Mixture Codebook Classification
Part 1: Method Outline

Mette Langaas
Department of Mathematical Sciences
The Norwegian Institute of Technology
The University of Trondheim
mette@imf.unit.no

Abstract

The term codebook is used in the field of vector quantization and neural networks, where a codebook is placed into the sample space to approximate various domains of an input vector by quantized values. Finite mixture models can be used to represent distributions, in particular class-conditional distributions. The term mixture codebook is an extension of the term codebook to include the parameters and weights of finite class-conditional mixture models. The method of mixture codebook classification (MC^2) consists in estimating a mixture codebook and using it to construct a classification rule. A small empirical study of mixture codebook classification is presented.

Some keywords: Finite Mixture Model; Alternating Likelihood Estimation; EM Algorithm; Supervised Classification; Discriminant Analysis.

Contents

1 Introduction

Linear discriminant analysis (LDA) is a popular tool for discrimination and classification. The method has high interpretability and it is optimal with respect to minimizing the Bayes decision risk when the class-conditional distributions are Gaussian distributions with equal covariance matrices and equal prior probabilities for the classes. LDA produces linear boundaries in the sample space. Despite the advantages of LDA in many applications it is important to be able to produce non-linear class-boundaries. Methods that can produce general non-linear class boundaries are flexible. Among these are the static neural networks, see for example Hertz et al. (1991) and Hush and Horne (1993), multivariate regression methods, among these projection pursuit regression of Friedman and Stuetzle (1981) and multivariate adaptive regression splines of Friedman (1991), flexible discriminant analysis of Hastie et al. (1994) and space partitioning methods, among these classification and regression trees of Breiman et al. (1984).

The main idea of this report is to combine the simplicity and powers of the LDA with the flexibility of producing non-linear class-boundaries. To achieve this the class-conditional distributions are represented by finite mixture models to produce a *mixture codebook*. The mixture codebook is placed in the sample space to minimize a chosen criterion. Different criteria are presented. A classification rule is constructed based on the mixture codebook, producing the method we call *mixture codebook classification* (MC^2). Among other methods for discrimination and classification linear discriminant analysis can be regarded as a special case of the MC^2 method.

The report is organized as follows. In section 2.1 the problem of classification is outlined. In section 2.2 the term codebook is explained. Distributions are represented by finite mixture models to produce a mixture codebook in section 3. Methods for creating and using a mixture codebook with the objective to discriminate and classify, the main topic of this report, are presented in section 4. In section 5 the methods for mixture codebook classification are used on one synthetic and one real data set. Conclusions and comments are presented in section 6.

2 Classification and Codebooks

2.1 Classification

An observation is a feature vector $\boldsymbol{X} = \boldsymbol{x}$ from some space $\mathcal{X}$, $\boldsymbol{x} \in \mathcal{X}$, the *sample space*. The true class membership is $g \in \mathcal{G} = \{g_1, g_2, ..., g_M\}$, M being the total number of classes (M is finite). We assume that pairs $(\boldsymbol{X}, g)$ are drawn independently from the joint probability distribution, $p(\boldsymbol{x}, g)$, on $\mathcal{X} \times \mathcal{G}$.

Given an observation (vector of features), $\boldsymbol{X} = \boldsymbol{x}$, we want to make a decision regarding the true class of the observation. This is our *classification problem*; predict the true class g for an observation $\boldsymbol{X} = \boldsymbol{x}$.

The Bayes decision rule with zero-one loss function, see for example Duda and Hart (1973), classifies an observation to the class with the maximum posterior probability. Let g_i denote a general class. The posterior probability for class g_i conditional on the observation $\boldsymbol{x}$ is given by

$$p(g_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|g_i)p(g_i)}{p(\boldsymbol{x})} \quad \textit{where} \quad p(\boldsymbol{x}) = \sum_{l=1}^{M} p(\boldsymbol{x}|g_l)p(g_l)$$

where $p(\boldsymbol{x}|g_i)$ is the class-conditional probability and $p(g_i)$ is the prior probability for class g_i. In the nonparametric density estimation approach to discriminant analysis the class-conditional distributions are estimated using nonparametric density estimation based on a training set of classified observations. Nonparametric density estimation can be performed using a non-negative kernel, see for example Tapia and Thompson (1978) or Silverman (1986). The method of kernel discriminant analysis estimates the class-conditional distributions using kernel density estimation, see for example Hand (1982) or chapter 9 of McLachlan (1992). This is also the idea of probabilistic neural networks of Specht (1990).

Instead of estimating the class-conditional distributions using nonparametric density estimation the class-conditional distributions can be represented by a parametrical model. This can be done using finite mixture distributions with a moderate number of mixture components, which is the building block for the mixture codebook classification method. The use of finite mixture distributions allows us to model multi-modal class-conditional distributions. The underlying component densities might or might not have a direct physical interpretation. Kernel based density estimation may be interpreted as equal-weights mixture densities where the number of mixture components equals the number of observations, see section 2.2 of Titterington et al. (1985)

2.2 What is a Codebook ?

The term codebook is used in the field of vector quantization (VQ), see Gersho and Gray (1992) and neural networks, see Kohonen (1990).

A vector quantizer, Q, is a mapping from the sample space $\mathcal{X}$ (usually $\mathbb{R}^d$) to the space of codewords $\mathcal{C}$, $\mathcal{C} \subset \mathcal{X}$.

$$Q : \mathcal{X} \to \mathcal{C}$$

This is a partition of the sample space into a finite number of disjoint quantized cells. The set of codewords is called a *codebook*. The codebook is placed into the sample space to approximate various domains of the input vector by quantized values. The performance of a vector quantizer is measured by the mean distortion (also called the average distortion), $D = E(d(\boldsymbol{X}, \boldsymbol{X}_q))$ between the sample vector $\boldsymbol{X}$ and its quantized value $\boldsymbol{X}_q$, where $d(\cdot, \cdot)$ is a chosen distance measure. Vector quantization is used for digital signal processing, speech recognition and compression and image compression, see Gersho and Gray (1992).

The term codebook can also be viewed from the field of clustering (unsupervised learning). Clustering is understood to be the grouping of similar objects and separation of dissimilar ones. To introduce the term codebook each of the clusters can be represented by a codeword. The codewords together form the codebook. The codeword is the centre of the cluster. To find clusters there exists well established techniques, see for example Hartigan (1975), Anderberg (1973) and Panel on Discriminant Analysis, Classification and Clustering (1989).

The term mixture codebook is an extension of the term codebook to include the parameters and weights of a finite mixture model. This is described further in section 3 and section 4.

Notation

The training set, $\mathcal{T}$, consists of n independent realizations of the random vector $\boldsymbol{X}$, together with the true class membership, $\mathcal{T} = (\boldsymbol{x}_j, g(\boldsymbol{x}_j))_{j=1}^n$. The dimension of $\boldsymbol{X}$ is d.

The following notation is adopted; the ith class (group) is denoted g_i, the lth component is denoted c_l and the lth component within the ith class is denoted c_{il}.

The term *density* is used for both continuous and discrete random variables.

3 Finite Mixture Models

This section gives an introduction to some aspects of finite mixture models. We will use the finite mixture models to represent class-conditional distributions. This is not a new idea and has been mentioned in the context of discrimination and classification by different authors, among others McLachlan (1992) chapter 2.11, Taxt et al. (1991), Lippmann (1994) and Streit and Luginbuhl (1994). The use of finite mixtures to model class-conditional distributions is also closely related to radial basis function networks, as reviewed in Xu et al. (1994). We will return to these references in section 4 when we look at constructing classification rules based on the class-conditional finite mixtures. The problem of constructing classification rules is not addressed in this section.

3.1 One and Two Level Mixture Models

We will in this section define one and two level mixtures and the mixture codebook, $\mathcal{M}$. General references to mixture models are Everitt and Hand (1981), McLachlan and Basford (1988) and Titterington et al. (1985).

3.1.1 One Level Mixture Models

Assume that $\boldsymbol{X}$ has a finite mixture distribution defined as

$$p(\boldsymbol{x}) = \sum_{l=1}^{K} p(c_l) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l) \tag{1}$$

with *mixing weights* $p(c_1), ..., p(c_K)$ and *component densities* $f_1(\boldsymbol{x}|\boldsymbol{\theta}_1), ..., f_K(\boldsymbol{x}|\boldsymbol{\theta}_K)$. The mixing weights are assumed to be non-negative and sum to one. Here $p(\boldsymbol{x})$ is written as a *one level mixture model*. We assume that each f_l is known up to its parameter vector $\boldsymbol{\theta}_l$. It is necessary to impose restrictions on the parameters and mixing weights in the mixture model to avoid problems of identifiability. That is, we may exchange $\boldsymbol{\theta}_1$ with $\boldsymbol{\theta}_2$ and rename $p(c_1)$ as $p(c_2)$ to get two representations of the same density. This problem is known as "label-switching", see Redner and Walker (1984).

A *mixture codebook* for the one level mixture is denoted $\mathcal{M} = \{\boldsymbol{\Theta}, f, K\}$ where $\boldsymbol{\Theta} = \{(\boldsymbol{\theta}_l, p(c_l)); l = 1, ..., K\}$ and $f = \{(f_l); l = 1, .., K\}$.

A one level mixture model can be used as the basic building block for constructing more complex mixture models, and can be used to represent a marginal density, $p(\boldsymbol{x})$, a class-conditional density, $p(\boldsymbol{x}|g_i)$, or a posterior class density, $p(g_i|\boldsymbol{x})$.

3.1.2 Two Level Mixture Models

The basic mixture model building block is the one level mixture model. A marginal distribution is called a *two level mixture* when the class-conditional distributions are defined to be one level mixtures. This distinction between a one level and a two level mixture is made strictly for estimative reasons. This makes possible the fact that parameters can be common for more than one class. A further division is made into what we call class specific component distributions and common component distributions for each class. This division distinguishes between components confined to each class and components overlapping the classes and is also made for estimative reasons.

Common Components

Assume that $\boldsymbol{X}$ has a finite class-conditional mixture distribution defined for each class, but that the mixture component distributions are common for all classes. Let g_i denote a general class.

$$p(\boldsymbol{x}|g_i) = \sum_{l=1}^{K} p(c_l|g_i) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l) \qquad for \;\; i = 1, ..., M \tag{2}$$

$$p(\boldsymbol{x}) = \sum_{i=1}^{M} p(g_i) \sum_{l=1}^{K} p(c_l|g_i) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l) \tag{3}$$

Here we have *class weights* $p(g_i)$, *mixing weights* $p(c_l|g_i)$ and *component densities* $f_l(\boldsymbol{x}|\boldsymbol{\theta}_l)$ for $i = 1, ..., M$ and $l = 1, ..., K$. The class weights and mixing weights are assumed to be non-negative. Equation (3) shows a *two level mixture model* where each component density is common to all classes.

It is also possible to regard the marginal density (3) as a one level mixture,

$$p(\boldsymbol{x}) = \sum_{l=1}^{K} (\sum_{i=1}^{M} p(g_i)p(c_l|g_i)) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l) = \sum_{l=1}^{K} (\sum_{i=1}^{M} p(g_i, c_l)) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l) = \sum_{l=1}^{K} p(c_l) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l).$$

A *mixture codebook* for this two level mixture is denoted $\mathcal{M} = \{\boldsymbol{\Theta}, f, K\}$ where $\boldsymbol{\Theta} = \{(\boldsymbol{\theta}_l, p(c_l|g_i), p(g_i)); i = 1, ..., M; l = 1, ..., K\}$ and $f = \{(f_l); l = 1, ..., K\}$.

Class Specific Components

Assume that $\boldsymbol{X}$ has a finite class-conditional mixture distribution defined for each class. Let g_i denote a general class.

$$p(\boldsymbol{x}|g_i) = \sum_{l=1}^{k_i} p(c_{il}|g_i) f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il}) \qquad for \;\; i = 1, ..., M \tag{4}$$

$$p(\boldsymbol{x}) = \sum_{i=1}^{M} p(g_i) \sum_{l=1}^{k_i} p(c_{il}|g_i) f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il}) \tag{5}$$

As before we have; *class weights* $p(g_i)$, *mixing weights* $p(c_{il}|g_i)$ and *component densities* $f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})$ for $i = 1, ..., M$ and $l = 1, ..., k_i$. Equation (5) shows a *two level mixture model* where each component density is specific for each class.

A *mixture codebook* for this two level mixture is denoted $\mathcal{M} = \{\boldsymbol{\Theta}, f, K\}$ where K is a vector giving the number of component distributions inside each class, $K = (k_1, k_2, ..., k_M)$ and $\boldsymbol{\Theta} = \{(\boldsymbol{\theta}_{il}, p(c_l|g_i), p(g_i)); i = 1, ..., M; l = 1, ..., k_i\}$ and $f = \{(f_{il}); i = 1, .., M; l = 1, .., k_i\}$.

The case of class specific components can be found as a special case of common components where some mixing weights are equal to zero.

Since the two level mixture models can be seen as special cases of the one level mixture model, conclusions about estimation in a one level mixture is valid also for the two level mixture models. We will in the following regard the number of component, the component architecture K, as known. The problem of choosing K is not addressed in this report.

3.2 Estimating the Parameters and Weights of a Mixture

The problem of estimating the parameters and weight of a one level mixture is one of the oldest estimation problems in the statistical literature, see Everitt and Hand (1981) and McLachlan and Basford (1988) for an historical account. Estimation is difficult even for a one level mixture of two univariate Gaussian component distributions, where the likelihood surface is littered with singularities. Karl Pearson used in 1894 the method of moments to estimate the five parameters and weights in a mixture of two univariate Gaussian component distributions with equal variances. The numerical solution involved finding the real roots of a ninth-degree polynomial.

In 1948 C.R. Rao suggested maximum likelihood estimation in the same situation, based on Fisher's method of scoring, Rao (1948). Finite mixture distributions do not belong to the exponential family. The loglikelihood can be unbounded and there are often multiple stationary points and singularities. Such an unbounded global maximum occur when the component distributions are (allowed to be) degenerated around observed points. Maximum likelihood estimates, $\boldsymbol{\Theta}^*$, are usually defined as the choice of $\boldsymbol{\Theta}$ (in the parameter space Ω) at which the loglikelihood attains its global maximum. Because of the boundary problems, the usual asymptotic arguments about the maximum likelihood estimates do not apply. When the loglikelihood is bounded, as in the case of two Gaussian components with equal covariance matrices, we will want to find its global maximum. But when the loglikelihood is unbounded, as for two Gaussian components with specific covariance matrices, we will search for the largest local maximum in the interior of the parameter space. Redner and Walker (1984) try to avoid this problem by defining the maximum likelihood estimate to be a point in parameter space where the loglikelihood attains its largest local maximum.

The traditional general approach to determining a maximum likelihood estimate is to arrive at a system of likelihood equations and try to obtain a maximum likelihood estimate by solving the likelihood equations. As we will see in the preceding sections, the likelihood equations in the mixture situation can not be solved explicitly. This raises the need for iterative optimization techniques. Conventional numerical optimization techniques can be used to find a local maximum of the loglikelihood. Popular methods for optimization are, among others, gradient descent methods, Newton's method, Fisher's method of scoring, various quasi-Newton methods, conjugate gradient methods, simulated annealing and downhill simplex. See Gill et al. (1981) for an introduction to practical optimization. Other approaches to mixture estimation also exists. Bernardo and Girón (1988) give a Bayesian treatment of finite mixture models. Diebolt and Robert (1994) consider estimation of finite mixture distributions through Bayesian sampling and Crawford (1994) uses the Laplace method.

We will concentrate on estimating the parameters and weights of mixture distributions based on the likelihood equations using an alternating likelihood estimation procedure, which is equivalent to using the Expectation Maximization (EM) algorithm of Dempster et al. (1977). Likelihood equations are presented for general component densities in section 3.3 and for Gaussian component densities in section 3.4. The alternating likelihood estimation (EM algorithm) based on these likelihood equations is treated in section 3.5.

3.3 Likelihood Equations for General Component Distributions

3.3.1 One Level Mixtures

The one level mixture is presented in section 3.1.1. We want to estimate the parameters, $\boldsymbol{\theta}_l$, and the mixing weights, $p(c_l)$, for $l = 1, ..., K$ based on a training set $\mathcal{T} = (\boldsymbol{x}_j)_{j=1}^n$ of n independent observations from the density $p(\boldsymbol{x})$. Then the loglikelihood can be written,

$$\log(\prod_{j=1}^{n} p(\boldsymbol{x}_j)) = \sum_{j=1}^{n} \log(\sum_{l=1}^{K} p(c_l) f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l)).$$

The loglikelihood is maximized with respect to the parameters and weights under the constraint that the mixing weights sum to one, $\sum_{l=1}^{K} p(c_l) = 1$. The parameter vector, $\boldsymbol{\theta}_l$, can be decomposed into $\boldsymbol{\theta}_l = (\boldsymbol{\phi}_l, \boldsymbol{\phi})$, where $\boldsymbol{\phi}_l$ denotes parameters specific to component density f_l and $\boldsymbol{\phi}$ denotes parameters common to all component densities. For Gaussian mixtures with different covariance matrices for each component density $\boldsymbol{\phi}_l = (\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$ and $\boldsymbol{\phi} = \emptyset$. A mixture of Gaussian component densities with common covariance matrices for all component distributions has $\boldsymbol{\phi}_l = \boldsymbol{\mu}_l$ and $\boldsymbol{\phi} = \boldsymbol{\Sigma}$.

Assuming f_l to be differentiable with respect to $\boldsymbol{\theta}_l$ the likelihood equations for the mixing weights and the parameters are obtained,

$$\sum_{j=1}^{n} p(c_l|\boldsymbol{x}_j) \frac{\partial}{\partial \boldsymbol{\phi}_l} \log f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l) = 0 \qquad for \quad l = 1, ..., K \tag{6}$$

$$\sum_{j=1}^{n} \sum_{l=1}^{K} p(c_l|\boldsymbol{x}_j) \frac{\partial}{\partial \boldsymbol{\phi}} \log f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l) = 0 \tag{7}$$

$$p(c_l) = \frac{1}{n} \sum_{j=1}^{n} p(c_l|\boldsymbol{x}_j) \tag{8}$$

where

$$p(c_l|\boldsymbol{x}_j) = \frac{p(c_l) f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l)}{\sum_{i=1}^{K} p(c_i) f_i(\boldsymbol{x}_j|\boldsymbol{\theta}_i)}. \tag{9}$$

Notice that the quantity $p(c_l|\boldsymbol{x}_j)$ depends on the parameters and weights of the mixture.

3.3.2 Two Level Mixtures with Common Components

We continue with the mixture model described in equation (3).

Let us assume that each component density f_l is known up to its parameter vector $\boldsymbol{\theta}_l$. The parameter vector $\boldsymbol{\theta}_l$ can be decomposed into $\boldsymbol{\theta}_l = (\boldsymbol{\phi}_l, \boldsymbol{\phi})$, where $\boldsymbol{\phi}_l$ denotes parameters specific to component density f_l and $\boldsymbol{\phi}$ denotes parameters common to all component densities. Two examples illustrating this decomposition is the mixture of Gaussian component densities with specific covariance matrices for all component densities $(\boldsymbol{\phi}_l = (\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l), \boldsymbol{\phi} = \emptyset)$ and the mixture of Gaussian component densities with equal covariance matrices for all component distributions $(\boldsymbol{\phi}_l = \boldsymbol{\mu}_l, \boldsymbol{\phi} = \boldsymbol{\Sigma})$.

With unknown class membership, the likelihood equations in section 3.1.1 apply. Here we consider the situation where the class membership is known. The training set consist of n independent observations $\boldsymbol{x}_j$ together with the encoded class membership $\boldsymbol{z}_j$; $\mathcal{T} = (\boldsymbol{y}_j)_{j=1}^n = (\boldsymbol{x}_j, \boldsymbol{z}_j)_{j=1}^n$ where each $\boldsymbol{z}_j$ is an indicator variable encoding of the classes, a vector of length M with one in the position corresponding to the appropriate class and zeros elsewhere. The ith element of $\boldsymbol{z}_j$ is z_{ji}.

$$z_{ji} = \begin{cases} 1 & \text{if } \boldsymbol{x}_j \in g_i \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

The loglikelihood for this situation can be written,

$$\log(\prod_{j=1}^{n} p(\boldsymbol{y}_j)) = \log(\prod_{j=1}^{n} p(\boldsymbol{x}_j, \boldsymbol{z}_j)) = \sum_{j=1}^{n} \sum_{i=1}^{M} (z_{ji} \log p(g_i) + z_{ji} \log(\sum_{l=1}^{K} p(c_l|g_i) f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l))).$$

The loglikelihood is maximized with respect to the parameters and weights under the constraints that the class weights and the mixing weights sum to one, $\sum_{i=1}^{M} p(g_i) = 1$ and $\sum_{l=1}^{K} p(c_l|g_i) = 1$ for $i = 1, ..., M$.

Assuming the component densities f_l to be differentiable with respect to $\boldsymbol{\theta}_l = (\boldsymbol{\phi}_l, \boldsymbol{\phi})$ results in the following likelihood equations,

$$\sum_{j=1}^{n}\sum_{i=1}^{M} z_{ji} p(c_l|g_i, \boldsymbol{x}_j) \frac{\partial \log f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l)}{\partial \boldsymbol{\phi}_l} = 0 \qquad for \quad l = 1, ..., K \tag{11}$$

$$\sum_{j=1}^{n}\sum_{i=1}^{M} z_{ji} \sum_{l=1}^{K} p(c_l|g_i, \boldsymbol{x}_j) \frac{\partial \log f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l)}{\partial \boldsymbol{\phi}} = 0 \tag{12}$$

$$p(g_i) = \frac{n_i}{n} \qquad for \quad i = 1, ..., M \tag{13}$$

$$p(c_l|g_i) = \frac{1}{n_i} \sum_{j=1}^{n} z_{ji} p(c_l|g_i, \boldsymbol{x}_j) \qquad for \quad i = 1, ..., M \quad and \quad l = 1, ..., K \tag{14}$$

where

$$p(c_l|g_i, \boldsymbol{x}_j) = \frac{p(c_l|g_i) f_l(\boldsymbol{x}_j|\boldsymbol{\theta}_l)}{p(\boldsymbol{x}_j|g_i)}. \tag{15}$$

Here the quantity $p(c_l|g_i, \boldsymbol{x}_j)$ depends on the parameters and weights of the mixture model which clearly complicates the estimation.

3.3.3 Two Level Mixtures with Class Specific Components

We continue with the mixture model described in equation (5). Let us assume that each component density f_{il} is known up to its parameter vector $\boldsymbol{\theta}_{il}$. In addition there are sum to one constraints for the class weights and the mixing weights,

$$\sum_{i=1}^{M} p(g_i) = 1 \quad and \qquad \sum_{l=1}^{k_i} p(c_{il}|g_i) = 1 \quad for \quad i = 1, ..., M.$$

The unknown parameters and weights are $\boldsymbol{\Theta} = \{(\boldsymbol{\theta}_{il}, p(g_i), p(c_{il}|g_i))$ for $i = 1, ..., M$ and $l = 1, ..., k_i\}$.

The parameter vector $\boldsymbol{\theta}_{il}$ can be decomposed into $\boldsymbol{\theta}_{il} = (\boldsymbol{\phi}_{il}, \boldsymbol{\phi}_i, \boldsymbol{\phi})$, where $\boldsymbol{\phi}_{il}$ denotes parameters specific to component density f_{il}, $\boldsymbol{\phi}_i$ denotes parameters common for component densities $f_{i1}, ..., f_{ik_i}$ and $\boldsymbol{\phi}$ denotes parameters common to all component densities. Two examples illustrating this decomposition is the mixture of Gaussian component densities with equal covariance matrices for the component densities inside each class ($\boldsymbol{\phi}_{il} = \boldsymbol{\mu}_{il}, \boldsymbol{\phi}_i = \boldsymbol{\Sigma}_i, \boldsymbol{\phi} = \emptyset$) and the mixture of Gaussian component densities with equal covariance matrices for all component densities ($\boldsymbol{\phi}_{il} = \boldsymbol{\mu}_{il}, \boldsymbol{\phi}_i = \emptyset, \boldsymbol{\phi} = \boldsymbol{\Sigma}$).

In classification we are presented with a training set, $\mathcal{T}$, with observations and their true class membership, $\mathcal{T} = (\boldsymbol{y}_j)_{j=1}^{n} = (\boldsymbol{x}_j, \boldsymbol{z}_j)_{j=1}^{n}$. The indicator variable encoding $\boldsymbol{z}_j$ is given in equation (10).

The loglikelihood for this situation can be written,

$$\log(\prod_{j=1}^{n} p(\boldsymbol{y}_j)) = \log(\prod_{j=1}^{n} p(\boldsymbol{x}_j, \boldsymbol{z}_j)) = \sum_{j=1}^{n}\sum_{i=1}^{M}(z_{ji}\log p(g_i) + z_{ji}\log(\sum_{l=1}^{k_i} p(c_{il}|g_i) f_{il}(\boldsymbol{x}_j|\boldsymbol{\theta}_{il}))).$$

The loglikelihood is maximized with the constraints that the class weights and the mixing weights sum to one, $\sum_{i=1}^{M} p(g_i) = 1$ and $\sum_{l=1}^{k_i} p(c_{il}|g_i) = 1$ for $i = 1, ..., M$.

Again assuming f_{il} to be differentiable with respect to $\boldsymbol{\theta}_{il} = (\boldsymbol{\phi}_{il}, \boldsymbol{\phi}_i, \boldsymbol{\phi})$ results in the following likelihood equations,

$$\sum_{j=1}^{n} z_{ji} p(c_{il}|g_i, \boldsymbol{x}_j) \frac{\partial \log f_{il}(\boldsymbol{x}_j|\boldsymbol{\theta}_{il})}{\partial \boldsymbol{\phi}_{il}} = 0 \qquad for \quad i = 1, ..., M \quad and \quad l = 1, ..., k_i \tag{16}$$

$$\sum_{j=1}^{n} z_{ji} \sum_{l=1}^{k_i} p(c_{il}|g_i, \boldsymbol{x}_j) \frac{\partial \log f_{il}(\boldsymbol{x}_j|\boldsymbol{\theta}_{il})}{\partial \boldsymbol{\phi}_i} = 0 \qquad for \quad i = 1, ..., M \tag{17}$$

$$\sum_{j=1}^{n}\sum_{i=1}^{M} z_{ji} \sum_{l=1}^{k_i} p(c_{il}|g_i, \boldsymbol{x}_j) \frac{\partial \log f_{il}(\boldsymbol{x}_j|\boldsymbol{\theta}_{il})}{\partial \boldsymbol{\phi}} = 0 \tag{18}$$

$$p(g_i) = \frac{n_i}{n} \qquad for \quad i = 1, ..., M \tag{19}$$

$$p(c_{il}|g_i) = \frac{1}{n_i} \sum_{j=1}^{n} z_{ji} p(c_{il}|g_i, \boldsymbol{x}_j) \qquad for \quad i = 1, ..., M \quad and \quad l = 1, ..., k_i \tag{20}$$

where

$$p(c_{il}|g_i, \boldsymbol{x}_j) = \frac{p(c_{il}|g_i) f_{il}(\boldsymbol{x}_j|\boldsymbol{\theta}_{il})}{p(\boldsymbol{x}_j|g_i)}. \tag{21}$$

As for the two previous situations, $p(c_{il}|g_i, \boldsymbol{x}_j)$ is dependent on the parameters and weights of the mixture model.

3.4 Likelihood Equations for Gaussian Mixtures

We here rewrite the likelihood equations for the case of Gaussian component distributions.

3.4.1 One Level Gaussian Mixtures

In section 3.3.1 the likelihood equations for a one level mixture are outlined. For a one level Gaussian mixture the likelihood equations can be written more explicitly. The equations (8) and (9) remain unchanged.

1. $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$; in the case of separate covariance matrices for each component density we have $\boldsymbol{\phi}_l = (\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$ and $\boldsymbol{\phi} = \emptyset$.

$$\log f_l(\boldsymbol{x}_j|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) = \log \frac{|\boldsymbol{\Sigma}_l^{-1}|^{\frac{1}{2}}}{(2\pi)^{\frac{d}{2}}} - \frac{1}{2}(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T \boldsymbol{\Sigma}_l^{-1} (\boldsymbol{x}_j - \boldsymbol{\mu}_l) \qquad for \quad l = 1, ..., K$$

Let $\boldsymbol{\mu}_l(p)$ denote the pth element of the vector $\boldsymbol{\mu}_l$, $\boldsymbol{\Sigma}_l(p,q)$ denote element (p,q) of the covariance matrix $\boldsymbol{\Sigma}_l$ and $I\boldsymbol{\Sigma}_l(p,q)$ denote element (p,q) of the the inverse covariance matrix $\boldsymbol{\Sigma}_l^{-1}$ (consistent with the notation in Duda and Hart (1973)),

$$\frac{\partial \log f_l(\boldsymbol{x}_j|\boldsymbol{\mu}_l,\boldsymbol{\Sigma}_l)}{\partial \boldsymbol{\mu}_l} = \boldsymbol{\Sigma}_l^{-1}(\boldsymbol{x}_j - \boldsymbol{\mu}_l) \qquad for \quad l = 1, ..., K$$

$$\frac{\partial \log f_l(\boldsymbol{x}_j|\boldsymbol{\mu}_l,\boldsymbol{\Sigma}_l)}{\partial I\boldsymbol{\Sigma}_l(p,q)} = (1-\frac{\delta_{pq}}{2})[\boldsymbol{\Sigma}_l(p,q)-(\boldsymbol{x}_j(p)-\boldsymbol{\mu}_l(p))(\boldsymbol{x}_j(q)-\boldsymbol{\mu}_l(q))] \quad for \quad l = 1, ..., K$$

where δ_{pq} is the Kronecker delta function. [1] Using equation (6) for $\boldsymbol{\mu}_l$ and $\boldsymbol{\Sigma}_l$ and solving for $\boldsymbol{\mu}_l$ and $\boldsymbol{\Sigma}_l$ leads to the following likelihood equations.

$$\boldsymbol{\mu}_l = \frac{\sum_{j=1}^n p(c_l|\boldsymbol{x}_j)\boldsymbol{x}_j}{\sum_{j=1}^n p(c_l|\boldsymbol{x}_j)} \qquad for \quad l = 1, ..., K \tag{22}$$

$$\boldsymbol{\Sigma}_l = \frac{\sum_{j=1}^n p(c_l|\boldsymbol{x}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T}{\sum_{j=1}^n p(c_l|\boldsymbol{x}_j)} \qquad for \quad l = 1, ..., K \tag{23}$$

These are together with the equations (8) and (9) the likelihood equations for this situation.

2. $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$; that is, $\boldsymbol{\phi}_l = \boldsymbol{\mu}_l$ and $\boldsymbol{\phi} = \boldsymbol{\Sigma}$. Equation (22) is unchanged. With equal covariance matrices for all component densities, equation (23) is replaced by

$$\boldsymbol{\Sigma} = \frac{1}{n}\sum_{j=1}^{n}\sum_{l=1}^{K} p(c_l|\boldsymbol{x}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T. \tag{24}$$

3.4.2 Two Level Mixtures with Common Gaussian Components

In section 3.3.2 the likelihood equations for this situation are outlined. For the special case that the component densities are Gaussian densities the equations can be written more explicitly. The likelihood equations (13) and (14) for the class weights and mixing weights remain unchanged. From equations (11) and (12) we obtain the likelihood equations inserting the derivative of the log-density with respect to the parameters.

1. $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$; we start with the case of separate covariance matrices for each component density, that is, $\boldsymbol{\phi}_l = (\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$ and $\boldsymbol{\phi} = \emptyset$. Using equation (11) for $\boldsymbol{\mu}_l$ and $\boldsymbol{\Sigma}_l$ and solving for $\boldsymbol{\mu}_l$ and $\boldsymbol{\Sigma}_l$ leads to the following likelihood equations,

$$\boldsymbol{\mu}_l = \frac{\sum_{j=1}^n \sum_{i=1}^M z_{ji} p(c_l|g_i,\boldsymbol{x}_j)\boldsymbol{x}_j}{\sum_{j=1}^n \sum_{i=1}^M z_{ji} p(c_l|g_i,\boldsymbol{x}_j)} \qquad for \quad l = 1, ..., K$$

$$\boldsymbol{\Sigma}_l = \frac{\sum_{j=1}^n \sum_{i=1}^M z_{ji} p(c_l|g_i,\boldsymbol{x}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T}{\sum_{j=1}^n \sum_{i=1}^M z_{ji} p(c_l|g_i,\boldsymbol{x}_j)} \qquad for \quad l = 1, ..., K.$$

[1]The derivation is straightforward if one makes use of the formula for the derivative of a determinant of a symmetric matrix and the relationship between the inverse and the determinant of a matrix. The formulas can be found in Appendix A of Mardia et al. (1979).

2. $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$; this is the case of equal covariance matrices for all component densities, that is, $\boldsymbol{\phi}_l = \boldsymbol{\mu}_l$ and $\boldsymbol{\phi} = \boldsymbol{\Sigma}$. The derivation is essentially the same as for the previous cases and the likelihood equation for $\boldsymbol{\mu}_l$ is unchanged. Using equation (12) leads to the following likelihood equation for $\boldsymbol{\Sigma}$,

$$\boldsymbol{\Sigma} = \frac{1}{n} \sum_{j=1}^{n} \sum_{i=1}^{M} \sum_{l=1}^{K} z_{ji} p(c_l|g_i, \boldsymbol{x}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T.$$

3.4.3 Two Level Mixtures with Class Specific Gaussian Components

In section 3.3.3 the likelihood equations for the case of class specific component distributions are outlined. For the special case that the component distributions are Gaussian distributions the equations can be written more explicitly. The likelihood equations (19) and (20) for the class weights and mixing weights remain unchanged. From equations (16)-(18) we obtain the likelihood equations inserting the derivative of the log-density with respect to the parameters.

1. $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_{il})$; we start with the case of separate covariance matrices for each component density, that is, $\boldsymbol{\phi}_{il} = (\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_{il}), \boldsymbol{\phi}_i = \emptyset$ and $\boldsymbol{\phi} = \emptyset$.

$$\log f_{il}(\boldsymbol{x}_j|\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_{il}) = \log \frac{|\boldsymbol{\Sigma}_{il}^{-1}|^{\frac{1}{2}}}{(2\pi)^{\frac{d}{2}}} - \frac{1}{2}(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})^T \boldsymbol{\Sigma}_{il}^{-1}(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})$$

$$for \ \ i = 1, ..., M \ \ and \ \ l = 1, ..., k_i$$

Let $\boldsymbol{\mu}_{il}(p)$ denote the pth element of the vector $\boldsymbol{\mu}_{il}$, $\boldsymbol{\Sigma}_{il}(p,q)$ denote element (p,q) of the covariance matrix $\boldsymbol{\Sigma}_{il}$ and $I\boldsymbol{\Sigma}_{il}(p,q)$ denote element (p,q) of the the inverse covariance matrix $\boldsymbol{\Sigma}_{il}^{-1}$. Then

$$\frac{\partial \log f_{il}(\boldsymbol{x}_j|\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_{il})}{\partial \boldsymbol{\mu}_{il}} = \boldsymbol{\Sigma}_{il}^{-1}(\boldsymbol{x}_j - \boldsymbol{\mu}_{il}) \quad for \ \ i = 1, ..., M \ \ and \ \ l = 1, ..., k_i$$

$$\frac{\partial \log f_{il}(\boldsymbol{x}_j|\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_{il})}{\partial I\boldsymbol{\Sigma}_{il}(p,q)} = (1 - \frac{\delta_{pq}}{2})[\boldsymbol{\Sigma}_{il}(p,q) - (\boldsymbol{x}_j(p) - \boldsymbol{\mu}_{il}(p))(\boldsymbol{x}_j(q) - \boldsymbol{\mu}_{il}(q))]$$

$$for \quad i = 1, ..., M \quad and \quad l = 1, ..., k_i$$

where δ_{pq} is the Kronecker delta function. Using equation (16) for $\boldsymbol{\mu}_{il}$ and $\boldsymbol{\Sigma}_{il}$ and solving for $\boldsymbol{\mu}_{il}$ and $\boldsymbol{\Sigma}_{il}$ leads to the following likelihood equations,

$$\boldsymbol{\mu}_{il} = \frac{\sum_{j=1}^{n} z_{ji} p(c_{il}|g_i, \boldsymbol{x}_j)\boldsymbol{x}_j}{\sum_{j=1}^{n} z_{ji} p(c_{il}|g_i, \boldsymbol{x}_j)} \qquad for \quad i = 1, ..., M \quad and \quad l = 1, ..., k_i$$

$$\boldsymbol{\Sigma}_{il} = \frac{\sum_{j=1}^{n} z_{ji} p(c_{il}|g_i, \boldsymbol{x}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})^T}{\sum_{j=1}^{n} z_{ji} p(c_{il}|g_i, \boldsymbol{x}_j)} \qquad for \ \ i = 1, ..., M \ \ and \ \ l = 1, ..., k_i.$$

These equations are together with the equations (19) and (20) the likelihood equations for the situation $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_{il})$.

2. $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$; we continue with the case of class-conditional covariance matrices, that is, $\boldsymbol{\phi}_{il} = \boldsymbol{\mu}_{il}, \boldsymbol{\phi}_i = \boldsymbol{\Sigma}_i$ and $\boldsymbol{\phi} = \emptyset$. The derivation is essentially the same as for the previous case and the likelihood equation for $\boldsymbol{\mu}_{il}$ is unchanged. Let n_i denote the number of observations in the training set from class g_i. Using equation (17) leads to the following likelihood equation for $\boldsymbol{\Sigma}_i$.

$$\boldsymbol{\Sigma}_i = \frac{1}{n_i} \sum_{j=1}^{n} z_{ji} \sum_{l=1}^{k_i} p(c_{il}|g_i, \boldsymbol{x}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})^T \qquad for \quad i = 1, ..., M.$$

3. $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$; finally we turn to the case of equal covariance matrices, that is, $\boldsymbol{\phi}_{il} = \boldsymbol{\mu}_{il}$, $\boldsymbol{\phi}_i = \emptyset$, $\boldsymbol{\phi} = \boldsymbol{\Sigma}$. The likelihood equation for $\boldsymbol{\mu}_{il}$ is unchanged. Using equation (18) leads to the following likelihood equation for $\boldsymbol{\Sigma}$,

$$\boldsymbol{\Sigma} = \frac{1}{n} \sum_{j=1}^{n} \sum_{i=1}^{M} z_{ji} \sum_{l=1}^{k_i} p(c_{il}|g_i, \boldsymbol{x}_j)(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})(\boldsymbol{x}_j - \boldsymbol{\mu}_{il})^T.$$

3.5 Alternating Likelihood Estimation and the EM Algorithm

For the case of a one level mixture the parameters and mixing weights can be estimated from the likelihood equations in section 3.3.1 using an alternating scheme. This scheme can easily be extended to the case of class-conditional mixtures where we want to estimate the parameters of a two level mixture with class specific component distributions from the likelihood equations (16)–(21) and for common component distributions for the classes from the likelihood equations (11)–(15).

The alternating likelihood estimation procedure is an algorithm for finding stationary points of the loglikelihood. This procedure is equivalent to the Expectation Maximization (EM) algorithm of Dempster et al. (1977) for finite mixture models. It is possible to regard the mixture data as coming from one of the component distributions. We can then regard a data set as *incomplete* with the *component* memberships as missing values, and thereby apply the EM algorithm.

We have in section 3.2, mentioned that the loglikelihood for finite mixture models can be unbounded and that it can have many local optima, saddle points and singularities. The parameter estimates found by using the EM algorithm do therefore depend upon the starting point of the algorithm and the possibility of multiple solutions is always present. Duda and Hart (1973) show, on page 195, that for a one level mixture of two univariate Gaussian components with unit (known) variances the EM algorithm will find both a saddle point and local maxima (in this case the loglikelihood is bounded). In most cases, however, the stationary points will be local maxima.

Convergence properties of the EM algorithm are discussed in Dempster et al. (1977), Boyles (1983) and Wu (1983). Each step of the EM algorithm will increase (not decrease) the loglikelihood (monotonicity property). The loglikelihood, L, will converge to some limit, L^*, if it is bounded above. Further regularity conditions are needed to prove that the limit is a stationary point and that the EM sequence of parameters and weights, $\{\boldsymbol{\Theta}^{(s)}\}$, converges, see Wu (1983).

Theoretical considerations about the EM algorithm for mixture problems are found in Redner and Walker (1984) and section 4.3 of Titterington et al. (1985). For mixtures of exponential family densities, among others the Gaussian distribution, Redner and Walker (1984) found conditions for local convergence of the EM algorithm. Titterington et al. (1985) have rewritten the results as follows. "If the Fischer information matrix is positive definite at the true parameters and weights, $\mathbf{\Theta}_0$, and the sequence $\{\mathbf{\Theta}^{(s)}\}$ is generated by the EM algorithm started from $\mathbf{\Theta}^{(0)}$ close enough to $\mathbf{\Theta}_0$, then the strongly consistent solution $\hat{\mathbf{\Theta}}$ of the likelihood equations is well defined and $\mathbf{\Theta}^{(s)} \rightarrow \hat{\mathbf{\Theta}}$ as $s \rightarrow \infty$ at a linear rate."

Titterington et al. (1985), section 4.3.3, find that in most examples a sensible local maximum of the likelihood surface is found by the EM algorithm as long as we for the unbounded loglikelihoods keep away from the boundary of the parameter space. The rate of convergence of an EM algorithm is in general proportional to the fraction of information in the observed data relative to the information in the observed and missing data. Redner and Walker (1984) found in simulation studies that 95 % of the change in the loglikelihood function between its starting point and the limit of the EM sequence was reached after only 5 iterations. This was regardless of the number of iterations required to approximate the limit very closely. Redner and Walker (1984) have a comment on the speed of convergence for well and poorly separated component populations. If the component populations in the mixture are "well-separated", an EM iteration sequence which converges will exhibit rapid linear convergence. If the component populations are "poorly separated" low linear convergence of the EM sequence can be expected. The speed of convergence is enhanced by larger sample size and better separation of the components.

There exists many empirical studies of the convergence of the EM algorithm for Gaussian mixtures. An early study is Hasselblad (1966). He found that the for a mixture of univariate Gaussian densities the EM algorithm (which he calls a generalized steepest decent method) was preferable to the the Newton-Raphson method on small data set. For larger data sets the Newton-Raphson method gave best results. Ingrassia (1992) compared the EM algorithm to simulated annealing for Gaussian mixtures. He found "no overwhelming superiority" of either method. But, in his study he found that both algorithms sometimes did not find the largest local maximum and ended up at a less desirable local maximum.

As we may understand, the EM algorithm does have many deficiencies; it has a slow convergence, is sensitive to starting values, converges to sometimes inferior local maxima and might experience problems for the case of unbounded loglikelihoods for poorly chosen starting points. But, it has a low cost per iteration, is economical when it comes to storage, is (as we will see) relatively easy to programme, it handles probabilistic constraints with ease and has a heuristic appeal.

One Level Mixtures

This is an outline of how to estimate the parameters and mixing weights, $\mathbf{\Theta} = \{(\boldsymbol{\theta}_l, p(c_l)); l = 1, ..., K\}$, for the one level mixture of section 3.1.1. This alternating likelihood estimation algorithm is found in Panel on Discriminant Analysis, Classification and Clustering (1989) and its equivalent Expectation Maximization (EM) algorithm is outlined for this situation in section 4.3.2 of Titterington et al. (1985).

1. Given estimates of the parameters, $\boldsymbol{\theta}_l$, and class weights, $p(c_l)$, estimate $p(c_l|\boldsymbol{x}_j)$ using equation (9) for $l = 1, ..., K$ and $j = 1, ..., n$.
2. Given $p(c_l|\boldsymbol{x}_j)$, estimate $\boldsymbol{\theta}_l$ from equations (6) and (7) and $p(c_l)$ from equation (8).
3. If the estimates $\boldsymbol{\theta}_l$ or $p(c_l)$ have changed more than a specified value goto 1.

Initial estimates of $\boldsymbol{\theta}_l$ can be found using K-means or one of the other initialization methods described in appendix B. Initial estimates of $p(c_l)$ are set to the fraction of observations being assigned to component c_l by that method.

Two Level Mixtures with Common Components

This is an outline of how to estimate the parameters and the weights, $\boldsymbol{\Theta} = \{(\boldsymbol{\theta}_l, p(g_i), p(c_l|g_i)); i = 1, ..., M; l = 1, ..., K\}$, for a two level mixture with common component distributions for the classes.

1. Given estimates of the parameters, $\boldsymbol{\theta}_l$, and the mixing weights, $p(c_l|g_i)$, estimate $p(c_l|g_i, \boldsymbol{x}_j)$ using equation (15), $i = 1, ..., M$, $l = 1, ..., K$ and $j = 1, ..., n$.
2. Given $p(c_l|g_i, \boldsymbol{x}_j)$, estimate $\boldsymbol{\theta}_l$ from equations (11) and (12) and $p(c_l|g_i)$ from equation (14).
3. If the estimates $\boldsymbol{\theta}_l$ or $p(c_l|g_i)$ have changed more than a specified value goto 1.
4. Calculate estimates of the class weights, $p(g_i)$, using equation (13)

Initial estimates are found as in the previous case.

Two Level Mixtures with Class Specific Components

When there are no common parameters across the classes, $\boldsymbol{\phi} = \emptyset$, the estimation procedure below can be conducted for each class separately. This is an outline of how to estimate the parameters, the class weights, and the mixing weights, $\boldsymbol{\Theta} = \{(\boldsymbol{\theta}_{il}, p(g_i), p(c_{il}|g_i)); i = 1, ..., M; l = 1, ..., k_i\}$, for a two level mixture with component distributions specific to each class.

1. Given estimates of the parameters, $\boldsymbol{\theta}_{il}$, and the mixing weights, $p(c_{il}|g_i)$, estimate $p(c_{il}|g_i, \boldsymbol{x}_j)$ from equation (21) for $i = 1, ..., M$, $l = 1, ..., k_i$ and $j = 1, ..., n$.
2. Given $p(c_{il}|g_i, \boldsymbol{x}_j)$, estimate $\boldsymbol{\theta}_{il}$ from equations (16)-(18) and $p(c_{il}|g_i)$ from equation (20).
3. If the estimates $\boldsymbol{\theta}_{il}$ or $p(c_{il}|g_i)$ have changed more than a specified value goto 1.
4. Calculate estimates of the class weights, $p(g_i)$, from equation (19).

Initial estimates can be found as in the previous cases

The two methods found above are applications of the EM algorithm. The step one is the E-step and the step two and four are the M-step. For class specific components the E-step is equation (21) and the M-step is equations (16)–(20). This is shown in appendix A.

4 Mixture Codebook Classification

The aim of this report is to construct methods for discrimination and classification based on a mixture codebook. After in the previous section having used finite mixture models to represent distributions, here the goal is to present methods to construct a mixture codebook with the aim to classify. We will here represent the class-conditional distributions by finite mixture models and therefore look exclusively at two level mixture models.

As defined in section 3, a mixture codebook consists of three parts; a set of parameters and weights, $\boldsymbol{\Theta}$, the component densities, f, and the component architecture, K. That is, $\mathcal{M} = \{\boldsymbol{\Theta}, f, K\}$. For mixtures with component distributions specific to each class the component architecture, K, is a vector giving the number of component densities in each class, $K = (k_1, k_2, ..., k_M)$. For component densities common for all classes the component architecture, K, gives the total number of components. We will here assume that the form of the component densities are known up to the parameters given in $\boldsymbol{\Theta}$ and that also the component architecture is known. The question of choosing the component architecture can be answered using knowledge of the complexity of the class-conditional distributions or by cross-validation and will not be studied in this report.

Representing the class-conditional distributions as mixtures leads to two different approaches to the estimation of the parameters and weights, $\boldsymbol{\Theta}$, of the mixture codebook, $\mathcal{M}$, based on the training set, $\mathcal{T} = (\boldsymbol{x}_j, \boldsymbol{z}_j)_{j=1}^n$.

Summarizing the Class-Conditional Distributions This approach consists of two separate steps. First we concentrate on the class-conditional distributions $p(\boldsymbol{x}|g_i, \boldsymbol{\Theta})$, that is, we estimate $\boldsymbol{\Theta}$ by maximizing $\log[\prod_{j=1}^n p(\boldsymbol{x}_j, \boldsymbol{z}_j|\boldsymbol{\Theta})]$. We impute this estimate to approximate $p(g_i|\boldsymbol{x}, \boldsymbol{\Theta})$ by $p(g_i|\boldsymbol{x}, \hat{\boldsymbol{\Theta}})$ and then classify to the class with the maximum posterior class probability. This is the topic of section 4.2.

Summarizing the Posterior Class Distributions This approach looks directly at the problem in hand. The parameters and weights of the mixture codebook, $\boldsymbol{\Theta}$, are estimated by optimizing a criterion function of the posterior class probabilities and the correct classification with respect to the parameters and weights, $\boldsymbol{\Theta}$; $\log[\prod_{j=1}^n J(p(g_1|\boldsymbol{x}_j, \boldsymbol{\Theta}), p(g_2|\boldsymbol{x}_j, \boldsymbol{\Theta}), ..., p(g_M|\boldsymbol{x}_j, \boldsymbol{\Theta}), \boldsymbol{z}_j)]$. Then classification is performed based on $p(g_i|\boldsymbol{x}, \hat{\boldsymbol{\Theta}})$. This is the topic of section 4.3.

4.1 Maximum Posterior Class Probability

The Bayes decision rule with zero-one loss function, see for example Duda and Hart (1973), classifies an observation to the class with the maximum posterior probability, which we will call MAP classification. We look more closely into the situation when $p(\boldsymbol{x}|g_i)$ is represented by a finite mixture model, $p(\boldsymbol{x}|g_i, \boldsymbol{\Theta})$.

4.1.1 Two Level Mixture with Common Components

Using a class-conditional mixture with common components for the classes, in (3), we find the posterior probability for the class g_i given the observation $\boldsymbol{x}$, $p(g_i|\boldsymbol{x}, \boldsymbol{\Theta})$:

$$p(g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \frac{p(g_i)\sum_{l=1}^{K} p(c_l|g_i) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l)}{p(\boldsymbol{x}|\boldsymbol{\Theta})} \tag{25}$$

Using MAP classification one then classifies to the class with the maximum posterior probability

$$\operatorname*{argmax}_{i=1,M} p(g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \operatorname*{argmax}_{i=1,M} p(g_i)\sum_{l=1}^{K} p(c_l|g_i) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l)$$

Notice that it is not necessary to compute the marginal distribution $p(\boldsymbol{x}|\boldsymbol{\Theta})$ to perform the classification.

We will comment on the relationship between the two level mixture with common components and radial basis function neural network, see Xu et al. (1994). The posterior probability for class g_i can be written

$$p(g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \frac{p(g_i)\sum_{l=1}^{K} p(c_l|g_i) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l)}{p(\boldsymbol{x}|\boldsymbol{\Theta})} = \frac{1}{p(\boldsymbol{x}|\boldsymbol{\Theta})}\sum_{l=1}^{K} p(c_l, g_i) f_l(\boldsymbol{x}|\boldsymbol{\theta}_l)$$

We thus see that the joint probability, $p(g_i, \boldsymbol{x}|\boldsymbol{\Theta})$, for a class-conditional mixture with common components for the classes can be written as a basis function expansion. When the component distributions are radial, as is the case for the Gaussian distribution with diagonal covariance matrix with equal variances, this is called a radial basis function expansion (RBF), see for example Broomhead and Lowe (1988). Radial basis function neural networks (RBFN) is a popular topic in the static neural networks literature, see for example Moody and Darken (1989) and Girosi (1994).

The posterior probability for a general class g_i can also be closely related to the *normalized* RBFN, as presented in the review by Xu et al. (1994). If we restrict the marginal probabilities for the components, $p(c_l) = \sum_{i=1}^{M} p(c_l|g_i)p(g_i)$, to be equal for all components, $p(c_l) = \frac{1}{K}$, then the posterior probability for class g_i can be written as follows,

$$p(g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \frac{\sum_{l=1}^{K} w_{il} f_l(\boldsymbol{x}|\boldsymbol{\theta}_l)}{\sum_{l=1}^{K} f_l(\boldsymbol{x}|\boldsymbol{\theta}_l)}$$

which is the general form of the normalized radial basis function network. The weights w_{il} equals $p(g_i|c_l)$ with $p(c_l) = \frac{1}{K}$, that is, $w_{il} = K \cdot p(g_i)p(c_l|g_i)$.

4.1.2 Two Level Mixture with Class Specific Components

Using a class-conditional mixture with class specific components, in (5), we find the posterior probability for the class g_i given the observation $\boldsymbol{x}$, $p(g_i|\boldsymbol{x}, \boldsymbol{\Theta})$:

$$p(g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \frac{p(g_i)\sum_{l=1}^{k_i} p(c_{il}|g_i) f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})}{p(\boldsymbol{x}|\boldsymbol{\Theta})} \tag{26}$$

Using MAP classification one then classifies to the class with the maximum posterior probability

$$\underset{i=1,M}{\operatorname{argmax}}\, p(g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \underset{i=1,M}{\operatorname{argmax}}\, p(g_i) \sum_{l=1}^{k_i} p(c_{il}|g_i) f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})$$

Again, it is not necessary to compute the marginal distribution, $p(\boldsymbol{x}|\boldsymbol{\Theta})$, to perform the classification. Also here the joint probability, $p(g_i, \boldsymbol{x}|\boldsymbol{\Theta})$, can be written as a basis function expansion, but here with separate basis functions for the classes,

$$p(g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \frac{1}{p(\boldsymbol{x}|\boldsymbol{\Theta})} \sum_{l=1}^{k_i} p(c_l, g_i) f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})$$

The posterior probability for the component c_{il}, $p(c_{il}|\boldsymbol{x}, \boldsymbol{\Theta})$, is given as:

$$p(c_{il}|\boldsymbol{x}, \boldsymbol{\Theta}) = p(c_{il}, g_i|\boldsymbol{x}, \boldsymbol{\Theta}) = \frac{p(g_i)p(c_{il}|g_i) f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})}{p(\boldsymbol{x}|\boldsymbol{\Theta})}$$

Instead of classifying to the *class* with the maximum probability another possibility could be to find the *component* with the maximum posterior probability and classify to the class of the component with the maximum posterior probability.

$$\underset{i=1,M,l=1,k_i}{\operatorname{argmax}}\, p(c_{il}|\boldsymbol{x}, \boldsymbol{\Theta}) = \underset{i=1,M,l=1,k_i}{\operatorname{argmax}}\, p(g_i)p(c_{il}|g_i) f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})$$

If the maximum posterior probability of a component is greater than 0.5, then the maximum posterior class probability and the class of the component with the maximum posterior component probability, will give the same classification. When the maximum posterior probability of a component is below 0.5, the two methods can give different results. Therefore classifying to the class with the maximum posterior probability is preferred.

4.2 Classification Based on Summarizing the Class Conditional Distributions

We will here look into estimation of the parameters and weights of the mixture codebook using two separate steps.

4.2.1 Direct Approach, Maximizing the Joint Likelihood

Consider summarizing the class-conditional distributions, $p(\boldsymbol{x}|g_i)$, by finite mixture models, $p(\boldsymbol{x}|g_i, \boldsymbol{\Theta})$. Without looking to the problem of discrimination and classification the parameters and weights of the mixture codebook are estimated by maximizing the joint likelihood of the observations and encoded class memberships in the training set; $\prod_{j=1}^{n} p(\boldsymbol{x}_j, \boldsymbol{z}_j|\boldsymbol{\Theta})$. This can be done using the alternating likelihood estimation algorithm (EM) of section 3.5.

The estimated parameters and weights, $\hat{\boldsymbol{\Theta}}$, are then used to estimate the posterior class probabilities, $p(g_i|\boldsymbol{x}, \boldsymbol{\Theta})$, by $p(g_i|\boldsymbol{x}, \hat{\boldsymbol{\Theta}})$, for $i = 1, ..., M$ and an observation, $\boldsymbol{x}$, is then classified to the class with the largest estimated posterior probability from equation (25) or (26).

This approach has been used by Taxt et al. (1991) and Lippmann (1994). Taxt et al. have chosen a class-conditional mixture with class specific components. Each class consisted of two Gaussian component distributions with common covariance matrix for all components. Lippmann (1994) uses class specific Gaussian component densities. Hastie and Tibshirani (1995) has also treated the case of using Gaussian mixtures with class specific components and common covariance matrix for all components. They have implemented the EM algorithm by optimal scoring. Their method is called Mixture Discriminant Analysis (MDA), which also includes a feature called subclass shrinkage. The MC^2 direct approach is more general than these accounts. The MC^2 direct approach has the ability to use other component distributions than the Gaussian, can use common components for all classes and component or class specific covariance matrices for the Gaussian case.

Related work is also found in Tråvén (1991), where the EM algorithm is rewritten as a stochastic gradient descent routine. His method is called a Gaussian clustering network and has the ability to use unclassified observations in the estimation. In Chou and Chen (1992) class-conditional distributions are approximated by class specific Gaussian potential functions with component specific variances. The method is presented as a two layer network (with input, hidden and output layers). The estimation procedure consists of a mode detection step using K-means and a relaxation procedure to adjust so called scale factors.

4.2.2 Class Decomposition

Below the method we call *class decomposition* is outlined. This is an alternative approach to performing the calculations in the previous section for the case of a two level mixture with class specific components. The reason for presenting this method is to supply a procedure that does not need special software to perform the calculations of the direct approach. The procedure is based on the training set, $\mathcal{T} = (\boldsymbol{x}_j, \boldsymbol{z}_j)_{j=1}^n$.

1. Let g_i denote a general class. Divide the observations in class g_i into clusters using a clustering method (for example K-means as presented in section B.1) or the EM algorithm for a one level Gaussian mixture. Repeat this procedure for all the classes, g_i, for $i = 1, ..., M$. Alternatively, learning vector quantizer (LVQ) can be used on the total data set to produce cluster centres (the codebook) and then the observations are labelled to be from the nearest cluster in Euclidean distance. For a short introduction to LVQ see appendix B.3.
2. The $K = \sum_{i=1}^{M} k_i$ clusters are now regarded as classes. Run a discrimination or classification method on the K cluster data set. Now the classification rule has been constructed.

To use this classification rule on new observations we do the following. If the discrimination or classification method produces probabilities for the clusters, use this to calculate probabilities for the classes and classify to the class with the highest probability. If the discrimination or classification method produces classification regions for the clusters, merge the classification regions for the clusters into classification regions for the classes.

The discrimination and classification methods yielding probabilities in step 3, are preferred. This is because classification to the class with the highest probability is preferred to classification to the class of the component with the highest probability, see section 4.2.1.

This method can be used with different clustering methods and different methods of discrimination and classification. One choice is to produce the clusters and then use LDA or QDA to produce the classification rule. This will produce piecewise linear and piecewise quadratic class boundaries, respectively.

4.3 Classification Based on Summarizing the Posterior Class Distributions

Again consider representing the class-conditional distributions, $p(\boldsymbol{x}|g_i)$, by finite mixture models, $p(\boldsymbol{x}|g_i, \boldsymbol{\Theta})$. But estimate the parameters and weights, $\boldsymbol{\Theta}$, of the mixture codebook, $\mathcal{M}$, by optimizing a criterion function of the posterior class probabilities, $p(g_i|\boldsymbol{x}, \boldsymbol{\Theta})$ for $i = 1, ..., M$ and the correct classification of the training set; $\mathcal{J}(\boldsymbol{\Theta}|\mathcal{T})$. The training set is denoted $\mathcal{T} = (\boldsymbol{x}_j, \boldsymbol{z}_j)_{j=1}^n$. Below four different criterion functions are presented.

The estimated parameters and weights, $\hat{\boldsymbol{\Theta}}$, are then used to estimate $p(g_i|\boldsymbol{x}, \boldsymbol{\Theta})$ by $p(g_i|\boldsymbol{x}, \hat{\boldsymbol{\Theta}})$, for $i = 1, ..., M$, and an observation, $\boldsymbol{x}$, is then classified to the class with the largest estimated posterior probability, that is, MAP classification using equation (25) or (26). We feel that the use of these criterion functions in combination with the class-conditional mixture models is a novel approach to classification.

The criterion functions:

1. **Minimize the misclassification rate on the training set under MAP classification**,

$$\mathcal{J}(\boldsymbol{\Theta}|\mathcal{T}) = \sum_{j=1}^{n} I(\underset{i=1,M}{\operatorname{argmax}}\, p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}) \neq \underset{i=1,M}{\operatorname{argmax}}\, \boldsymbol{z}_j) \tag{27}$$

   where $I(A) = 1$ if A is true and $I(A) = 0$ otherwise.

   Minimizing this criterion will usually not result in a unique solution for $\boldsymbol{\Theta}$. Many different choices of $\boldsymbol{\Theta}$ will in general give the same misclassification rate on the training set. This criterion is also difficult to minimize, but it can be done using exhaustive and intelligent search routines.

2. **Maximize the posterior probabilities of the correct class of each observation in the training set (the conditional likelihood)**,

$$\prod_{j=1}^{n}\prod_{i=1}^{M}[p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})]^{z_{ji}}$$

$$\mathcal{J}(\boldsymbol{\Theta}|\mathcal{T}) = \sum_{j=1}^{n}\sum_{i=1}^{M} z_{ji} \log p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}). \tag{28}$$

   This criterion is the likelihood for conditional sampling; given the observation $\boldsymbol{x}$ the class of the observation is g_i with probability $p(g_i|\boldsymbol{x})$ (multinomial situation). This is the natural likelihood of $g_i|\boldsymbol{x}$, $i = 1, ..., M$. This criterion will not necessary minimize the criterion (27) above.

The summation in (28) only includes the probabilities $p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})$ where $z_{ji} = 1$, that is, the probabilities of the correct classes. But since $\sum_{i=1}^{M} p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}) = 1$, maximizing the probabilities $p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})$ where $z_{ji} = 1$ is also indirectly to minimize the probabilities $p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})$ where $z_{ji} = 0$.

The criterion (28) can be made more flexible by substituting the indicator variable encoding of the classes, $\boldsymbol{z}_j$, by a weight function, $\boldsymbol{v}_j(\boldsymbol{\Theta})$. For given values of the constants a, b and c the choice

$$v_{ji}(\boldsymbol{\Theta}) = \begin{cases} a & \text{if } z_{ji} = 1 \text{ and } \operatorname{argmax}_{i=1,M} p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}) = \operatorname{argmax}_{i=1,M} \boldsymbol{z}_j \\ b & \text{if } z_{ji} = 1 \text{ and } \operatorname{argmax}_{i=1,M} p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}) \neq \operatorname{argmax}_{i=1,M} \boldsymbol{z}_j \\ c & \text{if } z_{ji} = 0 \end{cases}$$

will take into account whether an observation is correctly classified or not using the current parameter choices. For $a = 1.0$, $b = 1.0$ and $c = 0.0$ we have the original criterion (28).

3. **Minimize the quadratic loss function**,

$$\mathcal{J}(\boldsymbol{\Theta}|\mathcal{T}) = \sum_{j=1}^{n} \sum_{i=1}^{M} (z_{ji} - p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}))^2. \tag{29}$$

This criterion is the well known least squares fitting of the regression of the class indications, z_{ji}, on the variables. This criterion is popular in the static neural network literature, see among other Ripley (1994b) and Hertz et al. (1991).

4. **Minimize the Kullback-Leibler divergence for independent Bernoulli variables** (also called entropy loss),

$$\mathcal{J}(\boldsymbol{\Theta}|\mathcal{T}) = \sum_{j=1}^{n} \sum_{i=1}^{M} (z_{ji} \log \frac{z_{ji}}{p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})} + (1 - z_{ji}) \log \frac{1 - z_{ji}}{1 - p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})}). \tag{30}$$

This criterion emerges in the situation when the class indicators $z_{ji}|\boldsymbol{x}_j$ are regarded as *independent* Bernoulli variables with probabilities $p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})$. The likelihood in this situation can be written:

$$\prod_{j=1}^{n} \prod_{i=1}^{M} p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta})^{z_{ji}} (1 - p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}))^{1-z_{ji}}$$

yielding the loglikelihood:

$$\sum_{j=1}^{n} \sum_{i=1}^{M} [z_{ji} \log p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}) + (1 - z_{ji}) \log(1 - p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}))]$$

The deviance is proportional to the loglikelihood minus the maximal value of the loglikelihood,

$$\sum_{j=1}^{n} \sum_{i=1}^{M} [z_{ji} \log z_{ji} + (1 - z_{ji}) \log(1 - z_{ji})] - \sum_{j=1}^{n} \sum_{i=1}^{M} [z_{ji} \log p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}) + (1 - z_{ji}) \log(1 - p(g_i|\boldsymbol{x}_j, \boldsymbol{\Theta}))]$$

which is equivalent to the expression (30).

This criterion is often used for multilayer perceptrons (MLPs), see Ripley (1993). When there are only two classes minimizing this criterion is equivalent to maximizing the criterion (28).

The four criterion functions have been implemented and tested on a small illustrative example in the next section and on one synthetic and one real data set in section 5.

4.4 A Small Illustrative Example

To compare the optimization of the joint likelihood (direct approach) and the four optimization criterion from the previous sections a small synthetic example is analysed.

A training set is created by drawing ten observations from each of two univariate Gaussian distributions with equal and known variances.

$$g_1: \quad x_{1,1}, x_{1,2}, ..., x_{1,10} \sim N(\mu_1 = -2, 2^2)$$

$$g_2: \quad x_{2,1}, x_{2,2}, ..., x_{2,10} \sim N(\mu_2 = 2, 2^2)$$

We assume that $p(g_1) = p(g_2) = \frac{1}{2}$.

We only look at the situation of one Gaussian component density for each class; class specific components. There are only two parameters to be estimated; $\mathbf{\Theta} = (\mu_1, \mu_2)$. Using MAP classification the class boundary, X_B, between class g_1 and g_2 is found at

$$p(g_1|X_B) = p(g_2|X_B)$$

$$X_B = \frac{\mu_1 + \mu_2}{2}$$

The theoretical boundary is at $X_B = 0.0$.

The total probability of misclassification (TPM) is defined as

$$TPM(X_B) = p(g_2) \int_{-\infty}^{X_B} f(x|\mu_2)dx + p(g_1) \int_{X_B}^{\infty} f(x|\mu_1)dx$$

$$= p(g_2)(1 - \Phi(\frac{X_B - \mu_2}{\sigma})) + p(g_1)\Phi(\frac{X_B - \mu_2}{\sigma})$$

Where Φ is the cumulative probability in the standard Gaussian distribution. Inserting the values for μ_1, μ_2, σ, $p(g_1)$ and $p(g_2)$ gives

$$TPM(X_B) = \frac{1}{2}[1 - \Phi(\frac{X_B}{2} - 1) + \Phi(\frac{X_B}{2} + 1)].$$

The total probability of misclassification when $X_B = 0.0$ is 15.9 %. The total probability of misclassification inserted an estimated class boundary, $\hat{X}_B$, is called the *actual error rate*, AER, see for example Johnson and Wichern (1982).

We assume that the parameters, $\mathbf{\Theta}$, are unknown and must be estimated from the training set. For each criterion the optimized value for the criterion, the estimated means, the class boundary, the misclassification rate on the training set and the actual error rate are given.

Maximizing the joint likelihood (direct approach) The likelihood equations have an unique solution at $\hat{\mu}_i = \frac{1}{10}\sum_{j=1}^{10} x_{i,j}$ for $i = 1, 2$.

$\hat{\mu}_1 = -2.13$ $\hat{\mu}_2 = 2.57$ $\hat{X}_B = 0.218$
Misclassification rate on the training set: 10.0 %
Actual error rate : 16.0 %

Minimum misclassification rate on the training set This problem has an indefinite number of solutions for $\hat{\mu}_1$ and $\hat{\mu}_2$. Since $\hat{\mu}_1 = 2\hat{X}_B - \hat{\mu}_2$, at a chosen boundary $\hat{X}_B$ we see that all choices of $\hat{\mu}_1$ have a corresponding choice of $\hat{\mu}_2$. This illustrates the problem of identifiability associated with this criterion. For the small synthetic example the boundary can be placed in the interval $[-0.170, -0.023]$ and will then give one misclassified observation in the training set and actual error rate of 15.9 %. The results below are for the minimum found using an implemented search routine.

Criterion=1.0 $\hat{\mu}_1 = -2.29$ $\hat{\mu}_2 = 2.21$ $\hat{X}_B = -0.038$
Misclassification rate on the training set: 5.0 %
Actual error rate: 15.9 %

Maximize the conditional likelihood The results below are the maximum found using an implemented search routine. This is the global maximum. Here we have only two classes and therefore the results using this criterion equals the results using criterion (30).

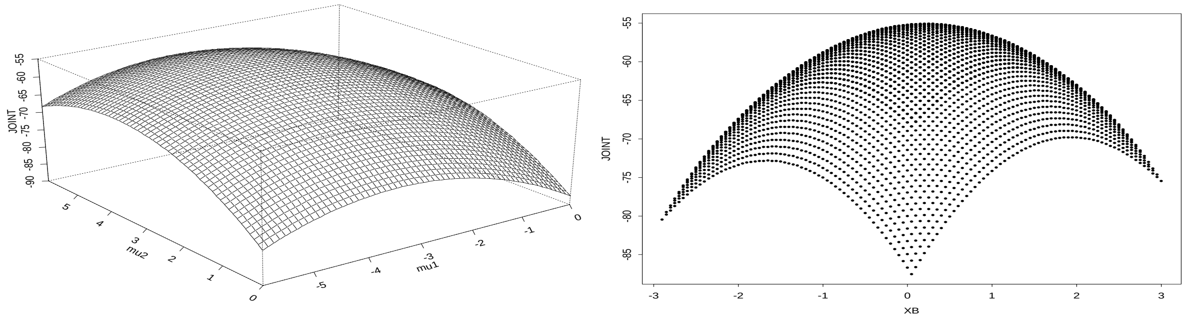
Criterion=-3.713 $\hat{\mu}_1 = -3.87$ $\hat{\mu}_2 = 4.45$ $\hat{X}_B = 0.293$
Misclassification rate on the training set: 10.0 %
Actual error rate: 16.1 %

Minimize the quadratic loss function As above, the results are found using an implemented search routine and is the global minimum.

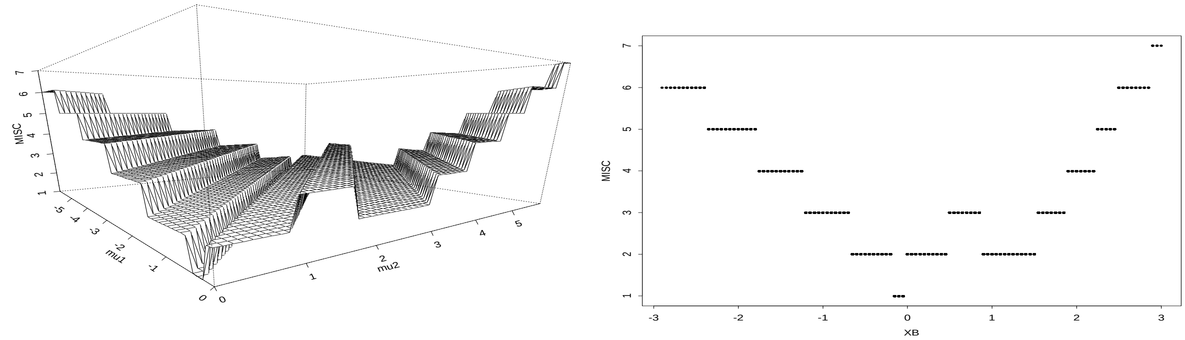
Criterion= 2.495 $\hat{\mu}_1 = -2.91$ $\hat{\mu}_2 = 3.37$ $\hat{X}_B = 0.229$
Misclassification rate on the training set: 10.0 %
Actual error rate: 16.0 %

The criteria are plotted as functions of the means, μ_1 and μ_2, and of the class boundary, X_B, in figure 1 for all combination of $\mu_1 \in \{-5.9, -5.8, ..., 0.0\}$ and $\mu_2 \in \{0.1, 0.2, ..., 6.0\}$. The estimated means and class boundaries are presented graphically in figure 2.

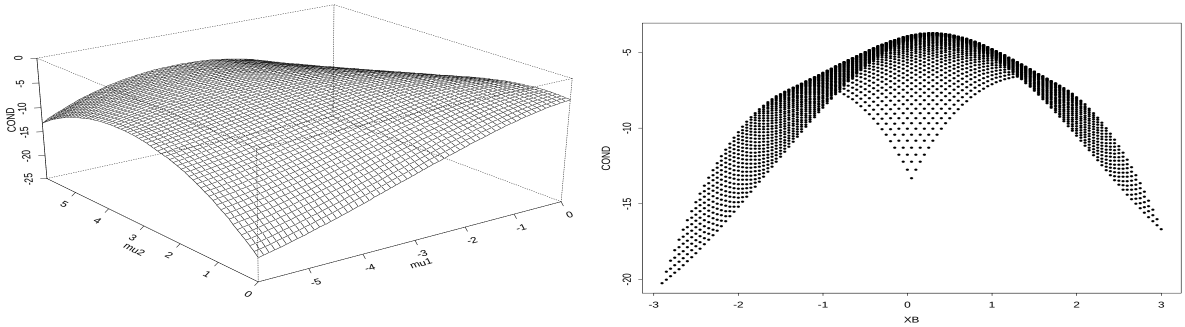
Some conclusions can be drawn from this small example. We see that the estimated means differ significantly for the different criteria. Only the joint loglikelihood aims directly at estimating the class-conditional distributions. The other criteria focus on estimating the posterior class distributions and are therefore not suited for estimating the class-conditional distributions. The estimated class boundary is similar for all criteria, and the actual error rates does not show significant differences for the estimated class boundaries.



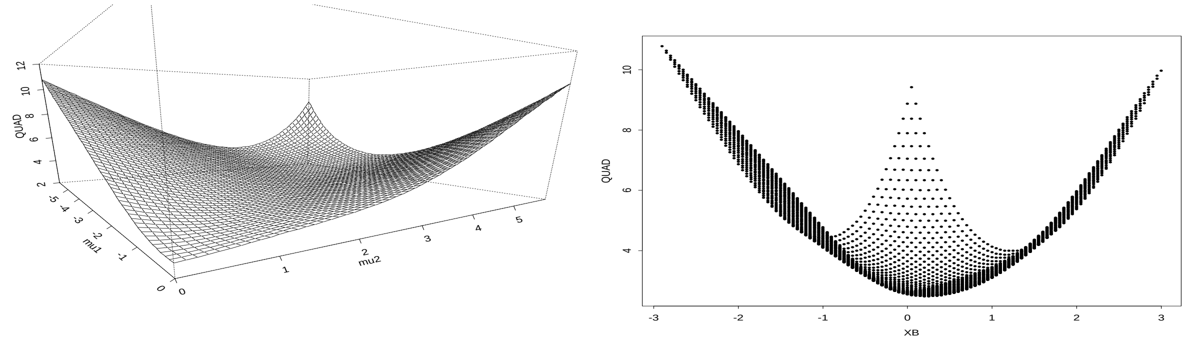
Maximize the joint loglikelihood JOINT



Minimize the misclassification rate, MISC



Maximize the posterior probabilities of correct classification, COND



Minimize the quadratic loss, QUAD

Figure 1: Plots of the criterion functions on the small example as functions of the means and as functions of the class boundary. Here JOINT denotes that the joint loglikelihood of section 3.3.3 is the criterion, MISC that (27) is the criterion, COND that (28) is the criterion and QUAD that (29) is the criterion.

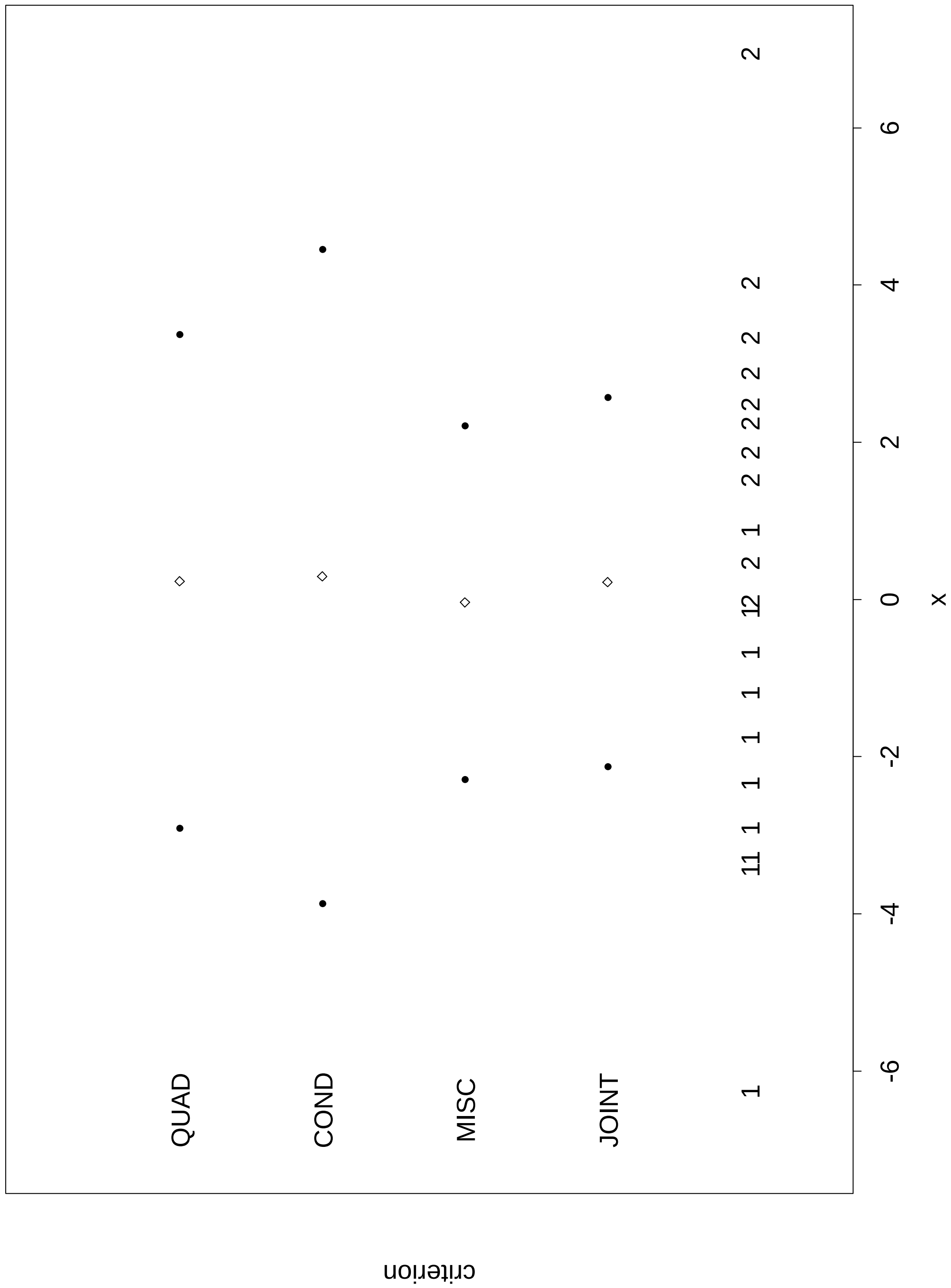


Figure 2: Parameter estimates (dots) and class boundaries (diamonds) for the small synthetic example. The lower line denotes the observations coded with the class membership. The JOINT denotes that the joint likelihood of section 4.2.1 is used, MISC that the criterion (27) is used, COND that the criterion (28) is used, QUAD that the criterion (29) is used.

4.5 Mixture Codebook Classification in Relation to Some Other Methods for Classification and Discrimination

Some well known methods for discrimination and classification fit into the framework of mixture codebook classification (MC^2).

Linear Discriminant Analysis (LDA) can be seen as a special case of the MC^2 with Gaussian class specific component densities for each class, $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$, with one component inside each class, $K = (1, 1, ..., 1)$, and equal class weights. In this case the results from the EM algorithm is the well-known Gaussian maximum likelihood estimates.

Quadratic Discriminant Analysis (QDA) can also be seen as a special case of the MC^2. The situation equals that of LDA with the difference that Gaussian component distributions with class specific matrices of covariance is used, $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$. LDA and QDA are presented for example in Johnson and Wichern (1982) chapter 11.

Logistic Discrimination For the MC^2 method using Gaussian class specific component densities with one component inside each class and equal covariance matrices for the classes maximizing the posterior probability of correct classification, (28), will result in logistic discrimination, as presented for example in Seber (1984) chapter 6.

Learning Vector Quantizer (LVQ) The method of Learning Vector Quantizer is presented briefly in appendix B.3. It can be seen as a MC^2 situation where the component distributions are class specific and uniform over the Dirichlet tessellation formed by the cluster (component) centres. The component architecture is general, $K = (k_1, k_2, ..., k_M)$.

Radial Basis Function Networks (RBFN) The name RBFN is used to describe different methods; see among others Broomhead and Lowe (1988), Hertz et al. (1991), Girosi (1994), Moody and Darken (1989), Lippmann (1994) and Xu et al. (1994). In classification a RBFN (or a normalized RBFN) can be seen as modelling the posterior probabilities for the classes by finite mixture models, as shown in section 4.1.1. Popular choices for component distributions are $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$ and $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$. The number of components, K, can take general values. The estimation procedure is often conducted in two step, as in Moody and Darken (1989), the first step estimates the parameters of the component distributions using the K-means or EM algorithm and the second step consists of estimating the equivalent of the mixing and class weights minimizing the quadratic loss function in (29).

kth Nearest Neighbour (k-NN) The kth nearest neighbour method is a special case of the kernel discriminant analysis, where the number of components equals the number of observations in the training set, $K = (n_1, n_2, ..., n_M)$, that is, each observation is the centre of a component distribution which is dependent on the choice of metric and the class priors are estimated by the proportions of each class in the data set. For kth nearest neighbour using Euclidean distance the component densities are uniform distributions over the Dirichlet tessellation formed by the observations. For more information on kth nearest neighbour see Fix and Hodges (1951) and Dasarathy (1991).

5 Examples

To evaluate and compare the performance of the mixture codebook (MC^2) routines a comparative study is performed using two data sets; one synthetic data set and one real data set.

To characterize a MC^2 situation we need to specify the *component architecture* and the choice of *component distributions*. The component architecture gives the number of components inside each class for class specific components (for example (1,3,2) for three classes with one, three and two components respectively) and the total number of components for common components densities.

Here only Gaussian component distributions are investigated. For Gaussian component distributions we need to specify the covariance structure. The covariance matrices can be equal for all classes or components or specific to each class or component. The structure of the covariance matrices can also be specified; full, diagonal or diagonal with equal variances.

5.1 Notes on Analysis and Implementation

The analyses are based on one training set and one test set. The training set is used to construct the classification rule and the performance of the classification rule is evaluated on the test set. Approximate 95 % confidence intervals can be constructed based on that the number of errors in the test set is assumed to be binomially distributed. These confidence intervals are the basis for unpaired comparisons between MC^2 situations. Paired comparisons are based on McNemar's test with an added condition. We assume that all the errors made by the method with the lower error rate are made by the method with the higher error rate. We then need the difference in the number of errors made by the two methods to be at least 5 to be significant on a 5% level, see Ripley (1994a).

Mixture Codebook Classification using the EM algorithm (JOINT)

The EM algorithm is implemented in the S-language for Gaussian component distributions, see section 3.4 and 3.5. Common components, $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$ and $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$, and specific components, $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$ and $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$, are implemented for full covariance matrices, $\boldsymbol{\Sigma}$, diagonal covariance matrices, $diag(\boldsymbol{\Sigma})$, and diagonal covariance matrices with equal variances, $\sigma^2\boldsymbol{I}$.

The JOINT routines use the K-means algorithm of Hartigan and Wong (1979) (as implemented in S-Plus) to find a good starting point for the EM algorithm. The K-means algorithm is initialized by drawing observations randomly from the training set. Only one starting point for the EM algorithm is tried for each MC^2 model. The EM algorithm is stopped when the average change in each parameter estimate (the means and covariance matrices) is less than 0.005. Let θ_r denote a general parameter, and $\theta_r^{(s)}$ the parameter estimates at the sth step of the algorithm. The stopping (convergence) criterion is as follows,

$$\frac{1}{R}\sqrt{\sum_{r=1}^{R}(\theta_r^{(s+1)} - \theta_r^{(s)})^2} \leq 0.005.$$

Mixture Codebook Classification Optimization (OPT)

Optimization of the four criteria given in section 4.3 is implemented in C using a robust routine called `amoeba` from Press et al. (1988). This is a downhill simplex method. Using gradient routines (CG and BFGS) from Press et al. (1988) have resulted in convergence problems. Also here only Gaussian component distributions are implemented. Common components, $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$ and $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$, and specific components, $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$ and $N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$, are implemented for diagonal covariance matrices, $diag(\boldsymbol{\Sigma})$, and diagonal covariance matrices with equal variances, $\sigma^2\boldsymbol{I}$. Full covariance matrices are not implemented due to problems with the requirement of positive definiteness. For the sake of simplicity, only constant uniform prior class and component weights, $p(g_i, c_l) = \frac{1}{\sum_{i=1}^{M} k_i}$, are considered for the class specific components. For the common components joint component and class weights, $p(g_i, c_l)$, are estimated.

The OPT routines are started each time with $antpar + 1$ initial starting points, where $antpar$ equals the number of parameters and weights to be estimated. These starting points are perturbed versions of the result from the JOINT routine. The OPT routines are run five times from these different perturbed start points. The OPT routines are stopped if the relative change in the criterion function used are smaller than 10^{-6}. Let the criterion function at the sth step of the routine be denoted $\mathcal{J}(\boldsymbol{\Theta}^{(s)}|\mathcal{T})$. The routine is then stopped (considered converged) when

$$\frac{|\mathcal{J}(\boldsymbol{\Theta}^{(s+1)}|\mathcal{T}) - \mathcal{J}(\boldsymbol{\Theta}^{(s)}|\mathcal{T})|}{\frac{1}{2}[\mathcal{J}(\boldsymbol{\Theta}^{(s+1)}|\mathcal{T}) + \mathcal{J}(\boldsymbol{\Theta}^{(s)}|\mathcal{T})]} \leq 10^{-6}.$$

5.2 A Synthetic Data Set of Gaussian Mixtures

This data set consists of three classes where each class is an equal mixture of two two-dimensional Gaussian distributions. Both the training set and the test set consist of 600 observations, 100 from each component distribution. The marginal distribution can be written:

$$p(\boldsymbol{x}) = \sum_{i=1}^{3} \frac{1}{3} \sum_{l=1}^{2} \frac{1}{2} f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})$$

where $f_{il}(\boldsymbol{x}|\boldsymbol{\theta}_{il})$ denotes the two-dimensional Gaussian distribution with parameters $\boldsymbol{\theta}_{il} = (\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$.

$$\boldsymbol{\mu}_{11} = \begin{bmatrix} 0 \\ -2 \end{bmatrix} \qquad \boldsymbol{\mu}_{12} = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \qquad \boldsymbol{\Sigma}_1 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.8 \end{bmatrix}$$

$$\boldsymbol{\mu}_{21} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \qquad \boldsymbol{\mu}_{22} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \qquad \boldsymbol{\Sigma}_2 = \begin{bmatrix} 1.2 & 0 \\ 0 & 1.3 \end{bmatrix}$$

$$\boldsymbol{\mu}_{31} = \begin{bmatrix} -2 \\ 4 \end{bmatrix} \qquad \boldsymbol{\mu}_{32} = \begin{bmatrix} 3 \\ 4 \end{bmatrix} \qquad \boldsymbol{\Sigma}_3 = \begin{bmatrix} 2.0 & 0 \\ 0 & 1.5 \end{bmatrix}$$

The training set is shown in the scatter plot of figure 3.

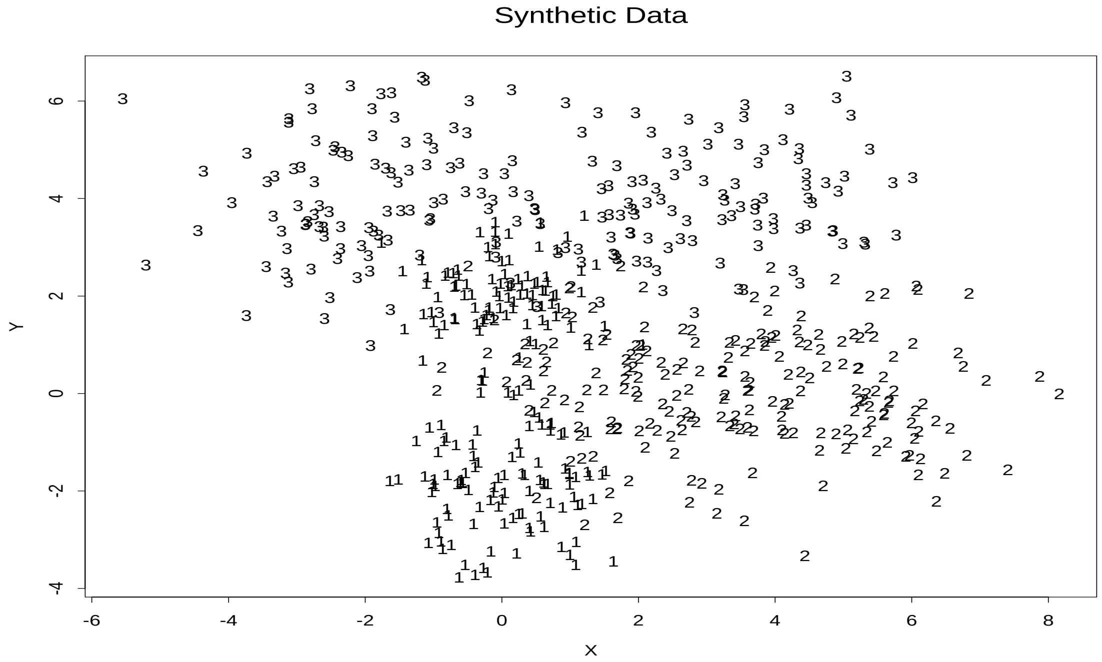


Figure 3: The three class synthetic training set of Gaussian mixtures. Each class is an equal mixture of two Gaussian component distribution. The training set contains 100 observations from each component distribution.

Since the population distributions are known, it is therefore possible to construct the classifier minimizing the Bayes decision rule with zero-one loss function. Error rates on the test set for some discrimination and classification methods are found in table 6 of Langaas (1995).

The results of the MC^2 routines are found in table 1. Four different component architectures are investigated; one component for each class (1,1,1), two components for each class (2,2,2), three components for each class (3,3,3) and four components for each class (4,4,4). For common components for all classes 3, 6, 9 and 12 components are used. JOINT denotes that the MC^2 JOINT routine is used and COND that the MC^2 OPT routine maximizing the conditional likelihood criterion of equation (28) in section 4.3 is used. The other criteria of section 4.3 have been tried for some of the component architectures and covariance situations, and gave similar results to the conditional likelihood criterion used here. Note that the MC^2 JOINT with component architecture (1,1,1) for the situation $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$ and equal class weights equals linear discriminant analysis (LDA) and the MC^2 JOINT with component architecture (1,1,1) for the situation $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$ with equal class weights equals quadratic discriminant analysis (QDA).

This example is constructed in favour of the MC^2 JOINT routines. Since the training set is relatively large, the JOINT routine with component architecture (2,2,2) for the component situation $N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$ should be able to estimate the class-conditional distributions well. The error rate has an approximate 95 % confidence interval of [12.0 $\pm$ 2.7] %. In this situation the summarizing of the class-conditional distributions is parametric density estimation. For this situation we do not expect the OPT routine to outperform the JOINT routine. This is confirmed by the results in table 1. The quadratic discriminant analysis classifier

Gaussian Mixtures													
MC^2		(1,1,1) components			(2,2,2) components			(3,3,3) components			(4,4,4) components		
specific components		min	mean	std	min	mean	std	min	mean	std	min	mean	std
JOINT	$N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$		21.000	(LDA)		13.333			13.500			12.667	
JOINT	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$		21.000			13.333			13.167			12.667	
COND	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$	19.500	19.500	0.000	11.833	12.267	0.226	11.500	12.033	0.356	11.167	11.500	0.236
JOINT	$N(\boldsymbol{\mu}_{il}, \sigma^2\boldsymbol{I})$		22.333			13.500			13.833			12.833	
COND	$N(\boldsymbol{\mu}_{il}, \sigma^2\boldsymbol{I})$	19.500	19.500	0.000	12.000	12.267	0.170	11.833	12.300	0.287	11.167	11.700	0.287
JOINT	$N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$		13.667	(QDA)		12.000			12.667			12.833	
JOINT	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$		13.333			12.167			12.167			12.500	
COND	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$	12.333	12.467	0.163	12.167	12.633	0.245	12.500	12.867	0.267	12.667	12.833	0.149
JOINT	$N(\boldsymbol{\mu}_{il}, \sigma_i^2\boldsymbol{I})$		19.833			12.333			12.833			12.500	
COND	$N(\boldsymbol{\mu}_{il}, \sigma_i^2\boldsymbol{I})$	18.667	21.900	3.988	11.667	12.367	0.371	12.000	13.433	2.159	11.667	16.500	7.020
MC^2		3 components			6 components			9 components			12 components		
common components		min	mean	std	min	mean	std	min	mean	std	min	mean	std
JOINT	$N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$		22.333			13.833			13.833			13.333	
JOINT	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}))$		22.500			13.833			13.333			12.833	
COND	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}))$	17.000	17.700	0.686	11.500	12.800	1.451	12.000	12.375	0.247	11.500	12.567	0.772
JOINT	$N(\boldsymbol{\mu}_l, \sigma^2\boldsymbol{I})$		23.000			13.667			12.833			12.833	
COND	$N(\boldsymbol{\mu}_l, \sigma^2\boldsymbol{I})$	17.333	20.861	3.258	12.167	13.067	0.916	12.833	13.367	0.464	12.000	12.800	0.452
JOINT	$N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$		13.500			12.833			12.500			12.833	
JOINT	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}_l))$		13.333			12.500			12.500			12.167	
COND	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}_l))$	11.667	12.200	0.323	11.667	12.367	0.552	11.667	12.667	0.767	11.833	12.633	0.678
JOINT	$N(\boldsymbol{\mu}_l, \sigma_l^2\boldsymbol{I})$		20.833			12.167			12.833			11.833	
COND	$N(\boldsymbol{\mu}_l, \sigma_l^2\boldsymbol{I})$	15.333	17.033	2.825	12.333	12.733	0.374	12.500	13.200	0.562	11.833	12.667	0.606

Table 1: Error rates (%) for the Gaussian mixtures test set of size 600. For the JOINT routines the results are from one run and for the COND from 5 runs. The columns "min" denote the best run of the COND with respect to the error rate. The columns "mean" denote the mean error rate for the 5 runs for the COND and the error rate for the one JOINT run. The columns "std" give the standard deviation of the 5 runs for the COND routines. Pairwise comparisons have a 95 % confidence interval of half-length about 0.8 %. See the text for more details and comparisons.

(JOINT $N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma}_i)$ with (1,1,1) components) has an approximate 95 % confidence interval of [13.7 ± 2.8] %. Using an unpaired comparison the estimates based on the underlying MC^2 situation ($N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$ with (2,2,2) component architecture) is not significantly better than the quadratic discriminant analysis classifier. Using paired comparisons the difference is significant on a 5 % significance level.

It is difficult to draw general conclusions from the results in table 1. The trend is that for a given component architecture and distributional situation the COND routine achieves a lower misclassification rate on the test set than the JOINT routine. Since the COND routines are started from the result from the JOINT routine, it may be natural to assume that all the errors made by the classifier constructed maximizing the conditional likelihood (COND routine) is also made by the classifier maximizing the joint likelihood (JOINT routine). This opens for conclusions based on paired comparisons. For each MC^2 situation we may compare the mean error rate achieved using the conditional likelihood with the error rate achieved using the joint likelihood. In nearly all MC^2 situation the error rates using conditional likelihood (COND) is significantly better from using the joint likelihood (JOINT) using a 5% significance level.

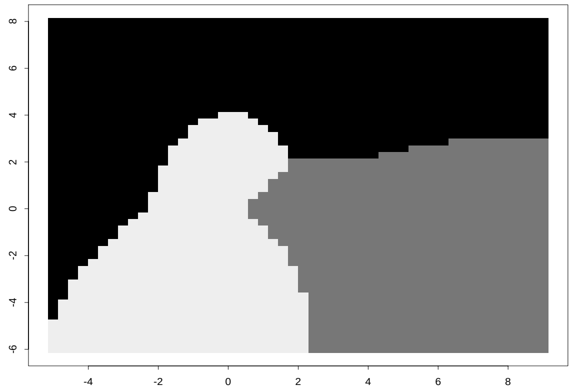
Figure 4 shows how some of the MC^2 routines divide the sample space for classification. This is meant as an aid for visualizing the class boundaries for the methods and does not reflect the sampling distribution of the classes. See Langaas (1995) for more graphical presentations.

5.3 The Forensic Glass Data Set

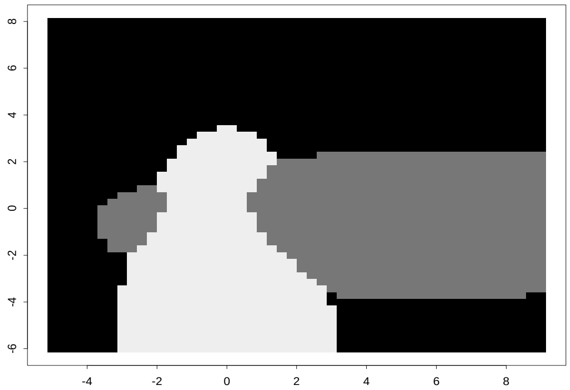
The forensic glass data set is collected by B. German and is described in Ripley (1994b). The data set consists of measurements on nine explanatory variables of 214 fragments of glass. The explanatory variables are refractive index and weight percent of oxides of sodium, magnesium, aluminium, silicon, potassium, calcium, barium and iron. The fragments were originally classified into seven groups, then one class was excluded (head-lamp glass which was found to be easy to discriminate) and the remaining six groups rearranged into four groups by Ripley (1994b); window float glass (WF), window non-float glass (WNF), vehicle window glass (VW) and other (O). The remaining observations were randomly split into a training set of size 89 and a test set of size 96.

The data is analysed in Ripley (1994b), where the lowest error rate on the test set was obtained using nearest neighbour classification with an approximate 95 % confidence interval of [26 ± 9] %. Allowing confusion between the two types of window glass, the best error rate on the test set was obtained using a classification tree, 95 % confidence interval of [15 ± 7.3] %. In Ripley (1994a) also predictive methods were included in the analysis and a lower error rate, 95 % confidence interval of [12 ± 6.6]% was achieved allowing confusion between the two types of window glass.

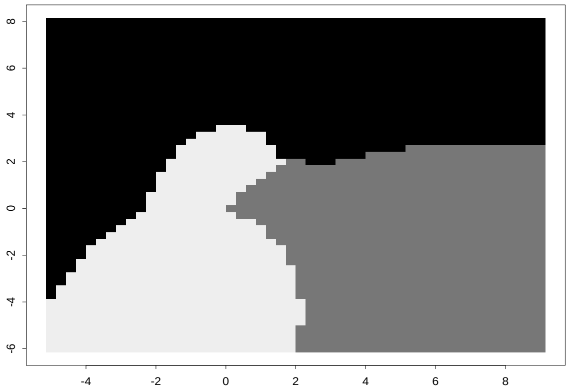
Here the data set is analysed by mixture codebook classification (MC^2) with Gaussian components and learning vector quantizer (LVQ). The fraction of observations from each class in the training set is (33.7 %, 43.8 %, 10.1 %, 12.4 %) and in the test set is (41.7 %, 38.5 %, 8.3 %, 11.5 %) for the classes (WF, WNF, VW, O). The class of vehicle window glass (VW) is too small to fit full class specific covariance matrices. Therefore only full common covariance matrices are tried. For the component specific and class specific covariance matrices only diagonal covariance matrices and diagonal covariance matrices with equal variances are investigated.



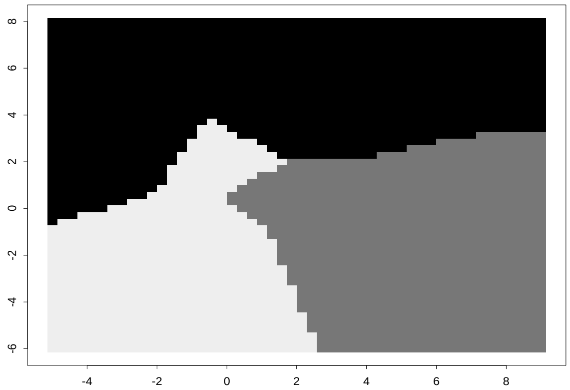
Bayes decision rule



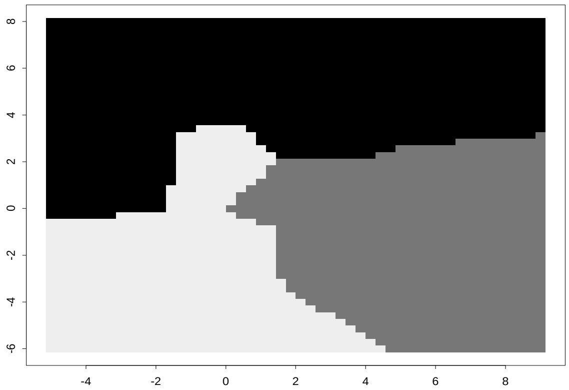
COND 3 components $N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}_l))$



JOINT (2,2,2) $N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$



COND (3,3,3) $N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$



COND (4,4,4) $N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$

Figure 4: Classification regions for the synthetic Gaussian data set. In the light grey areas of the plots observations are classified as class 1, the dark grey areas as class 2 and the black areas as class 3. COND denotes that the estimated classification regions are based on maximizing the conditional likelihood (28) and JOINT that the joint likelihood of section 4.2.1 is used. The component architecture is given as the number of components for common component and as a vector of the number of component for each class for class specific components.

Forensic Glass																
MC^2		(1,1,1,1) components			(1,1,1,2) components			(1,1,2,2) components			(2,2,2,1) components			(2,2,2,2) components		
specific components		min	mean	std	min	mean	std	min	mean	std	min	mean	std	min	mean	std
JOINT	$N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$		40.625			42.708			42.708			39.583			39.583	
JOINT	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$		51.042			47.917			47.917			47.917			45.833	
COND	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$	40.625	42.500	1.215	40.625	42.917	1.531	44.791	46.875	1.473	32.292	38.750	3.385	33.333	39.375	4.290
JOINT	$N(\boldsymbol{\mu}_{il}, \sigma^2 \boldsymbol{I})$		64.583			60.417			60.417			45.833			43.750	
COND	$N(\boldsymbol{\mu}_{il}, \sigma^2 \boldsymbol{I})$	41.667	43.334	1.413	36.458	40.417	3.511	38.542	40.833	2.412	36.458	38.542	1.743	36.458	41.667	3.090
JOINT	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$		52.083			51.042			58.333			48.958			48.958	
COND	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$	38.542	42.083	2.244	37.500	41.458	2.748	35.417	40.833	4.083	36.458	42.500	3.692	36.458	41.042	3.875
JOINT	$N(\boldsymbol{\mu}_{il}, \sigma_i^2 \boldsymbol{I})$		56.250			55.208			77.083			47.917			51.042	
COND	$N(\boldsymbol{\mu}_{il}, \sigma_i^2 \boldsymbol{I})$	50.000	60.417	8.565	50.000	56.667	3.762	58.333	66.042	11.851	56.250	64.167	12.265	39.583	57.083	12.137
MC^2		4 components			5 components			6 components			7 components			8 components		
common components		min	mean	std	min	mean	std	min	mean	std	min	mean	std	min	mean	std
JOINT	$N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$		51.042			47.917			47.917			44.792			43.750	
JOINT	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}))$		55.208			53.125			43.750			54.167			34.375	
COND	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}))$	33.333	38.750	3.692	34.375	38.750	2.826	36.458	40.000	2.517	31.250	42.708	10.032	34.375	34.896	0.521
JOINT	$N(\boldsymbol{\mu}_l, \sigma^2 \boldsymbol{I})$		50.000			50.000			44.792			43.750			43.750	
COND	$N(\boldsymbol{\mu}_l, \sigma^2 \boldsymbol{I})$	38.542	41.875	2.020	38.542	40.417	1.531	39.583	41.667	2.185	35.417	38.958	2.917	35.417	38.125	2.602
COND	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}_l))$	41.667	54.375	6.401	39.583	52.708	6.960	57.292	58.542	1.021	57.292	59.167	1.792	39.583	52.708	7.500
JOINT	$N(\boldsymbol{\mu}_l, \sigma_l^2 \boldsymbol{I})$		47.917			48.958			50.000			40.625			*	
COND	$N(\boldsymbol{\mu}_l, \sigma_l^2 \boldsymbol{I})$	29.167	36.250	4.029	35.417	37.500	1.141	33.333	41.667	8.590	32.292	36.667	2.976	33.333	38.958	3.062
LVQ		47.423	58.144	5.994	45.361	53.402	7.045	47.423	52.784	4.276	39.175	47.423	6.641	31.959	42.268	10.488

Table 2: Error rates (%) for the forensic glass test set (96 observations). The quoted rates are over five runs for COND and LVQ and one run for JOINT. The columns "min", "mean" and "std" give the minimum, mean and standard deviation of the 5 COND and LVQ runs. Pairwise comparisons have a 95 % confidence interval of half-length about 5.2 %. The $*$ denotes that no result was obtained.

Forensic Glass																
MC^2 specific components		(1,1,1,1) components			(1,1,1,2) components			(1,1,2,2) components			(2,2,2,1) components			(2,2,2,2) components		
		min	mean	std	min	mean	std	min	mean	std	min	mean	std	min	mean	std
JOINT	$N(\boldsymbol{\mu}_{il}, \boldsymbol{\Sigma})$		21.875			23.958			23.958			20.833			20.833	
JOINT	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$		19.791			19.791			19.791			15.625			14.583	
COND	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}))$	15.625	16.042	0.510	14.583	15.833	0.780	17.708	18.542	0.778	14.583	15.208	0.510	13.542	15.833	1.531
JOINT	$N(\boldsymbol{\mu}_{il}, \sigma^2\boldsymbol{I})$		20.833			18.750			18.750			14.583			14.583	
COND	$N(\boldsymbol{\mu}_{il}, \sigma^2\boldsymbol{I})$	14.583	16.042	1.062	13.542	15.208	1.062	14.583	17.292	1.559	11.458	14.375	1.531	11.458	16.458	2.585
JOINT	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$		19.792			18.750			29.167			16.667			15.625	
COND	$N(\boldsymbol{\mu}_{il}, diag(\boldsymbol{\Sigma}_i))$	20.833	22.917	1.318	12.500	18.542	3.632	14.583	18.958	3.572	14.583	21.667	4.340	16.667	21.042	3.187
JOINT	$N(\boldsymbol{\mu}_{il}, \sigma_i^2\boldsymbol{I})$		28.125			27.083			53.125			25.000			28.125	
COND	$N(\boldsymbol{\mu}_{il}, \sigma_i^2\boldsymbol{I})$	15.625	23.333	7.895	19.792	20.000	0.417	18.750	20.833	2.443	18.750	22.656	5.016	15.625	19.010	2.000
MC^2 common components		4 components			5 components			6 components			7 components			8 components		
		min	mean	std	min	mean	std	min	mean	std	min	mean	std	min	mean	std
JOINT	$N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma})$		17.708			18.750			16.667			14.583			14.583	
JOINT	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}))$		19.792			19.792			13.542			19.792			16.667	
COND	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}))$	11.458	16.667	2.872	16.667	17.500	0.780	16.667	18.333	1.062	12.500	21.667	8.524	15.625	16.146	0.521
JOINT	$N(\boldsymbol{\mu}_l, \sigma^2\boldsymbol{I})$		16.667			16.667			14.583			14.583			15.625	
COND	$N(\boldsymbol{\mu}_l, \sigma^2\boldsymbol{I})$	16.667	18.750	1.863	15.625	16.458	1.021	14.583	18.750	2.716	15.625	19.375	3.584	13.542	15.417	1.382
COND	$N(\boldsymbol{\mu}_l, diag(\boldsymbol{\Sigma}_l))$	14.583	16.042	0.833	13.541	17.708	2.872	15.625	16.875	1.021	15.625	17.500	1.792	16.667	17.292	0.833
JOINT	$N(\boldsymbol{\mu}_l, \sigma_l^2\boldsymbol{I})$		19.792			19.792			21.875			21.875			*	
COND	$N(\boldsymbol{\mu}_l, \sigma_l^2\boldsymbol{I})$	15.625	17.917	1.909	13.541	16.458	1.531	15.625	17.083	0.833	14.583	15.833	1.021	15.625	17.292	1.413
LVQ		15.625	18.958	1.863	14.583	17.500	2.696	16.667	19.167	1.580	13.542	18.333	4.270	17.708	18.958	0.872

Table 3: Error rates (%) for the forensic glass test set (96 observations) allowing confusion between the two types of window glass. The quoted rates are over five runs for COND and LVQ and one run for JOINT. The columns "min", "mean" and "std" give the minimum, mean and standard deviation of the 5 COND and LVQ runs. Pairwise comparisons have a 95 % confidence interval of half-length about 5.2 %. The $*$ denotes that no result was obtained.

The results of the analyses are found in table 2 giving misclassification rates on the test set for classification into the four classes. The best result, 95 % confidence interval of [29 ± 9.3] %, was obtained minimizing the conditional likelihood optimization criterion of equation (28) for four common components and constant diagonal component specific matrices of covariance. The results in table 3 give the error rates on the test set allowing confusion between the two types of window glass. The best result with a 95 % confidence interval of [11.5 ± 6.5] % was obtained minimizing the conditional likelihood optimization criterion of equation (28) for four common components and diagonal common component matrix of covariance. The same result was also obtained for component architectures (2,2,2,1) and (2,2,2,2) for components specific for each class with common constant diagonal matrix of covariance. Using both unpaired and paired comparisons it is clear that there is no significant difference between the best method quoted in Ripley (1994b) and Ripley (1994a) and the best MC^2 methods for both full classification and classification allowing confusion between the two types of window glass. Comparing the LDA with the best MC^2 method there is a significant difference if we use paired comparisons (but not for unpaired) for both full classification and classification allowing confusion between the two types of window glass.

The error rate (both on the training and on the test set) for the vehicle window glass (VW) is in most cases 100% both for the MC^2 and other methods for discrimination and classification that has been tried. When confusion between the two types of window glass is allowed, the MC^2 with common components for the classes give superior results to the situations with class specific components. The reason for this might be that no component is placed inside the VW class and therefore the number of components available for the other classes are larger than for class specific components.

6 Conclusions and Comments

A mixture codebook, $\mathcal{M} = \{\mathbf{\Theta}, f, K\}$, is a novel flexible framework from which to build classification rules. After specifying the component architecture, K, and the choice of component distributions, f, the problem of estimating the parameters and weights, $\mathbf{\Theta}$, of the mixture codebook remains. Two different approaches to estimating the parameters and weights are presented; maximizing the joint loglikelihood of the observation and their class membership, of section 4.2 and 3.5, and the criterion function optimization of section 4.3.

In this report only Gaussian component distributions are studied empirically. From the empirical comparisons between the two mixture codebook estimation approaches, based on the examples of section 4.4, 5.2 and 5.3, we find that the second approach (section 4.3) in general gives a lower misclassification rate on a separate test set than the first approach (section 4.2). It is also possible to combine these two approaches; for example by estimating a portion of the parameters using the first approach and estimating the remaining parameters and weights using the second approach. We also find that the MC^2 methods give good results compared to other methods for discrimination and classification. Many methods for discrimination and classification fit into the mixture codebook framework, among these are the linear discriminant analysis (LDA) and radial basis function networks (RBFN).

The decision boundaries created by the MC^2 method are in general flexible and non-linear. Linear boundaries can also be produced, since linear discriminant analysis (LDA) can be seen as a special case of the MC^2.

Since the total number of components are kept at a moderate level the MC^2 is a parametric method. By increasing the number of components the MC^2 will become "more nonparametric" in nature. When the number of components equals the number of observations the MC^2 will become more like a kernel discriminant method, with equality when the component centers are placed in each observation.

The optimization of the criterion functions of section 4.3 have caused some numerical difficulties. The number of iterations required to find a solution is large using the downhill simplex method implemented here. Compared to this methods the EM algorithm need a substantially smaller number of iterations to converge. In some situations problems are experienced for the optimization of the criterion functions because the current value of parameters and weights cause the marginal probability of some observations in the training set to be small (because the variances are too small), in a few cases below the precision used in the implemented routine. This problem can be overcome by penalizing such solutions. For the empirical studies of this report in a few situations the EM algorithm has ended up with a singular covariance matrix, but convergence has in most cases been to a stationary point of the loglikelihood with positive definite covariance matrices. The quadratic criterion (29) and the Kullback-Leibler divergence (30) have not been studied thoroughly in the empirical analyses, but we believe that the performance of these criteria are similar to the performance of the conditional likelihood criterion (28).

Many questions remain unanswered and need to be investigated further. Among those are the topics of model selection, interpretability, robustness, speed of convergence, choice of starting values for the EM algorithm and regularization. Model selection includes both the problem of selecting the component architecture and the component distributions. The choice of component architecture is of special importance when the components have a direct physical interpretation, as in Roeder (1990). Other choices of component distributions could include the multivariate Student-t distribution which would provide heavier elliptical tails. There are many solutions to the problem of regularization, see for example Wabha (1994). These includes early stopping, see Sjöberg (1993), and penalization, see Friedman (1994). More work is also needed on combining the two mixture codebook classification approaches of section 4. Some of these questions are topics of further study by the author.

Acknowledgements

I would like to thank to my supervisor Bo Lindqvist for helpful discussions and suggestions. Thanks to Brian D. Ripley for valuable comments on an earlier version of this report and for advocating the use of the optimization criterion in (28). Thanks to Håvard Rue for help with the C and shell programming. This report was written while the author was funded by a Dr. Ing. grant from the Norwegian Research Council.

A The EM algorithm for a Two Level Mixture with Class Specific Components

The training set consists of observations and encoded class memberships, $\mathcal{T} = (\boldsymbol{x}_j, \boldsymbol{z}_j)_{j=1}^n$. As before $\boldsymbol{z}_j$ denotes the encoded class of observation $\boldsymbol{x}_j$. The data can be regarded as incomplete, the component membership is missing. The component membership is encoded as $\boldsymbol{æ}_j$, where $\boldsymbol{æ}_j$ is a set of indicator variables for the component with one in the position corresponding to the appropriate class and component and zeros elsewhere,

$$æ_{jil} = \begin{cases} 1 & \text{if } \boldsymbol{x}_j \text{ is a member of the component } c_{il} \text{ inside class } g_i \\ 0 & \text{else} \end{cases} .$$

We start by finding the loglikelihood for the *complete data set*. The parameters are denoted $\boldsymbol{\Theta} = \{(\boldsymbol{\theta}_{il}, p(c_{il}|g_i), p(g_i))$ for $i = 1, .., M$ and $l = 1, ..., k_i\}$.

$$l = \log(\prod_{j=1}^{n} p(\boldsymbol{x}_j, \boldsymbol{z}_j, \boldsymbol{æ}_j))$$

$$= \sum_{j=1}^{n} \{(\sum_{i=1}^{M} \sum_{l=1}^{k_i} z_{ji} æ_{jil} \log p(\boldsymbol{x}_j | g_i, c_{il})) + (\sum_{i=1}^{M} \sum_{l=1}^{k_i} z_{ji} æ_{jil} \log p(c_{il}|g_i)) + (\sum_{i=1}^{M} z_{ji} \log p(g_i))\}$$

The EM algorithm alternates between averaging the loglikelihood, l, over the unobserved component membership, $\boldsymbol{æ}_j$, and maximizing the averaged loglikelihood over the parameters and weights, $\boldsymbol{\Theta}$. That is, the expectation step of the algorithm (E-step) calculates the expectation of the complete data loglikelihood given the data $\mathcal{T} = (\boldsymbol{x}_j, \boldsymbol{z}_j)_{j=1}^{n}$ and the current estimate of the parameters and weights, $\boldsymbol{\Theta}^{(s)}$.

The component membership is regarded as a stochastic variable.

$$E(æ_{jil} | \mathcal{T}, \boldsymbol{\Theta}^{(s)}) = p(c_{il} | g_i, \boldsymbol{x}_j)^{(s)}$$

$$E(l | \mathcal{T}, \boldsymbol{\Theta}^{(s)}) = \sum_{j=1}^{n} \sum_{i=1}^{M} \sum_{l=1}^{k_i} p(c_{il} | g_i, \boldsymbol{x}_j)^{(s)} z_{ji} \log f_{il}(\boldsymbol{x}_j | g_i, c_{il})$$

$$+ \sum_{j=1}^{n} \sum_{i=1}^{M} \sum_{l=1}^{k_i} p(c_{il} | g_i, \boldsymbol{x}_j)^{(s)} z_{ji} \log p(c_{il} | g_i) + \sum_{j=1}^{n} \sum_{i=1}^{M} z_{ji} \log p(g_i)$$

The maximization step (M-step) consists in maximizing the function $E(l | \mathcal{T}, \boldsymbol{\Theta}^{(s)})$ with respect to $\boldsymbol{\Theta}$ under the constraints that the mixing weights and the class weights sum to one, $\sum_{i=1}^{M} p(g_i) = 1$ and $\sum_{l=1}^{k_i} p(c_{il} | g_i) = 1$ for $i = 1, ..., M$. This leads to the following M-step equations.

$$\sum_{j=1}^{n} z_{ji} p(c_{il} | g_i, \boldsymbol{x}_j)^{(s)} \frac{\partial \log f_{il}(\boldsymbol{x}_j | \boldsymbol{\theta}_{il})}{\partial \boldsymbol{\phi}_{il}} = 0 \qquad for \quad i = 1, ..., M \quad and \quad l = 1, ..., k_i$$

$$\sum_{j=1}^{n} z_{ji} \sum_{l=1}^{k_i} p(c_{il} | g_i, \boldsymbol{x}_j)^{(s)} \frac{\partial \log f_{il}(\boldsymbol{x}_j | \boldsymbol{\theta}_{il})}{\partial \boldsymbol{\phi}_i} = 0 \qquad for \quad i = 1, ..., M$$

$$\sum_{j=1}^{n} \sum_{i=1}^{M} z_{ji} \sum_{l=1}^{k_i} p(c_{il} | g_i, \boldsymbol{x}_j)^{(s)} \frac{\partial \log f_{il}(\boldsymbol{x}_j | \boldsymbol{\theta}_{il})}{\partial \boldsymbol{\phi}} = 0$$

$$p(g_i) = \frac{n_i}{n} \qquad for \quad i = 1, ..., M$$

$$p(c_{il}|g_i) = \frac{1}{n_i}\sum_{j=1}^{n} z_{ji} p(c_{il}|g_i, \boldsymbol{x}_j)^{(s)} \qquad for \quad i = 1, ..., M \quad and \quad l = 1, ..., k_i$$

We see that these equation are equivalent to the update rules of the alternating likelihood estimation algorithm of section 3.5 for the two level mixture with class specific components.

B Methods Used for Initialization

When using the EM algorithm to estimate the parameters and weights of mixture models we have in section 3.5 seen the importance of choosing a good starting point for the algorithm. Two methods that can be used to find initial estimates are the K-means and the Learning Vector Quantization (LVQ). We will here present these two methods briefly. We will also present the fuzzy K-means and fuzzy maximum likelihood algorithm because of their strong relationship to the K-means algorithm and the alternating likelihood estimation algorithm (EM algorithm) for mixtures.

B.1 K-means

The K-means algorithm dates back to MacQueen (1967). In the vector quantization (VQ) literature the K-means algorithm is called the generalized Lloyd algorithm (GLA), see Gersho and Gray (1992). In MacQueen (1967) only Euclidean distance measure is used, but the general K-means algorithm allows for other distance measures than the Euclidean distance, see Hartigan (1975). Modified versions of the K-means algorithm are presented later in this section.

The K-means method operates on a one level mixture with K components, where K is known. When K-means is to be used on a two-level mixture then one can use K-means on each class separately for the situation with class specific components and use K-means on the total data set for the situation with common components. The basic K-means method is outlined below for a one level mixture.

Aim Divide observations into K clusters such that the observations within a cluster are closer to one another than the observations in different clusters. Close is defined in terms of a distance measure. Usually the squared Euclidean distance measure is used under the assumption that the variables are scaled so that the Euclidean distance between observations is appropriate. The number of clusters, K, is assumed known.

Notation The K-means algorithm is constructed to find both the cluster centroids, $\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K$, and the membership matrix, $\boldsymbol{Z} = \{z_{jl}\}$. The membership matrix is defined so that $z_{jl} = 1$ if the observation $\boldsymbol{x}_j$ is closest to the cluster c_l with centroid $\boldsymbol{\mu}_l$, otherwise $z_{jl} = 0$. The number of observations assigned to cluster c_l is denoted $N(c_l)$.

Minimization Problem

$$J = \sum_{j=1}^{n}\sum_{l=1}^{K} z_{jl} d(\boldsymbol{\mu}_l, \boldsymbol{x}_j)$$

The aim is to find a partition of the observations that minimizes the criterion J. Here $d(\cdot,\cdot)$ is a chosen distance measure. Observations are "moved" from one cluster to another and the cluster centres are updated until no such movements reduce J.

The General K-means Algorithm This algorithm is also called adaptive K-means.

1. Choose K initial cluster centres $\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K$. MacQueen (1967) choose the first K observations as the initial cluster centres. Another possibility is to choose K observations randomly and use these as initial centres. See Anderberg (1973) for other suggestions.
2. Calculate the distance from each observation to each cluster centre. Assign each observation to the nearest cluster centre (using the appropriate distance measure). This produces the membership matrix, $\boldsymbol{Z}$. This obeys what is called "the nearest neighbour condition"; if $\boldsymbol{x}$ has $\boldsymbol{\mu}_l$ as its unique nearest neighbour among the set of cluster centres, then the observation $\boldsymbol{x}$ must be a member of the cluster with centre $\boldsymbol{\mu}_l$.
3. Recalculate the cluster centres as the cluster centroids for the cluster receiving an observation and for the cluster losing an observation. This obeys what is called "the centroid condition"; the optimal centre of the cluster is the centroid.
4. If a reassignment has taken place, goto step 2.

It is also possible to partition the observations into K initial clusters instead of choosing initial cluster centres. Hartigan (1975) initializes the observations into K clusters based on the sum over all variables for each observation. Let Min denote the minimum sum and Max the maximum sum over all variables for each observation. Observation $\boldsymbol{x}_j$, with sum over all variables equal to $sum(\boldsymbol{x}_j)$, is assigned to the cluster $[\frac{K(sum(\boldsymbol{x}_j)-Min)}{Max-Min}+1]$ where $[a]$ denotes the integer part of a.

Optimality Conditions Gersho and Gray (1992), chapter 11, define three conditions that are necessary for optimality (minimum J); the nearest neighbour condition, the centroid condition and the zero-probability boundary condition. The zero-probability boundary condition states for a given distribution of the observations that the boundary points between the clusters occur with zero probability (with continuous random variables this is always satisfied). The set of cluster centres is locally optimal if every small perturbation of the cluster centres does not lead to a decrease in the distortion criterion, J. It is globally optimal if there exists no other set of cluster centres that gives a lower value of J.

A set of cluster centres satisfying the nearest neighbour condition, the centroid condition and the zero-probability boundary condition is widely believed to be *locally* optimal, but no theoretical derivation of this result has ever been obtained, Gersho and Gray (1992).

Relation to the EM algorithm for mixtures Consider a one level mixture where the component distributions are Gaussian with $\boldsymbol{\theta}_l = (\boldsymbol{\mu}_l, \boldsymbol{\Sigma} = \boldsymbol{I})$. Let the mixing weights, $p(c_l)$, only take binary values. That is, the mixing weights are treated as membership functions. This means that we assume that all components are well-separated within each class and that the distance used is the Euclidean distance. Under these conditions the alternating likelihood estimation (EM) algorithm for a one level mixture is reduced to the basic K-means algorithm.

K-means using Steepest Descent A variant of the adaptive K-means algorithm uses steepest descent gradient-step optimization of the error function, J, Moody and Darken (1989) and Kohonen (1990). Initial cluster centres are randomly chosen from the observations. At each time step, an observation, $\boldsymbol{x}$ is chosen and the nearest cluster centre, $\boldsymbol{\mu}$, is moved by an amount

$$\triangle\boldsymbol{\mu} = \eta(\boldsymbol{x} - \boldsymbol{\mu})$$

where η is the learning rate. (That is, an observation is chosen randomly without replacement at each time step until all observations are chosen, then the procedure is repeated.) The learning rate can be adaptively chosen at each time step, $\eta = \eta(t)$, see Moody and Darken (1989) and Kohonen (1990) for more details.

K-means with an Adaptive Distance Measure Instead of using the Euclidean distance measure in the minimization problem the Mahalanobis distance can be used. This requires that the covariance matrices for the clusters are updated together with the cluster centres in step 3 in the general K-means algorithm. The following update rules are applied. Let the covariance matrix for cluster c_l be denoted $\boldsymbol{\Sigma}_l$.

$$\boldsymbol{\Sigma}_l = \frac{1}{N(c_l)} \sum_{j=1}^{n} z_{jl}(\boldsymbol{x}_j - \boldsymbol{\mu}_l)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T$$

If all clusters have equal covariance matrices the update rule is modified to

$$\boldsymbol{\Sigma} = \frac{1}{n} \sum_{j=1}^{n} \sum_{l=1}^{K} z_{jl}(\boldsymbol{x}_j - \boldsymbol{\mu}_l)(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T$$

We see that these equations are equal to the likelihood equations for a one level Gaussian mixture with known component membership, which can be found as a special case of the equations given in the end of section 3.4.

K-means with Variable Number of Clusters MacQueen (1967) has a version of the basic K-means algorithm which includes coarsening and refining parameters. The method chooses the number of clusters with initial cluster centres so that no centres are closer to each other than the coarsening parameter and no centres are further apart than the refining parameter. In the ISODATA algorithm, see for example Anderberg (1973), the number of clusters are chosen adaptively using an iterative scheme. Lumping and splitting actions depend on the number of clusters, the distance between cluster centres and the within cluster standard deviation for each variable.

The K-means Algorithm of Hartigan and Wong (1979) Hartigan and Wong (1979) present an efficient version of the K-means algorithm. The algorithm seek solutions such that no movement of a point from one cluster to another will reduce the within-cluster sum of squares.

B.2 Fuzzy K-means and Fuzzy Maximum Likelihood Algorithms

In fuzzy K-means the criterion to minimize is

$$J_q = \sum_{j=1}^{n} \sum_{l=1}^{K} (u_{jl})^q d(\boldsymbol{\mu}_l, \boldsymbol{x}_j)$$

where u_{jl} gives the degree of membership for observation $\boldsymbol{x}_j$ to cluster c_l. The constant q is the weighting exponent for u_{jl} and controls the fuzziness of the resulting clusters, Bezdek and Dunn (1975).

Fuzzy Algorithm

1. Choose K initial cluster centres.
2. The degree of membership, u_{jl}, is calculate from

$$u_{jl} = \frac{\left(\frac{1}{d(\boldsymbol{\mu}_l, \boldsymbol{x}_j)}\right)^{\frac{1}{q-1}}}{\sum_{l=1}^{K} \left(\frac{1}{d(\boldsymbol{\mu}_l, \boldsymbol{x}_j)}\right)^{\frac{1}{q-1}}}$$

3. Recalculate the cluster centres

$$\boldsymbol{\mu}_l = \frac{\sum_{j=1}^{n} (u_{jl})^q \boldsymbol{x}_j}{\sum_{j=1}^{n} (u_{jl})^q}$$

   and update the degree of membership using the equation from step 2.
4. If the maximal change in the the degree of membership is larger than a predefined value goto step 3.

If the distance measure is the Euclidean distance the above algorithm is the *fuzzy K-means algorithm.*

The fuzzy maximum likelihood algorithm is based on the algorithm listed above using a distance measure proportional to the inverse of the Gaussian density, Bezdek and Dunn (1975). That is, the distance measure is given as $d_e(\boldsymbol{\mu}_l, \boldsymbol{x}_j)$.

$$d_e(\boldsymbol{\mu}_l, \boldsymbol{x}_j) = \frac{|\boldsymbol{\Sigma}_l|^{\frac{1}{2}}}{p(c_l)} \exp\left(\frac{1}{2}(\boldsymbol{x}_j - \boldsymbol{\mu}_l)^T \boldsymbol{\Sigma}_l^{-1} (\boldsymbol{x}_j - \boldsymbol{\mu}_l)\right)$$

Here $p(c_l)$ is the mixing weight for cluster c_l. In addition the degree of membership, u_{jl}, is substituted by the probability $p(c_l|\boldsymbol{x}_j)$.

$$p(c_l|\boldsymbol{x}_j) = \frac{\frac{1}{d_e(\boldsymbol{\mu}_l, \boldsymbol{x}_j)}}{\sum_{l=1}^{K} \frac{1}{d_e(\boldsymbol{\mu}_l, \boldsymbol{x}_j)}}$$

This equals the likelihood equation (9). The mixing weight $p(c_l)$ is updated using equation (8). The covariance matrices are updated using equation (23). Gath and Geva (1989) combine the fuzzy K-means and the fuzzy maximum likelihood algorithm sequentially.

It is easy to see that the fuzzy maximum likelihood algorithm equals the alternating likelihood estimation algorithm for a one level Gaussian mixture.

We see that the formula for $p(c_l|\boldsymbol{x}_j)$ equals the formula for u_{jl} when $q = 2$. When $q \neq 2$ the fuzzy algorithm is difficult to interpret form a statistical point of view.

B.3 Learning Vector Quantizer

Learning Vector Quantizer (LVQ) is a supervised neural network method developed by T. Kohonen, see for example Kohonen (1990). There exists different LVQ variants; LVQ1, LVQ2.1, LVQ3 and OLVQ1, all described in Kohonen et al. (1992). The presentation here is based on these two references.

Aim LVQ is a supervised classification method. A *codebook* is created from the training set. The codebook is used in place of the training set for first nearest codeword classification.

General Algorithm for all LVQ methods. The distance measure is Euclidean distance, so it is assumed that the observations are scaled in an appropriate manner.

1. Initialization and categorizing (also called labelling) of the codebook: Two different approaches can be taken. The first is to find a codebook using K-means (or related methods) and then classify each codeword using kth nearest neighbour classification on the training set. The second approach is to choose an initial codebook using the observations in the training set, specifying the number of codewords within each class.
2. Move codewords to achieve a better nearest codeword classification (demarcate the class borders more accurately): The recommended procedure is to run OLVQ1 until convergence and then a few steps of LVQ1 or LVQ3. LVQ2 is only for use in a small time interval. For each method an observation $\boldsymbol{x}$ is chosen randomly without replacement from the training set until all observations have been chosen. Then the procedure is repeated. A codeword is denoted $\boldsymbol{c}$.
   - LVQ1: only the nearest codeword to the observation, $\boldsymbol{x}$, is updated.

     $$\boldsymbol{c}_{closest} \leftarrow \boldsymbol{c}_{closest} \pm \alpha(t)[\boldsymbol{x} - \boldsymbol{c}_{closest}]$$

     The positive sign is used if the observation $\boldsymbol{x}$ and the closest codeword $\boldsymbol{c}_{closest}$ is of the same class (moving the codeword towards the observation), else the negative sign is used (moving the codeword away from the observation). A time dependent learning rate, $\alpha(t)$, is used for all codewords.
   - OLVQ1: an optimized modification of LVQ1 using separate time dependent learning rates for each codebook.
   - LVQ2.1: the two nearest codewords are updated if the observation is inside a window of fixed size and if only one of the codewords are of the same class as the observation. The window is of relative width w and is defined by the distances d_i and d_j from $\boldsymbol{x}$ to the two closest codewords $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$. The observation is inside the window if

     $$min(\frac{d_i}{d_j}, \frac{d_j}{d_i}) > \frac{1-w}{1+w}$$

The two closest codewords are updated:

$$\boldsymbol{c}_l \leftarrow \boldsymbol{c}_l \pm \epsilon\alpha(t)[\boldsymbol{x} - \boldsymbol{c}_l]$$

The positive sign is used for the codeword $\boldsymbol{c}_l$ of the same class as the observation and the negative sign for the codeword $\boldsymbol{c}_l$ not of the same class as the observation.

- LVQ3: the two nearest codewords are updated if they are inside a window of fixed size. The window is the same as in LVQ2.1 and the codewords are updated as in LVQ2.1 if only one of the codewords are of the same class as the observation. Here the two nearest codewords are also updated if they both have the same class as the observation:

$$\boldsymbol{c}_l \leftarrow \boldsymbol{c}_l + \epsilon\alpha(t)[\boldsymbol{x} - \boldsymbol{c}_l]$$

where ϵ depends on the size of the window. No update is made if both the two closest codewords have different classes from the observation.

3. After the codebook is constructed, an observation is classified to the class of the nearest codeword.

The class boundaries of the LVQ classification method are piecewise linear because they are composed by midplanes between codewords of neighbouring classes, they are the borders of the Dirichlet (Voroni) tessellation with the codewords as the centroids. The LVQ method is easily modified to allow for kth nearest codeword classification.

References

Anderberg, M. R. (1973), *Cluster Analysis for Applications*, New York: Academic Press.

Bernardo, J. M. and Girón, F. J. (1988), "A Bayesian analysis of simple mixture problems", in Bernardo, J. M., DeGroot, M. H., Lindley, D. V., and Smith, A. F. M., editors, *Bayesian Statistics 3*, pages 67–78, Oxford: Oxford University Press.

Bezdek, J. C. and Dunn, J. C. (1975), "Optimal fuzzy partitions: A heuristic for estimating the parameters in a mixture of normal distributions", *IEEE Transactions on Computers*, C-24, 835–838.

Boyles, R. A. (1983), "On the convergence of the EM algorithm", *Journal of the Royal Statistical Society, Series B*, 45, 47–50.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984), *Classification and Regression Trees*, Montery: Wadsworth and Brooks/Cole.

Broomhead, D. S. and Lowe, D. (1988), "Multivariable functional interpolation and adaptive networks", *Complex Systems*, 2, 321–355.

Chou, W. S. and Chen, Y. C. (1992), "A new fast algorithm for effective training of neural classifiers", *Pattern Recognition*, 25, 423–429.

Crawford, S. L. (1994), "An application of the Laplace method to finite mixture distributions", *Journal of the American Statistical Association*, 89, 259–267.

Dasarathy, B. V. (1991), *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*, Los Alamitos, California: IEEE Computer Society Press.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum likelihood from incomplete data via the EM algorithm (with discussion)", *Journal of the Royal Statistical Society, Series B*, 39, 1–38.

Diebolt, J. and Robert, C. P. (1994), "Estimation of finite mixture distributions through Bayesian sampling", *Journal of the Royal Statistical Society, Series B*, 56, 363–375.

Duda, R. O. and Hart, P. E. (1973), *Pattern Classification and Scene Analysis*, New York: Wiley.

Everitt, B. S. and Hand, D. J. (1981), *Finite Mixture Distributions*, London: Chapman & Hall.

Fix, E. and Hodges, J. L. (1951), "Discriminatory analysis - nonparametric discrimination: consistency properties", Report Number 4, U.S. Air Force School of Aviation Medicine, Randolf Fields, Texas, reprinted in Dasarathy (1991).

Friedman, J. H. (1991), "Multivariate adaptive regression splines (with discussion)", *The Annals of Statistics*, 19, 1–141.

Friedman, J. H. (1994), "Adaptive techniques for machine learning and function approximation", in Cherkassky, V., Friedman, J. H., and Wechsler, H., editors, *From Statistics to Neural Networks. Theory and Pattern Recognition Applications*, ASI Proceedings, Series F: Computer and Systems Sciences, pages 1–82, New York: Springer-Verlag.

Friedman, J. H. and Stuetzle, W. (1981), "Projection pursuit regression", *Journal of the American Statistical Association*, 76, 817–823.

Gath, I. and Geva, A. B. (1989), "Unsupervised optimal fuzzy clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11, 773–781.

Gersho, A. and Gray, R. M. (1992), *Vector Quantization and Signal Compression*, Boston: Kluwer Academic Publications.

Gill, P. E., Murray, W., and Wright, M. H. (1981), *Practical Optimization*, London: Academic Press.

Girosi, F. (1994), "Regularization theory, radial basis functions and networks", in Cherkassky, V., Friedman, J., and Wechsler, H., editors, *From Statistics to Neural Networks. Theory and Pattern Recognition Applications*, NATO ASI Series F: Computer and Systems Sciences, vol. 136, pages 166–187, New York: Springer-Verlag.

Hand, D. J. (1982), *Kernel Discriminant Analysis*, Chichester: Research Studies Press.

Hartigan, J. A. (1975), *Clustering Algorithms*, New York: Wiley.

Hartigan, J. A. and Wong, M. A. (1979), "A K-means clustering algorithm", *Applied Statistics*, 28, 100–108.

Hasselblad, V. (1966), "Estimation of parameters for a mixture of normal distributions", *Technometrics*, 8, 431–444.

Hastie, T. and Tibshirani, R. (1995), "Discriminant analysis by Gaussian mixtures", *Journal of the Royal Statistical Society, Series B*, in press, available via WWW at URL http://playfair.standford.edu/reports/hastie.

Hastie, T., Tibshirani, R., and Buja, A. (1994), "Flexible discriminant analysis by optimal scoring", *Journal of the American Statistical Association*, 89, 1255–1270.

Hertz, J., Krogh, A., and Palmer, R. G. (1991), *Introduction to the Theory of Neural Computation*, Redwood City, California: Addison-Wesley.

Hush, D. R. and Horne, B. G. (1993), "Progress in supervised neural networks", *IEEE Signal Processing Magazine*, January, 8–39.

Ingrassia, S. (1992), "A comparison between the simulated annealing and the EM algorithms in normal mixture decompositions", *Statistics and Computing*, 2, 203–211.

Johnson, R. A. and Wichern, D. W. (1982), *Applied Multivariate Statistical Analysis*, Englewood Cliffs, New Jersey: Prentice-Hall, 2nd edition.

Kohonen, T. (1990), "The self-organizing map", *Proceedings of the IEEE*, 78, 1464–1480, reprinted in Lau (1992).

Kohonen, T., Kangas, J., Laaksonen, J., and Torkkola, K. (1992), *LVQ PAK. The learning vector quantization package version 2.1*, LVQ Programming Team of the Helsinki University of Technology and Laboratory of Computer and Information Science, Espoo.

Langaas, M. (1995), "Discrimination and classification", Technical Report Statistics 1/95, Department of Mathematical Sciences, The Norwegian Institute of Technology, The University of Trondheim.

Lau, C., editor (1992), *Neural Network, Theoretical Foundations and Analysis*, Piscataway: IEEE Press, includes papers published in two IEEE proceedings special issues and in other publications from 1986–1990.

Lippmann, R. P. (1994), "Neural networks, Bayesian a posteriori probabilities, and pattern classification", in Cherkassky, V., Friedman, J. H., and Wechsler, H., editors, *From Statistics to Neural Networks. Theory and Pattern Recognition Applications*, NATO ASI Series F: Computer and Systems Sciences, vol. 136, pages 83–104, New York: Springer-Verlag.

MacQueen, J. B. (1967), "Some methods for classification and analysis of multivariate observations", in Le Cam, L. M. and Neyman, J., editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, Berkeley, California: University of California Press.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979), *Multivariate Analysis*, London: Academic Press.

McLachlan, G. J. (1992), *Discriminant Analysis and Statistical Pattern Recognition*, New York: John Wiley & Sons.

McLachlan, G. J. and Basford, K. E. (1988), *Mixture Models, Inference and Applications to Clustering*, New York: Marcel Dekker.

Moody, J. E. and Darken, C. J. (1989), "Fast learning in networks of locally-tuned processing units", *Neural Computation*, 1, 281–294.

Panel on Discriminant Analysis, Classification and Clustering (1989), "Discriminant analysis and clustering", *Statistical Science*, 4, 34–69.

Pearson, K. (1894), "Contributions to the mathematical theory of evoluation", *Philosophical Transactions of the Royal Society of London, Series A*, 185, 71–110, cited in McLachlan and Basford (1988).

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1988), *Numerical Recipes in C*, Cambridge: Cambridge University Press, 1st edition.

Rao, C. R. (1948), "The utilization of multiple measurements in problems of biological classification", *Journal of the Royal Statistical Society, Series B*, 10, 159–203.

Redner, R. A. and Walker, H. F. (1984), "Mixture densities, maximum likelihood and the EM algorithm", *SIAM Review*, 26, 195–239.

Ripley, B. D. (1993), "Statistical aspects of neural networks", in Barndorff-Nielsen, O. E., Jensen, J. L., and Kendall, W. S., editors, *Networks and Chaos - Statistical and Probabilistic Aspects*, pages 40–123, London: Chapman & Hall.

Ripley, B. D. (1994a), "Flexible non-linear approaches to classification", in Cherkassky, V., Friedman, J. H., and Wechsler, H., editors, *From Statistics to Neural Networks. Theory and Pattern Recognition Applications*, NATO ASI Series F: Computer and Systems Sciences, vol. 136, pages 105–126, New York: Springer-Verlag.

Ripley, B. D. (1994b), "Neural networks and related methods for classification (with discussion)", *Journal of the Royal Statistical Society, Series B*, 56, 409–456.

Roeder, K. (1990), "Density estimation with confidence sets exemplified by superclusters and voids in galaxies", *Journal of the Americal Statistical Association*, 85, 617–624.

Seber, G. A. F. (1984), *Multivariate Observations*, New York: Wiley.

Silverman, B. W. (1986), *Density Estimation for Statistics and Data Analysis*, London: Chapman & Hall.

Sjöberg, J. (1993), "Regularization issues in neural network models of dynamical systems", Licentiate Thesis No. 366, Department of Electrical Engineering, Linköping University.

Specht, D. F. (1990), "Probabilistic neural networks", *Neural Networks*, 3, 109–118.

Streit, R. L. and Luginbuhl, T. E. (1994), “Maximum likelihood training of probabilistic neural networks”, *IEEE Transactions on Neural Networks*, 5, 764–783.

Tapia, R. A. and Thompson, J. R. (1978), *Nonparametric Probability Density Estimation*, Baltimore/London: The Johns Hopkins University Press.

Taxt, T., Hjort, N. L., and Eikvil, L. (1991), “Statistical classification using a linear mixture of two multinormal probability densities”, Technical Report 5/1991, Institute of Mathematics, University of Oslo.

Titterington, D. M., Smith, A. F. M., and Makov, U. E. (1985), *Statistical Analysis of Finite Mixture Distributions*, Chichester: John Wiley & Sons.

Tråvén, H. G. C. (1991), “A neural network approach to statistical pattern classification by “semiparametric” estimation of probability density functions”, *IEEE Transactions on Neural Networks*, 2, 366–377.

Wabha, G. (1994), “Generalization and regularization in nonlinear learning systems”, Technical Report 921, Department of Statistics, University of Wisconsin, Madison.

Wu, C. F. J. (1983), “On the convergence properties of the EM algorithm”, *The Annals of Statistics*, 11, 95–103.

Xu, L., Krzyzak, A., and Yuille, A. (1994), “On radial basis function nets and kernel regression: statistical consistency, convergence rates, and receptive field size”, *Neural Networks*, 7, 609–628.