MODEL-BASED CLUSTERING ALGORITHMS, PERFORMANCE AND APPLICATION

By

JUN LIU

M.Eng., Shanghai Jiao Tong University

B.Sc., Huazhong University of Science & Technology

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Ph. D.

McMaster University

MODEL-BASED CLUSTERING
ALGORITHMS, PERFORMANCE AND APPLICATION

PH. D. (2000) MCMASTER UNIVERSITY

(Electrical and Computer Engineering) Hamilton, Ontario

TITLE: **Model-Based Clustering Algorithms, Performance And Application**

AUTHOR: Jun Liu

M.Eng., Shanghai Jiao Tong University

B.Sc., Huazhong University of Science & Technology

SUPERVISOR(S): Dr. K.M. Wong and Dr. Z.-Q. Luo

Professors,

Department of Electrical and Computer Engineering

NUMBER OF PAGES: xv, 120

ABSTRACT

The main contributions of this thesis are the development of new clustering algorithms (with cluster validation) both off-line and on-line, the performance analysis of the new algorithms and their applications to intrapulse analysis.

Bayesian inference and minimum encoding inference including Wallace's minimum message length (MML) and Rissanen's minimum description length (MDL), are reviewed for model selection. It is found that the MML coding length is more accurate than the other two in the view of quantization. By introducing a penalty weight, all criteria considered here are cast into the framework of a penalized likelihood method.

Based on minimum encoding inference, an appropriate measure of coding length is proposed for cluster validation, and the coding lengths under four different Gaussian mixture models are fully derived. This provides us with a criterion for the development of a new clustering algorithm. Judging from the performance comparison with other algorithms, the new clustering algorithm is more suitable to process high dimensional data with satisfactory performance on small and medium samples. This clustering algorithm is off-line because it requires all the data available at the same time.

The theoretical error performance of our clustering algorithm is evaluated under reasonable assumptions. It is shown here how the dimension of data space, the sample size, the mixing portion and the inter-cluster distance affect the performance of our clustering algorithm to detect the true number of clusters. Furthermore, we examine the impact of the penalty weight under the framework of the penalized likelihood method. It is found that there is a range of the penalty weight within which the best performance of our clustering

algorithm can be achieved. Therefore, with some supervision we could adjust the penalty weight to further improve the performance of our clustering algorithm.

The application of our clustering algorithm to intrapulse analysis is investigated in detail. We first develop the pre-processing techniques including data compression for received pulses and formulated the problem of emitter number detection and pulse-emitter association into a multivariate clustering problem. After applying the above (off-line) clustering algorithm here, we further develop two on-line clustering algorithms, one is based on some known thresholds while the other is based on a model-based detection scheme. Performance on intrapulse data by using our pre-processing techniques and clustering algorithms is reported, and the results demonstrate that our new clustering algorithms are very effective for intrapulse analysis, especially the model-based on-line algorithm.

Finally, the DSP implementation for intrapulse analysis is considered. Some relevant physical parameters are estimated such as the likely maximal incoming pulse rate. Then a suitable system diagram is proposed and its system requirements are investigated. Our on-line clustering algorithm is implemented as a core classification module on a TMS320C44 DSP board.

Acknowledgement

I have received both constant encouragement and expert supervision from my supervisors Dr. K.M. Wong and Dr. Z.Q. Luo. From both I have learned a great deal, of which only a part is in this thesis.

I wish to express sincere gratitude to supervisory committee members Dr. J.P. Reilly, Dr. S. Qiao and Dr. T. Todd for their encouragement and supervision. I am especially pleased to thank my external advisor Dr. J.P.Y. Lee, from the Defense Research Establishment Ottawa, for his encouragement and expert guidance.

I was fortunate to have been part of the Advanced Signal Processing for Communications (ASPC) group led by Dr. K.M. Wong and Dr. Z.Q. Luo. I would like to thank fellow ASPCers: Dr. T.N. Davidson, Dr. S.Q. Wu, Dr. S.W. Gao, Dr. J. Wu, Mr. L. Li, Mr. J. Zhang for their valuable help. I would also like to acknowledge the financial support provided by the Department of Electrical and Computer Engineering and the Defense Research Establishment Ottawa.

Finally, I am indebted to my parents Qiji Liu and Ande Chen, especially my wife Dieqian Han, for their understanding and great support.

Acronyms

BIC	Bayesian Inference Criterion
MML	Minimum Message Length
MDL	Minimum Description Length
LRT	Likelihood Ratio Test
p.d.f.	probability density function
ML	Maximum Likelihood
LPF	Low Pass Filter
HPF	High Pass Filter
PRI	Pulse Repetition Interval
PRF	Pulse Repetition Frequency
rpm	rotation per minute
flop	floating point operation
SRAM	Static Random Access Memory
EDF	Empirical Distribution Function

Notations

log	Natural logarithm
$\sum_n$	Summation over n
$\prod_n$	Product over n
$\boldsymbol{W}^T$	Transpose of $\boldsymbol{W}$
$\boldsymbol{W}^{-1}$	Inverse of $\boldsymbol{W}$
$tr(\boldsymbol{W})$	Trace of $\boldsymbol{W}$
$\|\boldsymbol{W}\|$	Determinant of $\boldsymbol{W}$
$diag(\boldsymbol{W})$	Diagonal matrix of $\boldsymbol{W}$
$\|\mu\|$	$\mathcal{L}_2$ norm of μ
$E[R]$	Expectation value of R
$Var[R]$	Variance of R
$N(\mu, \Sigma)$	Multivariate normal distribution
	with mean vector μ and covariance matrix Σ
$F_{v1,v2}(\delta)$	Noncentral F distribution
	with $v1$, $v2$ degrees of freedom and noncentral parameter δ
$W_M(n, \Sigma, \Omega)$	Noncentral Wishart distribution
	with the number of variates M, freedom degree n,
	covariance matrix Σ and noncentral matrix Ω

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Intrapulse Analysis

Radar emitter classification based on a collection of received radar signals is a subject of wide interest in both civil and military applications. The signals received usually consist of sequences of pulses emitted from multiple radar transmitters. If different radars transmit pulses with different carrier frequencies or pulse repetition intervals (PRIs), then it is not difficult to distinguish them from one another. However, in modern radar systems, more sophisticated signal waveforms have been used and inter-pulse information alone may not be enough to separate those received pulses according to their originations. To classify radar emitters in such an environment, we need to explore the detailed structure inside each pulse, i.e. the so called *intrapulse information*. This is because each emitter has its own electrical signal structure inside each of its transmitted pulses due to both intentional and unintentional modulations. This structure motivates us to explore the possibility of using intrapulse information of a collection of pulses to determine the number of emitters present and to classify those pulses according to their originations. In other words, the objectives of the research are to: (1) determine the number of emitters present; (2) classify the incoming pulses according to the emitters. The physical scenario in detail is illustrated in Section 5.2.

There are three important issues in the design of a processing algorithm for intrapulse analysis:

- The algorithm is suitable to process high dimensional data because in most cases more than 40 data points are required to describe a pulse.
- The performance of the algorithm is satisfactory for small or medium sample cases since it is desirable to identify the emitters present by using a few received pulses.
- The algorithm is computationally effective and on-line clustering is required for near real-time implementation.

In practice, a radar pulse intercepted by a passive receiver may be contaminated by an absolute amplitude and phase, time delay and residual carrier frequency. We first develop pre-processing techniques including data compression for received pulses and then formulate our objectives into a multivariate clustering problem. In the cluster analysis literature [1, 23, 31, 39, 49, 69], the first objective is known as *cluster validation* while the second is called *clustering*. Generally speaking, the current clustering methods range from those that are largely heuristic to more formal procedures based on statistical models. One major advantage of model-based methods is that they provide a precise theoretical framework for assessing the clustering structure of a given data set, especially for determining a relevant number of clusters. In next section, model-based cluster analysis is discussed in detail.

1.2 Model-Based Cluster Analysis

In model-based cluster analysis [28,39], it is assumed that the data under consideration are generated from a finite mixture of probability distributions (e.g. normal distributions) and each component of the mixture represents a different cluster. Given N observations from K clusters, there are two ways to formulate the mixture model: one is the so-called classification approach which assigns an observation to one of the K clusters deterministically, and the other is the so-called mixture approach which assigns an observation to the K clusters

probabilistically. An empirical comparison [15] in a finite sample setting between these two approaches suggested that the classification approach is preferred for small sample cases, although from the studies in [13, 14, 30], asymptotically, the mixture approach tends to perform better than the classification approach when classifying ill-separated components, with a sufficiently large sample.

Given a data set and underlying models, the first question to be considered is how to select a model which best fits the data? This is a critical question common to the fields of Statistics, Machine Learning and Artificial Intelligence. Two principles can be applied to search for the answer, one is Bayesian Inference [17,28,35,48,54] and the other is Minimum Encoding Inference [50, 52, 62, 64]. In the Bayesian framework, a model is chosen as the best if it has highest posterior probability. In the minimum encoding inference framework, the best model is the one that yields the minimal coding length of the data. For the latter principle to interpret the encoding process, there are two different approaches, one by Wallace [64], termed the Minimum Message Length (MML) criterion; and the other by Rissanen [52], termed the Minimum Description Length (MDL) criterion. A comparative study of Bayesian inference, MML and MDL was reported in [8, 44, 45]. There will be a further discussion of this important issue in the next chapter.

After establishing a criterion, either an ad hoc or a model-based, is chosen for cluster validation. There is the question of how to classify the observations under an assumed known number of clusters. A simple and common method is k-means [33] which minimizes the within-group sums of squares. It starts with an initial estimate, and then regroup the given data set a few times until the cluster centers are convergent. The k-means method can be used for the classification approach. Another common method is the EM algorithm [66] which maximizes the underlying likelihood. Starting with an initial estimate, the EM algorithm consists of two steps: the expectation step which estimates the model parameters including the probabilities of an observation belonging to each cluster, and then the maximization step which evaluates the resulting likelihood. The EM algorithm is suitable for both the classification approach and the mixture approach. However, the EM algorithm is

more computationally intensive than the k-means algorithm.

In model-based cluster analysis, cluster validation and clustering are combined by first formulating a statistical model for the problem which is parameterized by k, the number of clusters, then selecting the hypothesis that best fits the data. Among statistical models for cluster analysis, Gaussian mixtures are widely used. Three Gaussian clustering algorithms are listed below:

1. MCLUST: Developed by Fraley and Raftery [6,27–29]. MCLUST incorporates eight different Gaussian mixture models in terms of the covariance matrix Σ, allows a choice of either the classification approach or the mixture approach, and applies an asymptotic criterion of Bayesian inference for assessing the number of clusters. This algorithm works well for medium and/or large sample cases but might not provide satisfactory results for small sample cases.

2. Autoclass: Developed by Cheeseman, Self and Kelly [16,17]. Autoclass only assumes the general covariance matrix structure, follows the mixture approach, and applies Bayesian inference for cluster validation. Since only the general covariance matrix structure is assumed, it weights the model complexity too much for high dimensional cases so that it is not suitable to process high dimensional data.

3. SNOB: Developed by Wallace and its co-workers [10,11,61,63]. SNOB assumes the covariance matrix is diagonal, follows the mixture approach, and applies MML inference for cluster validation. It works for both high dimensional cases and small sample cases.

Another important issue is the error performance analysis. Following the terminology in statistical signal processing [58], if one cluster is present for a given data set but a clustering algorithm detects two clusters, then a *false alarm* occurs; On the other hand, if two clusters are present but the clustering algorithm only says one, then a *miss* occurs. The error analysis of model-based clustering is related to the question of the number of components in a mixture, which is a problem that has not been completely solved in statistics. A general

method to this question is the Likelihood Ratio Test (LRT). Suppose that a random sample $\boldsymbol{Y}$ is available and we wish to test the following two hypotheses:

$$H_0: \quad \boldsymbol{Y} \text{ is generated from a mixture of } K \text{ normals}$$

$$H_1: \quad \boldsymbol{Y} \text{ is generated from a mixture of } K^* \text{ normals } (K^* > K)$$

However, the regularity condition for the usual asymptotic theory fail when the null hypothesis H_0 is true, see details in [39, Page 21] and [56]. It is indeed a difficult problem, even for detecting a univariate normal mixture with two components. For the aforementioned "simple" case, empirical tabulation and an asymptotic analysis were presented in [24, 32] by following the classification approach; Empirical tabulation was presented in [41, 42] and an asymptotic analysis was derived in [9] by following the mixture approach.

1.3 Major Contributions of The Thesis

Motivated by exploring the possibility of using intrapulse information of a collection of pulses to identify the emitters present, extensive research on model-based clustering have been conducted. The major contributions are twofold: model-based cluster analysis and intrapluse analysis.

Model-Based Cluster Analysis

Bayesian inference and minimum encoding inference including Wallace's minimum message length (MML) and Rissanen's minimum description length (MDL), are reviewed and compared for model selection. It is found that the MML coding length is more accurate than the other two in the view of quantization. All model selection criteria considered here consist of two parts, one is the log-likelihood function which measures the goodness of fit between the data and the model, and another is a penalty function which measures the complexity of the model. An inference aims to balance the trade-off between goodness of fit and model complexity. Hence in practice, we can introduce a penalty weight for the penalty function to control the trade-off. We call this approach the penalized likelihood method.

Applying minimum encoding inference to the classification approach of model-based clustering, we propose an appropriate measure of coding length for cluster validation. The coding lengths under four different Gaussian mixture models in terms of the covariance matrix Σ are derived. The first covariance structure is the simplest and mainly used for the purpose of a theoretical error performance analysis. The second and fourth are successfully applied to intrapulse analysis. The third one might be useful for other applications so we include it for completeness. Correspondingly, we develop an effective clustering algorithm which starts with viewing a given data set as a cluster and then repartitions and regroups the data to get a new cluster in each step. The new algorithm is off-line since it requires all the data available at the same time. Extensive empirical results show that the new clustering algorithm (with cluster validation) is more suitable than SNOB to process high dimensional data with better performance on small sample cases. In fact, our algorithm is well designed for the clustering problem in intrapulse analysis, in terms of the first two issues pointed out in Section 1.1.

The theoretical error performance of our clustering algorithm is evaluated under reasonable assumptions. It is shown in this thesis how the dimension of the data space, the sample size, the mixing portion and the inter-cluster distance affect the performance of the clustering algorithm to detect the true number of clusters. Furthermore, by introducing a penalty weight, we investigate our clustering algorithm as a penalized likelihood method. The impact of the penalty weight is investigated. With some supervision, we could adjust the penalty weight to further improve the performance of our algorithm. Testing our clustering algorithm on intrapulse data, we have found that the best performance is usually achieved by using the fourth covariance structure when no supervision is available (i.e., the penalty weight is 1, the default value), and that the best performance is usually achieved by using the second covariance structure when supervision is available (i.e., the penalty weight can be adapted).

Intrapulse Analysis

First, we develop pre-processing techniques to remove nuisance parameters from received

pulses such as an absolute amplitude and phase, time delay and residual carrier frequency. As a result, we formulate the problem of emitter number detection and pulse-emitter association into a multivariate clustering problem. In order to reduce the computational cost for clustering, a suitable data compression method based on a wavelet decomposition is also included in pre-processing. The pre-processing techniques are intuitive in nature and are carried out so that after pre-processing, the pulses received from the same emitter maintain the resemblance to each other, while those from different emitters maintain their distinctive features.

Second, after applying the above new clustering algorithm to the clustering problem, we investigate how to achieve on-line clustering, that is, to perform classification dynamically as pulses arrive. To solve this problem, we propose to set up some thresholds and distance measures which can be used to indicate to which existing cluster an incoming pulse should be assigned, or whether it should form a new cluster. To achieve an accurate classification result, we have to adapt the thresholds as the statistics of the received pulses changes in time. Unfortunately, it is usually difficult to modify the thresholds appropriately when *a priori* knowledge of the incoming pulses is not available. To overcome this drawback, a novel on-line algorithm based on a model-based detection scheme is developed in which no explicit thresholds are required. This new on-line algorithm dynamically incorporates cluster splitting, merging and regrouping operations by using the model-based detection. The performance of this on-line model-based clustering algorithm is almost the same as that of the off-line model-based algorithm but is much faster.

Third, to effectively implement our pre-processing techniques and clustering algorithms for the emitter number detection and the pulse classification in near real-time, we estimate the relevant physical parameters such as the likely maximal incoming pulse rates. Based on these estimates, we then propose a suitable system diagram and investigate the system requirements. Finally, we implement our on-line clustering algorithm as a core classification module on a TMS320C44 DSP board.

1.4 Outline of The Thesis

This introduction chapter has been mainly concerned with placing this thesis in context. We have reviewed the problems in intrapulse analysis and model-based cluster analysis, and outlined our major contributions to these two fields.

In next chapter, we review some criteria for model selection, compare Bayesian inference and minimum encoding inference (including MDL and MML), and cast them into the framework of a penalized likelihood method.

In Chapter 3, applying minimum encoding inference to the classification approach to model-based clustering, we propose an appropriate measure of coding length for cluster validation, and derive the coding lengths under four different Gaussian mixture models. Then we describe a new clustering algorithm, compare it with SNOB, and demonstrate by extensive simulations that our algorithm is more suitable than SNOB to process high dimensional data with better performance on small sample cases. In addition, we also examine the performance of the coding length measure based on an asymptotic method for Bayesian Inference.

In Chapter 4, we conduct the theoretical performance analysis of our clustering algorithm, in terms of two types of errors: miss and false alarm. We also study the impact of the penalty weight under the framework of the penalized likelihood method. The conclusion is that there is a range of the penalty weight within which the best performance of our clustering algorithm can be achieved.

Intrapulse analysis is carried out in Chapter 5. We first describe the pre-processing techniques including data compression for received pulses and formulate our objectives into a multivariate clustering setting. After applying the model-based clustering algorithm developed in Chapter 3 to the clustering problem, we further develop two on-line clustering algorithms, one is based on known thresholds while the other is based on a model-based detection. Performance on intrapulse data by using our clustering algorithms and SNOB are reported, and the results demonstrate that our new clustering algorithms are very effective for intrapulse analysis, especially the on-line model-based algorithm.

In Chapter 6, the DSP implementation for intrapulse analysis is considered. we estimate the relevant physical parameters such as the likely maximal incoming pulse rate, then propose a suitable system diagram and investigate the system requirements. The benchmark of DSP coding of our on-line clustering algorithm is reported.

Finally, the last chapter concludes the thesis with a summary of what has been achieved, and outlines areas of future research.

Chapter 2

Model Selection Criteria

2.1 Introduction

In this chapter, we review Bayesian Inference [17,35,48,54] and Minimum Encoding Inference [50,52,62,64]. For Bayesian Inference, two inference techniques are introduced: one is using Laplace's method [17,36] and the other is an asymptotic method [54]. For Minimum Encoding Inference, there are two approaches: one is called the Minimum Description Length (MDL) criterion [52] and the other is called the Minimum Message Length (MML) criterion [64]. For MDL, its coding steps are briefly described and the idea of a universal prior is introduced. For MML, its coding steps are briefly described and a sensible prior is required. In the end of this chapter, Bayesian Inference, MDL and MML are cast into the framework of a penalized likelihood method.

2.2 Bayesian Inference

Given a data set $\boldsymbol{Y} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N\}$ and a set of model classes [1] parameterized by K ($K = 1, \cdots, K_{\max}$), let $\boldsymbol{\theta}$ denote the model parameter vector under a model class, $f(\boldsymbol{Y}|\boldsymbol{\theta}, K)$ denote the conditional probability density function (p.d.f.) of the data given $\boldsymbol{\theta}$ and K, and

[1] For instance, in model-based cluster analysis, K is the number of clusters, and different partitions which form K clusters belong to the same model class.

$h(\boldsymbol{\theta}|K)$ denote the prior p.d.f. of $\boldsymbol{\theta}$ given K. Then the conditional p.d.f. of $\boldsymbol{Y}$ given K is

$$f(\boldsymbol{Y}|K) = \int f(\boldsymbol{Y}|\boldsymbol{\theta}, K)h(\boldsymbol{\theta}|K)d\boldsymbol{\theta}. \tag{2.1}$$

Further let $f(\boldsymbol{Y})$ denote the p.d.f. of $\boldsymbol{Y}$ occurring and let $P(K)$ be the prior probability of the model class K. Then Bayes' theorem tells us $P(K|\boldsymbol{Y})$, the posterior probability of K given $\boldsymbol{Y}$, is given by

$$P(K|\boldsymbol{Y}) = \frac{f(\boldsymbol{Y}|K)P(K)}{f(\boldsymbol{Y})}. \tag{2.2}$$

If we take a uniform prior for K, then $P(K|\boldsymbol{Y})$ is proportional to $f(\boldsymbol{Y}|K)$, i.e.,

$$P(K|\boldsymbol{Y}) \propto f(\boldsymbol{Y}|K). \tag{2.3}$$

By using Laplace's method [17,36] to approximate the integral in Eq. (2.1), we have

$$f(\boldsymbol{Y}|K) \simeq \frac{(2\pi)^{\ell/2}}{\sqrt{|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}|}} f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, K)h(\hat{\boldsymbol{\theta}}|K) \tag{2.4}$$

where $\hat{\boldsymbol{\theta}}$ is the maximum likelihood (ML) estimate of $\boldsymbol{\theta}$, ℓ is the number of free parameters in $\boldsymbol{\theta}$, $|\cdot|$ is the determinant of a matrix and $\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}$ is the Fisher information matrix evaluated at $\hat{\boldsymbol{\theta}}$. The Fisher information matrix is defined by

$$\boldsymbol{F}_{\boldsymbol{\theta}} = -\frac{\partial^2 \log f(\boldsymbol{Y}|\boldsymbol{\theta}, K)}{\partial \boldsymbol{\theta}^2}. \tag{2.5}$$

Usually we examine Eq. (2.4) in the logarithm form. Hence, Bayesian inference criterion can be described by

$$\begin{aligned} & \arg\min_K[-\log P(K|\boldsymbol{Y})] \\ = \; & \arg\min_K[-\log f(\boldsymbol{Y}|K)] \\ = \; & \arg\min_K \left[-\log f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, K) + \frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| + \frac{\ell}{2}\log\frac{1}{2\pi} - \log h(\hat{\boldsymbol{\theta}}|K)\right]. \end{aligned} \tag{2.6}$$

It is shown in [54] that asymptotically

$$-\log f(\boldsymbol{Y}|K) = -\log f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, K) + \frac{\ell}{2}\log N. \tag{2.7}$$

The above asymptotic criterion is usually referred as Bayesian Inference Criterion (BIC). The advantage of using BIC is that it does not depend on the prior distribution $h(\hat{\boldsymbol{\theta}}|K)$.

However, this large-sample criterion may not work satisfactorily for small or medium sample cases. This drawback can be compensated to some extent by specifying a sensible prior probability $h(\hat{\boldsymbol{\theta}}|K)$ in Eq. (2.6) if we have some knowledge of the given data set.

A model selection is actually performed in two levels. The first level is to choose the best model class to fit the data and the second level is to choose the best model under the chosen model class. The ML estimate $\hat{\boldsymbol{\theta}}$ under the chosen model class is usually taken as the best model.

Notations

Throughout this thesis, if a model $\boldsymbol{\theta}$ is considered, it is under some model class K. For notational simplicity, we choose not to record this dependence explicitly with the understanding that $\boldsymbol{\theta}$ is dependent on K implicitly.

2.3 Minimum Encoding Inference

There are two major approaches of minimum encoding inference: one by Wallace [62, 64] and the other by Rissanen [50, 52]. Wallace termed his inference method the Minimum Message Length (MML) criterion while Rissanen termed his the Minimum Description Length (MDL) criterion. MDL appears more widely known in engineering fields [47, 65, 70–72]. Wallace and Rissanen's Royal Statistical Society meeting review papers on MML and MDL were shown side by side in 1987 [52, 64]. A comprehensive comparison between them was presented in 1994 [8]. The fundamental ideas of MML and MDL are the same:

> Given a data set and a family of competing statistical models, the best model is the one that yields the minimal coding length of the data.

We assume that there are a data set $\boldsymbol{Y} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N\}$ and a statistical model determined by ℓ parameters which is described as $\boldsymbol{\theta}$, $\boldsymbol{\theta} \in R^{\ell}$. To assess the goodness of fit between the model and the data, we construct a code length $L(\boldsymbol{\theta})$ of the model and a code length $L(\boldsymbol{Y}|\boldsymbol{\theta})$ of the data in terms of the model under a proper encoding scheme. A good model is one leading a concise total description length $L(\boldsymbol{Y}, \boldsymbol{\theta})$ which is the sum of $L(\boldsymbol{\theta})$ and $L(\boldsymbol{Y}|\boldsymbol{\theta})$

— the shorter the better. i.e. the minimum encoding inference is to find

$$\theta^* = \arg\min L(\boldsymbol{Y}, \theta) \tag{2.8}$$

where

$$L(\boldsymbol{Y}, \theta) = L(\boldsymbol{Y}|\theta) + L(\theta) \tag{2.9}$$

Let $f(\boldsymbol{Y}|\theta)$ be the conditional p.d.f. of $\boldsymbol{Y}$ given θ, we regard $-\log f(\boldsymbol{Y}|\theta)$, known in information theory as self-information, to be the number of "nits"[2] it takes to encode $\boldsymbol{Y}$ with an ideal code relative to the assumed model of the data. i.e.,

$$L(\boldsymbol{Y}, \theta) = -\log f(\boldsymbol{Y}|\theta) + L(\theta). \tag{2.10}$$

To encode θ, we need to know its prior probability. In addition, we can only encode θ to a limited precision, so an optimal quantization is needed to yield the total coding length as short as possible. Briefly, the differences between Rissanen's MDL and Wallace's MML are the view of the prior and the selection of the optimal quantization.

2.3.1 Rissanen's Description Length

The major steps of Rissanen's Minimum Description Length procedure [51] are described as follows:

1. **Quantization:** partition the parameter space into regions with centers θ_i, $\theta_i \in R^\ell$, $i \in \mathcal{N}$ and quantization volumes $V(\theta_i)$.
2. **Indexing:** map θ_i into a positive integer j.
3. **Encoding a prior:** use a so-called universal prior for positive integers to encode j by the length $L(j)$.
4. **Total description length:**

$$-\log f(\boldsymbol{Y}|\theta_i) + L(j). \tag{2.11}$$

[2]The unit is called "nit" by using the natural logarithm. In fact, we are concerned with calculating the length of description for the inference but we do not need really to transmit it. Therefore, we use codes that are efficient in terms of code length, but may not be efficient in the time required to encode/decode data.

Note that for each i, we have a probability model $f(\boldsymbol{Y}|\boldsymbol{\theta}_i)$.

The procedure is not complete until we specify:

- the optimal quantization volumes $V(\boldsymbol{\theta}_i)$, $i \in \mathcal{N}$.
- the mapping from $\boldsymbol{\theta}_i$ to a positive integer j.
- the universal prior for integers.

A universal prior for integers

Since the codes of both j and $\boldsymbol{Y}$ are strings, we can not just attach them next to each other. If the decoder always reads the code string from left to right, then a necessary and sufficient condition for the decoder to be able to separate the codeword from whatever string follows it, is that the codewords for the integers form a *prefix* set. This means that no codeword is allowed to be a prefix of another. Based on this point, the optimum length of a positive integer j used by Rissanen is

$$L(j) = \log^* j + \log c \tag{2.12}$$

where $\log^* j = \log j + \log\log j + \log\log\log j + \cdots$, only including positive terms, and c is a small constant ($\simeq 2.865064$). Therefore, the universal prior is defined as

$$P(j) = \frac{1}{j} \times \frac{1}{\log j} \times \cdots \frac{1}{\log \cdots \log j} \times \frac{1}{c}. \tag{2.13}$$

For simplicity, the first-order approximation of the length $L(j)$ is used, i.e.,

$$L(j) \simeq \log j. \tag{2.14}$$

Furthermore, a lattice quantization is used here. In his approximation to the optimal quantization, Rissanen obtained his description length as

$$L(\boldsymbol{Y}, \boldsymbol{\theta}) \simeq -\log f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}) + \frac{\ell}{2}\log(\hat{\boldsymbol{\theta}}^T \boldsymbol{F}_{\hat{\boldsymbol{\theta}}} \hat{\boldsymbol{\theta}}) - \frac{\ell+1}{2}\log \ell + \frac{\ell}{2}\log(2\pi e N) \tag{2.15}$$

where $\hat{\boldsymbol{\theta}}$ is the ML estimate and the Fisher information matrix is defined by

$$\boldsymbol{F}_{\boldsymbol{\theta}} = -\frac{\partial^2 \log f(\boldsymbol{Y}|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^2}. \tag{2.16}$$

2.3.2 Wallace's Message Length

The major steps of Wallace's Minimum Message Length procedure [64] are briefly described as follows:

1. **Quantization:** partition the parameter space into regions with centers $\boldsymbol{\theta}_i$, $\boldsymbol{\theta}_i \in R^\ell$, $i \in \mathcal{N}$, and quantization volumes $V(\boldsymbol{\theta}_i)$.

2. **Choosing a prior:** specify some sensible prior $h(\boldsymbol{\theta}_i)$.

3. **Total message length:**

$$-\log f(\boldsymbol{Y}|\boldsymbol{\theta}_i) - \log[\int_{V(\boldsymbol{\theta}_i)} h(\boldsymbol{\theta}) d\boldsymbol{\theta}].$$

Note that for each i, we have a probability model $f(\boldsymbol{Y}|\boldsymbol{\theta}_i)$. It is convenient to approximate the integral (for sufficiently small $V(\boldsymbol{\theta}_i)$) as follows

$$-\log f(\boldsymbol{Y}|\boldsymbol{\theta}_i) - \log[V(\boldsymbol{\theta}_i)h(\boldsymbol{\theta}_i)]. \tag{2.17}$$

The procedure is not complete until we specify:

- the optimal quantization volume $V(\boldsymbol{\theta}_i)$.
- a prior probability density $h(\boldsymbol{\theta}_i)$ over the parameter space.

By a calculation similar to the optimal lattice quantization, Wallace derived his message length as

$$L(\boldsymbol{Y}, \boldsymbol{\theta}) \simeq -\log f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}) + \frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| + \frac{\ell}{2} + \frac{\ell}{2}\log G_\ell - \log h(\hat{\boldsymbol{\theta}}) \tag{2.18}$$

where $\hat{\boldsymbol{\theta}}$ is the ML estimate, $|\cdot|$ denotes the determinant of a matrix and G_ℓ is the ℓ-dimensional optimal lattice quantization constant which can be found in [18, Page 61], and $h(\hat{\boldsymbol{\theta}})$ is the prior p.d.f. of $\hat{\boldsymbol{\theta}}$.

2.4 Framework: Penalized Likelihood Method

By comparing Eqs. (2.6) and (2.18), we observe that the major difference between them is a quantization constant. Specifically, the hyper-sphere constant $\frac{1}{2\pi}$ is used in Laplace's

method for Bayesian inference but the optimal lattice constant G_ℓ is used in the MML message length. In the view of optimal quantization, the MML coding length is more accurate. In addition, we believe that we should specify a sensible prior if we have some knowledge of a given data set, instead of some universal prior. Hence, we prefer the MML message length formula Eq. (2.18) for our application to model-based cluster analysis.

We also notice that all model selection criteria considered here consist of two parts, one is the log-likelihood function which measures the goodness of fit between the data and the model, and another is a penalty function which measures the complexity of the model. An inference aims to balance the trade-off between goodness of fit and model complexity. Hence in practice, we can introduce a penalty weight for the penalty function to control the trade-off as follows:

$$L(\boldsymbol{Y}, \boldsymbol{\theta}) = L(\boldsymbol{Y}|\boldsymbol{\theta}) + \lambda L(\boldsymbol{\theta}) \tag{2.19}$$

where λ is the penalty weight.

Roughly speaking, an inference tends to underestimate when λ is large, and it tends to overestimate when λ is small. Therefore, we have to determine the suitable range λ which guarantee the true estimation. This is investigated in detail in Section 4.4.

Chapter 3

Model-Based Clustering

3.1 Introduction

In model-based clustering [28, 39], it is assumed that the data under consideration are generated from a finite mixture of probability distributions; each component of the mixture represents a different group or cluster. Therefore, given a set of observed data vectors, our objectives are

- cluster validation, to determine the number of components in the mixture;
- clustering, to determine which data vectors arise from each component.

In the previous chapter, Bayesian inference and minimum encoding inference (MDL and MML) for model selection were discussed. In addition, different clustering algorithms for Gaussian mixture models were briefly compared in Section 1.2. These algorithms include MCLUST, Autoclass and SNOB. Only SNOB is suitable for both high dimensional cases and small sample cases.

In this chapter, we apply minimum encoding inference to model-based clustering, propose an appropriate coding length measure for cluster validation, and fully derive the coding lengths under four different Gaussian mixture models. Then we describe our model-based clustering algorithm, compare it with SNOB, and demonstrate by extensive simulations that

our algorithm is more suitable than SNOB to process high dimensional data with better performance on small sample cases. In addition, we also investigate the performance of the coding length measure based on the asymptotic method to Bayesian Inference. Part of this chapter has been published in [68].

3.2 General Coding Length Measure

Given a data set $\boldsymbol{Y}$ consisting of N observed data vectors $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N$, each of dimension M, the data vector $\boldsymbol{y}_n$ $(n = 1, 2, \ldots, N)$ is to be assigned among $\hat{K}$ clusters. Let an association parameter vector $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_N]^T$, such that if $\alpha_n = k$, then the data vector $\boldsymbol{y}_n$ is assigned to the kth cluster. In a model-based method, the k-th cluster ($k = 1, \cdots, \hat{K}$) is assumed to be a sample of a simple distribution, denoted by $f_k(\cdot)$ with its parameter vector $\boldsymbol{\theta}_k$. Therefore, the conditional density function for the data is

$$f(\boldsymbol{Y}|\boldsymbol{\theta}, \boldsymbol{\alpha}) = \prod_{n=1}^{N} f_{\alpha_n}(\boldsymbol{y}_n) \tag{3.1}$$

where the mixture model parameter vector $\boldsymbol{\theta}$ consists of independent parameters in the set $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{\hat{K}}\}$ and ℓ is the dimension of $\boldsymbol{\theta}$.

Now, from Shannon's coding theorem [19], the minimum code length is given by the entropy of the data. Thus, using the natural logarithm, the minimum code length in "nits" (see the footnote of Section 2.3) is

$$\begin{aligned} L(\boldsymbol{Y}, \hat{K}) &= E[-\log f(\boldsymbol{Y})] \\ &\simeq -\log f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}}) - \log f(\hat{\boldsymbol{\theta}}) - \log P(\hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.2}$$

In Eq. (3.2), we have used the evaluation of the coding length at the maximum likelihood (ML) estimates $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\alpha}}$ to approximate the expected coding length, where $f(\hat{\boldsymbol{\theta}})$ is the probability density function evaluated at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$, and $P(\hat{\boldsymbol{\alpha}})$ is the probability of $\boldsymbol{\alpha} = \hat{\boldsymbol{\alpha}}$.

First, let us examine the last term in Eq. (3.2). $\boldsymbol{\alpha}$ is a particular association vector, the nth element α_n of which denotes the association of the nth data vector with the α_nth cluster. Now, to partition N data vectors into $\hat{K}$ clusters, the number of different ways as

shown in Appendix A is

$$S(N,\hat{\alpha}) = \frac{N!}{N_1!N_2!\cdots N_{\hat{K}}!(\hat{K}-1)!} \times \frac{1}{m_1!m_2!\cdots m_N!}, \tag{3.3}$$

where N_k is the number of data vectors assigned to the kth cluster ($k = 1,\cdots,\hat{K}$; $N_1 + N_2 + \cdots + N_{\hat{K}} = N$), and m_n is the number of clusters with n data vectors ($n = 1, 2, \cdots, N$).

If a uniform *a priori* probability is assumed for α, then

$$-\log P(\hat{\alpha}) = \log S(N,\hat{\alpha}). \tag{3.4}$$

The first and second terms in Eq. (3.2) can be described by the message length formula of Eq. (2.18). Therefore, we have the total coding length

$$L(\boldsymbol{Y},\hat{K}) = -\log f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\alpha}) + \frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| + \frac{\ell}{2} + \frac{\ell}{2}\log G_\ell - \log h(\hat{\boldsymbol{\theta}}) + \log S(N,\hat{\alpha}) \tag{3.5}$$

where ℓ is the number of independent parameters in $\boldsymbol{\theta}$, G_ℓ is the ℓ-dimensional optimal lattice quantization constant which can be found in [18, Page 61], $h(\hat{\boldsymbol{\theta}})$ is the prior p.d.f. of $\hat{\boldsymbol{\theta}}$, $|\cdot|$ is the determinant of a matrix and $\boldsymbol{F}_{\boldsymbol{\theta}}$ is the Fisher information matrix defined by

$$\boldsymbol{F}_{\boldsymbol{\theta}} = -\frac{\partial^2 \log f(\boldsymbol{Y}|\boldsymbol{\theta},\alpha)}{\partial\boldsymbol{\theta}^2}. \tag{3.6}$$

The first term in Eq. (3.5) is the negative log-likelihood which measures the goodness of fit between the data and the model. We denote it by $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\alpha})$, i.e.,

$$L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\alpha}) = -\log f(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\alpha})$$

The rest of the terms in Eq. (3.5) forms the penalty function which measures the model complexity. If the dominant penalty term $\frac{\ell}{2}\log N$ shown in Eq. (2.7) is used, we have the asymptotic coding length

$$L_{\text{asy}}(\boldsymbol{Y},\hat{K}) = L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\alpha}) + \frac{\ell}{2}\log N + \log S(N,\hat{\alpha}). \tag{3.7}$$

3.3 Coding Lengths Under Gaussian Mixture Models

In this section we investigate how to calculate each term in Eq. (3.5) under Gaussian mixture models. We start with $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\alpha})$, the log-likelihood term. In this case, $f_k(\cdot)$ is assumed to

be the density function of a multivariate normal distribution with its mean vector μ_k and its covariance matrix Σ_k. Suppose that a particular association vector $\alpha = [\alpha_1\ \alpha_2\ \ldots\ \alpha_N]^T$ partitions N data vectors into $\hat{K}$ groups such that we have

$$Y_\alpha = \{\ \{y_1(1),\ldots,y_{N_1}(1)\},\ldots,\{y_1(\hat{K}),\ldots,y_{N_{\hat{K}}}(\hat{K})\}\ \},$$

the conditional density function for the data [43] is

$$f(Y|\theta,\alpha) = (2\pi)^{-MN/2}\prod_{k=1}^{\hat{K}}|\Sigma_k|^{-N_k/2}\exp\left[-\frac{1}{2}\sum_{k=1}^{\hat{K}}\sum_{n=1}^{N_k}(y_n(k)-\mu_k)^T\Sigma_k^{-1}(y_n(k)-\mu_k)\right]. \tag{3.8}$$

Hence, the $L(Y|\hat{\theta},\hat{\alpha})$ term is expressed as

$$\begin{aligned} L(Y|\hat{\theta},\hat{\alpha}) &= \sum_{k=1}^{\hat{K}}N_k\log|\Sigma_k| + \sum_{k=1}^{\hat{K}}tr\left[\Sigma_k^{-1}\sum_{n=1}^{N_k}(y_n(k)-\mu_k)(y_n(k)-\mu_k)^T\right] + \frac{MN}{2}\log(2\pi) \\ &= \sum_{k=1}^{\hat{K}}N_k\log|\Sigma_k| + \sum_{k=1}^{\hat{K}}tr\left[\Sigma_k^{-1}W_k\right] + \frac{MN}{2}\log(2\pi) \end{aligned} \tag{3.9}$$

where

$$\hat{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k}y_n(k),\quad k=1,\ldots,\hat{K}; \tag{3.10}$$

and

$$W_k = \sum_{n=1}^{N_k}(y_n(k)-\hat{\mu}_k)(y_n(k)-\hat{\mu}_k)^T. \tag{3.11}$$

The Fisher information matrix $F_{\hat{\theta}}$ in the second term of Eq. (3.5) is very sophisticated for a general structure of Σ_k. Furthermore, the dimension M for our application is usually higher than 40 so there are more than 40×40 parameters in just one covariance matrix Σ_k! This general structure will generate severe numerical problems when only small, or medium samples are available. To avoid the above limitations, we assume that the covariance matrix Σ_k is diagonal. In this case, It is easy to verify that

$$\left.\frac{\partial tr(W_k)}{\partial\mu_k}\right|_{\mu_k=\hat{\mu}_k} = O,\quad \text{for } k=1,\ldots,\hat{K}. \tag{3.12}$$

Therefore, $F_{\hat{\theta}}$ is a diagonal matrix according to that

$$\left.\frac{\partial^2 L(Y|\theta,\alpha)}{\partial\theta_i\partial\theta_j}\right|_{\theta=\hat{\theta}} = 0,\quad \text{for } i\neq j,\ i,j=1,\ldots,\ell. \tag{3.13}$$

As we see, coding is usually based on $\hat{\boldsymbol{\theta}}$ since the true $\boldsymbol{\theta}$ is unknown. $\hat{\boldsymbol{\theta}}$ is a vector with ℓ elements $\hat{\theta}_i$, $i = 1, \cdots, \ell$. These elements are statistically independent when $\boldsymbol{\Sigma}_k$, $k = 1, \cdots, \hat{K}$, are diagonal, so each $\hat{\theta}_i$ is quantized independently. In other words, the quantization here is actually performed in one-dimension, instead of ℓ-dimension. For the one-dimension case, the optimal quantization constant is $\frac{1}{12}$ [18, Page 61]. Hence, the optimal quantization constant we used is

$$G_\ell = G_1 = \frac{1}{12}. \tag{3.14}$$

Below we consider four different covariance structures:

- Covariance Structure 1: $\boldsymbol{\Sigma}_k = \sigma^2 \boldsymbol{I}, \quad \forall k$
- Covariance Structure 2: $\boldsymbol{\Sigma}_k = \sigma_k^2 \boldsymbol{I}, \quad \forall k$
- Covariance Structure 3: $\boldsymbol{\Sigma}_k = \boldsymbol{D}, \quad \forall k$
- Covariance Structure 4: $\boldsymbol{\Sigma}_k = \boldsymbol{D}_k, \quad \forall k$

The first covariance structure is the simplest, and mainly used for the purpose of theoretical performance analysis in Chapter 4. The second and fourth structures have been successfully applied to intrapulse analysis in Chapter 5. The third one might be useful for other applications so we include it here for completeness.

To fully derive a coding length, we assume a uniform prior probability for each parameter in $\boldsymbol{\mu}_k = [\mu_{k1}, \cdots, \mu_{kM}]^T$ and the underlying covariance matrix $\boldsymbol{\Sigma}_k$ over some certain regions. Let $\hat{\boldsymbol{\mu}}_0$ and $\boldsymbol{W}_0$ be the mean vector and the covariance matrix of the whole data set $\boldsymbol{Y}$ respectively, i.e.,

$$\hat{\boldsymbol{\mu}}_0 = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{y}_n, \tag{3.15}$$

$$\boldsymbol{W}_0 = \sum_{n=1}^{N} (\boldsymbol{y}_n - \hat{\boldsymbol{\mu}}_0)(\boldsymbol{y}_n - \hat{\boldsymbol{\mu}}_0)^T. \tag{3.16}$$

Parametric regions will be determined by $\hat{\boldsymbol{\mu}}_0$ and $\boldsymbol{W}_0$ according to the underlying covariance structure, as detailed in the following subsections.

3.3.1 Covariance Structure 1: $\Sigma_k = \sigma^2 I, \quad \forall k$

Under the assumed covariance structure, we have only one parameter σ to characterize all covariance matrices, and $\hat{K}M$ parameters $\{\mu_{km} | \quad k = 1, \ldots, \hat{K}; \quad m = 1, \ldots, M\}$ to characterize all mean vectors. Therefore, the number of free parameters are

$$\ell = \hat{K}M + 1. \tag{3.17}$$

A. The log-likelihood term $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})$

Let $\hat{\sigma}$ be the ML estimate of σ. From Eq. (3.9), we obtain

$$L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}}) = \frac{MN}{2}\log(\hat{\sigma}^2) + \frac{1}{2\hat{\sigma}^2}tr(\boldsymbol{W}) + \frac{MN}{2}\log(2\pi) \tag{3.18}$$

where $tr(\cdot)$ is the trace of a matrix and

$$\boldsymbol{W} = \sum_{k=1}^{\hat{K}} \sum_{n=1}^{N_k} (\boldsymbol{y}_n(k) - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{y}_n(k) - \hat{\boldsymbol{\mu}}_k)^T \tag{3.19}$$

with $\hat{\boldsymbol{\mu}}_k$ given by Eq. (3.10).

Differentiating Eq. (3.8) with respect to σ and equating it to zero, we have the ML estimate of σ given by

$$\hat{\sigma} = \sqrt{\frac{tr(\boldsymbol{W})}{MN}}. \tag{3.20}$$

Substituting Eq. (3.20) into Eq. (3.18), we have

$$L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}}) = \frac{MN}{2}\log\frac{tr(\boldsymbol{W})}{MN} + \frac{MN}{2} + \frac{MN}{2}\log(2\pi). \tag{3.21}$$

B. The Fisher term $\frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}|$

Under this mixture model,

$$\frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| = \frac{1}{2}\log\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})}{\partial\sigma^2} + \frac{1}{2}\sum_{k=1}^{\hat{K}}\sum_{m=1}^{M}\log\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})}{\partial\mu_{km}^2}. \tag{3.22}$$

Differentiating Eq. (3.8) twice with respect to σ and using Eq. (3.20), we have

$$\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})}{\partial\sigma^2} = \frac{2(MN)^2}{tr(\boldsymbol{W})}. \tag{3.23}$$

Differentiating Eq. (3.8) twice with respect to μ_k and using Eq. (3.20), we have

$$\begin{aligned}\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\alpha}})}{\partial \mu_{km}^2} &= \frac{1}{\hat{\sigma}^2} tr\left(\frac{\partial^2 \boldsymbol{W}}{\partial \mu_{km}^2}\right) = \frac{N_k}{\hat{\sigma}^2} \\ &= \frac{N_k NM}{tr(\boldsymbol{W})}. \end{aligned} \tag{3.24}$$

Substituting Eqs. (3.23) and (3.24) into Eq. (3.22), we have

$$\frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| = -\frac{(\hat{K}M+1)}{2}\log tr(\boldsymbol{W}) + \frac{M}{2}\sum_{k=1}^{\hat{K}}\log(MNN_k) + \log(\sqrt{2}MN). \tag{3.25}$$

C. The prior term $-\log h(\hat{\boldsymbol{\theta}})$

Denoting the ranges of $\hat{\mu}_{km}$ and $\hat{\sigma}$ by r_{km} and ρ respectively and assuming a uniform distribution for each, then

$$h(\hat{\boldsymbol{\theta}}) = \frac{1}{\rho}\prod_{k=1}^{\hat{K}}\prod_{m=1}^{M}\frac{1}{r_{km}}. \tag{3.26}$$

Let $\hat{\sigma}_0$ be the standard deviation of the whole data set $\boldsymbol{Y}$. From Eq. (3.20),

$$\hat{\sigma}_0 = \sqrt{\frac{tr(\boldsymbol{W}_0)}{MN}}. \tag{3.27}$$

where $\boldsymbol{W}_0$ has been defined by Eq. (3.16). For simplicity, we further assume that

$$\begin{aligned} r_{km} &= 2\hat{\sigma}_0,\ m = 1,\cdots,M; \\ \rho &= \hat{\sigma}_0. \end{aligned}$$

Hence, the prior term contributes to the coding length by

$$-\log h(\hat{\boldsymbol{\theta}}) = \hat{K}M\log 2 + \frac{\hat{K}M+1}{2}\log\frac{tr(\boldsymbol{W}_0)}{MN}. \tag{3.28}$$

D. The total coding length $L(\boldsymbol{Y},\hat{K})$

Eqs. (3.21), (3.25), (3.17), (3.14) and (3.28) form the total coding length defined in Eq. (3.5). By removing the parts independent of $\hat{K}$ (because they are inconsequential to the subsequent minimization), we can simplify its expression to

$$\begin{aligned} L_1(\boldsymbol{Y},\hat{K}) &= \frac{MN}{2}\log tr(\boldsymbol{W}) + \frac{\hat{K}M+1}{2}\log\frac{tr(\boldsymbol{W}_0)}{tr(\boldsymbol{W})} \\ &\quad + \frac{M}{2}\sum_{k=1}^{\hat{K}}\log N_k + \frac{\hat{K}M}{2}\log\frac{e}{3} + \log S(N,\hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.29}$$

where $\boldsymbol{W}, \boldsymbol{W}_0$ and $S(N, \hat{\boldsymbol{\alpha}})$ are defined in Eqs. (3.19), (3.16) and (3.3), respectively.

3.3.2 Covariance Structure 2: $\boldsymbol{\Sigma}_k = \sigma_k^2 \boldsymbol{I}, \quad \forall k$

Under the assumed covariance structure, we have $\hat{K}$ parameters $\{\sigma_k | \ k = 1, \ldots, \hat{K}\}$ to characterize all covariance matrices, and $\hat{K}M$ parameters $\{\mu_{km} | \ k = 1, \ldots, \hat{K}; \ m = 1, \ldots, M\}$ to characterize all mean vectors. Therefore, the number of free parameters are

$$\ell = \hat{K}M + \hat{K}. \tag{3.30}$$

A. The log-likelihood term $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})$

Let $\hat{\sigma}_k$ be the ML estimate of σ_k. From Eq. (3.9), we obtain

$$L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}}) = \frac{M}{2} \sum_{k=1}^{\hat{K}} N_k \log(\hat{\sigma}_k^2) + \sum_{k=1}^{\hat{K}} \frac{1}{2\hat{\sigma}_k^2} \log tr(\boldsymbol{W}_k) + \frac{MN}{2} \log(2\pi) \tag{3.31}$$

where $\boldsymbol{W}_k$ is given by Eq. (3.11).

Differentiating Eq. (3.8) with respect to σ_k and equating it to zero, we have the ML estimate of σ_k given by

$$\hat{\sigma}_k = \sqrt{\frac{tr(\boldsymbol{W}_k)}{MN_k}}. \tag{3.32}$$

Substituting Eq. (3.32) into Eq. (3.31), we have

$$L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}}) = \frac{M}{2} \sum_{k=1}^{\hat{K}} N_k \log tr(\boldsymbol{W}_k) - \frac{M}{2} \sum_{k=1}^{\hat{K}} N_k \log(MN_k) + \frac{MN}{2} + \frac{MN}{2} \log(2\pi). \tag{3.33}$$

B. The Fisher term $\frac{1}{2} \log |\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}|$

Under this covariance structure,

$$\frac{1}{2} \log |\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| = \frac{1}{2} \sum_{k=1}^{\hat{K}} \log \frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})}{\partial \sigma_k^2} + \frac{1}{2} \sum_{k=1}^{\hat{K}} \sum_{m=1}^{M} \log \frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})}{\partial \mu_{km}^2}. \tag{3.34}$$

Differentiating Eq. (3.8) twice with respect to σ_k and using Eq. (3.32), we have

$$\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})}{\partial \sigma_k^2} = \frac{2(MN_k)^2}{tr(\boldsymbol{W}_k)}. \tag{3.35}$$

Differentiating Eq. (3.8) twice with respect to μ_k and using Eq. (3.32), we have

$$\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\alpha}})}{\partial \mu_{km}^2} = \frac{MN_k^2}{tr(\boldsymbol{W}_k)}. \tag{3.36}$$

Substituting Eqs. (3.35) and (3.36) into Eq. (3.34), we have

$$\frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| = -\frac{M+1}{2}\sum_{k=1}^{\hat{K}}\log tr(\boldsymbol{W}_k) + (M+1)\sum_{k=1}^{\hat{K}}\log N_k + (\frac{M}{2}+1)\hat{K}\log M + \frac{\hat{K}}{2}\log 2. \tag{3.37}$$

C. The prior term $-\log h(\hat{\boldsymbol{\theta}})$

Denoting the ranges of $\hat{\mu}_{km}$ and $\hat{\sigma}_k$ by r_{km} and ρ_k respectively and assuming a uniform distribution for each, then

$$h(\hat{\boldsymbol{\theta}}) = \prod_{k=1}^{\hat{K}}\left(\frac{1}{\rho_k}\prod_{m=1}^{M}\frac{1}{r_{km}}\right). \tag{3.38}$$

Similarly to Covariance Structure 1, we further assume that

$$\begin{aligned} r_{km} &= 2\hat{\sigma}_0,\ m = 1,\cdots,M; \\ \rho_k &= \hat{\sigma}_0,\ k = 1,\cdots,\hat{K}. \end{aligned}$$

Hence, the prior term contributes to the coding length by

$$-\log h(\hat{\boldsymbol{\theta}}) = \hat{K}M\log 2 + \frac{\hat{K}M+\hat{K}}{2}\log\frac{tr(\boldsymbol{W}_0)}{MN}. \tag{3.39}$$

D. The total coding length $L(\boldsymbol{Y},\hat{K})$

Eqs. (3.33), (3.37), (3.30), (3.14) and (3.39) form the total coding length defined in Eq. (3.5). By removing the parts independent of $\hat{K}$ (because they are inconsequential to the subsequent minimization), we can simplify its expression to

$$\begin{aligned} L_2(\boldsymbol{Y},\hat{K}) &= \frac{M}{2}\sum_{k=1}^{\hat{K}}N_k\log tr(\boldsymbol{W}_k) - \frac{M}{2}\sum_{k=1}^{\hat{K}}N_k\log N_k + \frac{M+1}{2}\sum_{k=1}^{\hat{K}}\log\frac{tr(\boldsymbol{W}_0)}{tr(\boldsymbol{W}_k)} \\ &\quad +(M+1)\sum_{k=1}^{\hat{K}}\log N_k + \frac{\hat{K}M}{2}\log\frac{e}{3N} + \frac{\hat{K}}{2}\log\frac{e}{6N} + \log S(N,\hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.40}$$

where $\boldsymbol{W}_k, \boldsymbol{W}_0$ and $S(N,\hat{\boldsymbol{\alpha}})$ are defined in Eqs. (3.11), (3.16) and (3.3), respectively.

Differentiating Eq. (3.8) twice with respect to σ_{km} and using Eq. (3.55), we have

$$\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\alpha}})}{\partial \sigma_{km}^2} = \frac{2N_k^2}{w_{km}}. \tag{3.58}$$

Differentiating Eq. (3.8) twice with respect to μ_k and using Eq. (3.55), we have

$$\frac{\partial^2 L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\alpha}})}{\partial \mu_{km}^2} = \frac{N_k^2}{w_{km}}. \tag{3.59}$$

Substituting Eqs. (3.58) and (3.59) into Eq. (3.57), we have

$$\frac{1}{2}\log|\boldsymbol{F}_{\hat{\boldsymbol{\theta}}}| = -\sum_{k=1}^{\hat{K}} \log|diag(\boldsymbol{W}_k)| + 2M\sum_{k=1}^{\hat{K}} \log N_k + \frac{\hat{K}M}{2}\log 2. \tag{3.60}$$

C. The prior term $-\log h(\hat{\boldsymbol{\theta}})$

Denoting the ranges of $\hat{\mu}_{km}$ and $\hat{\sigma}_{km}$ by r_{km} and ρ_{km} respectively and assuming a uniform distribution for each, then

$$h(\hat{\boldsymbol{\theta}}) = \prod_{k=1}^{\hat{K}}\prod_{m=1}^{M} \frac{1}{\rho_{km} r_{km}}. \tag{3.61}$$

Similarly to Covariance Structure 3, we further assume that

$$\begin{aligned} r_{km} &= 2\hat{\sigma}_{0m},\ k = 1,\cdots,\hat{K}; \\ \rho_{km} &= \hat{\sigma}_{0m},\ m = 1,\cdots,M. \end{aligned}$$

Hence, the prior term contributes to the coding length by

$$-\log h(\hat{\boldsymbol{\theta}}) = \hat{K}M\log 2 + \hat{K}\log\left|\frac{diag(\boldsymbol{W}_0)}{N}\right|. \tag{3.62}$$

D. The total coding length $L(\boldsymbol{Y},\hat{K})$

Eqs. (3.56), (3.60), (3.53), (3.14) and (3.62) form the total coding length defined in Eq. (3.5). By removing the parts independent of $\hat{K}$, we can simplify its expression to

$$\begin{aligned} L_4(\boldsymbol{Y},\hat{K}) &= \frac{1}{2}\sum_{k=1}^{\hat{K}} N_k \log|diag(\boldsymbol{W}_k)| - \frac{M}{2}\sum_{k=1}^{\hat{K}} N_k \log N_k + \sum_{k=1}^{\hat{K}} \log\frac{|diag(\boldsymbol{W}_0)|}{|diag(\boldsymbol{W}_k)|} \\ &\quad + 2M\sum_{k=1}^{\hat{K}} \log N_k + \hat{K}M\log\frac{\sqrt{2}e}{6N} + \log S(N,\hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.63}$$

where $\boldsymbol{W}_k, \boldsymbol{W}_0$ and $S(N,\hat{\boldsymbol{\alpha}})$ are defined in Eqs. (3.11), (3.16) and (3.3), respectively.

3.3.5 Summary of Coding Lengths

For easy comparison and reference, the coding lengths under the above four covariance structures are listed below:

Covariance Structure 1: $\boldsymbol{\Sigma}_k = \sigma^2 \boldsymbol{I}, \quad \forall k$

$$\begin{aligned} L_1(\boldsymbol{Y}, \hat{K}) &= \frac{MN}{2} \log tr(\boldsymbol{W}) + \frac{\hat{K}M+1}{2} \log \frac{tr(\boldsymbol{W}_0)}{tr(\boldsymbol{W})} \\ &\quad + \frac{M}{2} \sum_{k=1}^{\hat{K}} \log N_k + \frac{\hat{K}M}{2} \log \frac{e}{3} + \log S(N, \hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.64}$$

where $\boldsymbol{W}, \boldsymbol{W}_0$ and $S(N, \hat{\boldsymbol{\alpha}})$ are defined in Eqs. (3.19), (3.16) and (3.3), respectively.

Covariance Structure 2: $\boldsymbol{\Sigma}_k = \sigma_k^2 \boldsymbol{I}, \quad \forall k$

$$\begin{aligned} L_2(\boldsymbol{Y}, \hat{K}) &= \frac{M}{2} \sum_{k=1}^{\hat{K}} N_k \log tr(\boldsymbol{W}_k) - \frac{M}{2} \sum_{k=1}^{\hat{K}} N_k \log N_k + \frac{M+1}{2} \sum_{k=1}^{\hat{K}} \log \frac{tr(\boldsymbol{W}_0)}{tr(\boldsymbol{W}_k)} \\ &\quad + (M+1) \sum_{k=1}^{\hat{K}} \log N_k + \frac{\hat{K}M}{2} \log \frac{e}{3N} + \frac{\hat{K}}{2} \log \frac{e}{6N} + \log S(N, \hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.65}$$

where $\boldsymbol{W}_k, \boldsymbol{W}_0$ and $S(N, \hat{\boldsymbol{\alpha}})$ are defined in Eqs. (3.11), (3.16) and (3.3), respectively.

Covariance Structure 3: $\boldsymbol{\Sigma}_k = \boldsymbol{D}, \quad \forall k$

$$\begin{aligned} L_3(\boldsymbol{Y}, \hat{K}) &= \frac{N}{2} \log |diag(\boldsymbol{W})| + \frac{\hat{K}+1}{2} \log \frac{|diag(\boldsymbol{W}_0)|}{|diag(\boldsymbol{W})|} \\ &\quad + \frac{M}{2} \sum_{k=1}^{\hat{K}} \log N_k + \frac{\hat{K}M}{2} \log \frac{e}{3} + \log S(N, \hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.66}$$

where $\boldsymbol{W}, \boldsymbol{W}_0$ and $S(N, \hat{\boldsymbol{\alpha}})$ are defined in Eqs. (3.19), (3.16) and (3.3), respectively.

Covariance Structure 4: $\boldsymbol{\Sigma}_k = \boldsymbol{D}_k, \quad \forall k$

$$\begin{aligned} L_4(\boldsymbol{Y}, \hat{K}) &= \frac{1}{2} \sum_{k=1}^{\hat{K}} N_k \log |diag(\boldsymbol{W}_k)| - \frac{M}{2} \sum_{k=1}^{\hat{K}} N_k \log N_k + \sum_{k=1}^{\hat{K}} \log \frac{|diag(\boldsymbol{W}_0)|}{|diag(\boldsymbol{W}_k)|} \\ &\quad + 2M \sum_{k=1}^{\hat{K}} \log N_k + \hat{K}M \log \frac{\sqrt{2}e}{6N} + \log S(N, \hat{\boldsymbol{\alpha}}) \end{aligned} \tag{3.67}$$

where $\boldsymbol{W}_k, \boldsymbol{W}_0$ and $S(N, \hat{\boldsymbol{\alpha}})$ are defined in Eqs. (3.11), (3.16) and (3.3), respectively.

3.4 A Model-Based Clustering Algorithm

Given N data vectors (denoted by $\boldsymbol{Y}$) and $\hat{K}$ clusters, we need an appropriate clustering procedure to determine the optimal partition (or grouping) of these vectors $\boldsymbol{Y}$ into $\hat{K}$ clusters. Here by "optimal", we mean that, for a given $\hat{K}$, the optimal partition $\hat{\boldsymbol{\alpha}}$ achieves the maximum likelihood. In other words, the negative log-likelihood $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}})$ is minimal. The procedure we propose to optimally repartition $\hat{K}$ existing clusters into $\hat{K}+1$ new clusters is as follows: we obtain $\hat{K}$ candidate partitions, each by a binary splitting of one of the $\hat{K}$ existing clusters, followed by a regrouping of the data into $\hat{K}+1$ clusters; The optimal clustering among the $\hat{K}$ candidates is the one which achieves the maximum likelihood.

Therefore, the clustering analysis consists of two loops: (a) In the outer loop $\hat{K}$ starts from 1 to $K_{\max}$, a pre-selected upper bound; (b) In the inner loop the optimal partition $\hat{\boldsymbol{\alpha}}$ of $\boldsymbol{Y}$ into $\hat{K}$ clusters is chosen and $L(\boldsymbol{Y}, \hat{K})$ is calculated according to $\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}}$. Finally, the number of clusters K^* is selected if it yields the minimal $L(\boldsymbol{Y}, \hat{K})$. In the following, the procedure of the clustering algorithm is presented in Section 3.4.1 and its computational complexity is analyzed in Section 3.4.2.

3.4.1 Procedure

The flow chart of the clustering algorithm is shown in Fig. 3.1. The algorithm is off-line since it requires all the data available at the same time.

1. Start from $\hat{K} = 1$, i.e., the whole data set $\boldsymbol{Y}$ is viewed as one cluster.

2. For each of the $\hat{K}$ existing clusters, compute the mean vector $\hat{\boldsymbol{\mu}}_k$ as the cluster center and the standard deviation vector $\hat{\boldsymbol{\sigma}}_k$ as the cluster deviation, $k = 1, \ldots, \hat{K}$.

3. In this step we will obtain $\hat{K}$ candidate partitions, each by a binary splitting of one of the $\hat{K}$ existing clusters, followed by a regrouping of the data into $\hat{K}+1$ clusters.

   For $k = 1, \ldots, \hat{K}$, compute

$$\hat{\boldsymbol{\mu}}_{\hat{K}+1} \leftarrow \hat{\boldsymbol{\mu}}_k + \hat{\boldsymbol{\sigma}}_k$$

$$\hat{\mu}_k \leftarrow \hat{\mu}_k - \hat{\sigma}_k. \tag{3.68}$$

(a) Use $\hat{\mu}_1, \hat{\mu}_2, ..., \hat{\mu}_{\hat{K}}, \hat{\mu}_{\hat{K}+1}$ as the initial centers to repartition the data into $(\hat{K}+1)$ new clusters and obtain the association vector, according to the minimum distance principle. In other words, each data vector will be classified into a cluster whose center is the closest. Here the distance measure is the $\mathcal{L}_2$ norm.

(b) Compute all $\hat{K}+1$ new cluster centers, and repeat the repartition process a few times (say N_t times) until the cluster centers converge. Thus, the association vector $\hat{\alpha}_k$ is obtained.

(c) Compute the negative log-likelihood $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\alpha}_k)$.

4. There are $\hat{K}$ different splittings in Step 3 to repartition $\hat{K}$ existing clusters into $\hat{K}+1$ new clusters. The optimal splitting rule here is to choose the best splitting p which yields the minimal $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\alpha}_k)$. i.e.,

$$p = \arg \min_{k \in [1,...,\hat{K}]} L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}}, \hat{\alpha}_k). \tag{3.69}$$

Set $\hat{\alpha} = \hat{\alpha}_p$; i.e., $\hat{\alpha}_p$ is the optimum association vector obtained from the above splittings and repartitions.

5. $\hat{K} = \hat{K}+1$ and $L(\boldsymbol{Y}, \hat{K})$ is calculated according to $\hat{\boldsymbol{\theta}}, \hat{\alpha}$. Go to Step 2 until $\hat{K}$ reaches $K_{\max}$.

6. Choose the optimal number K^* such that $L(\boldsymbol{Y}, K^*)$ is minimal among all $L(\boldsymbol{Y}, \hat{K})$.

3.4.2 Computational Complexity

Given N M-dimensional data vectors, we start by viewing the whole data set as a cluster and then repartition the data to get a new cluster in each step until the number of clusters $\hat{K}$ reaches $K_{\max}$. Below one addition, subtraction, multiplication and division are counted 1 flop (floating point operation), respectively.

As we observed in Section 3.4.1, the dominant cost occurs in Step 3. At the k-th stage $(k = 1, \ldots, \hat{K})$:

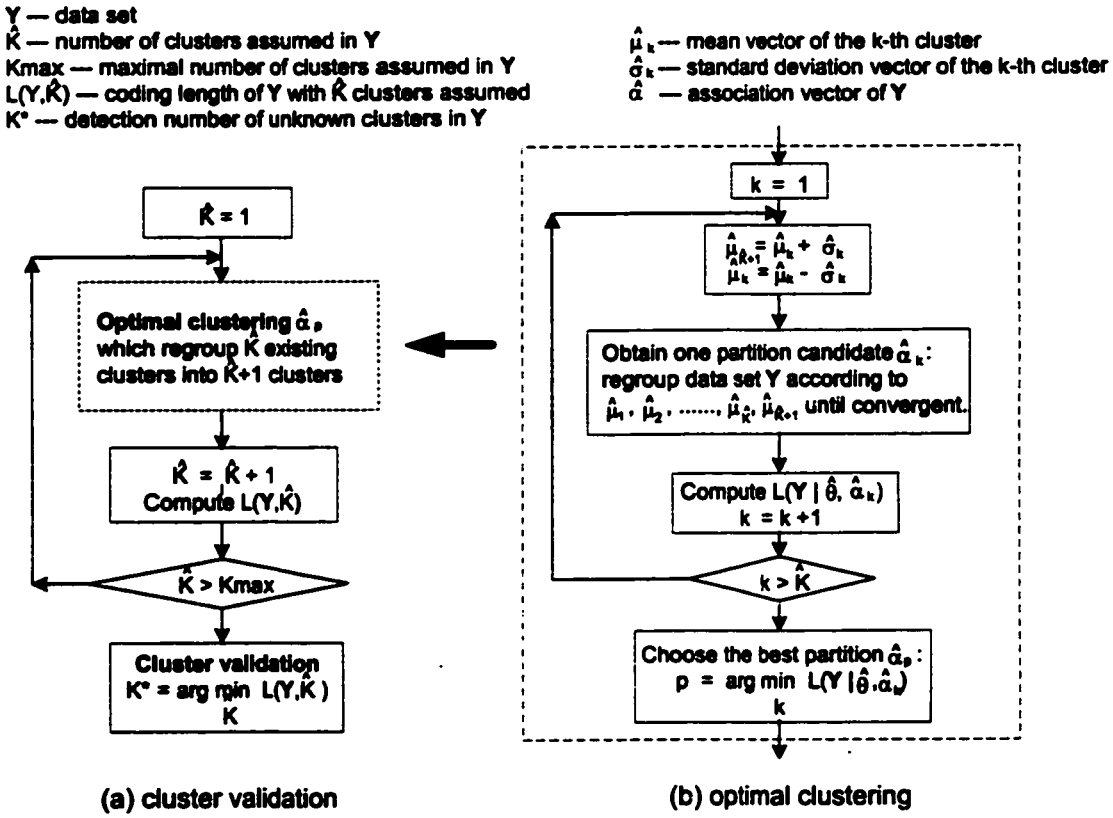


Figure 3.1: The diagram of our model-based clustering algorithm

(a) Repartition N data vectors according to the $\hat{K}+1$ cluster centers.

To compute the distance square [1] between a data vector and a cluster center, it requires M subtractions, M multiplications and M additions. Thus, $3M(\hat{K}+1)$ flops are required for computing the distance squares between one data vector and all $\hat{K}+1$ cluster centers. To choose the minimum distance square, it approximately takes $\log_2(\hat{K}+1)$ comparisons. The cost of these comparisons is negligible compared to $3M(\hat{K}+1)$ flops. Thus, to assign the N data vectors into $\hat{K}+1$ clusters, it requires

$$N\left[3M(\hat{K}+1)+\log_2(\hat{K}+1)\right] \simeq 3MN(\hat{K}+1).$$

(b) Update all $\hat{K}+1$ cluster centers, and repeat the repartition process N_t times until the cluster centers converge.

To compute the k-th cluster center, it requires MN_k additions and M divisions. Thus, to compute all $\hat{K}+1$ cluster centers, it requires

$$\sum_{k=1}^{\hat{K}+1}(MN_k+M) = M(N+\hat{K}+1) \simeq MN,$$

where we assume $N \gg \hat{K}$. Hence, to repeat (a) and (b) by N_t times, it requires

$$N_t\left[MN+3MN(\hat{K}+1)\right] = MNN_t(3\hat{K}+4).$$

(c) Compute the negative log-likelihood $L(\boldsymbol{Y}|\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\alpha}}_k)$.

The computational cost in (c) is negligible compared to those in (a) and (b).

There are $\hat{K}$ candidate partitions in Step 3. Hence, the computational cost for Step 3 is

$$\hat{K}\left[MNN_t(3\hat{K}+4)\right] = MNN_t(3\hat{K}^2+4\hat{K}).$$

Step 3 is repeated from that $\hat{K}=1$ to that $\hat{K}=K_{\max}-1$. Therefore, the total computational cost of the clustering algorithm described in Section 3.4.1 is approximately

$$\begin{aligned} C_{\text{off}} &= \sum_{\hat{K}=1}^{K_{\max}-1} MNN_t(3\hat{K}^2+4\hat{K}) \\ &= MNN_t\left[(K_{\max}-1)^3+3.5(K_{\max}-1)^2+2.5(K_{\max}-1)\right]. \end{aligned} \tag{3.70}$$

[1] To find the minimum distance of a data vector to all cluster centers, it is equivalent to find the minimum distance square. In this way, one square root operation is saved.

It is shown in Eq. (3.70) that the computational complexity is approximately proportional to $K_{\max}^3$. Thus, the computational cost increases dramatically as $K_{\max}$ increases. To alleviate the computational burden of the model-based clustering scheme, a fast algorithm is developed in Section 5.5. In this chapter, we emphasize on the development of our clustering scheme and comparison with existing algorithms.

3.5 Comparison with SNOB

Wallace [62] started his idea on the minimum message length (MML) criterion from classification. Over the past three decades, Wallace and his co-workers [10, 11, 61, 63] have developed and maintained a clustering program called SNOB. Basically, SNOB probabilistically assigns a data vector $\boldsymbol{y}_n$ to a cluster k, say in probability $P(\alpha_n = k)$. Hence, the data is conditionally modeled by

$$f(\boldsymbol{Y}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \sum_{k=1}^{\hat{K}} p_k \times f_k(\boldsymbol{y}_n) \tag{3.71}$$

where $\boldsymbol{\theta} = \{p_1, \cdots, p_{\hat{K}}, \boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_{\hat{K}}, \boldsymbol{\Sigma}_1, \cdots, \boldsymbol{\Sigma}_{\hat{K}}\}$ and

$$p_k = \frac{1}{N} \sum_{n=1}^{N} P(\alpha_n = k).$$

It is also assumed in SNOB that each covariance matrix $\boldsymbol{\Sigma}_k$ is diagonal, and then the message length formula of Eq. (2.18) is directly applied for cluster validation. Detailed descriptions of the SNOB approach were presented in [46] and [7, Chapter 7]. Here the differences between SNOB and our method are summarized below:

- We follow the classification approach using the deterministic assignment which only allows $P(\alpha_n = k)$ to be 0 or 1, instead of the mixture approach using the probabilistic assignment in the SNOB program, so the mixture models are different (compare Eqs. (3.71) and (3.1)).
- Our model parameter vector $\boldsymbol{\theta}$ does not require p_k. This results in a simpler coding length.

- The theoretic analysis of our approach is mathematically tractable but the analysis of SNOB's approach is far more difficult (see next chapter for details).
- The deterministic assignment can be done by a simple procedure such as the *k-means* algorithm [33] but the probabilistic assignment requires a more complicated procedure such as the EM algorithm [66], Therefore, our clustering algorithm is computationally simpler than SNOB.
- For clustering, we start by viewing the whole data set as a cluster and then repartition and regroup the data to get a new cluster in each step. The SNOB program starts with a randomly initial estimate of the clustering structure and then split a cluster or merge two clusters recursively.

3.6 Experimental Results

An empirical comparison of Autoclass, SNOB and two neural network classifiers (Kohonen's network and Adaptive Resonance Theory) was made in [59,60], where the conclusion is that, overall, statistical classifiers, especially SNOB, perform better than the neural network classifiers on both cluster validation and clustering. For our coding length measures described in Section 3.3, we demonstrated that it outperforms some well-known non-parametric criteria in [37, 67]. Here to fairly compare the performance of our clustering algorithm with that of SNOB, we incorporate the same prior specification $h(\boldsymbol{\theta})$ as SNOB, which has been described in Section 3.3. We also examine the performance of the coding length measure of Eq. (3.7) based on the Bayesian Inference Criterion (BIC).

To simulate a mixture of two clusters, let N be the sample size of the two clusters in total and c be the mixing portion. Then the populations of the two clusters are cN and $(1-c)N$ respectively. We define the following measure for the inter-cluster distance:

$$D = \left\| \boldsymbol{\Sigma}_1^{-1/2}\boldsymbol{\mu}_1 - \boldsymbol{\Sigma}_2^{-1/2}\boldsymbol{\mu}_2 \right\|. \tag{3.72}$$

For simplicity, we have chosen the covariance matrices of the two clusters to be identical such that $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \sigma^2 \boldsymbol{I}_M$. Let the inter-cluster distance be D, we define $\boldsymbol{\mu}_1 = [0, 0, \cdots, 0]^T$

and $\mu_2 = [\frac{D}{\sqrt{M}}, \frac{D}{\sqrt{M}}, \cdots, \frac{D}{\sqrt{M}}]^T$. Furthermore, for the high dimension case ($M = 22$), data is generated by adding Gaussian noises in two noiseless pulse patterns, such examples are shown in Fig. 3.2.

In the following, we create various two-cluster mixtures when the data vector dimension $M = 1, 2, 22$, the mixing portion $c = 0.5, 0.2$, the sample size $N = 40, 100, 1000$ and the distance measure $D = 2, 3, 4, 6, 8$. We also create the case of one cluster when $M = 1, 2, 22$ and the sample size $N = 40, 100, 1000$. We employ our clustering algorithm under four covariance structures which are denoted by L_1, L_2, L_3 and L_4 respectively, and the SNOB algorithm to perform cluster validation and clustering. We also employ our clustering algorithm under the BIC of Eq. (3.7) based on Covariance Structure 4. In this way, we can fairly compare the performance of L_4, BIC and SNOB. Ten trials of each mixture are carried out using L_1, L_2, L_3, L_4, BIC and SNOB. In each trial, we assume the number of clusters from 1 to 4 and then choose the number at which the corresponding criterion is minimal. The results of cluster validation for all criteria examined here, as represented by the number of times out of 10 trials that the correct number of clusters is determined, are shown in Tables 3.1 - 3.9. To further compare our clustering algorithm and SNOB, the accuracy of the corresponding clustering results are shown in Tables 3.10 - 3.12 for $N = 100$ when both algorithms have made the correct decision on cluster validation out of 10 trials.

From Tables 3.1 - 3.12, it can be observed that

1. None of the criteria is reliable for small and medium samples ($N = 40, 100$) when $D < 3$.

2. The performance of all criteria is improved as the distance D and the sample size N increase.

3. The performance of L_1, L_2 and L_3 is very similar to that of L_4.

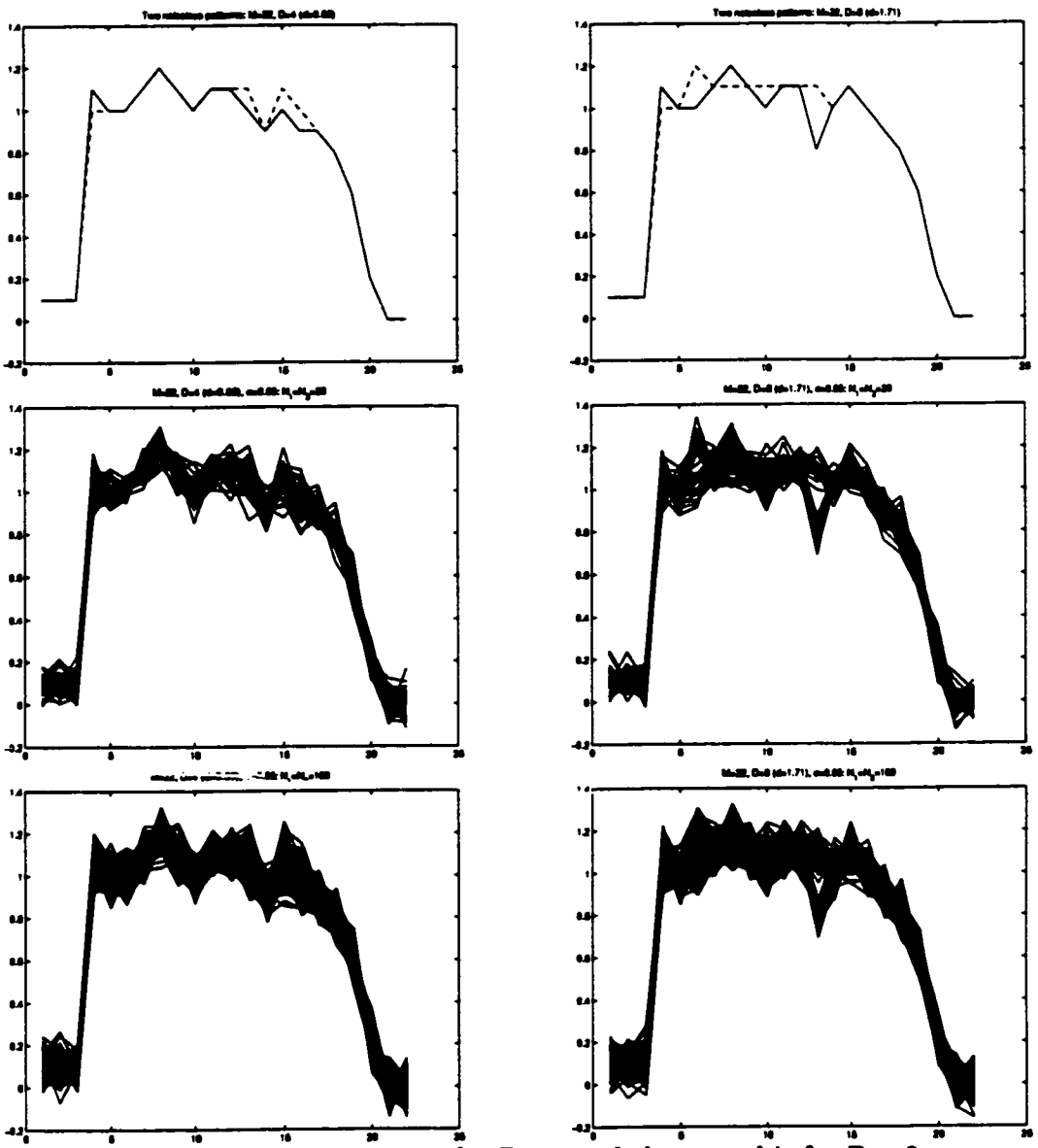
4. BIC is inferior to L_4 and SNOB in cluster validation in general but performs very well for the medium and large samples ($N = 100, 1000$).

5. The performance of L_4 is superior to that of SNOB in cluster validation for the small and medium samples ($N = 40, 100$).

6. In the cases where both algorithms perform perfectly in cluster validation, SNOB yields slightly more accurate results in clustering than L_4.

Observation 1 is easy to justify since if $D < 3$, the overlap between two clusters is very extensive, which makes it difficult for any of the criteria to work properly. Observation 2 is intuitively clear: the larger is the inter-cluster distance D, the less overlap is there between the two clusters, and the higher accuracy is achieved in parameter estimation. Also, since the data here is generated by using $\Sigma_1 = \Sigma_2 = \sigma^2 I_M$, the performance of L_1, L_2, L_3 and L_4 is likely to be more or less the same, as confirmed by Observation 3. Observation 4 is natural because the BIC of Eq. (3.7) is an asymptotic criterion. In fact, BIC has been used in many applications due to its simplicity and the fact that no prior knowledge is required.

From Observations 5 and 6, our clustering algorithm shows much higher reliability in cluster validation than SNOB, while sacrificing marginally on the accuracy in clustering. Recall that an extra set of parameters p_k is included in SNOB whereas such parameters have not been taken into consideration in the development of our algorithm. These parameters prescribe the probabilities of the nth data vector being generated by the kth cluster and must be estimated. This difference between SNOB and our algorithm has profound implications in their performance. For cluster validation, the probability that the nth data vector associates with the kth cluster is irrelevant, i.e., regardless of the values of these probabilities, the number of clusters remains the same. Therefore, for cluster validation, these extra parameters are nuisance parameters and their inclusion will lower the accuracy of the determination of the number of clusters, and hence the new algorithm shows better performance than SNOB in cluster validation. On the other hand, the probabilities of associating the data vectors to the clusters are highly relevant parameters in the process of clustering. Therefore, their inclusion provides more information gained from the data and renders SNOB the more accurate algorithm in clustering.



The first column is for $D = 4$ and the second is for $D = 8$

Figure 3.2: Simulated data for M=22, where x-axis is the index of data sample points and y-axis is the amplitude of simulated data.

$\frac{\|\mu_1-\mu_2\|}{\sigma}$	Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
D=2	N=40	1/10	1/10	1/10	1/10	1/10	0/10
	N=100	0/10	0/10	0/10	0/10	0/10	0/10
	N=1000	0/10	0/10	0/10	0/10	0/10	1/10
D=3	N=40	3/10	3/10	3/10	3/10	3/10	0/10
	N=100	4/10	4/10	4/10	4/10	4/10	2/10
	N=1000	3/10	3/10	3/10	3/10	3/10	10/10
D=4	N=40	8/10	9/10	8/10	9/10	6/10	1/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=6	N=40	9/10	9/10	9/10	10/10	6/10	9/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=8	N=40	9/10	10/10	9/10	10/10	6/10	10/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.1: Cluster validation results for two true clusters: M=1, c=0.5

$\frac{\|\mu_1-\mu_2\|}{\sigma}$	Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
D=2	N=40	1/10	1/10	1/10	1/10	0/10	0/10
	N=100	0/10	0/10	0/10	0/10	0/10	0/10
	N=1000	0/10	0/10	0/10	0/10	0/10	2/10
D=3	N=40	2/10	3/10	2/10	3/10	2/10	0/10
	N=100	3/10	3/10	3/10	3/10	3/10	0/10
	N=1000	0/10	0/10	0/10	0/10	0/10	10/10
D=4	N=40	7/10	8/10	7/10	8/10	5/10	0/10
	N=100	10/10	10/10	10/10	10/10	10/10	7/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=6	N=40	9/10	10/10	9/10	10/10	7/10	9/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=8	N=40	9/10	10/10	9/10	10/10	7/10	10/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.2: Cluster validation results for two true clusters: M=1, c=0.2

Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
N=40	8/10	10/10	8/10	10/10	8/10	10/10
N=100	10/10	10/10	10/10	10/10	10/10	10/10
N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.3: Cluster validation results for one true cluster: M=1

$\frac{\|\mu_1-\mu_2\|}{\sigma}$	Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
D=2	N=40	1/10	1/10	1/10	1/10	1/10	0/10
	N=100	1/10	1/10	1/10	1/10	0/10	0/10
	N=1000	0/10	0/10	0/10	0/10	0/10	10/10
D=3	N=40	6/10	7/10	6/10	7/10	7/10	0/10
	N=100	10/10	10/10	10/10	10/10	10/10	5/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=4	N=40	8/10	9/10	8/10	10/10	9/10	4/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=6	N=40	10/10	10/10	9/10	10/10	9/10	10/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=8	N=40	10/10	10/10	10/10	10/10	9/10	10/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.4: Cluster validation results for two true clusters: M=2, c=0.5

$\frac{\|\mu_1-\mu_2\|}{\sigma}$	Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
D=2	N=40	1/10	1/10	1/10	1/10	0/10	0/10
	N=100	0/10	0/10	0/10	0/10	0/10	0/10
	N=1000	0/10	0/10	0/10	0/10	0/10	6/10
D=3	N=40	4/10	5/10	4/10	5/10	4/10	0/10
	N=100	6/10	6/10	6/10	6/10	5/10	3/10
	N=1000	9/10	9/10	9/10	9/10	9/10	10/10
D=4	N=40	7/10	8/10	7/10	9/10	8/10	2/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=6	N=40	9/10	10/10	9/10	10/10	9/10	10/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=8	N=40	10/10	10/10	9/10	10/10	9/10	10/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.5: Cluster validation results for two true clusters: M=2, c=0.2

Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
N=40	9/10	10/10	9/10	10/10	10/10	10/10
N=100	10/10	10/10	10/10	10/10	10/10	10/10
N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.6: Cluster validation results for one true cluster: M=2

$\frac{\|\mu_1-\mu_2\|}{\sigma}$	Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
D=2	N=40	1/10	0/10	0/10	0/10	0/10	0/10
	N=100	0/10	0/10	0/10	0/10	0/10	0/10
	N=1000	0/10	0/10	0/10	0/10	0/10	0/10
D=3	N=40	3/10	3/10	1/10	0/10	0/10	0/10
	N=100	9/10	9/10	4/10	0/10	0/10	0/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=4	N=40	8/10	8/10	7/10	6/10	0/10	0/10
	N=100	10/10	10/10	10/10	10/10	5/10	0/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=6	N=40	9/10	10/10	8/10	10/10	10/10	0/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=8	N=40	10/10	10/10	9/10	10/10	10/10	9/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.7: Cluster validation results for two true clusters: M=22, c=0.5

$\frac{\|\mu_1-\mu_2\|}{\sigma}$	Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
D=2	N=40	0/10	0/10	0/10	0/10	0/10	0/10
	N=100	0/10	0/10	0/10	0/10	0/10	0/10
	N=1000	0/10	0/10	0/10	0/10	0/10	0/10
D=3	N=40	0/10	0/10	0/10	0/10	0/10	0/10
	N=100	2/10	2/10	0/10	0/10	0/10	0/10
	N=1000	10/10	10/10	6/10	2/10	0/10	10/10
D=4	N=40	3/10	4/10	3/10	2/10	0/10	0/10
	N=100	10/10	10/10	9/10	8/10	0/10	0/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=6	N=40	9/10	9/10	9/10	7/10	6/10	1/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10
D=8	N=40	10/10	9/10	9/10	8/10	10/10	4/10
	N=100	10/10	10/10	10/10	10/10	10/10	10/10
	N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.8: Cluster validation results for two true clusters: M=22, c=0.2

Sample Size	L_1	L_2	L_3	L_4	BIC	SNOB
N=40	10/10	10/10	10/10	10/10	10/10	10/10
N=100	10/10	10/10	10/10	10/10	10/10	10/10
N=1000	10/10	10/10	10/10	10/10	10/10	10/10

Table 3.9: Cluster validation results for one true cluster: M=22

D	SNOB				L_1			
	$N_1 = 50, N_2 = 50$		$N_1 = 20, N_2 = 80$		$N_1 = 50, N_2 = 50$		$N_1 = 20, N_2 = 80$	
	validation	clustering	validation	clustering	validation	clustering	validation	clustering
D=2	0/10		0/10		0/10		0/10	
D=3	2/10		0/10		4/10		3/10	
D=4	10/10	97.3%	7/10		10/10	97.0%	10/10	95.7%
D=6	10/10	99.9%	10/10	99.9%	10/10	99.9%	10/10	99.6%
D=8	10/10	100.0%	10/10	100.0%	10/10	100.0%	10/10	99.9%

Table 3.10: Comparison of performance of SNOB and our algorithm, M=1

D	SNOB				L_1			
	$N_1 = 50, N_2 = 50$		$N_1 = 20, N_2 = 80$		$N_1 = 50, N_2 = 50$		$N_1 = 20, N_2 = 80$	
	validation	clustering	validation	clustering	validation	clustering	validation	clustering
D=2	0/10		0/10		1/10		0/10	
D=3	5/10		3/10		10/10	92.1%	6/10	
D=4	10/10	97.4%	10/10	98.2%	10/10	97.3%	10/10	96.6%
D=6	10/10	99.9%	10/10	99.9%	10/10	99.9%	10/10	99.9%
D=8	10/10	100.0%	10/10	100.0%	10/10	100.0%	10/10	100.0%

Table 3.11: Comparison of performance of SNOB and our algorithm, M=2

D	SNOB				L_1			
	$N_1 = 50, N_2 = 50$		$N_1 = 20, N_2 = 80$		$N_1 = 50, N_2 = 50$		$N_1 = 20, N_2 = 80$	
	validation	clustering	validation	clustering	validation	clustering	validation	clustering
D=2	0/10		0/10		0/10		0/10	
D=3	0/10		0/10		0/10		0/10	
D=4	0/10		0/10		10/10	96.8%	8/10	
D=6	10/10	99.9%	10/10	99.9%	10/10	99.9%	10/10	99.7%
D=8	10/10	100.0%	10/10	100.0%	10/10	100.0%	10/10	100.0%

Table 3.12: Comparison of performance of SNOB and our algorithm, M=22

3.7 Summary

In this chapter, model-based clustering has been considered. Based on minimum encoding inference, an appropriate coding length measure is proposed for cluster validation, and the coding lengths under four different Gaussian mixture models are fully derived. The corresponding clustering algorithm is developed.

Judging from the performance comparison, our coding length measure outperforms the BIC in cluster validation since it is not based on the large sample assumption. More importantly, our clustering algorithm shows much higher reliability in cluster validation than SNOB, while sacrificing marginally on the accuracy in clustering. Indeed, our clustering algorithm is well designed to effectively process high dimensional data with satisfactory performance on small and medium samples. Thus, the new algorithm is an attractive approach for the clustering problem in intra-pulse analysis, in terms of the first two issues pointed out in Section 1.1.

Chapter 4

Detection Performance Analysis

4.1 Introduction

The error analysis of model-based clustering concerns about the estimation accuracy of the number of components in a Gaussian mixture. This is a problem that has not been completely solved in statistics, as stated in the end of Section 1.2. In this chapter, we conduct a binary detection performance analysis of our clustering algorithm under Covariance Structure 1 described in Section 3.3.1, by estimating the two types of errors: miss and false alarm. We also examine the impact of the penalty weight on the error probability under the framework of the penalized likelihood method described in Section 2.4.

Under Covariance Structure 1, each of all $\hat{K}$ clusters is assumed to be a sample from a multivariate normal distribution $N(\mu, \sigma^2 \boldsymbol{I}_M)$. The log-likelihood function Eq. (3.21) is rewritten here

$$L(\hat{K}) = \frac{MN}{2} \log(tr\boldsymbol{W}) \tag{4.1}$$

and the penalty function, obtained by combining Eqs. (3.25), (3.17), (3.14) and (3.28), is given by

$$\Pi(\hat{K}) = +\frac{\hat{K}M+1}{2} \log \frac{tr\boldsymbol{W}_0}{tr\boldsymbol{W}} + \frac{M}{2} \sum_{k=1}^{\hat{K}} \log N_k + \frac{\hat{K}M}{2} \log \frac{e}{3} + \log S(N, \hat{\alpha}) \tag{4.2}$$

where N_k is the number of members in the kth cluster; $\boldsymbol{W}, \boldsymbol{W}_0$ and $S(N, \hat{\alpha})$ are defined in Eqs. (3.19), (3.16) and (3.3), respectively.

Therefore, the total description length is that

$$DL(\hat{K}) = L(\hat{K}) + \Pi(\hat{K}). \tag{4.3}$$

The detection will select the number of clusters to be K^* if

$$K^* = arg \min_{1 \le \hat{K} \le N} DL(\hat{K}). \tag{4.4}$$

We investigate a binary detection performance here. Given a data set with N observation vectors, let $\boldsymbol{W}$ be the sample covariance matrix when taking the whole data set as a cluster. By some classification, the data set is partitioned into two clusters, one with a sample size N_1 and a sample covariance matrix $\boldsymbol{W}_1$ and the other with a sample size N_2 and a sample covariance matrix $\boldsymbol{W}_2$. In addition, we let c be the mixing portion, i.e., $N_1 = cN$ and $N_2 = (1-c)N$, where $0 < c < 1$. Define

$$\Delta L = L(2) - L(1) = \frac{MN}{2} \log \frac{tr(\boldsymbol{W}_1 + \boldsymbol{W}_2)}{tr(\boldsymbol{W})} \tag{4.5}$$

and

$$\begin{aligned} \Delta\Pi = \Pi(2) - \Pi(1) \quad &= \quad -\frac{2M+1}{2} \log \frac{tr(\boldsymbol{W}_1 + \boldsymbol{W}_2)}{tr(\boldsymbol{W})} \\ &\quad + \frac{M}{2} \log \frac{c(1-c)Ne}{3} + \log \frac{N!}{(cN)!((1-c)N)!}. \end{aligned} \tag{4.6}$$

Then

$$DL(2) - DL(1) = \Delta L + \Delta\Pi. \tag{4.7}$$

Therefore, to evaluate the error performance, we have to know the distribution of the trace ratio $\frac{tr(\boldsymbol{W}_1+\boldsymbol{W}_2)}{tr(\boldsymbol{W})}$ or its variation.

4.2 Probability of A Miss

A miss occurs when two clusters are embedded in a data set but the binary detection says that only one cluster exists. A simple illustration is shown in Fig. 4.1. Let H_1 and H_2 denote respectively the hypotheses of one cluster and two clusters, the miss probability given H_2 is

$$P_m = P\{DL(2) - DL(1) > 0 | H_2\} = P\{\Delta L + \Delta\Pi > 0 | H_2\}. \tag{4.8}$$

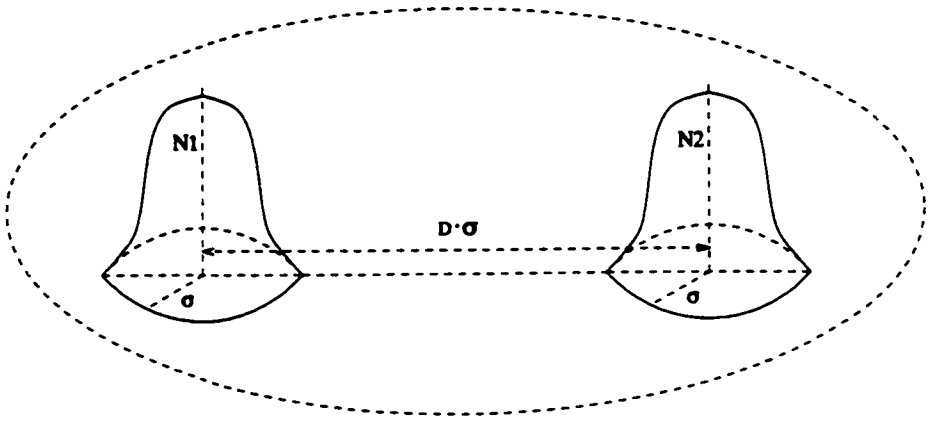


Figure 4.1: Two Gaussian clusters

Let

$$R_m = (N-2)\frac{tr[\boldsymbol{W} - (\boldsymbol{W}_1 + \boldsymbol{W}_2)]}{tr(\boldsymbol{W}_1 + \boldsymbol{W}_2)}. \tag{4.9}$$

Different partition criteria may result in different values of R_m. Here we assume that our clustering algorithm can separate two Gaussian clusters perfectly. This assumption is reasonable when two clusters are well separated. We expect more deviation from the assumption when two clusters are closer to each other in distance. Under the perfect separation assumption, it is proven in Appendix B that R_m is distributed as a noncentral F distribution $F_{M,M(N-2)}(\delta)$ with the noncentral parameter

$$\delta = \frac{N_1 N_2}{N}\left\|\frac{\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2}{\sigma}\right\|^2 = c(1-c)ND^2, \tag{4.10}$$

and D is the normalized inter-cluster distance defined by

$$D = \left\|\frac{\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2}{\sigma}\right\|. \tag{4.11}$$

Substituting Eqs. (4.5) and (4.6) into Eq. (4.8), we have

$$P_m = P\{ \quad \frac{MN-2M-1}{2}\log\frac{tr(\boldsymbol{W}_1+\boldsymbol{W}_2)}{tr(\boldsymbol{W})} + \frac{M}{2}\log\frac{c(1-c)Ne}{3} + \log\frac{N!}{(cN)!((1-c)N)!} > 0 \quad |H_2\}.$$

Rewriting the above equation in terms of R_m, we then have

$$P_m = P\{R_m < F_{P_m} \,|H_2\} \tag{4.12}$$

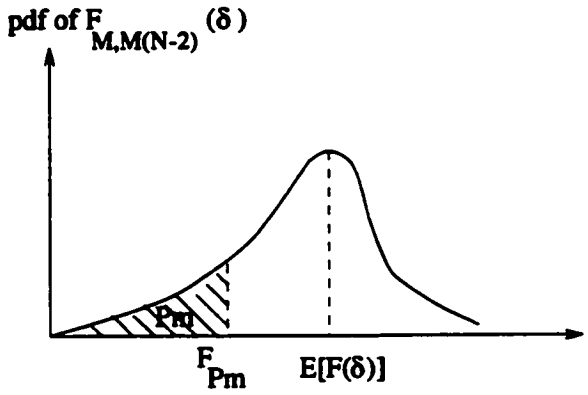


Figure 4.2: The illustration of P_m

where the threshold

$$F_{P_m} = (N-2)\left\{\left[\frac{e}{3}c(1-c)N\right]^{\frac{M}{MN-2M-1}} \cdot \left[\frac{N!}{(cN)!((1-c)N)!}\right]^{\frac{2}{MN-2M-1}} - 1\right\} \tag{4.13}$$

and the value of P_m is represented by the shadow area in Fig. 4.2.

From [34, Chapter 30], the mean and variance of a non-central F distribution with v_1, v_2 degrees of freedom and a non-central parameter δ are given respectively by

$$E[F_{v_1,v_2}(\delta)] = \frac{v_2(v_1+\delta)}{v_1(v_2-2)} \qquad (v_2 > 2), \tag{4.14}$$

$$Var[F_{v_1,v_2}(\delta)] = 2\left(\frac{v_2}{v_1}\right)^2 \frac{(v_1+\delta)^2 + (v_1+2\delta)(v_2-2)}{(v_2-2)^2(v_2-4)} \quad (v_2 > 4). \tag{4.15}$$

Here $v_1 = M$, $v_2 = M(N-2)$ and δ is specified by Eq. (4.10). substituting these values into Eqs. (4.14) and (4.15) and assuming that $N \gg 1$, we have

$$E[R_m] \simeq 1 + \frac{\delta}{M} = 1 + \frac{c(1-c)D^2}{M}N, \tag{4.16}$$

$$Var[R_m] \simeq \frac{2c^2(1-c)^2D^4 + 4c(1-c)D^2M}{M^3}N. \tag{4.17}$$

Property 1: Given the dimension M, the mixing portion c, we define D_0 such that

$$D_0 = \sqrt{\frac{M}{c(1-c)}\left\{\left[c^{-c}(1-c)^{-(1-c)}\right]^{\frac{2}{M}} - 1\right\}}. \tag{4.18}$$

If $D > D_0$, then P_m tends to 0 as the number of observations N increases. If $D < D_0$, then P_m tends to 1 as the number of observations N increases.

Proof: By some mathematical manipulations, we have

$$\lim_{N\to\infty} \frac{F_{P_m}}{E[R_m]} = \frac{M}{c(1-c)D^2}\left\{\left[c^{-c}(1-c)^{-(1-c)}\right]^{\frac{2}{M}} - 1\right\}. \tag{4.19}$$

To obtain Eq. (4.19), we have used the knowledge that $\frac{N!}{(cN)!((1-c)N)!} \simeq [c^{-c}(1-c)^{-(1-c)}]^N$ for large N, as shown in Appendix C.

First, let us check the case when $D > D_0$. Given M and c, we know from Eq. (4.19) that $F_{P_m} < E[R_m]$ asymptotically if and only if $D > D_0$. By using Chebyshev's inequality [53, Page 69], we have

$$P\{|R_m - E[R_m]| \geq (E[R_m] - F_{P_m})\} \leq \frac{Var[R_m]}{(E[R_m] - F_{P_m})^2}. \tag{4.20}$$

From Eqs. (4.13),(4.16) and (4.17), we know that $Var[R_m]$ is proportional to N but $(E[R_m] - F_{P_m})^2$ is proportional to N^2. Thus,

$$\lim_{N\to\infty} P\{|R_m - E[R_m]| \geq (E[R_m] - F_{P_m})\} = 0. \tag{4.21}$$

The left hand side of Eq. (4.20) is the area under the p.d.f. of R_m over the intervals $(-\infty, F_{P_m}]$ and $[2E[R_m] - F_{P_m}, +\infty)$. Hence,

$$\lim_{N\to\infty} P_m = \lim_{N\to\infty} P\{R_m < F_{P_m}\} = 0. \tag{4.22}$$

Second, we check the case when $D < D_0$. In this case, $F_{P_m} > E[R_m]$ for a sufficiently large N. Similarly by using Chebyshev's inequality for $N \to \infty$, we have

$$\lim_{N\to\infty} P\{|R_m - E[R_m]| \geq (F_{P_m} - E[R_m])\} = 0. \tag{4.23}$$

Thus,

$$\lim_{N\to\infty} P\{2E[R_m] - F_{P_m} \leq R_m \text{ or } R_m \geq F_{P_m}\} = 0.$$

This further implies

$$\lim_{N\to\infty} P\{2E[R_m] - F_{P_m} < R_m < F_{P_m}\} = 1. \tag{4.24}$$

Consequently,

$$\lim_{N\to\infty} P_m = \lim_{N\to\infty} P\{R_m < F_{P_m}\} = 1. \tag{4.25}$$

M	1		2		22	
c	0.5	0.2	0.5	0.2	0.5	0.2
D_0	3.47	3.28	2.83	2.85	2.40	2.53

Table 4.1: The critical distance D_0

Therefore, Property 1 holds. **Q.E.D.**

If it happens that $D = D_0$, then F_{P_m} is asymptotically equal to $E[R_m]$ and P_m is a positive number between 0 and 1. Some values of D_0 given M and c are listed in Table 4.2. Indeed, D_0 is a critical distance. While $D > D_0$, the two clusters are well separated, the probability density function (p.d.f.) of the mixture is bimodal and our clustering algorithm can successfully separate these two clusters. While $D \leq D_0$, the p.d.f. of the mixture becomes unimodal, the overlap between the two clusters is very extensive and the similarity makes it difficult for our algorithm to work properly, as for other existing algorithms.

Property 2: P_m is a monotonically decreasing function of the normalized inter-cluster distance $D = \|\frac{\mu_1 - \mu_2}{\sigma}\|$.

Proof: $P\{F_{M,M(N-2)}(\delta) < F_{P_m}\}$ is a decreasing function of δ, see [34, Page 193] for details. Property 2 holds since δ is proportional to the square of D as shown in Eq. (4.10). **Q.E.D.**

Figs. 4.3 - 4.8 show the theoretical P_m and the testing P_m vs. the number of observations N for the mixtures of two clusters considered in Section 3.6. The dotted lines in these figures are the theoretical P_m curves and the solid lines are the testing P_m curves based on 1000 trials for each N. From these figures, Properties 1 and 2 are clearly observed. We also notice that there is a discrepancy between the theoretical P_m and the testing one. The reason is that the actual partition may deviate more or less from the ideal partition which separates the two clusters perfectly. This discrepancy increases as the two clusters become closer in distance.

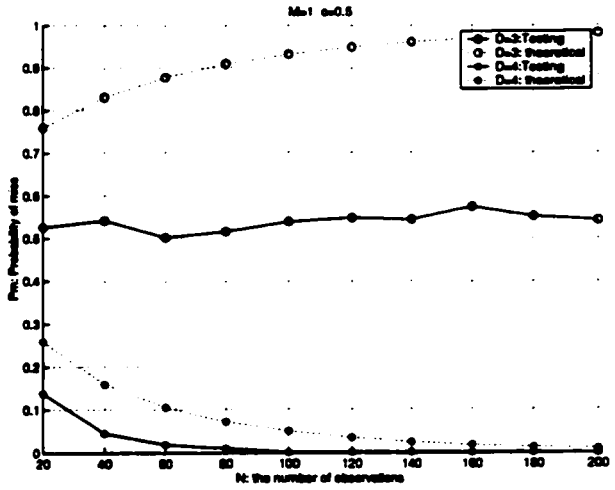


Figure 4.3: Miss probability curves for two true clusters: M=1 c=0.5

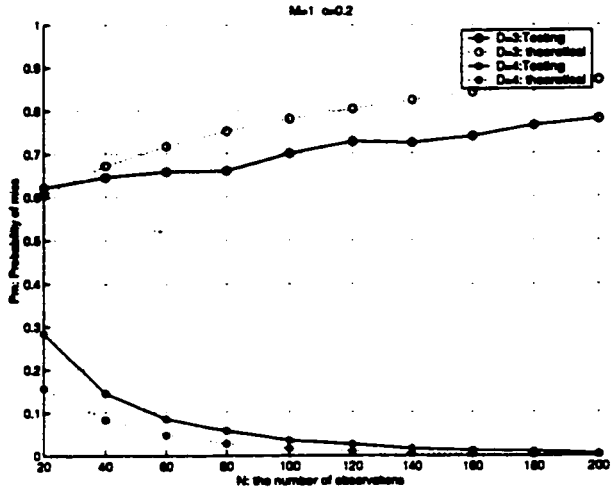


Figure 4.4: Miss probability curves for two true clusters: M=1 c=0.2

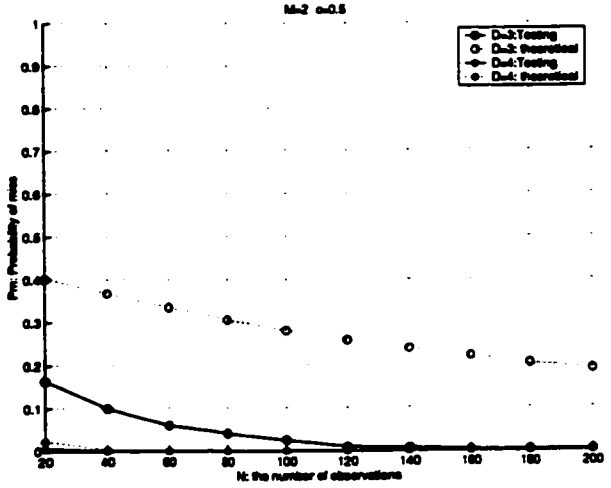


Figure 4.5: Miss probability curves for two true clusters: M=2 c=0.5

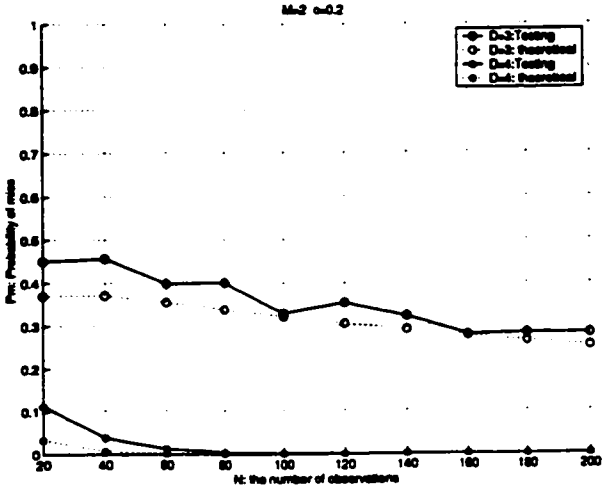


Figure 4.6: Miss probability curves for two true clusters: M=2 c=0.2

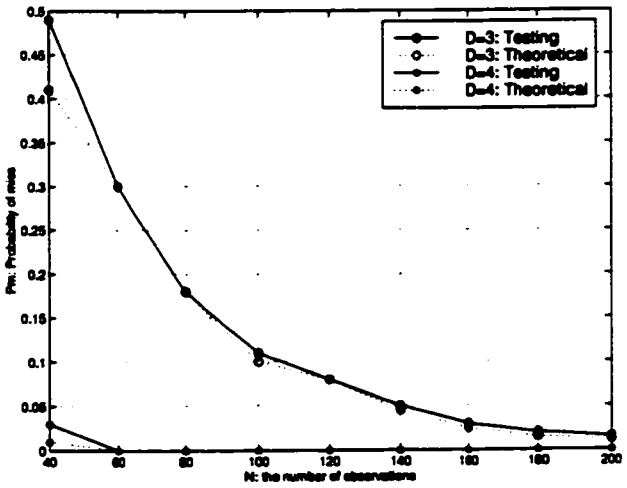


Figure 4.7: Miss probability curves for two true clusters: M=22 c=0.5

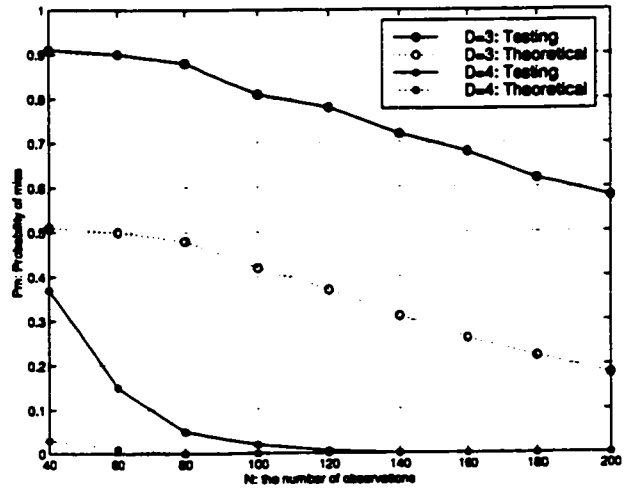


Figure 4.8: Miss probability curves for two true clusters: M=22 c=0.2

4.3 Probability of A False Alarm

A false alarm occurs when one cluster is embedded in a data set but the binary detection says that two clusters exist. A simple illustration is shown in Fig. 4.9. So the false alarm probability, given H_1 (the hypothesis of one cluster), is

$$P_f = P\{DL(2) - DL(1) < 0|H_1\} = P\{\Delta L + \Delta\Pi < 0|H_1\}. \tag{4.26}$$

In the miss case, P_m is exactly analyzed according to the Gaussian mixture model. However, in the false alarm case, the model is a mixture of truncated normal distributions, whose exact analysis is still incomplete in statistics.

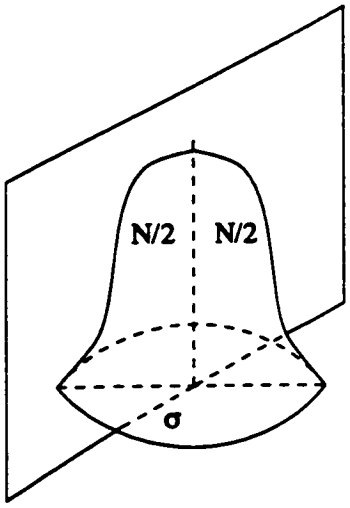


Figure 4.9: One Gaussian cluster

The key to the analysis is the understanding of how our clustering algorithm will partition a $N(\mu, \sigma^2 I_M)$ sample data into two clusters. Given a large sample, the sample mean vector $\hat{\mu}$ and the sample standard deviation vector $\hat{\sigma}$ of the data are close to the true ones μ and σ respectively. By our partition described in Section 3.4, two new clusters are centered in $\hat{\mu} - \hat{\sigma}$ and $\hat{\mu} + \hat{\sigma}$ respectively. Therefore, the partition is near symmetric to the true mean vector and produces two clusters of about equal size, i.e., the mixing portion $c \simeq 0.5$. Define

$$R_f = \log \frac{tr[W - (W_1 + W_2)]}{tr(W_1 + W_2)}. \tag{4.27}$$

Assuming that the sample is reasonably large and following Hartigan's work [32], Hawkins [23, Page 339] suggested that asymptotically, R_f is approximately normal $N(\mu_{R_f}, \sigma^2_{R_f})$ with the mean

$$\mu_{R_f} = \log(\frac{2}{\pi M - 2}) \tag{4.28}$$

and the variance

$$\sigma^2_{R_f} = \frac{2\pi^2 M(\pi M - 2M - 1)}{N(\pi M - 2)^2}. \tag{4.29}$$

Substituting Eqs. (4.5) and (4.6) into Eq. (4.26), we have

$$P_f = P\{ \quad \frac{MN - 2M - 1}{2} \log \frac{tr(\boldsymbol{W}_1 + \boldsymbol{W}_2)}{tr(\boldsymbol{W})} + \frac{M}{2} \log \frac{eN}{12} + \log \frac{N!}{(0.5N)!(0.5N)!} < 0 \quad |H_1\}.$$

Rewriting the above equation in terms of R_f, we then have

$$P_f = P\{R_f > F_{P_f} \ |H_1\} \tag{4.30}$$

where the threshold

$$F_{P_f} = \log \left\{ \left[\frac{eN}{12}\right]^{\frac{M}{MN-2M-1}} \cdot \left[\frac{N!}{(0.5N)!(0.5N)!}\right]^{\frac{2}{MN-2M-1}} - 1 \right\} \tag{4.31}$$

and the value of P_f is represented by the shadow area in Fig.(4.10).

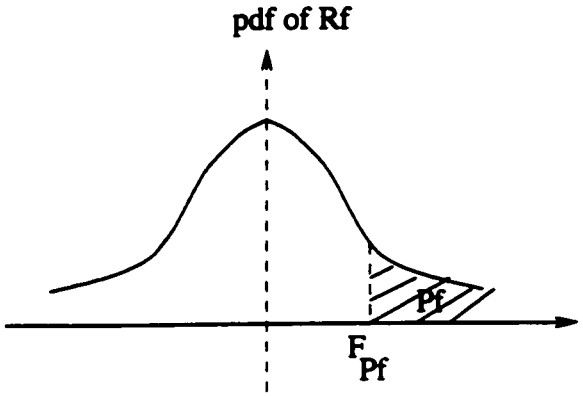


Figure 4.10: The illustration of P_f

Property 1: P_f tends to 0 as the number of observations N increases.

Proof: It is easy to verify that

$$\lim_{N\to\infty}(F_{P_f}-\mu_{R_f}) = \log(2^{\frac{2}{M}}-1) - \log(\frac{2}{\pi M-2}) > 0, \tag{4.32}$$

$$\lim_{N\to\infty}\sigma_{R_f} = 0. \tag{4.33}$$

Some of these limiting values are listed in Table 4.3. By using Chebyshev's inequality [53,

M	1	2	22
$\lim_{N\to\infty}(F_{P_f}-\mu_{R_f})$	0.54	0.77	0.78

Table 4.2: Limiting values of $(F_{P_f}-\mu_{R_f})$

Page 69], we have

$$P\left\{|R_f - E[R_f]| \geq (F_{P_f} - E[R_f])\right\} \leq \frac{Var[R_f]}{(F_{P_f}-E[R_f])^2}. \tag{4.34}$$

From Eqs. (4.32) and (4.33), we know that

$$\lim_{N\to\infty} P\left\{|R_f - E[R_f]| \geq (F_{P_f} - E(R_f])\right\} = 0. \tag{4.35}$$

The left hand side of Eq. (4.34) is the area under the p.d.f. of R_f over the intervals $(-\infty, 2E[R_f] - F_{P_f}]$ and $[F_{P_f}, +\infty)$. Hence,

$$\lim_{N\to\infty} P_f = \lim_{N\to\infty} P\{R_f > F_{P_f}\} = 0. \tag{4.36}$$

Therefore, P_f tends to 0 asymptotically. **Q.E.D.**

Figs. 4.11 - 4.13 show the testing P_f vs. the number of observations N for the one truth cluster cases considered in Section 3.6. The solid lines are the testing P_f curves based on 1000 trials for each N. The corresponding theoretical values of P_f are very very small, less than 10^{-3}. It is observed from Figs. 4.11 - 4.13 that the testing P_f is negligible for medium and large samples, although a small probability of false alarm does exist for small samples.

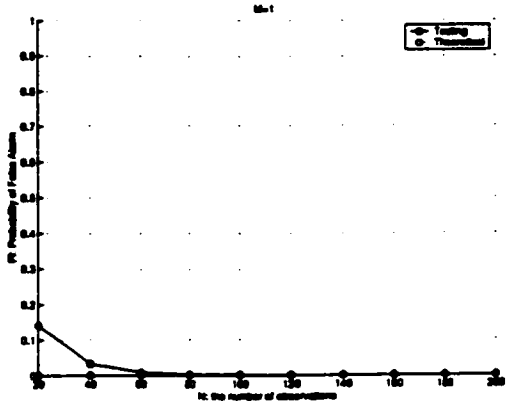


Figure 4.11: False alarm probability curves for one true cluster: M=1

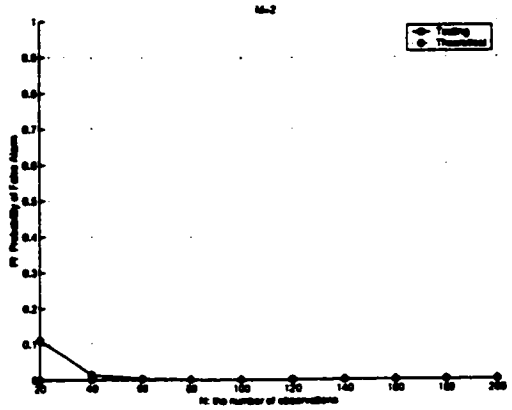


Figure 4.12: False alarm probability curves for one true cluster: M=2

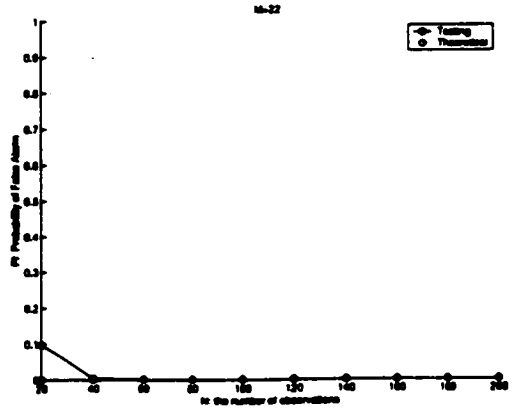


Figure 4.13: False alarm probability curves for one true cluster: M=22

4.4 Optimal Range of Penalty Weight

If the penalty function in Eq. (4.3) is multiplied by a penalty weight λ as described in Section 2.4, then under Covariance Structure 1, we have

$$\lim_{N\to\infty} \frac{F_{P_m}}{E[F_{M,M(N-2)}(\delta)]} = \frac{M}{c(1-c)D^2}\left\{\left[c^{-c}(1-c)^{-(1-c)}\right]^{\frac{2\lambda}{M}} - 1\right\}, \tag{4.37}$$

$$\lim_{N\to\infty} (F_{P_f} - \mu_{R_f}) = \log(2^{\frac{2\lambda}{M}} - 1) - \log(\frac{2}{\pi M - 2}). \tag{4.38}$$

By the same reasoning as that used for Property 1 in Section 4.2, we require that

$$\lim_{N\to\infty} \frac{F_{P_m}}{E[F_{M,M(N-2)}(\delta)]} < 1$$

in order to make $P_m \to 0$ asymptotically. This requirement lets us determine an upper bound for the penalty weight

$$\lambda_{max} = -\frac{0.5M\log(1+c(1-c)D^2/M)}{c\log c + (1-c)\log(1-c)}. \tag{4.39}$$

Similarly, by the same reasoning used to prove Property 1 in Section 4.3, we require that

$$\lim_{N\to\infty} (F_{P_f} - \mu_{R_f}) > 0$$

in order to make $P_f \to 0$ asymptotically. This requirement lets us determine a lower bound for the penalty weight

$$\lambda_{min} = -\frac{M}{2}\log\frac{\pi M}{\pi M - 2}. \tag{4.40}$$

According to Eqs. (4.39) and (4.40), we can tabulate in Table 4.3 the (asymptotically) optimal penalty weight ranges for the cases considered in Section 3.6.

M	1				2				22			
D	3		4		3		4		3		4	
c	0.5	0.2	0.5	0.2	0.5	0.2	0.5	0.2	0.5	0.2	0.5	0.2
λ_{max}	0.85	0.89	1.16	1.26	1.08	1.08	1.58	1.64	1.54	1.39	2.65	2.42
λ_{min}	0.54				0.77				0.78			

Table 4.3: Optimal ranges of the penalty weight

Apparently, penalty weights within the optimal ranges listed above may have different impacts on our clustering algorithm performance for processing small or medium samples,

$\frac{\|\mu_1-\mu_2\|}{\sigma}$	Sample Size	L_1	L_2	L_3	L_4	BIC
$D=2$	$N=40$	0/10	0/10	0/10	0/10	0/10
	$N=100$	0/10	0/10	0/10	0/10	0/10
	$N=1000$	0/10	0/10	0/10	0/10	0/10
$D=3$	$N=40$	3/10	3/10	0/10	0/10	0/10
	$N=100$	8/10	8/10	1/10	0/10	0/10
	$N=1000$	10/10	10/10	10/10	10/10	9/10
$D=4$	$N=40$	9/10	10/10	8/10	6/10	0/10
	$N=100$	10/10	10/10	10/10	10/10	3/10
	$N=1000$	10/10	10/10	10/10	10/10	10/10
$D=6$	$N=40$	10/10	10/10	9/10	10/10	8/10
	$N=100$	10/10	10/10	10/10	10/10	10/10
	$N=1000$	10/10	10/10	10/10	10/10	10/10
$D=8$	$N=40$	10/10	10/10	9/10	10/10	10/10
	$N=100$	10/10	10/10	10/10	10/10	10/10
	$N=1000$	10/10	10/10	10/10	10/10	10/10

Table 4.4: Cluster validation results for two true clusters: M=22 c=0.5 $\lambda = 1.1$

Sample Size	L_1	L_2	L_3	L_4	BIC
$N=40$	10/10	10/10	10/10	10/10	10/10
$N=100$	10/10	10/10	10/10	10/10	10/10
$N=1000$	10/10	10/10	10/10	10/10	10/10

Table 4.5: Cluster validation results for one true cluster: M=22 $\lambda = 1.1$

although their impacts are the same in the asymptotic sense. In practice, the number of clusters is searched from 1 to a pre-selected upper bound $K_{\max}$ (> 2), one true cluster might be partitioned into a few groups if the penalty weight is not large enough. In this case, we could choose the penalty weight slightly larger. For example, Tables 4.4 and 4.5 show the clustering results by using $\lambda = 1.1$ for the case $M = 22$. It is observed that the performance of our clustering algorithm on small samples are slightly improved when the penalty weight increases from 1 (see Tables 3.7 and 3.9) to 1.1 (see Tables 4.4 and 4.5).

4.5 Summary

In this chapter, we have conducted a binary detection performance analysis of our clustering algorithm under Covariance Structure 1. We assumed that a partition can separate two Gaussian clusters perfectly in order to analyze the miss probability, and that the partition of one Gaussian cluster is nearly symmetric to the cluster center in order to analyze the false alarm probability. Extensive tests show that these two assumptions are satisfied fairly well by using our clustering algorithm developed in Section 3.4. Among four factors considered here (the dimension of the data space M, the sample size N, the mixing portion c and

the inter-cluster distance D), D is the most important factor. There is a critical distance D_0 defined in Eq. (4.18), when $D > D_0$, our clustering algorithm can successfully separate two clusters. On the other hand, when $D \leq D_0$, the overlap between the two clusters is very extensive and it is difficult for our algorithm to work properly, as for other existing algorithms.

Furthermore, we have examined the impact of the penalty weight under the framework of the penalized likelihood method as described in Section 2.4. It is found that there is a range of the penalty weight within which the best performance of our clustering algorithm can be achieved. Therefore, with some supervision, we can adjust the penalty weight to further improve the performance of our clustering algorithm.

Chapter 5

Application to Intrapulse Analysis

5.1 Introduction

We consider the situation where a radar intercept receiver collects incoming pulse samples from a number of unknown emitters. Our objectives are to (1) determine the number of emitters present (cluster validation); (2) classify the incoming pulses according to the emitters from which they originate (clustering). The concept of intrapulse analysis has been introduced in Section 1.1. Briefly, the determination in intrapulse analysis is only based on intrinsic pulse shapes, without any inter-pulse information such as pulse repetition intervals, directions of arrival, carrier frequencies, or Doppler shifts.

In this chapter, we first describe the pre-processing techniques including data compression for received pulses, and then formulate the problem of emitter number detection and pulse-emitter association into a multivariate clustering problem. After applying the new clustering algorithm developed in Chapter 3 to the clustering problem, we develop two on-line clustering algorithms, one is based on known thresholds while the other is based on a model-based detection scheme. Performance on intrapulse data by using our clustering algorithms and SNOB are reported, and the results demonstrate that our new clustering algorithms are very effective for intrapulse analysis, especially the on-line model-based algorithm.

5.2 Signal Model And Pre-Processing of Received Pulses

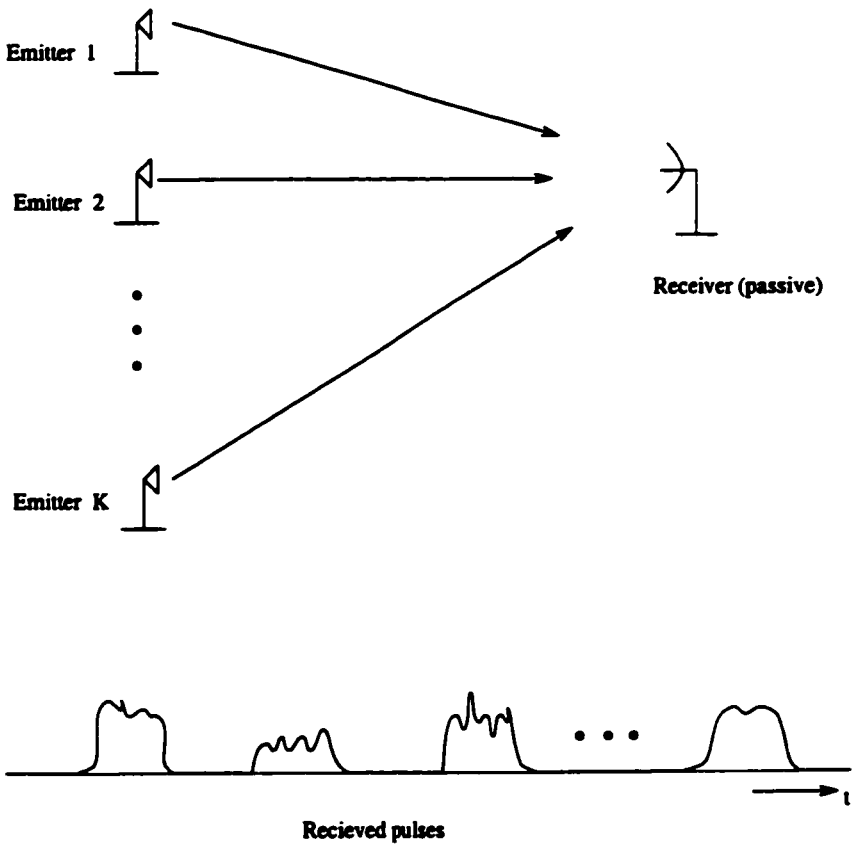
Let us first examine the signal representation of the received pulses. The physical scenario is illustrated in Fig. 5.1 in which there are, in total, K distinct emitters. The radar intercept receiver receives altogether N non-overlapping pulses from the emitters. We designate the nth received pulse by $x_n(t;\alpha_n)$, $n = 1,\ldots,N$. Here α_n is an association parameter which assumes an integer value, $\alpha_n \in \{1,\ldots,K\}$, such that if $\alpha_n = k$, then the nth pulse is determined to be from the kth emitter. We can therefore express the nth pulse as

$$x_n(t;\alpha_n) = \eta_n a_{\alpha_n}(t-\tau_n)e^{j[\psi_n+\omega_n(t-\tau_n)+\phi_{\alpha_n}(t-\tau_n)]} + \nu_n(t-\tau_n) \tag{5.1}$$

where

- η_n denotes the absolute amplitude of the received pulse;
- ψ_n denotes the added phase of the received pulse after transmission;
- τ_n denotes the time delay of the received pulse with respect to the reference;
- ω_n denotes the residual carrier frequency of the nth pulse;
- $a_{\alpha_n}(t)$ is the original envelope for the nth pulse such that
$$a_{\alpha_n}(t) \in \{a_1(t), a_2(t), ..., a_K(t)\};$$
- $\phi_{\alpha_n}(t)$ is the original phase for the nth pulse such that
$$\phi_{\alpha_n}(t) \in \{\phi_1(t), \phi_2(t), ..., \phi_K(t)\};$$
- $\nu_n(t)$ is the Gaussian noise accompanying the nth pulse.

The received pulse in Eq. (5.1) contains several nuisance parameters: η_n, ψ_n, τ_n, and ω_n. These parameters are of no use in intra-pulse analysis and should be removed. This is carried out by the pre-processing techniques which are introduced in the following paragraphs. These pre-processing techniques are intuitive in nature and are carried out so that after pre-processing, the pulses received from the same emitter maintain the resemblance to each other, while those from different emitters maintain their distinctive features.



K distinct emitters 1 receiver receiving N pulses

* Inter-band and inter-pulse information not usable

Figure 5.1: Radar pulses received for intrapulse analysis

Noise Suppression

The received pulses are passed through a band pass filter suppressing the out-of-band noise. We now define the amplitude and phase profiles of the received signal respectively as:

$$s_n^a(t;\alpha_n) = \eta_n a_{\alpha_n}(t-\tau_n), \tag{5.2}$$

$$s_n^p(t;\alpha_n) = \psi_n + \omega_n(t-\tau_n) + \phi_{\alpha_n}(t-\tau_n). \tag{5.3}$$

In the rest of the pre-process, it is assumed that the SNR is reasonably large so that the noise contribution has negligible effects.

5.2.1 Amplitude Normalization

Let $S_n^a(\omega;\alpha_n)$ and $A_{\alpha_n}(\omega)$ be the Fourier transforms of $s_n^a(t;\alpha_n)$ and $a_{\alpha_n}(t)$, respectively. We remove the parameter η_n by a simple procedure of normalization resulting in $\tilde{S}_n^a(\omega;\alpha_n)$ such that

$$\begin{aligned}\tilde{S}_n^a(\omega;\alpha_n) &= \frac{S_n^a(\omega;\alpha_n)}{S_n^a(0;\alpha_n)} = \frac{\eta_n A_{\alpha_n}(\omega)e^{-j\omega\tau_n}}{\eta_n A_{\alpha_n}(0)} \\ &= A_{\alpha_n}^{-1}(0)A_{\alpha_n}(\omega)e^{-j\omega\tau_n}.\end{aligned} \tag{5.4}$$

Therefore, $\tilde{s}_n^a(t;\alpha_n)$, the inverse Fourier transform of $\tilde{S}_n^a(\omega;\alpha_n)$, can be viewed as the normalized amplitude profile.

5.2.2 Time Alignment Based on Thresholding

After amplitude normalization, the removal of time shift is considered. The time shift can be removed by locating the first point, $\tilde{s}_n^a(t_0;\alpha_n)$, of $\tilde{s}_n^a(t;\alpha_n)$, whose magnitude is larger than a pre-set threshold Δ, i.e.,

$$t_0 = \min\{t : \tilde{s}_n^a(t;\alpha_n) > \Delta\}. \tag{5.5}$$

We then set $\tau_n = t_0$, and define

$$\begin{aligned}\hat{s}_n^a(t;\alpha_n) &= \tilde{s}_n^a(t+\tau_n;\alpha_n) \\ &= A_{\alpha_n}^{-1}(0)a_{\alpha_n}(t)\end{aligned} \tag{5.6}$$

and

$$\begin{aligned} \tilde{s}_n^p(t;\alpha_n) &= s_n^p(t+\tau_n;\alpha_n) \\ &= \psi_n + \omega_n t + \phi_{\alpha_n}(t). \end{aligned} \tag{5.7}$$

We use the same threshold Δ for all amplitude profiles to align the pulses on the time axis.

5.2.3 Phase Adjustment Based on Polynomial Fitting

After time alignment, the linear slope in the phase profile should be removed. The linear function can be estimated by polynomial curve fitting. Let

$$g(t) = \omega_n t + \psi_n, \tag{5.8}$$

by minimizing the error between $g(t)$ and $\tilde{s}_n^p(t;\alpha_n)$ in a least square sense, the coefficients ω_n and ψ_n can be determined. Then, the original phase $\phi_{\alpha_n}(t)$ can be recovered in a new coordinate system with the time axis being $\psi_n + \omega_n t$ (Fig. 5.2) when we define

$$\begin{aligned} \hat{s}_n^p(t';\alpha_n) &= |\tilde{s}_n^p(t;\alpha_n) - \psi_n - \omega_n t| \cos\gamma \\ &= \phi_{\alpha_n}(t'), \end{aligned} \tag{5.9}$$

where $\gamma = \arctan\omega_n$.

We denote by $y_n(t;\alpha_n)$ the resulting nth pulse with noise after pre-processing to remove the nuisance parameters, i.e.,

$$\begin{aligned} y_n(t;\alpha_n) &= \hat{s}_n^a(t;\alpha_n)e^{j\hat{s}_n^p(t;\alpha_n)} + \tilde{\nu}_n(t) \\ &= s_{\alpha_n}(t) + \tilde{\nu}_n(t), \end{aligned} \tag{5.10}$$

where $\tilde{\nu}_n(t)$ is the noise after pre-processing; $s_{\alpha_n}(t) = A_{\alpha_n}^{-1}(0)a_{\alpha_n}(t)e^{j\phi_{\alpha_n}(t)}$ is the ideally pre-processed pulse waveform. Notice that $s_{\alpha_n}(t)$ is only dependent on α_n, the index for the emitter with which this pulse signal associates.

In practice, the above pre-processing procedures of the received pulses are carried out in discrete time. Thus, Eq. (5.10) can be written as

$$y_n(mT;\alpha_n) = s_{\alpha_n}(mT) + \tilde{\nu}_n(mT), \qquad m = 1, 2, ..., M' \tag{5.11}$$

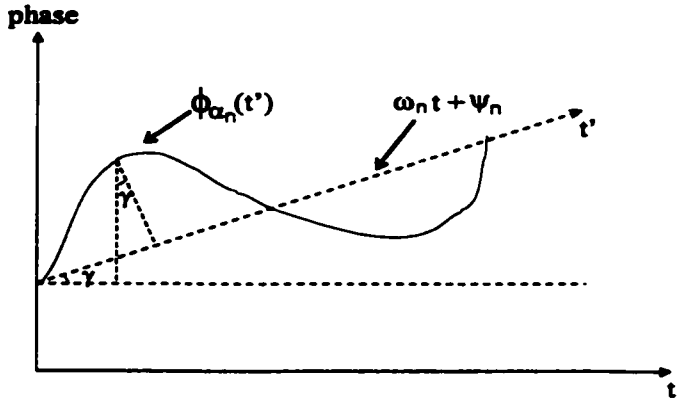


Figure 5.2: Polynomial fitting for phase adjustment

where T is the sampling interval of the pulses and M' is the number of samples in a pulse; or in a vector form

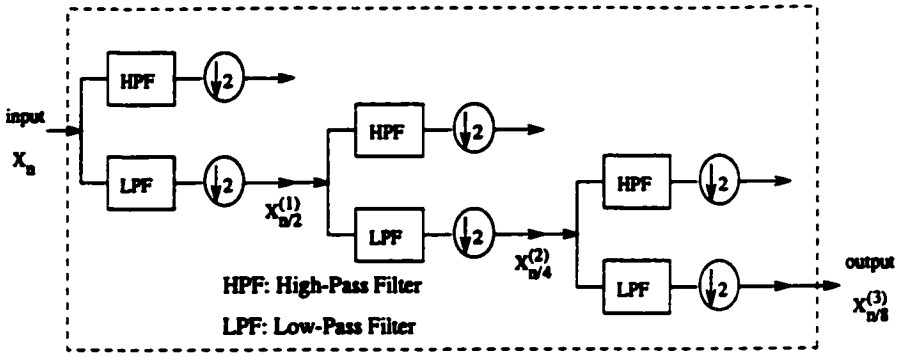
$$y_n(\alpha_n) = s_{\alpha_n} + \tilde{\nu}_n \tag{5.12}$$

with

$$y_n(\alpha_n) = [y_n(1;\alpha_n), ..., y_n(M';\alpha_n)]^T,\ s_{\alpha_n} = [s_{\alpha_n}(1), ..., s_{\alpha_n}(M')]^T,\ \tilde{\nu}_n = [\tilde{\nu}_n(1), ..., \tilde{\nu}_n(M')]^T.$$

5.2.4 Data Compression Using Wavelet Decomposition

The number of samples M' in each pre-processed pulse is typically over 100. Thus, the number of samples to be processed for cluster validation and clustering is very large. In order to lighten the computational and processing load, we have to compress the data. We note that the classification of pulses by intra-pulse analysis is based on pulse shapes (amplitude and phase). Now, the low frequency components of a pulse reflect its basic shape. A suitable technique for compressing the data while retaining the basic pulse shape is by means of wavelet decomposition [21, 38] by which the low frequency coefficients can be extracted retaining the pulse shape information. Wavelet decomposition is carried out by using a chosen filter bank in which each one of the filters is followed by down-sampling by 2. In our case, only the low frequency components are needed and undergo further



Symlet4 filter band coefficients

LPF:	0.0322	-0.0126	-0.0992	0.2979	0.8037	0.4976	-0.0296	-0.0758
HPF:	-0.0758	0.0296	0.4976	-0.8037	0.2979	0.0992	-0.0126	-0.0322

Figure 5.3: Data compression using wavelet decomposition

decomposition as shown in Fig. 5.3. Due to the process of down-sampling by 2, the size of each of the outputs of the low pass filter (LPF) and of the high pass filter (HPF) after one stage of decomposition is only half that of the input. Therefore, for a three-staged wavelet decomposition, the output sample size is only one-eighth of the original input sample size. The number of stages in a wavelet decomposition is a trade-off between the amount of data compression and the details of the pulse shape information retained. Here, we employ a three-staged filter bank with *symlet4* filters [21]. The coefficients of these LP and HP filters are indicated in Fig. 5.3.

We denote the data vector of Eq. (5.12) after compression by $y_n(\alpha_n)$ which is comprised of compressed signal and noise. Each of these complex data vectors is of dimension M which is only a fraction of the dimension of the original data vector.

5.3 Clustering Algorithms for Intrapulse Analysis

After pre-processing, all received pulses are aligned well. Then intrapulse analysis is considered as a multivariate clustering problem: given a compressed data set $\boldsymbol{Y}$ consisting of N observed data vectors $\boldsymbol{y}_1(\alpha_1), ..., \boldsymbol{y}_N(\alpha_N)$, each of which dimension is M, our objectives are

1. cluster validation, to determine the number of emitters present K;
2. clustering, to determine the association parameter α_n so that for $\alpha_n = k$, $\boldsymbol{y}_n(\alpha_n)$ is determined to be from the k-th emitter.

The pre-process is done in the complex domain because amplitudes and phases are considered separately. For clustering, we form each data vector in the real domain by putting its real part first and then its imaginary part, i.e.,

$$\left[real(\boldsymbol{y}_n^T), imag(\boldsymbol{y}_n^T)\right]^T$$

From now on, each data vector $\boldsymbol{y}_n$ is assumed to be in the real domain.

Generally, the noise accompanying a radar pulse vector is Gaussian. Hence, a set of pulse vectors emitted by the kth emitter is the sample of a multivariate normal distribution. Is Gaussianity still maintained after pre-processing? In Appendix D, Monte Carlo simulations show that compressed pre-processed pulses can be still assumed as Gaussian in relatively high confidence. Hence, a set of pulse vectors from $\hat{K}$ distinct emitters can be modeled by a mixture of $\hat{K}$ multivariate normal distributions [43]. Therefore, the model-based clustering algorithm developed in Chapter 3 can be directly deployed.

As introduced in Section 1.1, a clustering algorithm for intrapulse analysis should be capable to process high dimensional data and to produce satisfactory results for small or medium sample cases. In Chapter 3, we developed a suitable model-based clustering algorithm and demonstrated by extensive simulations that it outperforms other existing algorithms such as SNOB. Indeed, the model-based clustering algorithm we developed is well designed in an off-line mode to effectively process high dimensional with satisfactory

performance on small or medium samples. In some cases, on-line clustering would be more desirable because we may wish to classify received pulses as they arrive dynamically. This motivates us to develop on-line clustering algorithms. One approach is to set up some thresholds, and then assign an incoming pulse to an existing cluster or form a new cluster according to the thresholds. One realization of this approach is described in Section 5.4. However, this simple approach may not provide satisfactory results when the statistics of received data change in time. To overcome this drawback, we develop in Section 5.5 a novel on-line clustering algorithm in which no explicit thresholds are required. Performance on intrapulse data is reported in Section 5.6 by using the model-based clustering algorithm, SNOB, and the two on-line algorithms.

5.4 An On-line Clustering Algorithm Using Thresholds

We first fix two thresholds t_1 and t_2, the possible candidates being the maximal intra-cluster dispersion and the minimal inter-cluster distance, respectively. Let d be the minimal distance between a pulse and each cluster center. Then, an incoming pulse is assigned to an existing cluster when $d \leq t_1$, or assigned to a new cluster when $d \geq t_2$, or held to some stores for subsequent classification. This algorithm is on-line.

5.4.1 Procedure

The diagram of this algorithm is shown in Fig. 5.4.

1. Initialization:

   i) Choose two thresholds

   t_1 — possibly maximal intra-cluster dispersion,

   t_2 — possibly minimal inter-cluster distance,

   with $0 < t_1 < t_2$. These two thresholds can be determined experimentally using existing data (with known ground truths).

   ii) Take data vector y_1 and assign it to C_1;

$m \leftarrow 1$; (m counts the number of clusters)

$n \leftarrow 1$; (n counts the number of data vectors)

2. The main process:

While there is another data vector do

begin

$n \leftarrow n + 1$; Take vector y_n from the input;

Compute $d = \min_{k \in [1,\cdots,m]} \|y_n - mean\ of\ C_k\|$;

Find k for which d is minimum;

if $d \leq t_1$,

assign y_n to C_k;

else if $d \geq t_2$

$m \leftarrow m + 1$;

assign y_n to C_m;

else

store y_n;

if the storage size $\geq T_s$, then goto step 3;

end;

end;

end;

3. The storage process:

If the store is empty,

goto step 2;

else

$n \leftarrow 0$;

execute step 2 but now the vector y_n is taken from the store;

if the store has changed,

goto step 3;

else

save the current y_n to a post-identification store;

end;

end;

4. The post-identification process:

   For each data vector y_n in the post-identification store,

   assign it to the cluster which is nearest.

5.4.2 Computational Complexity

Suppose that after the on-line process N M-dimensional data vectors are assigned to $\hat{K}$ clusters, resulting in that the kth cluster has N_k data vectors ($k = 1, \cdots, \hat{K}$). Obviously, $N_1 + N_2 + \cdots + N_{\hat{K}} = N$. The store size T_s is usually set to be a small number so that the computational cost of the storage process is negligible compared to that of the main process. However, The computational cost of the main process may vary when the same set of pulses arrives in a different sequence. This makes the exact complexity analysis difficult. In this subsection, An upper complexity bound and an average complexity are analyzed; Then an example is illustrated in the end.

Upper Bound

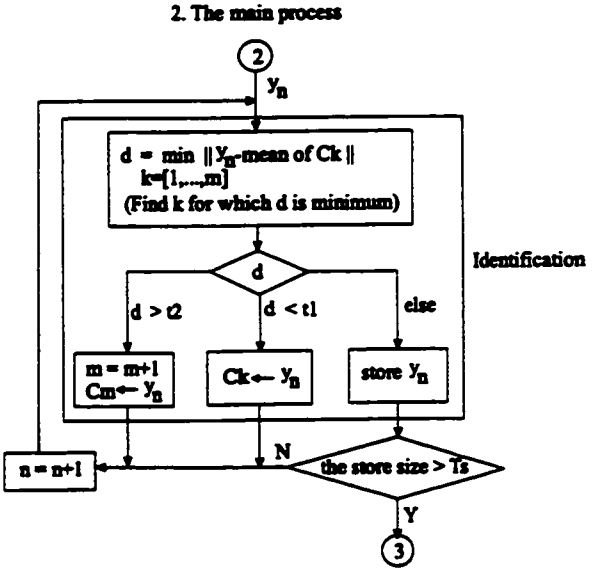
In the main process, there are two dominant operations:

1. For each incoming data vector, its distances to all existing k clusters are computed and then the minimum distance is chosen.

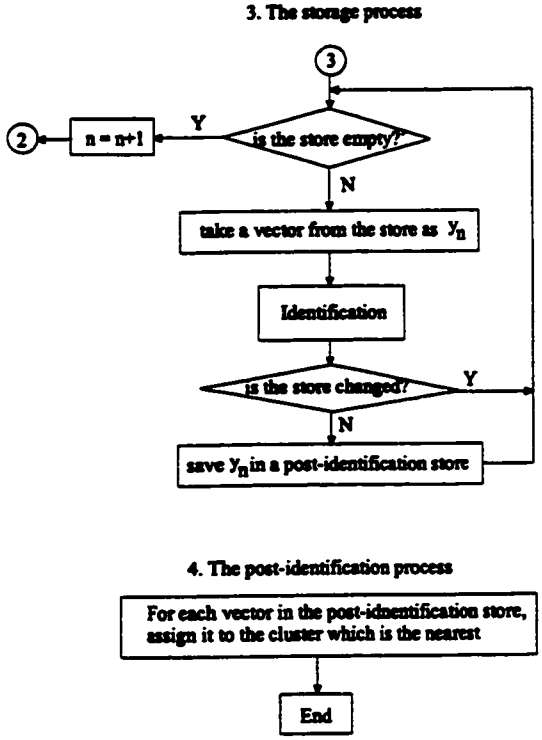
   As discussed in Step (a) of Section 3.4.2, this operation approximately requires $3Mk$ flops. There are at most $\hat{K}$ clusters so that the computational cost of this operation to process all N data vectors is upper-bounded by

   $$N(3M\hat{K}) = 3MN\hat{K}.$$

2. After an incoming data vector is assigned to an existing cluster, the cluster center has to be updated.



(a) The main process



(b) The storage and post-identification processes

Figure 5.4: The diagram of our on-line clustering algorithm using thresholds

As discussed in Step (b) of Section 3.4.2, if the nth member of a cluster arrives, it requires $M(n+1)$ flops to update its center. Note that there is no cluster center update as the first member of a cluster arrives. Thus, the computational cost of this operation to process all N data vectors is

$$\begin{aligned}\sum_{k=1}^{\hat{K}}\sum_{n=2}^{N_k} M(n+1) &= 0.5M\sum_{k=1}^{\hat{K}} N_k^2 - 0.5MN \\ &\leq 0.5MN^2 - 0.5MN\end{aligned}$$

where we have used the fact that

$$N_1^2 + N_2^2 + \cdots + N_{\hat{K}}^2 \leq (N_1 + N_2 + \cdots + N_{\hat{K}})^2 = N^2.$$

Therefore, the upper complexity bound of the on-line clustering algorithm described in Section 5.4.1 is

$$B_{\text{on1}} = MN(3\hat{K} - 0.5) + 0.5MN^2. \tag{5.13}$$

Average Complexity

Let us examine the case where each cluster has the same number of members (i.e., $N_1 = N_2 = \cdots = N_{\hat{K}} = \frac{N}{\hat{K}}$). Since the computational cost of Step 1 in the main process is sensitive to the sequence in which the pulses arrive , we assume that the members of the first cluster arrive first, and then those of the second cluster arrive next, and so on. The computational cost for this case is roughly an average complexity of the on-line clustering algorithm. Similarly as in the upper-bound analysis, the computational costs of Step 1 and Step 2 in the main process here are given respectively by

$$\begin{aligned}3M\frac{N}{\hat{K}}\sum_{k=1}^{\hat{K}} k &= 1.5MN(\hat{K}+1) \\ \sum_{k=1}^{\hat{K}}\sum_{n=2}^{N/\hat{K}} M(n+1) &= \frac{0.5MN^2}{\hat{K}} - 0.5MN\end{aligned}$$

Therefore, the average complexity of the on-line clustering algorithm described in Section 5.4.1 is approximately

$$C_{\text{on1}} = MN(1.5\hat{K}+1) + \frac{0.5MN^2}{\hat{K}}. \tag{5.14}$$

Example: Let us consider the example shown in Section 5.6.2, there are 100 44-dimensional preprocessed pulse vectors ($N = 100, M = 44$) and 5 emitters ($\hat{K} = 5$). Substituting the values of N, M and $\hat{K}$ in Eqs. (5.13) and (5.14), we have the upper bound and the average complexity as follows:

$$B_{\text{onl}} = 0.28\text{Mflops},$$
$$C_{\text{onl}} = 0.08\text{Mflops}.$$

Now consider using the off-line model-based algorithm developed in Section 3.4. Given that $N_t = 5$ (N_t is the number of iterations for cluster center convergence) and that $K_{\max} = 8$ ($K_{\max}$ is the maximal number of clusters to be searched), then according to Eq. (3.70) we have

$$C_{\text{off}} = 11.70\text{Mflops}$$

which is the computational cost by using the model-based algorithm for the above example. Obviously the on-line algorithm is much faster than the off-line model-based algorithm. Unfortunately, the performance of the on-line algorithm is usually inferior to that of the model-based algorithm as will be shown in Section 5.6.2.

5.5 A On-line Model-Based Clustering Algorithm

A major advantage of the following new algorithm is that no explicit thresholds are required. The algorithm dynamically incorporates cluster splitting, merging and regrouping operations by using the model-based detection modified from the clustering algorithm in Section 3.4.

On-line Process

As incoming data vectors are being classified into clusters, the sizes of clusters will continue to grow. A size increment counter is set for each cluster, and a cluster is checked if its size increment counter reaches a preset threshold, T_c. In the very beginning, the first T_c pulses are assigned to the first cluster. Checking a cluster or two clusters involves a binary

detection function defined as follows:

$$[K^*, C_{x1}, C_{x2}] = \texttt{bdetector}\ (\texttt{data}).$$

- If data comes from a single cluster, the bdetector function returns
  either $K^* = 1$ indicating **no splitting** the cluster;
  or $K^* = 2$ indicating **splitting** the cluster into C_{x1} and C_{x2}.

- If data comes from two clusters, the bdetector function returns
  either $K^* = 1$ indicating **merging** the two clusters into one;
  or $K^* = 2$ indicating **regrouping** into the clusters C_{x1} and C_{x2}.

This binary detection function is easily formed by setting $K_{\max} = 2$ in the clustering algorithm describe in Section 3.4.

As mentioned previously, a cluster is checked when its size increment counter reaches a preset threshold T_c. If the returned value $K^* = 1$, then the size increment counter is reset to 0; If $K^* = 2$, then the cluster is split into two new ones C_{x1} and C_{x2}. Since these two new clusters need to be merged or regrouped with other existing clusters, the closest existing cluster is found for each new cluster and both of them are sent to bdetector to check if they should be merged or regrouped. Therefore, as pulses arrive, the sizes of clusters increase, and the algorithm conducts cluster splitting, merging and regrouping operations appropriately by using the above scheme.

Post-processing

In fact, the same set of pulses arriving in a different sequence may result in a different clustering structure since each pulse is classified into a cluster according to the previously arrived pulse information during the on-line process. In other words, the pure on-line process is coarse, though it is very fast. Therefore, a post-processing scheme is introduced below to improve the classification accuracy. After the on-line process, we regroup all received pulses according to the existing cluster centers and then compute the new centers. The regrouping process is repeated a few times (say N_t times) until the centers converge. Then each cluster is checked if it should be split, merged or regrouped with another cluster. This

final splitting, merging and regrouping process is repeated a few times (say N_r times) until the clustering structure converges. After the post-processing, the performance of the on-line clustering is almost the same as the off-line algorithm in Section 3.4 but it is much faster.

5.5.1 Procedure

The flow diagram of this algorithm is shown in Fig. 5.5.

1. Initialization:

   i) Define a binary detection function: $[\mathtt{K}^*, \mathtt{C}_{\mathtt{x1}}, \mathtt{C}_{\mathtt{x2}}] = \mathtt{bdetector}\ (\mathtt{data})$.

   ii) Take data vector $\boldsymbol{y}_1$ and assign it to C_1;

   $m \leftarrow 1$; (m counts the number of clusters)

   $n \leftarrow 1$; (n counts the number of pulse samples)

   $isc\_1 \leftarrow 0$; (increment of the size of C_1)

2. The main process:

   While there is another sample do

   begin

   $n \leftarrow n + 1$; Take data vector $\boldsymbol{y}_n$ from the input;

   Compute $d = \min_{k \in [1,\cdots,m]} \|\boldsymbol{y}_n - mean\ of\ C_k\|$;

   Find k for which d is minimum;

   Assign $\boldsymbol{y}_n$ to C_k and increase isc_k by 1;

   if $isc\_k < \mathtt{T_c}$,

   goto Step 2;

   else

   $data = C_k$, check $data$ by the binary detector;

   if $k_0^* = 1$

   $isc\_k \leftarrow 0$ (reset isc_k);

   goto Step 2;

   end

   if $k_0^* = 2$, then goto Step 3;

end;

end;

3. The cluster splitting/regrouping/merging process:

After Step 3, the number of the clusters, m, may either increase by 1, or decrease by 1, or remain the same. The counter index still ranges from 1 to m.

$m \leftarrow m + 1$;

$C_m \leftarrow C_{x2}$;

$C_k \leftarrow C_{x1}$;

Find C_j which is closest to C_k;

$data = C_j + C_k$, check $data$ by the binary detector;

if $k_1^* = 1$ (indicates merging)

$C_j \leftarrow C_j + C_k$; $isc\_j \leftarrow 0$;

$C_k \leftarrow C_m$;

delete C_m; $m \leftarrow m - 1$;

Find C_j which is closest to C_k;

$data = C_j + C_k$, check $data$ by the binary detector;

if $k_2^* = 1$ (indicates merging)

$C_j \leftarrow C_j + C_k$; $isc\_j \leftarrow 0$;

$C_k \leftarrow C_m$; $isc\_k \leftarrow 0$;

delete C_m; $m \leftarrow m - 1$;

end

if $k_2^* = 2$ (indicates regrouping)

$C_j \leftarrow C_{x1}$; $isc\_j \leftarrow 0$;

$C_m \leftarrow C_{x2}$; $isc\_m \leftarrow 0$;

end

end

if $k_1^* = 2$ (indicates regrouping)

$C_j \leftarrow C_{x1}$; $isc\_j \leftarrow 0$;

$C_k \leftarrow C_{x2}$; $isc\_k \leftarrow 0$;

Find C_j which is closest to C_m;

$data = C_j + C_m$, check $data$ by the binary detector;

if $k_2^* = 1$ (indicates merging)

$C_j \leftarrow C_j + C_m$; $isc\_j \leftarrow 0$;

delete C_m; $m \leftarrow m - 1$;

end

if $k_2^* = 2$ (indicates regrouping)

$C_j \leftarrow C_{x1}$; $isc\_j \leftarrow 0$;

$C_m \leftarrow C_{x2}$; $isc\_m \leftarrow 0$;

end

end

5.5.2 Computational Complexity

Suppose that after the on-line process N M-dimensional data vectors are assigned to $\hat{K}$ clusters, resulting in that the kth cluster has N_k data vectors ($k = 1, \cdots, \hat{K}$). As mentioned in Section 5.4.2, the exact complexity analysis is difficult due to the fact that the computational cost of on-line process may vary when the same set of pulses arrives in a differnet sequence. In this subsection, An upper complexity bound and an average complexity are analyzed; Then an example is illustrated in the end.

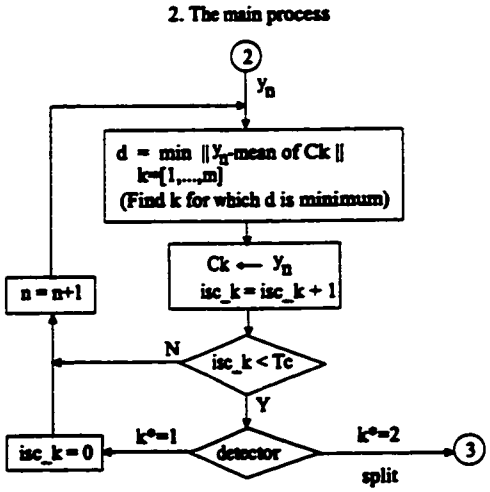
Upper Bound

There are three steps in this on-line clustering algorithm:

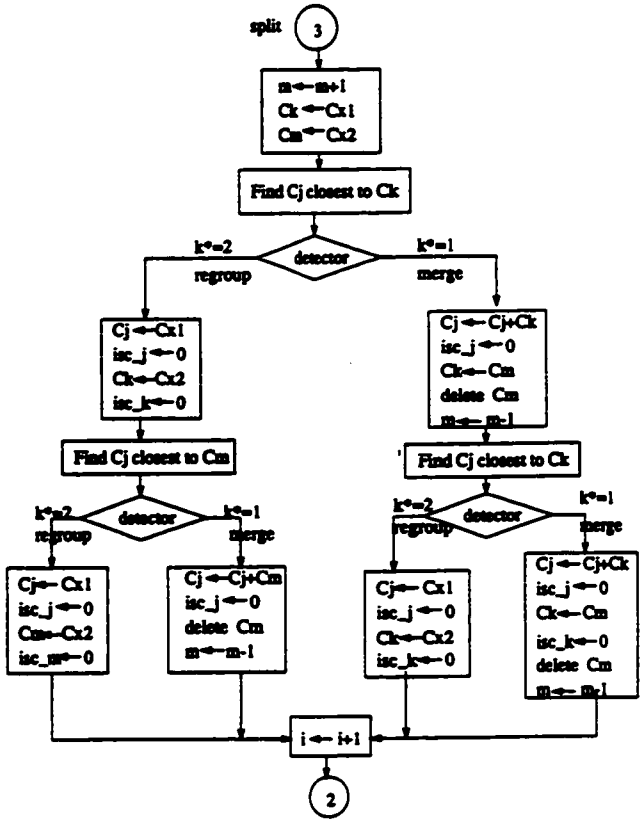
1. The first is the main process.

   The main process here is the same as the one considered in Section 5.4.2. Hence, the computational cost of this process according to Eq. (5.13) is upper-bounded by

$$MN(3\hat{K} - 0.5) + 0.5MN^2.$$



(a) The main process



(b) The cluster splitting, merging and regrouping process

Figure 5.5: The diagram of our on-line model-based clustering algorithm

2. The second is the cluster splitting/regrouping/merging process.

   The dominant complexity for the binary detector is to partition the given number of data vectors into two clusters, i.e., $K_{\max} = 2$. According to Eq. (3.70) in Section 3.4.2, this computational cost is $7MnN_t$, where n is the number of data vectors sent to the binary detector and N_t is the number of iterations for cluster center convergence. We need to use the binary detector $\frac{N}{T_c}$ times to check if a cluster should be split, and such a checking may result in one or two more binary detections to pursue cluster regrouping and merging. Thus, the total number of times (the binary detector being used) varies between $\frac{N}{T_c}$ and $3\frac{N}{T_c}$. Obviously the number of data vectors sent to the binary detector is at most N. Hence, the computational cost of the cluster splitting/regrouping/merging process is upper-bounded by

$$3\frac{N}{T_c}(7MNN_t) = 21\frac{MN^2N_t}{T_c}.$$

3. The third is the post-process.

   To regroup all data vectors according the $\hat{K}$ cluster centers by N_t times (see Steps (a) and (b) in Section 3.4.2), the computational cost is

$$N_t\left[MN + 3MN\hat{K}\right] = MNN_t(3\hat{K}+1). \tag{5.15}$$

   The final splitting on $\hat{K}$ clusters requires using the binary detector $\hat{K}$ times so the computational cost of the final splitting is

$$7MN_t(N_1 + N_2 + \cdots + N_{\hat{K}}) = 7MN_tN.$$

   There are $0.5\hat{K}(\hat{K}-1)$ different combinations to send two clusters to the binary detector for the final regrouping/merging purpose so the computational cost for the final regrouping/merging is

$$\begin{aligned} 7MN_t[(N_1+N_2)+(N_1+N_3)+\cdots\cdots &+ (N_1+N_{\hat{K}}) \\ +(N_2+N_3)+\cdots\cdots &+ (N_2+N_{\hat{K}}) \\ &+ (N_{\hat{K}-1}+N_{\hat{K}}) = 7MN_t(\hat{K}-1)N. \end{aligned}$$

Furthermore, The final splitting/regrouping/merging operation is repeated N_r times. Hence, the computational cost of the final splitting/regrouping/merging operation is approximately

$$N_r(7MN_tN + 7MN_t(\hat{K}-1)N) = 7MNN_tN_r\hat{K}.$$

Therefore, the total computational cost of the on-line clustering algorithm described in Section 5.5.1 is upper-bounded by

$$\begin{aligned} B_{\text{on2}} &= MN(3\hat{K}-0.5) + 0.5MN^2 \\ &\quad +21\frac{MN^2N_t}{T_c} \\ &\quad +MNN_t(3\hat{K}+1) + 7MNN_tN_r\hat{K}. \end{aligned} \tag{5.16}$$

Average Complexity

As in Section 5.4.2, we examine the case where each cluster has the same number of members (i.e., $N_1 = N_2 = \cdots = N_{\hat{K}} = \frac{N}{\hat{K}}$). Since the computational cost of the on-line process is sensitive to the sequence in which the pulses arrive, we assume that the members of the first cluster arrive first, and then those of the second cluster arrive next, and so on. The computational cost for this case is roughly an average complexity of the on-line clustering algorithm. Now we examine the three steps in this on-line clustering algorithm again:

1. The first is the main process. The computational cost of this process according to Eq. (5.14) is approximately
$$MN(1.5\hat{K}+1) + \frac{0.5MN^2}{\hat{K}}.$$

2. The second is the cluster splitting/regrouping/merging process. We need to estimate the times the binary detector being used, and the number of data vectors involved each time. For simplicity, an average number $1.5\frac{N}{T_c}$ is used here, and an average size of data sent to the binary detector is assumed to be $\frac{N}{\hat{K}}$. Hence, the computational cost for the cluster splitting/regrouping/merging process is approximately
$$1.5\frac{N}{T_c}(7M\frac{N}{\hat{K}}N_t) = 10.5\frac{MN^2N_t}{T_c\hat{K}}.$$

3. The third is the post-process. The computational cost of the post-process is the same as the one in the upper-bound analysis, i.e., $MNN_t(3\hat{K}+1)+7MNN_tN_r\hat{K}$.

Therefore, the average complexity of the on-line clustering algorithm described in Section 5.5.1 is approximately

$$\begin{aligned} C_{\text{on2}} &= MN(1.5\hat{K}+1)+\frac{0.5MN^2}{\hat{K}} \\ &\quad +10.5\frac{MN^2N_t}{T_c\hat{K}} \\ &\quad +MNN_t(3\hat{K}+1)+7MNN_tN_r\hat{K}. \end{aligned} \tag{5.17}$$

Example: Let us consider the same example as in Section 5.4.2, there are 100 44-dimensional preprocessed pulse vectors ($N=100, M=44$) and 5 emitters ($\hat{K}=5$). Given that $T_c=10$, $N_t=5$ and $N_r=2$, Substituting the values of N, M, $\hat{K}$, T_c, N_t and N_r in Eqs. (5.16) and (5.17), we have the upper bound and the average complexity as follows:

$$\begin{aligned} B_{\text{on2}} &= 6.79\text{Mflops}, \\ C_{\text{on2}} &= 2.43\text{Mflops}. \end{aligned}$$

As illustrated in Section 5.4.2, the computational cost by using the off-line model-based algorithm for the same example is 11.70 Mflops. Obviously the on-line algorithm is faster than the off-line algorithm while its performance is almost the same as that of the off-line counterpart as will be shown in Section 5.6.2.

5.6 Numerical Experiments on Intra-pulse Data

To illustrate the data pre-processing techniques and the effectiveness of the clustering algorithms developed in this chapter, we have carried out numerous experiments using computer simulated data. All programs including those for pre-processing and data compression are written in MATLAB and the simulations are run on a Pentium PC (400MHz).

5.6.1 Pulse Generation

We generate the pulses according to the signal representations Eq. (5.1), such that

- the distribution of absolute amplitude η_n is uniform in $[0.5, 1]$;
- the distribution of initial phase ψ_n is uniform in $[-\pi, \pi]$;
- the distribution of time delay τ_n is uniform in $[0, 5T]$;
- the distribution of carrier frequency ω_n is Gaussian and its standard deviation is 10 percent of its normalized mean value;
- the distribution of additive noise $\nu_n(t)$ is Gaussian with zero mean and the standard deviation about 0.05.

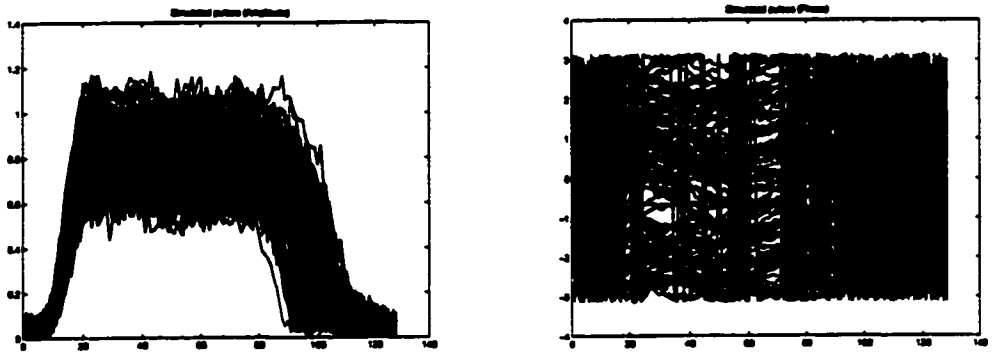
From a given set of signature signals $\{s_k(t)\}$, we can generate received pulses using the above parameter distributions.

5.6.2 Example 1

Fig. 5.6 (a) and (b) show the amplitude and phase of a group of 100 pulses received by the detector. These are from 5 different emitters each transmitting 20 pulses. The five emitters are numbered by $1, 2, \cdots, 5$. If a pulse is from the kth emitter ($k = 1, 2, \cdots, 5$), then its emitter index is k. Suppose that $\hat{K}$ clusters are determined by a clustering algorithm and that these $\hat{K}$ clusters are numbered by $1, 2, \cdots, \hat{K}$. if a pulse is assigned to the kth cluster ($k = 1, 2, \cdots, \hat{K}$), then its cluster index is k. By cross-checking cluster indices against emitter indices, we can count the classification accuracy in percentage.

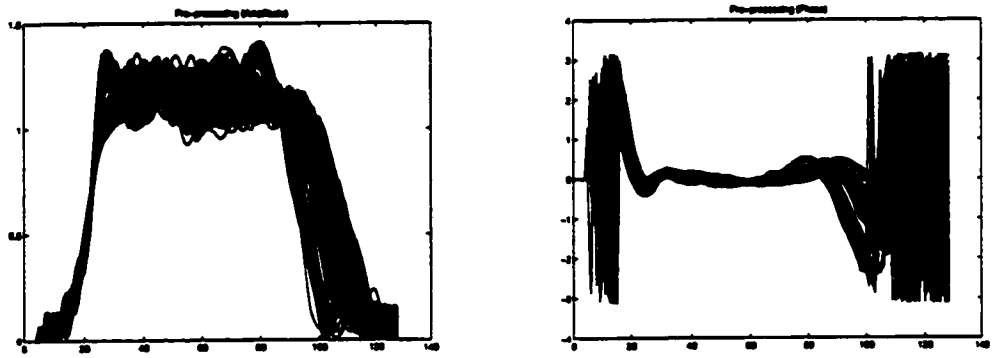
Pre-processing

For the pulses shown in Fig. 5.6 (a) and (b), the pre-processing techniques of amplitude normalization, time alignment and phase adjustment as described in Section 5.2 are applied to remove the nuisance parameters. The amplitude and phase of the pre-processed pulses are shown in Fig. 5.7 (a) and (b) respectively. Each pulse is represented by 128 time samples. A 3-level wavelet decomposition using *symlet4* filters is then applied to each of these pre-processed pulses and only the low frequency filter output samples are retained. Each pulse is now represented by 22-dimensional samples (The number $22 > 128/8$ is used because of



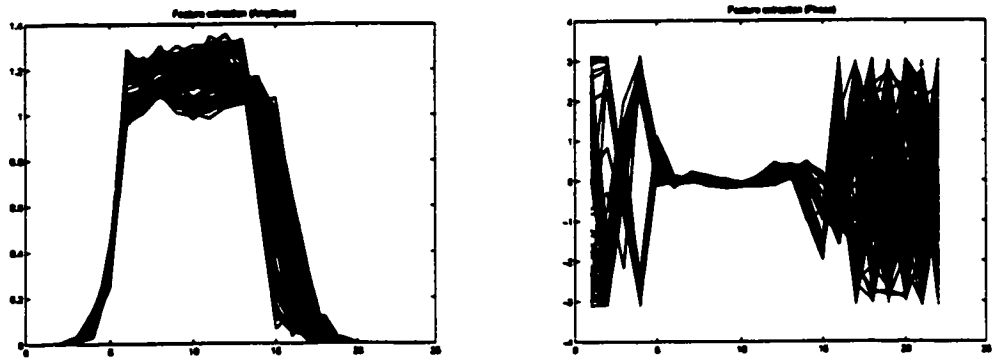
(a) (b)

Figure 5.6: Amplitude and phase of 100 received pulses from 5 unknown emitters, where x-axis is the index of data sample points.



(a) (b)

Figure 5.7: Amplitude and phase of the pre-processed pulses, where x-axis is the index of data sample points.



(a) (b)

Figure 5.8: Amplitude and phase of the compressed pre-processed pulses, where x-axis is the index of data sample points.

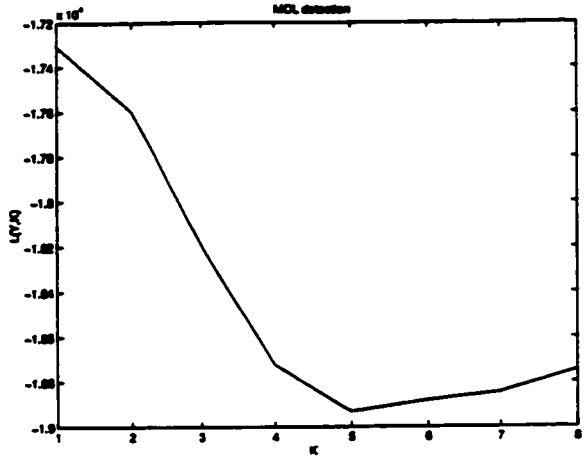


Figure 5.9: Determination of the number of emitters using our (off-line) clustering algorithm

Cluster index	Number of pulses	Emitter index assigned to the pulses
1	20	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2	19	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3
3	21	2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
4	20	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
5	20	5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5

Table 5.1: Clustering results for Example 1 by the off-line model-based clustering algorithm

the transient effect of the filters). As a result, data compression has been achieved. The amplitude and phase of the compressed pulses are shown in Fig. 5.8 (a) and (b) respectively. In the following, clustering is based on the compressed data. Furthermore, to compare our clustering algorithm and SNOB on a fair basis, Covariance Structure 4 is assumed.

The off-line model-based clustering algorithm

The clustering algorithm developed in Section 3.4 is applied to the compressed data. The evaluation of $L(\boldsymbol{Y}, \hat{K})$ for various values of $\hat{K}$ is plotted in Fig. 5.9. The number of clusters which is the value of $\hat{K}$ at which $L(\boldsymbol{Y}, \hat{K})$ is minimum is correctly determined to be 5. The association of the pulses using the clustering algorithm is shown in Table 5.1. It can be observed that apart from the seven pulses in Emitters 2 and 3, all the other pulses have been correctly associated. The classification accuracy in this case is 93%. For this example,

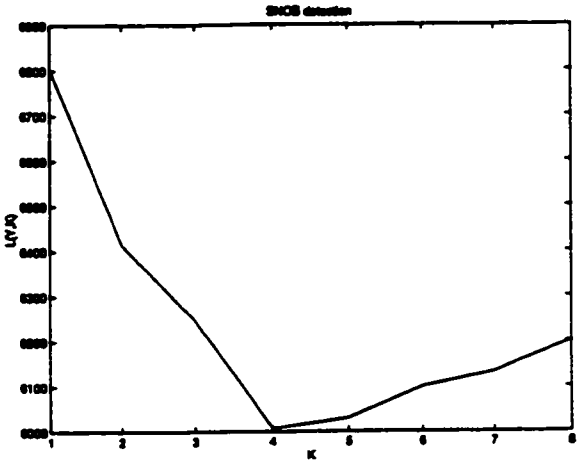


Figure 5.10: Determination of the number of emitters using the SNOB program

Cluster index	Number of pulses	Emitter index assigned to the pulses
1	20	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2	40	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3	20	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
4	20	5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5

Table 5.2: Clustering results for Example 1 by the SNOB algorithm

the clustering algorithm takes approximate 20 seconds to produce the above results.

SNOB

The SNOB algorithm has also been applied to this example, and the evaluation for different values of $\hat{K}$ is plotted in Fig. 5.10 while the clustering results are shown in Table 5.2. It is observed in Table 5.2 that the SNOB algorithm fails to identify all the emitters and the signals from Emitters 2 and 3 cannot be distinguished. SNOB is written in Fortran so it is difficult to compare its speed with our MATLAB program. As discussed in Section 3.5, SNOB in principle is more complex than our off-line clustering algorithm.

The on-line algorithm using known thresholds

We apply the on-line algorithm developed in Section 5.4 to the example with $t_1 = 0.04$, $t_2 = 0.08$, $T_s = 20$. The clustering result is shown in Table 5.3. The number of

Cluster index	Number of pulses	Emitter index assigned to the pulses
1	12	1 1 1 1 1 1 1 1 1 1 1 1
2	8	1 1 1 1 1 1 1 1
3	21	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3
4	19	2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
5	20	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
6	20	5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5

Table 5.3: Clustering results for Example 1 by the on-line clustering algorithm using known thresholds

clusters is 6 and the classification accuracy is 87%. Comparing the results of the on-line with the off-line methods, we find that this on-line result is slightly inferior. However, from the computational standpoint, the on-line algorithm based on the two thresholds is much simpler than the off-line counterpart. This on-line algorithm takes less than 1 second to produce the above results. Recall that the off-line clustering procedure described in Section 3.4 includes cluster splitting and regrouping operations. Obviously, we may improve the performance of this original on-line algorithm by introducing cluster splitting, merging and regrouping operations appropriately. However, more thresholds are needed.

The on-line model-based algorithm

We apply the on-line algorithm developed in Section 5.5 to the example. The clustering result is shown in Table 5.4. The number of clusters is 5 and the classification accuracy is 93%. The results are the same as those in Table 5.1 produced by the off-line model-based algorithm. This on-line algorithm takes approximately 6 seconds to produce the above results. We also note that, for a given data set, the computation cost of the on-line algorithm is much lower than that of the off-line. From Eq. (3.70), we know that the computational complexity of the off-line model-based algorithm is approximately proportional to $K_{\max}^3$. The computational cost is reduced significantly by binary partitions $K_{\max} = 2$ involved in the on-line model-based algorithm. In this sense, we conclude that this on-line algorithm is a fast version of the off-line model-based algorithm.

Cluster index	Number of pulses	Emitter index assigned to the pulses
1	20	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2	19	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3
3	21	2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
4	20	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
5	20	5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5

Table 5.4: Clustering results for Example 1 by the on-line model-based clustering algorithm

5.6.3 Conclusions

Many other simulation examples have been carried out using different number of received signals, different number of emitters, and different distributions of random signal parameters. The following are general observations drawn from the results of cluster validation and clustering:

1. The results of clustering employing compressed data from a 3-stage *symlet4* wavelet filter bank are in general the same as those employing uncompressed data.

2. Judging from performance, our model-based off-line clustering algorithm shows much higher reliability in cluster validation than SNOB, while sacrificing marginally on the accuracy in clustering.

3. The performance of our model-based on-line clustering algorithm is almost the same as that of the model-based off-line algorithm but is much faster.

For Observation 1, it seems that the original pulse contains redundant information, so by compressing it, adequate information is still retained. The performance of the new off-line clustering algorithm and SNOB has been compared intensively in Section 3.6; The results there well justifies the second observation. The last observation show that our on-line model-based clustering algorithm is a faster version of model-based clustering while retaining the quality of performance.

Furthermore, it is found that the best performance of our clustering algorithm for intrapulse analysis is usually achieved by using Covariance Structure 4 described in Section

3.3.4 if the default penalty weight ($\lambda = 1$) is used, and that the best performance is usually achieved by using Covariance Structure 2 described in Section 3.3.2 when supervision is available (i.e., the penalty weight can be adapted).

5.7 Summary

First, we have developed pre-processing techniques to remove some nuisance parameters from received pulses. These include the absolute amplitude, initial phase, time delay and residual carrier frequency. As a result, we have formulated the problem of emitter number estimation and pulse-emitter association as a multivariate Gaussian clustering problem. In order to reduce the computational cost for clustering, a data compression method based on wavelet decomposition has also been included in pre-processing. The pre-processing techniques are intuitive in nature and are carried out so that after pre-processing, the pulses received from the same emitter maintain the resemblance to each other, while those from different emitters maintain their distinctive features.

After applying the new off-line clustering algorithm developed in Chapter 3 to the clustering problem, we have developed two on-line clustering algorithms: one is based on known thresholds while the other makes use of the model-based detection modified from the off-line clustering algorithm. Although the on-line clustering based on thresholds is computationally very effective, it is difficult to adapt the thresholds as the statistics of received pulses changes in time. In contrast, the on-line model-based clustering algorithm does not require explicit thresholds; It can dynamically incorporate cluster splitting, merging and regrouping operations as the statistics of the received pulses changes.

The performance of our clustering algorithms and SNOB on intrapulse data have been reported. The results demonstrate that our new clustering algorithms are very effective for intrapulse analysis, especially the on-line model-based algorithm. Therefore, the on-line model-based clustering algorithm is suitable for near real-time implementation, which will be explored in the next chapter.

Chapter 6

DSP Implementation

6.1 Introduction

In the previous chapters, we have developed several radar pulse classification algorithms based on Minimum Encoding Inference, and described the framework of the penalized likelihood method. Extensive simulations show that the performance of our new algorithms is promising, especially the on-line model-based algorithm which is well suited for dynamically classifying incoming radar pulses. As a result, we have implemented this on-line clustering algorithm as a core classification module on a TMS320C44 DSP board.

In this chapter, the DSP implementation for intrapulse analysis is described. We do a simple analysis of the physical scenarios at a radar intercepter and estimate the likely maximal incoming pulse rate. We then propose a suitable system diagram and investigate the system requirements. The benchmark of the DSP coding of our on-line clustering algorithm is reported.

6.2 Physical Scenario Analysis

In this section, we describe some examples and discuss how radar characteristics [57] can affect the operation of a typical radar intercepter. Here and throughout we assume that the radar intercepter is passive in all directions.

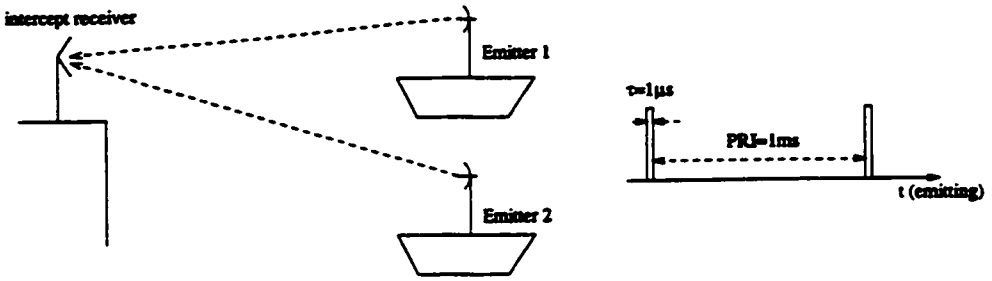


Figure 6.1: Physical scenario example 1

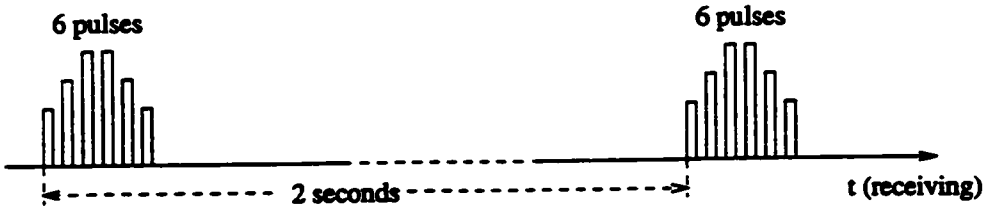


Figure 6.2: Physical scenario example 2

6.2.1 Probability of Receiving Overlapped Pulses

Given a radar, let PRF (Pulse Repetition Frequency) be 1000 pulses/sec and c be the speed of light, then the maximum range of the radar is

$$R_{\max} = \frac{c}{2 \times \mathrm{PRF}} = 150 \text{ km}.$$

Let range resolution ΔR be 150 m, then the pulse width is

$$\tau = \frac{2 \times \Delta R}{c} = 1\mu s.$$

So the duty cycle

$$\tau \times \mathrm{PRF} = \frac{1}{1000}.$$

It means that even if a radar interceptor receives pulses from two radars with the same specifications given above, the probability of receiving overlapped pulses is only 1/1000, assuming a perfect time synchronization. The illustration is shown in Fig. 6.1.

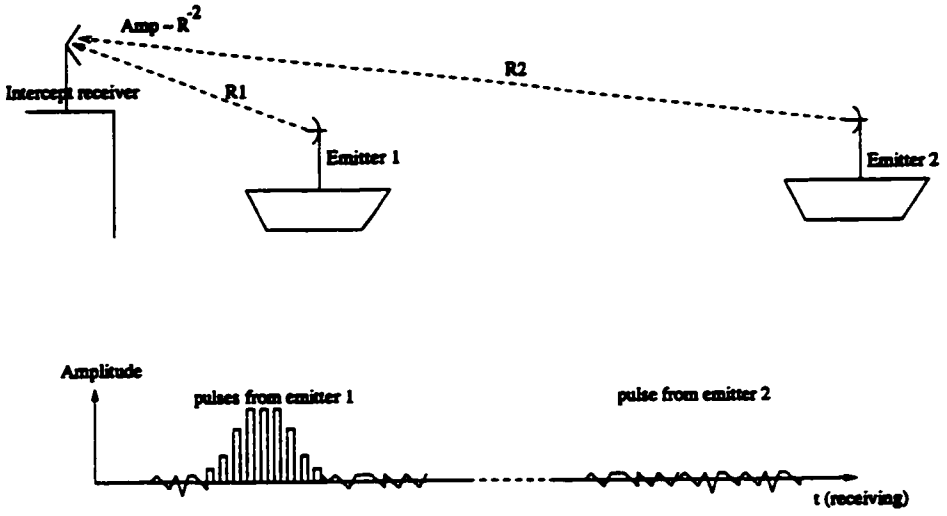


Figure 6.3: Physical Scenario example 3

6.2.2 Receiving Pulse Sequence

Given a radar with antenna beamwidth BW, the average incoming pulse rate to the intercept receiver generated by the radar emitter is

$$\text{rate}_i = \frac{\text{PRF}}{360°/\text{BW}} = \frac{\text{PRF} \times \text{BW}}{360°}. \tag{6.1}$$

Let a radar emitter with PRF = 1000 pulses/sec and BW = 1°, then at the intercept receiver, $\text{rate}_i \simeq 3$ pulses/sec. If the antenna rotation rate RPM is 30 rpm (rotations per minute), we will observe 6 incoming pulses from the intercept receiver in every two-second period. The illustration is shown in Fig. 6.2.

6.2.3 Near-Far Phenomenon

The signal power of an incoming pulse at the intercept receiver is inversely proportional to the square of the distance between the radar and the interceptor. Consider the case that one emitter is close to the interceptor and another is far away, it is possible that at the interceptor, the incoming pulse amplitude generated by a sidelobe of the close emitter overwhelms the pulse amplitude generated by the mainlobe of the far away emitter. This means that in a given geographic area, the number of detectable emitters is limited by this

Frequency Band: 3GHz or 9GHz		
PRF	Pulse Width (τ)	Duty Cycle (τ·PRF)
3.40KHz	$0.05\mu s$	0.17×10^{-3}
1.70KHz	$0.25\mu s$	0.43×10^{-3}
0.85KHz	$0.5 \sim 1\mu s$	$0.43 \sim 0.85 \times 10^{-3}$
Antenna Beamwidth (BW): $0.8 \sim 2.5$ degrees Antenna Rotation (RPM): typical 30 rpm (rotation per minute)		
Sidelobes relatives to main beam: around -25 dB		

Table 6.1: DECCA Groups 9A and 12A relative motion marine radars

near-far phenomenon. The illustration is shown in Fig. 6.3.

6.3 Maximal Incoming Pulse Rate

Some important parameters for a set of typical marine radars [73] are listed in Table 6.1. Suppose there are a number of ships in the surveillance area, each equipped with two radars. Under normal circumstances, the number of ships may be around 20. However, the ships usually have different types of radars, which can be identified according to inter-pulse information such as carrier frequency and pulse width. It is very rare that 40 same type radars are operating at the same time in the same area. Nevertheless, we assume for the worst case that the maximum number of the same type radars present is 40.

Assume that the radar types are given in Table 6.1. Then the maximal incoming pulse rate generated by a radar emitter is attained by using Eq. (6.1) when PRF = 3.4 KHz and BW = 2.5°:

$$\text{rate}_{\text{imax}} = 3400 \times 2.5/360 \simeq 24 \text{ pulses/sec.}$$

The minimal incoming pulse rate generated by a radar emitter is attained by using Eq. (6.1) when PRF = 0.85 KHz and BW = 0.8°:

$$\text{rate}_{\text{imin}} = 850 \times 0.8/360 \simeq 2 \text{ pulses/sec.}$$

Therefore, when 40 same type radars are operating at the same time in the same area, the incoming pulse rate in the intercept receiver is at most $40 * 24 = 960$ pulses/sec.

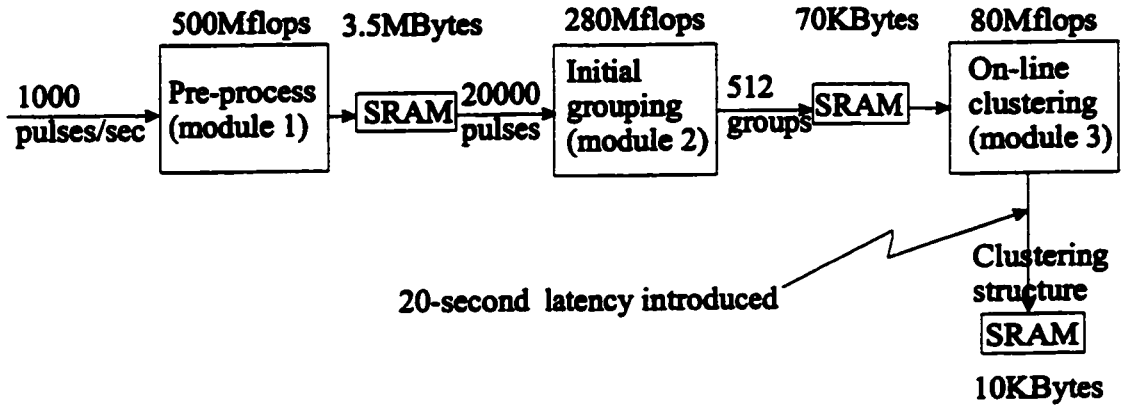


Figure 6.4: The DSP system diagram for intrapulse analysis

In the following sections of this chapter, we will assume the maximal incoming pulse rate at the intercept receiver is 1000 pulses/sec.

6.4 System Diagram And Requirements

As described in Section 5.5, the on-line clustering algorithm dynamically performs cluster splitting, merging and regrouping operations as pulses arrive. It is also suggested that we can not determine the pulse-emitter association right away since the clustering structure is being updated continuously. Here we assume that the maximal incoming pulse rate is 1000 pulses/sec and we output the pulse-emitter association after 20 seconds, i.e., a 20-second latency is introduced. Therefore, we need to process up to 20,000 pulses in every 20-second period. The whole process is divided into three independent modules: pre-process, initial grouping and on-line clustering; see the system diagram in Fig. 6.4. Next let us discuss the computational cost and memory requirement on each module.

6.4.1 Pre-processing

The pre-processing described in Section 5.2 consists of four steps: curve rotation, amplitude normalization, time alignment and data compression. Counted by our MATLAB programs, it requires 25 Kflops to pre-process a pulse with around 128 sampling points to a 44-dimensional vector. So 20,000 pulses require 500 Mflops. To totally store the 20,000 pre-

processed pulses, we need a memory:

$$20,000 \text{ pulses} \times 44 \text{ floating points/pulse} \times 4 \text{ Bytes/floating point} \simeq 3.5 \text{ MBytes}$$

6.4.2 Initial Grouping

The purpose of this module is to reduce the workload of the on-line clustering (Module 3). This is achieved by grouping 20,000 pulses into a certain number (say 512) of groups. Then in Module 3, we only deal with the mean vectors for the 512 groups, instead of the whole data set. In the following, we consider how to group N M-dimensional vectors into $\hat{K}$ groups.

One simple approach is by using the following batch scheme: split the whole data set into two groups, and then split each group into another two groups, and so on. If a group size is less than $\frac{N}{\hat{K}} = 50$, then this group is not split any more. The illustration is shown in Fig. 6.5. In the first stage, N M-dimensional vectors are split into two groups with the sizes being N_1 and N_2 respectively ($N_1 + N_2 = N$). One effective method is the *k-means* algorithm which has been used in Section 3.4.1. Let N_t be the number of iterations for cluster center convergence, then the computational cost for this stage is roughly $7MNN_t$ by using Eq. (5.15). At the second stage, each group is further split into two new groups, and the computational cost is

$$7MN_1N_t + 7MN_2N_t = 7MNN_t$$

Assume that the average number of splitting stages is L, then the total computational cost for initial grouping is roughly $7MNN_tL$.

To split 20,000 pulses into 512 groups, we need do in average 9 stage splitting ($512 = 2^9$, i.e., $L = 9$). Hence, given $N = 20,000$, $M = 44$, $\hat{K} = 512$, $N_t = 5$, and $L = 9$, the computation cost is roughly

$$7N_tMN \times L \simeq 280 \text{ Mflops}$$

To store the mean vectors of the 512 groups, we need a memory: $400 \times 44 \times 4 \simeq 70$ KBytes.

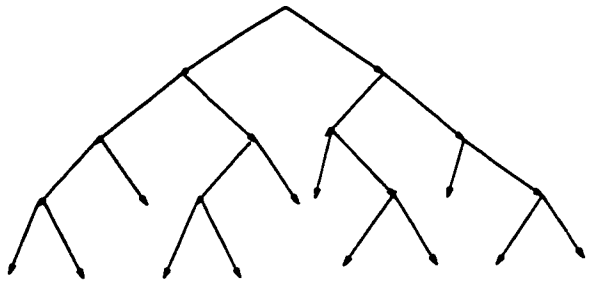


Figure 6.5: The tree structure for initial grouping

6.4.3 On-line Clustering

The procedure of the on-line clustering algorithm has been described in Section 5.5.1. When a pulse arrives, it is assigned to one of existing clusters according to the minimum distance principle. A cluster is checked if its size increment counter reaches a preset number T_c. After on-line process, the final regrouping is repeated by N_t times and the final splitting/merging is repeated by N_r times. The computational cost of the on-line algorithm has been analyzed in detail in Section 5.5.2. The maximal number of same type radar emitters in a surveillance area is assumed to be 40 (see details in Section 6.3) so that the largest number of clusters here is around 40. Given that $N = 512$, $M = 44$, $\hat{K} = 40$, $T_c = 10$, $N_t = 5$ and $N_r = 2$, Substituting them in Eq. (5.17), we have

$$C_{\text{on2}} \simeq 80\text{Mflops}$$

which is the average complexity for the on-line clustering module. The memory for keeping the clustering structure is $40 \times 44 \times 4 = 7\text{KBytes} < 10\text{KBytes}$.

6.5 C/DSP Coding of On-line Clustering

In this section, we briefly introduce our C/DSP coding of the model-based on-line clustering algorithm as a core clustering module on a TMS320C44 DSP board. The tools to complete this task are

- TMS320C3x/C4x floating-point DSP code generation tools [74–77].

Program size	2.5Kwords
Maximum data size	448Kwords
CPU processing time (example 1 in Section 5.6)	on-line processing: 0.2 seconds post-processing: 0.3 seconds

Table 6.2: The benchmark of DSP codes of the on-line clustering

- Code Composer [78] (coding and debugging) for C and Assembly from GO DSP Corporation.
- TMS320C44 (60Mflops) development board (Dakar) and its SDK (software development kits) [79,80] from Spectrum Signal Processing Inc.

The MATLAB programs developed before are redesigned into C programs which are converted to DSP codes by TI DSP code generation tools [74–77].

To efficiently locate memory for data and program, we need to study the physical configuration of the DSP board [79, 80], the local memory (SRAM) provided for the C44 is 2M bytes, i.e., 512K words. We could use the small memory model because all large arrays in our programs are allocated at run-time from a global pool, or heap (using `malloc`). Therefore, we partition the local 512K-word memory into two parts: one with 64K words is reserved for the small memory model and another with the rest 448K words is provided for the heap. With this memory allocation, our codes will run without any problem in the small memory model even when 10,000 pulses and each with 40 data points are processed (Note: $10,000 \times 40 < 448K$). In addition, the system stack is allocated in a 2K word internal RAM of the C44 chip.

To process the compressed data of Example 1 in Section 5.6. The C44 processing time itself takes 0.5 seconds, counted by the profiling features of the Code Composer [78]. In fact, the on-line process takes 0.2 seconds and the post-processing takes the remaining 0.3 seconds. Thus, there is a tradeoff between the speed and the performance.

As shown in Table 6.2, our on-line clustering module for the TMS320C44 development board is very efficient in size and speed. It has the capability to process data array as large

as 448K words. This on-line clustering module is ready for deployment on a multi-processing DSP board together with other modules such as pre-processing.

Chapter 7

Conclusions

In this thesis, we have reviewed Bayesian inference and minimum encoding inference including Wallace's minimum message length (MML) and Rissanen's minimum description length (MDL) for model selection. It is found that the MML coding length is more accurate than the other two from the standpoint of quantization. All model selection criteria considered here consist of two parts, one is the log-likelihood function which measures the goodness of fit between the data and the model, and the other is a penalty function which measures the complexity of the model. An inference method aims to balance the trade-off between goodness of fit and model complexity. As such, a penalty weight for the penalty function to control the trade-off has been introduced.

Based on minimum encoding inference, an appropriate measure of coding length has been proposed for cluster validation. Furthermore, the coding lengths under four different Gaussian mixture models have been fully derived. This provides us with a criterion for the development of a new clustering algorithm. Judging from the performance comparison, our coding length measure outperforms the Bayesian Inference Criterion (BIC) in cluster validation since it is not based on the large sample assumption as is BIC. More importantly, the new clustering algorithm shows much higher reliability in cluster validation than a well-known clustering algorithm SNOB, while sacrificing only marginally on the accuracy in clustering. Indeed, our clustering algorithm is well designed to effectively process high

dimensional data with satisfactory performance on small and medium samples.

The error performance of our clustering algorithm has been evaluated under reasonable assumptions. Two types of errors have been analyzed: miss and false alarm. Among four factors considered here (the dimension of the data space M, the sample size N, the mixing portion c and the inter-cluster distance D), D is the most important factor. There is a critical distance D_0 defined in Eq. (4.18) such that when $D > D_0$, our clustering algorithm can successfully separate two clusters and that when $D \leq D_0$, the extensive overlap between the two clusters will cause our algorithm to fail, as like other clustering algorithms such as SNOB. Furthermore, we have examined the impact of the penalty weight under the framework of the penalized likelihood method. It is found that there is a range of penalty weights within which the best performance of our clustering algorithm can be achieved. Therefore, it is suggested that with some supervision, we could adjust the penalty weight to further improve the performance of our clustering algorithm.

Another important contribution of this thesis is the application of our clustering algorithm to intrapulse analysis. We have developed the pre-processing techniques to remove nuisance parameters from received pulses and formulated the problem of emitter number detection and pulse-emitter association as a multivariate clustering problem. In order to reduce the computational cost for clustering, a suitable data compression method based on wavelet decomposition has also proposed. These pre-processing techniques are intuitive in nature and are carried out so that after pre-processing, the pulses received from the same emitter maintain the resemblance to each other, while those from different emitters retain their distinctive features.

There are several factors that make the task of clustering a challenging one: (1) the dimension of data vectors is high; (2) satisfactory performance on small samples is desirable; (3) near real-time implementation is required. The model-based clustering algorithm developed in Chapter 3 well addresses the first two factors. Furthermore, it is found that the best performance of our clustering algorithm for intrapulse analysis is usually achieved by using Covariance Structure 4 (see Section 3.3.4) when no supervision is available (i.e., the penalty

weight is 1, the default value), and that the best performance is usually achieved by using Covariance Structure 2 (see Section 3.3.2) when supervision is available (i.e., the penalty weight can be adapted). To achieve on-line clustering, that is, to perform classification dynamically as pulses arrive, we have further developed two new algorithms, one is based on known thresholds while the other is based on a model-based detection scheme. Although the on-line algorithm based on thresholds is computationally very effective, it is difficult to adapt the thresholds as the statistics of received pulses changes in time. In contrast, the on-line model-based algorithm does not require explicit thresholds and dynamically incorporates cluster splitting, merging and regrouping operations as the statistics of the received pulses changes. Performance of our clustering algorithms and SNOB on intrapulse data have been reported. It demonstrates that our new clustering algorithms (the on-line model-based algorithm in particular) are very effective for intrapulse analysis due to their low computation costs and high performance.

Our model-based clustering algorithms have been further implemented on a DSP board for intrapulse analysis. Some relevant physical parameters have been estimated such as the likely maximal incoming pulse rate. Then a suitable system diagram has been proposed and its system requirements have been suggested. The on-line model-based algorithm has been implemented as a core classification module on a TMS320C44 DSP board.

7.1 Future Work

In this thesis, we have developed new model-based clustering algorithms both off-line and on-line, and successfully applied them to intrapulse analysis. There are several issues worthy of further investigations in future research:

1. Applicability of other statistical models to clustering. In Chapter 5, Gaussian mixture models are applied for the clustering problem in intrapulse analysis since the noise accompanying a radar pulse is usually Gaussian. For different applications, other statistical models may be more suitable. For example, a mixture of uniform distributions

was explored in [3–5] where observations are edge elements in an image. Statistical models using von Mises distributions [26] were applied in [22] to find clusters in data of several thousand sets of protein dihedral angles.

2. Other promising criteria for cluster validation. As introduced in Section 1.2, the number of clusters in Gaussian clustering is the number of components in a Gaussian mixture. The determination of the number is formulated as a likelihood ratio test. However, the analysis for the null hypothesis case is very difficult since the regularity condition does not hold. In fact, the asymptotics that justify the penalized likelihood criteria (BIC, MDL and MML) are the same as those underlying the likelihood ratio test. One different approach is the use of Monte Carlo (bootstrap) tests. A study for bootstrapping the case of Gaussian mixtures was reported in [40]. However, empirically observed rejection rates may not quite match the expected levels under the null hypothesis, and it would be of interest to investigate the discrepancies involved.

3. Radar pulse classification by using inter-pulse information. In the field of applications, we have focused on using intrapulse information of a collection of pulses to identify the emitters present. However, radar emitter classification in practice is also based on interpulse information. Our model-based clustering schemes can be directly applied if the statistic of the interpulse information are provided. Furthermore, it would be of great interest to achieve the maximum integration gain by combining interpulse and intrapulse information. A simple approach is layered classifications by using interpulse and intrapulse information separately. Another approach is to form an integrated data vector for classification by using interpulse and intrapulse information together. If this is the case, the weighting between the interpulse part and the intrapulse part should be examined carefully.

Appendix A

The Value of $S(N, \hat{\alpha})$

$S(N, \hat{\alpha})$ is the number of different ways to partition N data vectors into $\hat{K}$ groups, resulting in that each group k ($k = 1, \cdots, \hat{K}$) has N_k data vectors. Obviously, $N_1 + N_2 + \cdots + N_{\hat{K}} = N$.

At the first stage, we take N_1 data vectors out of the whole data set, the number of different combinations is $C_N^{N_1}$; Then at the second stage, we take N_2 data vectors out of the rest of the whole data set, the number of different combinations is $C_{N-N_1}^{N_2}$, and so on. At the $(\hat{K}-1)$ stage, we! take $(N_{\hat{K}-1})$ data vectors out of the rest, the number of different combinations is $C_{N-N_1-\cdots-N_{\hat{K}-2}}^{N_{\hat{K}-1}}$. The last group is already determined in the end when the first $\hat{K}-1$ groups have been selected. Hence, the total number of different combinations is

$$\begin{aligned} & C_N^{N_1} C_{N-N_1}^{N_2} \cdots C_{N-N_1-\cdots-N_{\hat{K}-2}}^{N_{\hat{K}-1}} \\ = \frac{N!}{N_1!(N-N_1)!} & \times \frac{(N-N_1)!}{N_2!(N-N_1-N_2)!} \times \cdots \times \frac{(N-N_1-\cdots-N_{\hat{K}-2})!}{N_{\hat{K}-1}!(N-N_1-\cdots-N_{\hat{K}-1})!} \\ = & \frac{N!}{N_1!N_2!\cdots N_{\hat{K}}!}. \end{aligned}$$

In fact, the same groups in a different order construct the same clustering structure. If m_n clusters contain the same number of data vectors, we can swap these cluster order without changing the clustering structure. Let m_n be the number of clusters with n data vectors, $n = 1, 2, \cdots, N$, then the number of different partitions is that

$$S(N, \hat{\alpha}) = \frac{N!}{N_1!N_2!\cdots N_{\hat{K}}!(\hat{K}-1)!} \times \frac{1}{m_1!m_2!\cdots m_N!}.$$

Appendix B

$$R_m \sim F_{N,M(N-2)}(\delta)$$

We need two theorems to derive it.

Theorem 1: Suppose that an association vector $\hat{\alpha}$ partitions the N data vectors into $\hat{K}$ groups such that we have

$$\boldsymbol{Y}_{\boldsymbol{\alpha}} = \{\ \{\boldsymbol{y}_1(1),\ldots,\boldsymbol{y}_{N_1}(1)\},\ldots,\{\boldsymbol{y}_1(\hat{K}),\ldots,\boldsymbol{y}_{N_{\hat{K}}}(\hat{K})\}\ \},$$

and assume that these groups are the samples from multivariate Gaussian distributions with different means but the same covariance matrices, respectively, i.e., the k-th group is a sample from $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$. The sample mean $\hat{\boldsymbol{\mu}}_k$ and the sample within-group scatter matrix $\boldsymbol{W}_k$ for the k-th group are defined in Eqs. (3.10) and (3.11) respectively, and the total scatter matrix $\boldsymbol{W}$ for the whole data set is defined in Eq. (3.19).

A likelihood ratio criterion ([25, Page 165]) for testing the hypothesis $H_0 : \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \cdots = \boldsymbol{\mu}_{\hat{K}}$, is

$$\Lambda^{N/2} = \frac{|\boldsymbol{W}_1 + \boldsymbol{W}_2 + \cdots + \boldsymbol{W}_{\hat{K}}|}{|\boldsymbol{W}|} \tag{B.1}$$

which is called Wilks' Λ-statistic.

Denote

$$\boldsymbol{E} = \boldsymbol{W}_1 + \boldsymbol{W}_2 + \cdots + \boldsymbol{W}_{\hat{K}} \tag{B.2}$$

and

$$\boldsymbol{B} = \boldsymbol{W} - (\boldsymbol{W}_1 + \boldsymbol{W}_2 + \cdots + \boldsymbol{W}_{\hat{K}}). \tag{B.3}$$

Then

$$\Lambda^{N/2} = \frac{|\boldsymbol{E}|}{|\boldsymbol{E}+\boldsymbol{B}|} \tag{B.4}$$

where

- $\boldsymbol{E} \sim$ central Wishart distribution $W_M(N-\hat{K}, \boldsymbol{\Sigma})$.
- $\boldsymbol{B} \sim$ non-central Wishart distribution $W_M(\hat{K}-1, \boldsymbol{\Sigma}, \boldsymbol{\Omega})$, where $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}\sum_{k=1}^{\hat{K}} N_k(\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^T$ and $\boldsymbol{\mu} = \frac{1}{N}\sum_{k=1}^{\hat{K}} N_k\boldsymbol{\mu}_k$. In addition, when H_0 is true $\boldsymbol{\Omega} = \boldsymbol{O}$ so that $\boldsymbol{B} \sim W_M(\hat{K}-1, \boldsymbol{\Sigma})$.
- $\boldsymbol{E}$ and $\boldsymbol{B}$ are statistically independent.

Theorem 2: If $\boldsymbol{A}$ is $W_M(N, \boldsymbol{\Sigma}, \boldsymbol{\Omega})$ where N is a positive integer and $\boldsymbol{\alpha}(\neq \boldsymbol{O})$ is a $M \times 1$ fixed vector, then $\boldsymbol{\alpha}^T\boldsymbol{A}\boldsymbol{\alpha}/\boldsymbol{\alpha}^T\boldsymbol{\Sigma}\boldsymbol{\alpha}$ is $\chi^2_N(\delta)$, with $\delta = \boldsymbol{\alpha}^T\boldsymbol{\Sigma}\boldsymbol{\Omega}\boldsymbol{\alpha}/\boldsymbol{\alpha}^T\boldsymbol{\Sigma}\boldsymbol{\alpha}$ ([43, Theorem 10.3.6]).

From Theorem 1, we know that, under the assumption of perfect separation, our situation is the special case of the above when $\hat{K} = 2$. Thus in our case,

$$\boldsymbol{E} = \boldsymbol{W}_1 + \boldsymbol{W}_2 \sim W_M(N-2, \sigma^2\boldsymbol{I}_M)$$

$$\boldsymbol{B} = \boldsymbol{W} - (\boldsymbol{W}_1 + \boldsymbol{W}_2) \sim W_M(1, \sigma^2\boldsymbol{I}_M, \boldsymbol{\Omega})$$

where $\boldsymbol{\Omega} = \frac{1}{\sigma^2}[N_1(\boldsymbol{\mu}_1 - \boldsymbol{\mu})(\boldsymbol{\mu}_1 - \boldsymbol{\mu})^T + N_2(\boldsymbol{\mu}_2 - \boldsymbol{\mu})(\boldsymbol{\mu}_2 - \boldsymbol{\mu})^T]$ and $\boldsymbol{\mu} = \frac{1}{N}(N_1\boldsymbol{\mu}_1 + N_2\boldsymbol{\mu}_2)$.

We can write

$$tr\boldsymbol{B} = [1\ 0\ \cdots\ 0]\boldsymbol{B}\begin{bmatrix}1\\0\\\cdot\\\cdot\\\cdot\\0\end{bmatrix} + [0\ 1\ \cdots\ 0]\boldsymbol{B}\begin{bmatrix}0\\1\\\cdot\\\cdot\\\cdot\\0\end{bmatrix} + \cdots + [0\ 0\ \cdots\ 1]\boldsymbol{B}\begin{bmatrix}0\\0\\\cdot\\\cdot\\\cdot\\1\end{bmatrix}. \tag{B.5}$$

Denote that $\boldsymbol{\mu}_1^T = [\mu_{11}, \mu_{12}, \cdots, \mu_{1M}]^T$, $\boldsymbol{\mu}_2^T = [\mu_{21}, \mu_{22}, \cdots, \mu_{2M}]^T$, and $\boldsymbol{\mu} = [\mu_1, \mu_2, \cdots, \mu_M]^T$. According to Theorem 2, we have

$$tr\boldsymbol{B} \sim \sigma^2\chi^2_1(\delta_1) + \sigma^2\chi^2_1(\delta_2) + \cdots + \sigma^2\chi^2_1(\delta_M) \tag{B.6}$$

where

$$\delta_m = N_1(\frac{\mu_{1m} - \mu_m}{\sigma})^2 + N_2(\frac{\mu_{2m} - \mu_m}{\sigma})^2, \quad m = 1, \cdots, M. \tag{B.7}$$

In addition, $\chi_1^2(\delta_1), \cdots, \sigma^2\chi_1^2(\delta_M)$ are mutually independent since $\boldsymbol{\Sigma}$ is diagonal. Therefore,

$$tr\boldsymbol{B} \sim \sigma^2\chi_M^2(\delta) \tag{B.8}$$

and

$$\begin{aligned} \delta &= \delta_1 + \delta_2 + \cdots + \delta_M \\ &= N_1 \left\| \frac{\mu_1 - \mu}{\sigma} \right\|^2 + N_2 \left\| \frac{\mu_2 - \mu}{\sigma} \right\|^2 \\ &= \frac{N_1 N_2}{N} \left\| \frac{\mu_1 - \mu_2}{\sigma} \right\|^2 \end{aligned} \tag{B.9}$$

where $\|\cdot\|$ represents the $\mathcal{L}_2$ norm.

By the same reasoning as the above, it is obvious that

$$tr\boldsymbol{E} \sim \sigma^2\chi_{M(N-2)}^2.$$

Hence,

$$\begin{aligned} R_m &= (N-2)\frac{tr[\boldsymbol{W} - (\boldsymbol{W}_1 + \boldsymbol{W}_2)]}{tr(\boldsymbol{W}_1 + \boldsymbol{W}_2)} \\ &= \frac{M^{-1}tr\boldsymbol{B}}{(M(N-2))^{-1}tr\boldsymbol{E}}. \end{aligned}$$

Since $\boldsymbol{E}$ and $\boldsymbol{B}$ are statistically independent according to Theorem 1, we have by the definition of a noncentral F distribution:

$$R_m \sim F_{N,M(N-2)}(\delta)$$

where

$$\delta = \frac{N_1 N_2}{N} \left\| \frac{\mu_1 - \mu_2}{\sigma} \right\|^2.$$

Appendix C

$$\frac{N!}{(cN)!((1-c)N)!} \simeq [c^{-c}(1-c)^{-(1-c)}]^N$$

The Stirling's formula for the Gamma function ([12, Page 70]) states that

$$\Gamma(t+1) \simeq e^{-t}t^{t+\frac{1}{2}}(2\pi)^{\frac{1}{2}}, \quad (t \to \infty).$$

Thus for large N, we have

$$\begin{aligned} \frac{N!}{(cN)!((1-c)N)!} &= \frac{\Gamma(N+1)}{\Gamma(cN+1)\Gamma((1-c)N+1)} \\ &\simeq c^{-(cN+\frac{1}{2})}(1-c)^{-((1-c)N+\frac{1}{2})}N^{-\frac{1}{2}}(2\pi)^{-\frac{1}{2}}. \end{aligned}$$

Then by simple mathematical manipulations, we have

$$\lim_{N\to\infty} \frac{1}{N} \log \frac{N!}{(cN)!((1-c)N)!} = -c\log c - (1-c)\log(1-c).$$

Therefore,

$$\frac{N!}{(cN)!((1-c)N)!} \simeq [c^{-c}(1-c)^{-(1-c)}]^N, \quad (N \to \infty).$$

Appendix D

Multivariate Normality

In this appendix, we introduce a set of empirical distribution function (EDF) statistics, and describe how to test multivariate normality based on the EDF statistics. Then we use Monte Carlo simulations to assess Gaussianity of compressed pre-processed pulses [1] when original pulses are generated from a Gaussian distribution.

D.1 EDF Statistics

Suppose a given random sample of size N is $x_1, x_2, \cdots, x_N$ and let $x_{(1)} < x_{(2)} < \cdots < x_{(N)}$ be the order statistics. Let $F(x)$ denote the distribution function of x, then the empirical distribution function (EDF) $F_N(x)$ is defined by

$$\begin{aligned} F_N(x) &= 0, \qquad x < x_{(1)}; \\ F_N(x) &= \frac{n}{N}, \qquad x_{(n)} \le x < x_{(n+1)}, \quad n = 1, \cdots, N-1; \\ F_N(x) &= 1, \qquad x_{(N)} \le x. \end{aligned} \tag{D.1}$$

A statistic measuring the difference between $F_N(x)$ and $F(x)$ is called an EDF statistic. To test a null hypothesis

$$H_0 : \text{ a random sample } x_1, x_2, \cdots, x_N \text{ comes from a distribution } F(x),$$

[1] The pre-process and compression procedures are presented in Section 5.2.

Statistic	Expression
D^+	$\max_n\{\frac{n}{N} - z_{(n)}\}$
D^-	$\max_n\{z_{(n)} - \frac{n-1}{N}\}$
V	$D^+ + D^-$
W^2	$\sum_n[z_{(n)} - \frac{2n-1}{2N}]^2 + \frac{1}{12N}$
U^2	$W^2 - N(\bar{z} - 0.5)^2$, where $\bar{z} = \frac{1}{N}\sum_n z_{(n)}$.
A^2	$-N - \frac{1}{N}\sum_n(2n-1)[\log_e z_{(n)} + \log_e(1 - z_{(N+1-n)})]$

Table D.1: Six EDF statistics

Statistic	Modified statistic	Significance level α 0.25	0.10	0.05
		Threshold in the upper tail		
D^+	$D^+(\sqrt{N} + 0.12 + \frac{0.11}{\sqrt{N}})$	0.828	1.073	1.224
D^-	$D^-(\sqrt{N} + 0.12 + \frac{0.11}{\sqrt{N}})$	0.828	1.073	1.224
V	$V(sqrtN + 0.155 + \frac{0.24}{\sqrt{N}})$	1.420	1.620	1.747
W^2	$(W^2 - \frac{0.4}{N} + \frac{0.6}{N^2})(1.0 + \frac{1.0}{N})$	0.209	0.347	0.461
U^2	$(U^2 - \frac{0.1}{N} + \frac{0.1}{N^2})(1.0 + \frac{0.8}{N})$	0.105	0.152	0.187
A^2	For all $N \geq 5$	1.248	1.933	2.492

Table D.2: Modified EDF statistics

an EDF test procedure was presented in [20, Section 4.4]:

(a) Put the x_n in ascending order, $x_{(1)} < x_{(2)} < \cdots < x_{(N)}$.

(b) Calculate $z_{(n)} = F(x_{(n)})$, $n = 1, \cdots, N$.

(c) Choose and calculate an appropriate test statistic listed in Table D.1.

(d) Modify the test statistic as in Table D.2. If the modified statistic exceeds the threshold in the upper tail at given level α, H_0 is rejected at significance level α.

In Step (b), by using the probability integral transformation (PIT), $z = F(x)$, the new variable z is uniformly distributed between 0 and 1 when $F(x)$ is the true distribution of x. Hence, the six EDF statistics [55] in Table D.1 are actually designed to test if the new variable z is from a uniform distribution between 0 and 1. In general, D^+ and D^- are powerful in detecting whether or not the z-set tends to be close to 0 or to 1, respectively;

W^2 and A^2 are powerful in detecting a shift of the mean in either direction; V and U^2 are powerful in detecting a change in variance, either a grouping of z values at one point, or a division into two groups near 0 and 1.

D.2 Multivariate Normality Test

The goodness-of-fit assessment is to test the null hypothesis

H_0' : a multivariate sample $\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N$ comes from a multivariate normal distribution.

Let the dimension of $\boldsymbol{y}$ be M, the sample mean be $\hat{\boldsymbol{y}}$ and the sample covariance matrix be $\hat{\boldsymbol{\Sigma}}$, then under the null hypothesis H_0' the values

$$x_n = (\boldsymbol{y}_n - \hat{\boldsymbol{y}})^T \hat{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{y}_n - \hat{\boldsymbol{y}}), \quad n = 1, \cdots, N; \tag{D.2}$$

will have approximately a χ^2 distribution with M degrees of freedom. As suggested in [20, Section 9.7] and [2], instead of directly assessing multivariate normality, we test the following hypothesis:

H_0'' : the values $x_1, x_2, \cdots, x_N$ come from a χ^2_M distribution.

Hence, the EDF test procedure described in Section D.1 can be directly applied to test H_0'', and correspondingly to assess H_0'.

D.3 Gaussianity Test of Compressed Pre-processed Pulses

In this section, we use the multivariate normality test described in the previous section to examine the problem encountered in Section 5.3, i.e., to test whether or not Gaussianity is still maintained after received pulses are pre-processed and compressed.

Here we simulate received pulses by using the covariance matrix $\boldsymbol{\Sigma} = \sigma^2 I_M$, $\sigma = 0.05$. Figs. D.1 and D.2 show the real and imaginary parts of 50 simulated pulses and those of the compressed pre-processed pulses, respectively. In the following, we generate 100 sets of received pulses when $N = 20, 50$ and 100, respectively. To make the test more convincibe,

Original (simulated) pulses	D^+	D^-	V	W^2	U^2	A^2
	Rejection rate based on 100 trials					
$M = 256,\ N = 20$	0/100	2/100	0/100	4/100	0/100	0/100
$M = 256,\ N = 50$	0/100	0/100	1/100	2/100	0/100	0/100
$M = 256,\ N = 100$	0/100	0/100	0/100	0/100	0/100	0/100

Table D.3: Gaussianity test of original (simulated) pulses at significance level 0.05

Compressed pre-processed pulses	D^+	D^-	V	W^2	U^2	A^2
	Rejection rate based on 100 trials					
$M = 44,\ N = 20$	4/100	5/100	6/100	10/100	7/100	4/100
$M = 44,\ N = 50$	2/100	2/100	5/100	6/100	2/100	4/100
$M = 44,\ N = 100$	0/100	3/100	2/100	7/100	3/100	2/100

Table D.4: Gaussianity test of compressed pre-processed pulses at significance level 0.05

all the six EDF statistics introduced in Section D.1 are used. Tables D.3 and D.4 list the EDF test results at significance level 0.05 on original (simulated) pulses and compressed pre-processed pulses, respectively. In both tables, a result like 2/100 means that Gaussianity is rejected 2 times out of 100 trials. From Tables D.3 and D.4, it can be observed at significance level 0.05 that

1. The rejection rates for the compressed pre-processed pulses are slightly higher than those of original (simulated) pulses.

2. The rejection rates for the compressed pre-processed pulses are still smaller than 10/100.

The pre-process and compression procedures presented in Section 5.2 include non-linear operations, Gaussianity of the compressed pre-processed pulses may not be strictly held, as justified by Observation 1. On the other hand, the rejection rates for the compressed pre-processed pulses are still very small ($< 10\%$) from Observation 2; Hence, Gaussianity of the compressed pre-processed pulses can be still assumed for simplicity.

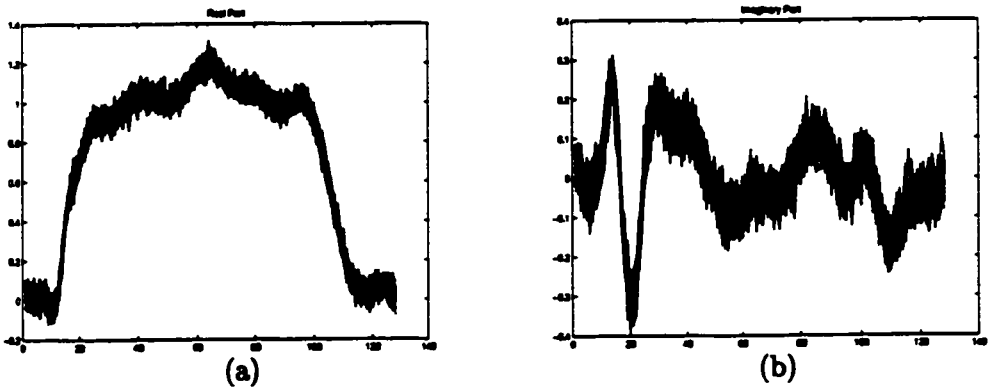


Figure D.1: Real and imaginary parts of 50 simulated pulses, where x-axis is the index of data sample points and y-axis is the amplitude.

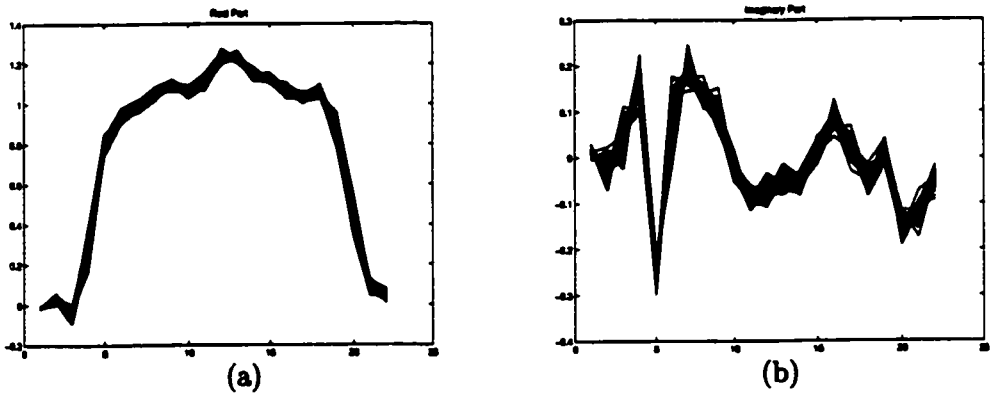


Figure D.2: Real and imaginary parts of the compressed pre-processed pulses,where x-axis is the index of data sample points and y-axis is the amplitude.

Bibliography

[1] M.R. Anderberg, *Cluster analysis for applications*, Academic Press, 1973.

[2] D.F. Andrews, R. Gnanadesikan, and J.L. Warner, *Transformation of multivariate data*, Biometrika **27** (1971), 825–840.

[3] J.D. Banfield and A.E. Raftery, *Automated tracking of ice floes: A statistical approach*, IEEE Tans. on Geoscience And Remote Sensing **29** (1991), 905–911.

[4] ———, *Ice floe identification in satellite images using mathematical morpholopy and clustering about principle curves*, Journal of the American Statistical Association **87** (1992), 7–16.

[5] ———, *Skeletal modeling of ice leads*, IEEE Tans. on Geoscience And Remote Sensing **30** (1992), 918–923.

[6] ———, *Model-based gaussian and non-gaussian clustering*, Biometrics **49** (1993), 803–821.

[7] R.A. Baxter, *Minimum message length inference: Theory and applications*, Ph.D. thesis, Dept. of Computer Science, Monash University, Clayton 3168, Australia, December 1996.

[8] R.A. Baxter and J.J. Oliver, *MDL and MML: similarities and differences*, Technical Report 207, Dept. of Computer Science, Monash University, Vic 3168, Australia, 1994.

[9] A. Berdai and B. Garel, *Detecting a univariate normal mixture with two components*, Statistics & Decisions **14** (1996), 35–51.

[10] D.M. Boulton and C.S. Wallace, *A program for numerical classification*, The Computer Journal **13** (1970), no. 1, 63–69.

[11] ———, *An information measure for hierarchic classification*, The Computer Journal **16** (1973), no. 3, 254–261.

[12] N.G.De. Bruijn, *Asymptotic methods in analysis*, North-Holland Publishing Company, 1970.

[13] P. Bryant, *Large-sample results for optimization based clustering methods*, Journal of classification **8** (1991), 31–44.

[14] P. Bryant and J.A. Williamson, *Aymptotic behaviour of classification maximum likelihood estimates*, Biometirka **65** (1978), 273–281.

[15] G. Celeux and G. Govaert, *Comparison of the mixture and the classification maximum likelihood in cluster analysis*, Journal of Statistical Computation and Simulation **47** (1993), 127–146.

[16] P. Cheeseman, J. Kelly, M. Self, W. Taylor, D. Freeman, and J. Stutz, *AutoClass: A Bayesian Classification*, Proceedings of the Fifth International Conference on Machine Learning (Ann Arbor, MI), 1988, pp. 54–64.

[17] P. Cheeseman, M. Self, J. Kelly, W. Taylor, D. Freeman, and J. Stutz, *Bayesian classification*, Seventh National Conference on Artificial Intelligence (Saint Paul, MN), 1988, pp. 607–611.

[18] J.H. Conway and N.J.A. Sloane, *Sphere packings, lattices and groups*, Springer-Verlag,New York, 1988.

[19] T.M. Cover and J.A. Thomas, *Elements of information theory*, John Wiley & Sons, 1991.

[20] R.B. Dagostino and M.A. Stephens, *Goodness-of-fit techniques*, Marcel Dekker, Inc., 1986.

[21] I. Daubechies, *Ten lectures on wavelets*, SIAM, 1992.

[22] D.L. Dowe, L. Allison, T.I. Dix, L. Hunter, C.S. Wallace, and T. Edgoose, *Circular clustering for protein dihedral angles by minimum message length*, In Proceedings of the 1st Pacifil Symposium on Biocomputing (Hawaii, U.S.A.), 1996, pp. 242–255.

[23] D. M. Hawkins (Editor), Topics in applied multivariate analysis, ch. 6. Cluster analysis, pp. 301–351, Cambridge University Press, 1982, pp. 301–351.

[24] L. Engelman and J.A. Hartigan, *Percentage points of a test for clusters*, J. Amer. Statist. Assoc. **64** (1969), 1647–1648.

[25] K.T. Fang and Y.T. Zhang, *Generalized multivariate analysis*, John Wiley & Sons, 1990.

[26] N.I. Fisher, *Statistical analysis of circular data*, Cambridge University Press, 1993.

[27] C. Fraley, *Algorithms for model-based gaussian hierarchical clustering*, SIAM Journal on Scientific Computing **20** (1998), 270–281.

[28] C. Fraley and A.E. Raftery, *How many clusters? which clustering method? - answers via model-based cluster analysis*, Computer Journal **41** (1998), 578–588.

[29] ———, *MCLUST: Software for model-based cluster analysis*, Technical Report 342, Department of Statistics, University of Washington, November 1998, to appear in Journal of Classification.

[30] S. Ganesalingam, *Classification and mixture approach to clustering via maximum likelihood*, Applied Statistics **38** (1989), 455–466.

[31] J.A. Hartigan, *Clustering algrithms*, John Wiley & Sons, 1975.

[32] ———, *Asymptotic distributions for clustering criteria*, The Annals of Statistics **6** (1978), 117–131.

[33] J.A. Hartigan and M.A. Wong, *A k-means clustering algorithm*, Applied Statistics **28** (1979), 100–108.

[34] N.L. Johnson and S. Kotz, *Distributions in statistics: continuous univariate distributions -2*, John Wiley & Sons, 1970.

[35] R.E. Kass and A.E. Raftery, *Bayes Factors*, Journal of the American Statistical Association **90** (1995), no. 430, 773–795.

[36] S.M. Lewis and A.E. Raftery, *Estimating bayes factors via posterior simulation with the lapace-metropolis estimator*, Technical Report 279, Department of Statistics, University of Washington, 1994.

[37] J. Liu, S.W. Gao, Z.Q. Luo, T.N. Davidson, and J.P.Y. Lee, *The minimum description length criterion applied to emitter number detection and pulse classication*, Proceeding of IEEE workshop on Statistical Signal and Array Processing (Portland, Oregon, USA), 1998.

[38] S. Mallat, *A theory for multi-resolution signal decomposition: the wavelet representation*, IEEE Trans. on PAMI **11** (1989), 674–693.

[39] G.I. McLachlan and K. Basford, *Mixture models: inference and applications to clustering*, Marcel Dekker. New York, 1988.

[40] G.J. Mclachlan, *On bootstapping the likelihood ratio test statistic for the number of components in a normal mixture*, Applied Statistics **36** (1987), 318–324.

[41] N.R. Mendell, S.J. Finch, and H.C. Thode, *Where is the likelihood ratio test powerful for detecting two component normal mixture? (the consultant's forum)*, Biometrics **49** (1993), 907–915.

[42] N.R. Mendell, H.C. Thode, and S.J. Finch, *The likelihood ratio test for the two-component normal mixture problem: power and siampe size analysis*, Biometrics **47** (1991), 1143–1148.

[43] R.J. Muirhead, *Aspects of multivariate statistical theory*, John Wiley & Sons, 1982.

[44] J.J. Oliver and R.A. Baxter, *MML and bayesianism: similarities and differences*, Technical Report 206, Dept. of Computer Science, Monash University, Vic 3168, Australia, 1994.

[45] J.J. Oliver and D.J. Hand, *Introduction to minimum encoding inference*, Technical Report 205, Dept. of Computer Science, Monash University, Vic 3168, Australia, 1994.

[46] J.D. Patrick, *A program for discriminating between classes*, Technical Report 151, Dept. of Computer Science, Monash University, Vic 3168, Australia, 1991.

[47] B. Pfahringer, *Practical uses of the minimum description length principle in inductive learning*, Ph.D. thesis, The Austrian Research Institute for Artificial Intelligence, 1995.

[48] S. Richardson and P.J. Green, *On Bayesian analysis of mixtures with an unknown number of components (with discussion)*, J. Royal Statistical Soc.,Ser. B **59** (1997), no. 4, 731–792.

[49] B.D. Ripley, *Pattern recognition and neural networks*, Cambridge University Press, 1996.

[50] J. Rissanen, *Modeling by shortest data description*, Automatica **14** (1978), 465–471.

[51] ———, *A universal prior for the integers and estimation by MDL*, Ann. of Statistics **11** (1983), no. 3, 416–431.

[52] ———, *Stochastic complexity*, J. R. Statist. Soc. B **49** (1987), no. 3, 223–239.

[53] S.M. Ross, *Introduction to probability models*, fourth ed., Academic Press, Inc., 1989.

[54] G. Schwarz, *Estimating the dimension of a model*, Annals of Statistics **6** (1978), 461–464.

[55] M.A. Stephens, *Use of the kolmogorov-smirnov, cramer-von mises and related statistics without extensive tables*, J. Roy. Statist. Soc., B **32** (1970), 115–122.

[56] D.M. Titterington, *Some recent research in the analysis of mixture distributions*, Statistics **21** (1990), 619–641.

[57] J.C. Toomay, *Radar principles for the non-specialist*, Wadsworth Inc., 1982.

[58] H.L. Van Trees, *Detection, estimation and modulation theory, part i*, New York: Wiley, 1968.

[59] M.A. Upal, *Montel Carlo comparison of non-hierarchical unsupervised classifiers*, Master's thesis, University of Saskatchewan, Saskatchewan, Canada, 1995.

[60] M.A. Upal and E.M. Neufeld, *Comparison of Unsupervised Classifiers*, Proceedings of the ISIS Information, Statistics and Induction in Science (Singapore), World Scientific, 20-23 August 1996, pp. 342–353.

[61] C.S. Wallace, *Classification by Minimum Message Length inference*, Advances in Computing and Information- ICCI 1990 (Niagara Falls) (S.G. Akl et al., eds.), 1990, pp. 72–81.

[62] C.S. Wallace and D.M. Boulton, *An information measure for classification*, Computer Journal **11** (1968), no. 2, 195–209.

[63] C.S. Wallace and D.L. Dowe, *Intrinsic classification by MML - the Snob program*, Proceedings of the 7th Australian Joint Conference on Artificial Intelligence (Singapore), World Scientific, 1994, pp. 37–44.

[64] C.S. Wallace and P.R. Freeman, *Estimation and inference by compact coding*, J. R. Statist. Soc B **49** (1987), no. 3, 240–265.

[65] M. Wax and T. Kailath, *Detection of signals by information theoretic criteria*, IEEE Trans. on ASSP **33** (1985), no. 2, 387–392.

[66] J.H. Wolfe, *Pattern clustering by multivariate mixture analysis*, Multivariate Behavioural Research **5** (1970), 329–350.

[67] K.M. Wong and Z.Q. Luo, *Emitter number detection and pulse classification in radar systems*, Tech. Report 356, Communications Research Labotary, McMaster University, Hamilton, Ontario, Canada, 1997.

[68] K.M. Wong, Z.Q. Luo, J. Liu, J.P.Y. Lee, and S.W. Gao, *Radar emitter classification using intrapulse data*, International Journal of Electronics and Communications, Germany (1999).

[69] J.K. Wu, *Neutral network and simulated methods*, New York: M. Dekker, 1994.

[70] J. Zhang and J.W. Modestino, *A model-fitting approach to cluster validation with application to stochastic model-based image segmentation*, IEEE Trans. on PAMI **12** (1990), no. 10, 1009–1017.

[71] Q.T. Zhang, K.M. Wong, P.C. Yip, and J.P. Reilly, *Statistical analysis of the performance of information theoretic criteria in the detection of the number of signals in array processing*, IEEE Trans. on ASSP **37** (1989), no. 10, 1557–1567.

[72] L.C. Zhao, P.R. Krishnaiah, and Z.D. Bai, *On detection of the number of signals in presence of white noise*, Journal of multivariate analysis **20** (1986), 1–25.

[73] *DECCA Ship's Manual: Groups 9A and 12A Relative Motion Marine Radars*, RACAL-DECCA Marine Radar Limited, 1975.

[74] *TMS320C3x/C4x code generation tools getting started guide*, Texas Instruments Inc., 1997.

[75] *TMS320C3x/C4x optimizing c compiler user's guide*, Texas Instruments Inc., 1997.

[76] *TMS320C3x/C4x assembly language tools user's guide*, Texas Instruments Inc., 1997.

[77] *TMS320C4x user's guide*, Texas Instruments Inc., 1997.

[78] *Code composer user's guide*, GO DSP Corporation, 1997.

[79] *Dakar F5 carrier board technical reference*, Spectrum Signal Processing Inc., 1997.

[80] *Dakar F5 carrier board user's guide*, Spectrum Signal Processing Inc., 1997.