, 2006.

[14] S. Sanjay-Gopal and T. J. Hebert, "Bayesian pixel classification using spatially variant finite mixtures and the generalized em algorithm," *IEEE Transactions on Image Processing*, vol. 7, no. 7, pp. 1014–1028, 1998.

[15] C. Nikou, N. P. Galatsanos, and C. Likas, "A class-adaptive spatially variant mixture model for image segmentation," *IEEE Transactions on Image Processing*, vol. 16, no. 4, pp. 1121–1130, 2007.

[16] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 696–710, 1997.

[17] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002.

[18] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, IEEE, 1999.

[19] D.-S. Lee, "Effective gaussian mixture learning for video background subtraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 827–832, 2005.

[20] J. D. Banfield and A. E. Raftery, "Model-based Gaussian and non-Gaussian clustering," *Biometrics*, vol. 49, pp. 803–821, 1993.

[21] C. Biernacki, G. Celeux, and G. Govaert, "Assessing a mixture model for clustering with the integrated completed likelihood," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 7, pp. 719–725, 2000.

[22] G. Celeux and G. Govaert, "Gaussian parsimonious clustering models," *Pattern Recognition*, vol. 28, no. 5, pp. 781–793, 1995.

[23] J.-P. Baudry, A. E. Raftery, G. Celeux, K. Lo, and R. Gottardo, "Combining mixture components for clustering," *Journal of Computational and Graphical Statistics*, vol. 19, no. 2, pp. 332–353, 2010.

[24] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, p. 14, California, USA, 1967.

[25] A. P. Dempster, N. M. Laird, D. B. Rubin, *et al.*, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[26] R. M. Neal and G. E. Hinton, "A view of the em algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models*, pp. 355–368, Springer, 1998.

[27] G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, vol. 382. John Wiley & Sons, 2007.

[28] C. Fraley and A. E. Raftery, "How many clusters? Which clustering method? Answers via model-based cluster analysis," *The Computer Journal*, vol. 41, no. 8, pp. 578–588, 1998.

[29] G. J. McLachlan, S.-K. Ng, and R. Bean, "Robust cluster analysis via mixture models," *Austrian Journal of Statistics*, vol. 35, no. 2, pp. 157–174, 2006.

[30] N. Neykov, P. Filzmoser, R. Dimova, and P. Neytchev, "Robust fitting of mixtures using the trimmed likelihood estimator," *Computational Statistics & Data Analysis*, vol. 52, pp. 299–308, September 2007.

[31] A. S. Hadi and A. Luceño, "Maximum trimmed likelihood estimators: a unified approach, examples, and algorithms," *Computational Statistics & Data Analysis*, vol. 25, pp. 251–272, Aug. 1997.

[32] M. C. Neykov N.M., "Breakdown point and computation of trimmed likelihood estimators in generalized linear models," *Developments in Robust Statistics*, pp. 277–286, 2003.

[33] C. Biernacki, G. Celeux, and G. Govaert, "Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models," *Computational Statistics & Data Analysis*, vol. 41, pp. 561–575, jan 2003.

[34] C. K. Reddy, H.-D. Chiang, and B. Rajaratnam, "Trust-tech-based expectation-maximization for learning finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp. 1146–1157, jul 2008.

[35] M. Figueiredo and A. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 381 –396, march 2002.

[36] S. Roberts, D. Husmeier, I. Rezek, and W. Penny, "Bayesian approaches to gaussian mixture modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 1133 –1142, nov 1998.

[37] J. J. Verbeek, N. Vlassis, and B. Krse, "Efficient greedy learning of gaussian mixture models," *Neural Computation*, vol. 15, pp. 469–485, 2003.

[38] M.-S. Yang, C.-Y. Lai, and C.-Y. Lin, "A robust em clustering algorithm for gaussian mixture models," *Pattern Recognition*, vol. 45, pp. 3950–3961, nov 2012.

[39] T. S. Ferguson, "A bayesian analysis of some nonparametric problems," *The Annals of Statistics*, pp. 209–230, 1973.

[40] J. Sethuraman, "A constructive definition of dirichlet priors," tech. rep., DTIC Document, 1991.

[41] C. E. Antoniak *et al.*, "Mixtures of dirichlet processes with applications to bayesian nonparametric problems," *The Annals of Statistics*, vol. 2, no. 6, pp. 1152–1174, 1974.

[42] M. D. Escobar and M. West, "Bayesian density estimation and inference using mixtures," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 577–588, 1995.

[43] R. M. Neal, "Markov chain sampling methods for dirichlet process mixture models," *Journal of Computational and Graphical Statistics*, vol. 9, no. 2, pp. 249–265, 2000.

[44] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, "Sharing clusters among related groups: hierarchical dirichlet processes," in *NIPS*, 2004.

[45] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[46] B. Iglewicz and D. Hoaglin, *How to Detect and Handle Outliers*. ASQC basic references in quality control, ASQC Quality Press, 1993.

[47] S. J. Roberts, "Novelty detection using extreme value statistics," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 146, no. 3, pp. 124–129, 1999.

[48] C. J. Mecklin and D. J. Mundfrom, "A monte carlo comparison of the type i and type ii error rates of tests of multivariate normality," *Journal of Statistical Computation and Simulation*, vol. 75, no. 2, pp. 93–107, 2005.

[49] J. P. Royston, "Some techniques for assessing multivarate normality based on the shapiro-wilk w," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 32, no. 2, pp. pp. 121–133, 1983.

[50] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for Normality (complete samples)," *Biometrika*, vol. 52, pp. 591–611, Dec. 1965.

[51] N. M. Razali and Y. B. Wah, "Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests," *Journal of Statistical Modeling and Analytics*, vol. 2, no. 1, pp. 21–33, 2011.

[52] S. Dasgupta, "Learning mixtures of gaussians," in *40th Annual Symposium on Foundations of Computer Science*, pp. 634–644, IEEE, 1999.

[53] K. Bache and M. Lichman, "UCI machine learning repository," 2013.

[54] A. Azzalini and A. Bowman, "A look at some data on the old faithful geyser," *Applied Statistics*, pp. 357–365, 1990.