RICE UNIVERSITY

Gaussian Mixture Regression and Classification

by

Hsi Guang Sung

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Doctor of Philosophy

Approved, Thesis Committee:

David W. Scott, Chairman
Noah G. Harding Professor of Statistics

Dennis D. Cox
Professor of Statistics

Danny C. Sorensen
Noah G. Harding Professor of
Computational and Applied Mathematics

Houston, Texas

May, 2004

Abstract

Gaussian Mixture Regression and Classification

by

Hsi Guang Sung

The sparsity of high dimensional data space renders standard nonparametric methods ineffective for multivariate data. A new procedure, Gaussian Mixture Regression (GMR), is developed for multivariate nonlinear regression modeling. GMR has the tight structure of a parametric model, yet still retains the flexibility of a nonparametric method.

The key idea of GMR is to construct a sequence of Gaussian mixture models for the joint density of the data, and then derive conditional density and regression functions from each model. Assuming the data are a random sample from the joint pdf $f_{X,Y}$, we fit a Gaussian kernel density model $\widehat{f}_{X,Y}$ and then implement a multivariate extension of the Iterative Pairwise Replacement Algorithm (IPRA) to simplify the initial kernel density. IPRA generates a sequence of Gaussian mixture density models indexed by the number of mixture components K. The corresponding regression function of each density model forms a sequence of regression models which covers a spectrum of regression models of varying flexibility, ranging from approximately the classical linear model ($K = 1$) to the nonparametric kernel regression estimator ($K = n$). We use mean squared error and prediction error for selecting K.

For binary responses, we extend GMR to fit nonparametric logistic regression models. Applying IPRA for each class density, we obtain two families of mixture density models. The logistic function can then be estimated by the ratio between pairs of members from each family. The result is a family of logistic models indexed by the number of mixtures in each density model. We call this procedure Gaussian Mixture Classification (GMC).

For a given GMR or GMC model, forward and backward projection algorithms are implemented to locate the optimal subspaces that minimize information loss. They serve as the model-based dimension reduction techniques for GMR and GMC.

In practice, GMR and GMC offer data analysts a systematic way to determine the appropriate level of model flexibility by choosing the number of components for modeling the underlying pdf. GMC can serve as an alternative or a complement to Mixture Discriminant Analysis (MDA). The uses of GMR and GMC are demonstrated in simulated and real data.

Acknowledgments

My first gratitude goes to my advisor, Dr. David Scott, who has been instrumental throughout the whole journey. Dr. Dennis Cox's criticism and pointed questions kept me honest and dug deep. Dr. Danny Sorensen and Dr. Mark Embree inspired me to look at statistical problems from a numerical analyst's perspective. That proved to be a fruitful angle. Dr. Jan Hewitt's professional editing made this thesis readable. Of course, I am solely responsible for any remaining blunder. My officemate, James Blair Christian, has been a tremendous resource.

I am forever in debt to the heros and heroines of the LaTeXcommunity. To name just a few: Donald Knuth started the revolution by introducing TeXto the world; Leslie Lamport implemented the user-friendly LaTeX; Diane Jamrog, Genetha Gray, Jennifer Wightman, and Daniel Reynolds created the thesis template and generously shared it with their fellow Rice graduate students. All in all, they made mathematical writing fun.

Finally, this research was supported in part by the National Science Foundation grants NSF EIA-9983459 (digital government) and DMS 02-04723 (non-parametric methodology).

"Always take a job that is too big for you."
Harry Emerson Fosdick

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rapid progress of modern data collection techniques has provided scientists with data ever increasing in size and dimension. While parametric models such as linear regression remain the most popular techniques in data modeling, they are often too rigid to model general nonlinear patterns hidden in a high-dimensional data space. In practice, a more flexible model is usually required. This motivates the development of nonparmatric regression procedures. Nonparametric regression methods achieve their flexibility by assuming only the continuity of the unknown regression function. However, nonparametric regression procedures face difficulty in extending themselves to model multivariate data. A regression procedure that is flexible enough to model nonlinear patterns for high dimensional data is in great demand.

In response to that demand, this research develops a procedure for multivariate regression that is applicable to high dimensional data and retains the flexibility of nonparametric methods. We achieve this goal by a new hybrid approach. Instead of modeling the regression function directly, we model the joint density of the data using a Gaussian mixture, which is a parametric density family flexible enough to model an arbitrary density function. From the Gaussian mixture model, we derive the regression function. We call this procedure Gaussian Mixture Regression (GMR).

In this chapter we review the problem of multivariate nonparametric regression and introduce a new template for high dimensional data analysis that connects density estimation and regression analysis, which we adopt for developing the GMR procedure. In Section 1.4 we briefly introduce the new IPRA/GMR and IPRA/GMC procedures.

Chapter 2 reviews multivariate nonparametric regression techniques and the theory and application of Gaussian mixture models. Chapter 3 extends the Iterative Pairwise Replacement Algorithm (IPRA) to multivariate data. The core of this thesis is Chapter 4, in which we describe our implementation of the Gaussian Mixture Regression (GMR) procedure and explore theoretical and practical aspects, including model selection algorithms and quantification of model uncertainty. In Chapter 5 we extend GMR to binary responses. The result is a new nonparametric logistic

regression procedure. We call this extension of GMR Gaussian Mixture Classification (GMC). We developed backward and forward projection algorithms for the optimal subspace of predictors, the details of which are in Chapter 6. Chapter 7 discusses applications of GMR and GMC on both simulated and real data sets. In some multivariate examples, we fit GMR models and then apply those dimensional reduction techniques we develop in Chapter 6. Chapter 8 concludes the dissertation with remarks on the key findings of the research and a discussion of our future research agenda.

1.1 The Problems of Multivariate Nonparametric Regression

In this thesis, we assume that the data $(X_i, Y_i)_{i=1}^n$ are iid samples from an unknown distribution $F_{X,Y}$, where $X_i \in \mathbb{R}^p$ and $Y_i \in \mathbb{R}$. A general data modeling problem is to identify the relationship between X and Y. Both standard parametric and nonparametric regression approaches start with the model $Y = m(X) + \varepsilon$, $\varepsilon \sim N(0, \sigma^2)$, which is equivalent to the distributional assumption that $Y|X \sim N(m(X), \sigma^2)$, where the regression function $m(x)$ is defined as the conditional expectation $E(Y|X = x)$. Notice that both parametric and nonparametric regression make no assumption of the marginal distribution of X; therefore the only way to link all the data is through the assumption of $m(x)$. Parametric regression achieves this connection by a specific assumption of the form of $m(x)$. The most common model is the linear form $m(x) = \beta^T x$.

Nonparametric regression is motivated by the demand for a flexible Exploratory Data Analysis (EDA) tool for data analysts in order to identify general nonlinear patterns in the data. Such flexibility is achieved by assuming only the continuity of the regression function $m(x)$. Some authors call nonparametric regression models smoothers, because they generate a smooth estimate of the unknown continuous regression function. The literature of nonparametric methods is abundant. Both Härdle (1990) and Hastie and Tibshirani (1990) give excellent surveys of the field. In this thesis, we will use the term "smoother" as a synonym for a nonparametric regression model.

All nonparametric smoothers are local methods by nature because they are based upon the continuity assumption of $m(x)$, which is a local property of $m(x)$. A straightforward approximation of $m(x)$ at a given x uses only the observations (x_i, y_i)'s where each x_i is within some neighborhood of x, for example, $B(x, \delta) = \{(x_i, y_i) : |x_i - x| < \delta\}$. The continuity of $m(x)$ implies that the each observed value y_i within $B(x, \delta)$ is close to the unknown value $m(x)$ on average. Hence, a natural estimator is to take the average of y_i within $B(x)$, that is,

$$\widehat{m}(x) = \sum_{i:|x_i - x| < \delta} \frac{y_i}{|B(x)|},$$

where $|B(x)|$ is the size of the set $B(x)$. Rewriting this estimator in a more general form, we have

$$\widehat{m}(x) = \frac{\sum_{i=1}^{n} y_i I(|x_i - x| < \delta)}{\sum_{i=1}^{n} I(|x_i - x| < \delta)},$$

which is a special case of a general estimator called the Nadaraya-Watson kernel smoother.

The Nadaraya-Watson kernel smoother was proposed independently by Nadaraya (1964) and Watson (1964). It takes the general form

$$\widehat{m}(x) = \frac{\sum_{i=1}^{n} y_i \, K(x, x_i)}{\sum_{i=1}^{n} K(x, x_i)}, \tag{1.1}$$

where $K(x, x_i)$ is called the kernel function. A kernel function measures the proximity of an observation at x_i to the given location x. In addition, $K(x, x_i)$ is usually a continuous, monotone decreasing function with respect to the distance between x and x_i. There are many choices for $K(x, x_i)$. A common practice is to specify K as a unimodal, symmetric density function, which centered at 0; that is,

$$K(x, x_i) = K(x_i, x) = K(x - x_i),$$

$$\int K(u)du = 1,$$

and

$$\int uK(u)du = 0.$$

A general discussion of the kernel methods can be found in Härdle (1990). Here we focus on the Nadaraya-Watson estimator using the Gaussian pdf as the kernel function. This gives us a special case of (1.1):

$$\widehat{m}(x) = \frac{\sum_{j=1}^{n} y_j \, \phi(x; x_j, h^2 I_p)}{\sum_{j=1}^{n} \phi(x; x_j, h^2 I_p)}, \tag{1.2}$$

where ϕ is the multivariate Gaussian pdf for $X \in \mathbb{R}^p$. The bandwidth, h, is a calibration parameter that controls the range of the local support and the resulting smoothness of the fitted curve. A smaller h implies tighter kernel coverage, which leads to a smaller neighborhood and a less smooth curve. On the other hand, a larger h implies broader kernel coverage, which leads to a wider neighborhood and a less wiggly curve.

The estimator (1.2) is very successful for univariate data; That is, $p = 1$. However, as the dimension p increases, the local sparsity begins to impede all neighborhood-based smoothing algorithms. For example, assume n data points are uniformly distributed in a cube $[0, d]^p$ in $\mathbb{R}^p$. The volume of the cube is d^p. Consider a neighborhood of a given x that contains a fraction, r, of the data. The volume of this neighborhood is roughly $r \times d^p$, or $(r^{1/p} \times d)^p$. Hence the edge of the neighborhood cube is

approximately $r^{1/p} \times d$. Consider, for example, $r = .01$, $p = 10$; we have $r^{1/p} = .631$. This simple calculation shows that a neighborhood that includes 1% of the data in $\mathbb{R}^{10}$ will cover about 63.1% of the range in each dimension. It is hardly local at all. This example also illustrates the curious phenomenon called local sparsity. It means that when looking at a location in a high dimensional space, there are generally very few points around or nearby. The direct consequence of the local sparsity in high dimensions is that local-based nonparametric regression methods such as (1.2) are inevitably over-smoothed. This phenomenon was first observed by Bellman (1961), who coined the term Curse of Dimensionality. A more elaborate illustration of the curse of dimensionality can be found in Hastie, Tibshirani, and Friedman (2001).

1.2 A New Template for Statistical Data Analysis

One way to overcome the curse of dimensionality is to have a global parametric model that utilizes every data point available for the model fitting. Standard parametric regression methods achieve this globalization by imposing a global form of $m(x)$ for all x. But parametric regression lacks flexibility to model novel features that do not agree with the assumptions of $m(x)$; in other words, parametric models are subject to extreme bias. Nonparametric methods attain flexibility, but their extension to higher dimensions is rather limited. As shown in the previous section, a local approximation algorithm is already ineffective in a mild $\mathbb{R}^{10}$ data space.

We propose a solution to this ineffectiveness by fitting a global, but more flexible, model for the joint density. This idea is based on a fundamental principle: **All the statistical information of the data is stored in the joint density function.** Based upon this principle, density estimation is a more fundamental task of data analysis than is regression analysis. Moreover, this principle indicates a new template to develop regression and classification procedures by means of density modeling.

Here is a demonstration of how the principle works. Assume that the data are iid samples from a distribution $F_{X,Y}$. Given the density function $f_{X,Y}(x,y) = \frac{\partial^2 F(x,y)}{\partial x \partial x}$ of the data $\{(X_i, Y_i)_{i=1}^n : X \in \mathbb{R}^p, Y \in \mathbb{R}\}$, we can derive any statistical quantity from $f_{X,Y}$. For example, the regression function $m(x)$ can be derived from $f_{X,Y}$ by the standard definition, as shown in equation (1.3):

$$m(x) = \int y\, f_{Y|X}(y|x)\, dy, \tag{1.3}$$

where

$$f_{Y|X}(y|x) = \frac{f_{X,Y}(x,y)}{\int f_{X,Y}(x,y)\, dy}.$$

Next we consider the classification problem. For binary responses $Y \in \{0, 1\}$, the optimal Bayes classifier is based upon the posterior odds

$$r(x) = \frac{Pr(Y = 1|X = x)}{Pr(Y = 0|X = x)}.$$

Applying Bayes theorem we obtain

$$r(x) = \frac{Pr(X = x|Y = 1)Pr(Y = 1)}{Pr(X = x|Y = 0)Pr(Y = 0)}. \tag{1.4}$$

Equation (1.3) suggests the strategy of regression modeling via density estimation. (1.4), on the other hand, suggests a classifier built upon the ratio of two class densities. They both demonstrate the fundamental principle we mentioned earlier. In this dissertation we pursue this line of attack and show that the fundamental principle does offer an effective template for high dimensional data analysis.

The most significant advantage of this template is that it is less susceptible to the curse of dimensionality. In principle, the procedures we develop under this template, GMR and GMC, are applicable to data of any dimension. That is, the procedure requires no modification as the dimension increases. In practice, the limitation is from the numerical computation software and hardware, not from the statistical procedure itself.

1.3 The Foundation of Gaussian Mixture Regression

To put this research in a broader context, the immediate motivation of GMR is its direct application to multivariate nonparametric regression problems. A deeper motivation, however, is to use GMR to demonstrate the use of a new template for high dimensional data analysis. The central principle of this new approach, as mentioned in previous section, is this: All statistical information of the data is stored in the density function. This principle implies that statistical modeling should start with the estimation of the density function. A parametric data model begins with a parametric assumption of the density function. When this parametric assumption is in place, the density function is expressed as a likelihood. Therefore our central principle is equivalent to the likelihood principle, which states that the likelihood function is the minimal sufficient statistic; see Casella and Berger (1990).

In addition to providing accurate prediction, a successful model must also offer an efficient way to combine information from each data point. This feature of a model is crucial when facing the sparsity in high dimensional data spaces. As Efron and Tibshirani (1993, p. 358) pointed out

> The likelihood plays a central role in model-based statistical inference ... the likelihood is a natural device for combining information across experiments ...

The likelihood function, which is the product of the density evaluated at each observation, provides a natural way to combine information of individual observations. This fact suggests that a data modeling approach based on the likelihood function

could offer a good mechanism to combine information and, therefore, a promising way to model high dimensional data. Specifically, a likelihood function summarizes the information from the data, an iid sample, $D = \{(x_i, y_i)_{i=1}^n\}$ as

$$L(\theta; D) = \prod_{i=1}^{n} f_{X,Y}(x_i, y_i; \theta) \tag{1.5}$$

under the density model $f_{X,Y}(x, y, \theta)$. In general, a parametric model specifies a global likelihood function $L(D; \theta)$ that governs every data point (x_i, y_i), as shown in equation (1.5). For example, a linear model specifies the conditional density $f_{Y|X}(y_i|x_i) = \phi(y_i; x_i^T\beta, \sigma^2)$, where ϕ is the Gaussian pdf. Since $f_{X,Y} = f_{Y|X}f_X$, we derive the likelihood for classical linear model as

$$L(\beta; D) = \prod_{i=1}^{n} \phi(y_i; x_i^T\beta, \sigma^2) f_X(x_i) = \prod_{i=1}^{n} f_X(x_i) \prod_{i=1}^{n} \phi(y_i; x_i^T\beta, \sigma^2). \tag{1.6}$$

Although we do not fit the MLE for the classical linear model using equation (1.6), it does provide us a broader perspective of how a global parametric model allows us to estimate its model parameters using every available observation. When facing local sparsity in high dimensional data space, we do need a global parametric model that can utilize data efficiently. The trick is to find a global parametric model that offers flexibility in modeling general nonlinear patterns of the data. Loader (1999) proposes a method to increase flexibility of the parametric regression called local likelihood. Instead of a global parameter β, local likelihood allows $\beta = \beta(x_0)$. Hastie et al. (2001, p. 179) describe the local likelihood regression model as

$$l(\beta(x_0)) = \sum_{i=1}^{n} K(x_0, x_i) l(y_i, x_i^T\beta(x_0)), \tag{1.7}$$

where $K(x_0, x_i)$ is a kernel that specifies the neighborhood of x_0. Obviously this model is a variation of kernel regression models. We do not pursue this line of attack because in high dimensional space, we need a global model. However, (1.7) does suggest that a mixture of local linear models can be very flexible. We know that a Gaussian pdf $f_{X,Y}$ implies a linear regression function; therefore a Gaussian mixture pdf should lead to a mixture of linear models. The finite Gaussian mixture is a global parametric model and it does provide the flexible mixture of linear regression. It is a perfect candidate for the procedure we need for the flexible regression modeling of high dimensional data.

1.4 IPRA/GMR and IPRA/GMC in a Nutshell

As mentioned in Section 1.2, both GMR and GMC procedures are based on the model of the underlying density. By modeling the underlying density as a Gaussian mixture, they inherit the power and versatility of the Gaussian mixture family. Using the

multivariate version of IPRA, we construct a sequence of Gaussian mixture models indexed by the number of components K. The regression function corresponds to the mixture densities formulate the family of GMR models: $\{\text{GMR}(K) : K = n, \ldots, 1\}$, where GMR($K$) denotes the regression function of the K-component Gaussian mixture density. On the one end, GMR(n) is the Nadaraya-Watson kernel smoother; on the other end, GMR(1) is approximately the classical linear model.

Two key parameters govern the IPRA/GMR procedure: the bandwidth h, which controls the initial kernel density; and the number of components K, which controls the smoothness of the GMR fit.

The model parameters of each GMR(K) are approximated by the method of moments (MoM) in IPRA. It provides a simple alternative to the EM algorithm. Although the GMR procedure does not fit the MLE, the simple MoM enables an efficient computation of the profiles of MSE(K), PE(K), and cv(K). They provide data analysts a glimpse of the entire GMR family and help them select reasonable K for further investigation, including the MLE update of the model parameters.

The role of the bandwidth h is less crucial. The basic rationale of choosing h is first to make sure it is sufficiently small, so that the initial model GMR(n) overfits the data. Since GMR(1) usually underfits the data, an overfit GMR(n) implies that the "right" model is covered by the GMR family. However, empirical evidence in this research shows that all reasonable values of h yield almost identical patterns in their MSE and PE profile curves, which implies the same conclusion in GMR model selection. Nevertheless, we employ the following rule of thumb for the bandwidth selection:

$$h^* = \max\{h_m, 1/20\, h_S\},$$

where $h_S = n^{-1/(d+4)}\sigma$ is the optimal bandwidth proposed by Scott (1992) for product kernel estimator and $h_m = \min_{j,k}\{|X_{k,j} - X_{k,j-1}|\}$ the closest distance between two consecutive observation. In our rule of thumb, h_m serves as the lower bound of the bandwidth selection to safeguard against a bandwidth that is too small.

For classification problems, we apply IPRA on each class density, $f_{X|Y=0}$ and $f_{X|Y=1}$, to generate two sequences of Gaussian mixture models: $\{\widehat{f}_0(x; K_0) : K_0 = n_0, \ldots, 1\}$ and $\{\widehat{f}_1(x; K_1) : K_1 = n_1, \ldots, 1\}$, where K_0 and K_1 are the number of components of each class density. These two sequences of Gaussian mixture allow us to approximate the posterior odds

$$r(x) = \frac{f_{X|Y=1}(x)\ \Pr(Y=1)}{f_{X|Y=0}(x)\ \Pr(Y=0)}$$

by the ratio between any two densities from each sequence. The result is an IPRA/GMC family of models indexed by (K_0, K_1). In practice, the IPRA/GMC procedure is an alternative implementation of Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani 1996). Because IPRA/GMC provides data analysts a simple way to choose the appropriate (K_0, K_1), it can also serve as a complement to MDA.

Chapter 2

Preliminaries

In this chapter we review the strengths and weaknesses of some available nonparametric procedures for multivariate regression. Several articles exploit the idea of building regression functions from a density model. This leads to our search for a parametric density family as the building blocks of our new procedure. We review the general kernel density estimation and its connection to the finite Gaussian mixture density models and the algorithm to estimate and simplify the Gaussian mixtures.

2.1 Current Landscape of the Multivariate Nonparametric Methods

Hastie and Tibshirani (1990, p. 85) review several approaches for multivariate nonparametric regression. One natural approach is to apply a univariate smoother on each variable independently and add them up. Hastie and Tibshirani pursued this approach to develop Generalized Additive Models (GAM), which has the form

$$Y = \alpha + \sum_{j=1}^{p} m_j(X_j) + \varepsilon, \tag{2.1}$$

where $\varepsilon \sim N(0, \sigma^2)$.

Friedman and Stuetzle (1981) propose another approach called projection pursuit regression (PPR), which combines the dimension reduction and regression as shown in Equation (2.2):

$$Y = \sum_{k=1}^{K} h_k(\beta_k^T X) + \varepsilon, \tag{2.2}$$

where β_k denotes a 1-D projection of X and $\varepsilon \sim N(0, \sigma^2)$. The idea of PPR is to assume that there are K variables that capture all the information of X regarding Y. In practice, K could exceed the dimension of X, especially when $m(x)$ has strong nonlinear components.

Both GAM and PPR are clever ways of applying univariate smoothers in the multivariate setting. While they are very effective tools, their basic strategy is to apply a series of univariate smoothers on each dimension of X. The GAM formulation has difficulty picking up nonlinear pattern involving terms such as $X^{(j)}X^{(k)}$.

Breiman et al. (1984) introduce one of the most versatile nonparametric methods called Classification and Regression Trees (CART). A regression tree model essentially approximates the regression surface using step functions. One problem with using stepping functions is that the fitted regression surface is not smooth. The Multivariate Adaptive Regression Splines (MARS) procedure, developed by Friedman (1991), is a generalization of CART to obtain much smoother fits. The basic idea of MARS is to build a set of basis functions of the form $B = \{(X_k - t)_+, (t - X_k)_+\}$, where X_k is the k-th dimension of X and t is the knot, $t \in \{x_{ki} : i = 1, \ldots, n\}$, where x_{ki} is the i-th observation of the k-th dimension of the feature space. Instead of using step functions as the basis, MARS uses the ramp functions $(x - t)_+$. In essense, t is the location of the cut. On one side of the cut, MARS fits the data with a linear hyperplane and a constant for the other side of the cut. Therefore there is no discontinuity at t.

MARS also includes the product of any two elements in B to be in the basis. This inclusion enables MARS to model first-order interactions $X_j\, X_k$. The MARS model has the form

$$Y = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X) + \varepsilon \tag{2.3}$$

where h_m is a basis function from B or a product of two elements in B. One significant feature of MARS is that it approximates the regression function by a rich set of basis functions B.

A vantage point shared by CART and MARS is that they both partition the feature space into square blocks (hypercubes in dimensions higher than 2), whose boundaries are determined by tree cuts in CART and knots in MARS. Within each block, CART fit the data with a constant which leads to unpleasant bumpy fit. MARS elimintes the bumps by fitting the data with a linear hyperplane. Although MARS achieves better smoothness by connecting the neighboring hyperplanes, there are still edges on the boundaries of the square blocks. Furthermore, because the knots are located on the axes of the feature space, both CART and MARS are limited to blocks of boundaries perpendicular to the axes. This is why MARS is coordinate sensitive. As pointed out by O'Sullivan (1991) in the discussion of Friedman (1991), a rotation of the coordinate axes can completely change the structure of the MARS model.

Both CART and MARS suggest a general method of nonparametric regression: partition the feature space of X into blocks and fit a simple regression model within each block. The key to acquire a smoother regression surface fit than MARS is to have a smooth transition between two neighboring blocks. And the blocks should not have to be hypercubes.

We take a different approach to the development of a procedure for multivariate regression. Instead of modeling the regression function directly, we model the joint

density $f_{X,Y}$, and then derive the regression function from the density model. This approach may be overkill in the univariate case, but there are several advantages of this regression-via-density approach: the same procedure applies to X in any dimension without modification. The model is invariant under any coordinate system. That is, rotating the coordinate axes of the feature space does not change the structure of the model. This coordinate invariance property implies that it is simple to update the model for any linear projection of X; Thus it allows us to implement straightforward alogrithms for the dimension reduction projections of X into a lower subspace. The details are in Chapter 6.

2.2 Density Estimation and Regression

In search of a straightforward way to build a regression procedure, we return to the standard definition of the regression function $m(x)$:

$$m(x) = \mathrm{E}[Y|X = x] = \int y f_{Y|X}(y|x)dy = \frac{\int y f_{X,Y}(x,y)dy}{\int f_{X,Y}(x,y)dy}. \tag{2.4}$$

Although the density function $f_{X,Y}$ and the regression function $m(x)$ are naturally connected by their definitions, the research utilizing this natural connection is surprisingly rare. An early article we found in pursuing this connection directly is Schmerling and Peil (1985), cited by Härdle (1990). They propose using kernel density models to approximate the bivariate joint density $f_{X,Y}$, then derive the regression from the density model. Schmerling and Peil (1986) improve kernel density estimation by local polynomial approximation to reduce the bias of their empirical regression model. There is no followup research along this line by those authors.

Scott (1992) points out that the Nadaraya-Watson kernel regression of the form

$$\widehat{m}(x) = \frac{\sum_{i=1}^{n} y_i \, K(x, x_i)}{\sum_{i=1}^{n} K(x, x_i)} \tag{2.5}$$

can be derived from (2.4) where the joint pdf $f_{X,Y}$ is estimated by the bivariate product kernel. That is

$$\widehat{f}_{X,Y}(x,y) = \sum_{j=1}^{n} n^{-1} K_h(x - x_j) K_h(y - y_j), \tag{2.6}$$

where $K_h(x - x_j) = h^{-1} K(\frac{x-x_j}{h})$. (2.6) implies that

$$\begin{aligned}
\int \widehat{f}_{X,Y}(x,y)dy &= \int \sum_{j=1}^{n} n^{-1} K_h(x - x_j) K_h(y - y_j) dy \\
&= \sum_{j=1}^{n} n^{-1} K_h(x - x_j) \int K_h(y - y_j) dy \\
&= \sum_{j=1}^{n} n^{-1} K_h(x - x_j)
\end{aligned}$$

and

$$\begin{aligned}
\int y\widehat{f}_{X,Y}(x,y)dy &= \int y \sum_{j=1}^{n} n^{-1} K_h(x - x_j) K_h(y - y_j) dy \\
&= \sum_{j=1}^{n} n^{-1} K_h(x - x_j) \int y K_h(y - y_j) dy \\
&= \sum_{j=1}^{n} n^{-1} y_j K_h(x - x_j).
\end{aligned}$$

Hence, the Nadaraya-Watson estimator of the regression function is

$$\widehat{m}(x) = \frac{\int y\widehat{f}_{X,Y}(x,y)dy}{\int \widehat{f}_{X,Y}(x,y)dy} = \frac{\sum_{j=1}^{n} n^{-1} y_j K_h(x - x_j)}{\sum_{j=1}^{n} n^{-1} K_h(x - x_j)} = \frac{\sum_{j=1}^{n} y_j \, K(x, x_j)}{\sum_{j=1}^{n} K(x, x_j)},$$

as shown in (2.5). This perspective of the famous Nadaraya-Watson estimator inspires the notion that by modeling $f_{X,Y}$ in the form of a kernel density estimator, we can derive a kernel smoother for $m(x)$. A natural extension of this notion is to use Gaussian pdf as the kernel K.

It is well-known that when $f_{X,Y}$ is Gaussian, the conditional pdf $f_{Y|X}$ is Gaussian and the regression function $m(x)$ is linear. A natural extension of the single Gaussian pdf for $f_{X,Y}$ is to model $f_{X,Y}$ as a K-component Gaussian mixture:

$$\widehat{f}_{X,Y}(x,y) = \sum_{j=1}^{K} \pi_j \phi(x, y; \mu_j, \Sigma_j). \tag{2.7}$$

The resulting regression function $m(x)$ of the pdf in (2.7) is a combination of linear functions $m_j(x)$. That is, $m(x) = \sum_{j=1}^{K} w_j(x) m_j(x)$. We will derive a procedure called Gaussian Mixture Regression (GMR) based on this idea, which is the core of this thesis. We defer the detailed discussion to Chapter 4.

Recently, Figueiredo (2000) has formulated the regression function from a finite Gaussian mixture density and uses the EM algorithm to fit the MLE of the mixture, given a pre-determined number of mixtures, K. However, Figueiredo's notion of fitting the underlying mixture density follows the standard EM approach. Figueiredo, Leitão, and Jain (1999) propose an algorithm using minimum description length (MDL) to select the number of components. They incorporate this MDL step into the standard EM algorithm. In general, we believe that the best mixture model for the underlying density $f_{X,Y}$ and the best mixture model for the regression function $m(X) = \mathrm{E}[Y|X]$ do not necessarily coincide. For regression purposes, we should pick the number K with respect to the goodness-of-fit on $\widehat{m}(x)$, not on $\widehat{f}_{X,Y}$. We will resume detailed discussion of this important point in Chapter 4.

There are several reasons why the regression via density approach never gained momentum. First of all, the empirical regression approach can be viewed as an alternative way to derive the Nadaraya-Watson kernel smoother. In this regard, it is absorbed into the huge literature of kernel regression. Vapnik (2000) observes that density estimation is a hard computational problem because the density estimation problem is ill-posed. On the other hand, a direct estimation of a smooth regression curve from the data is a well-studied problem with abundant software available. Hence, to work out a univariate regression curve from the density function is probably overkill at best. And since the standard univariate nonparametric methods are very successful, there is no reason to dip into this line of research for a univariate smoother.

D. W. Scott (personal communication) argues that density estimation is more straightforward than regression curve fitting because there are more quantities to specify in the regression problem than in the density estimation problem. For example, in regression, analysts must specify the form of the regression function and the variance of the residuals. On the other hand, in histogram and kernel density estimation, the only parameter is the bandwidth h. Conceptually, density estimation is more intriguing than direct data fitting because more information of the data is stored in the density function.

Whatever the case may be, the density model approach may be inevitable, especially in facing the curse of dimensionality. Because all the statistical information of X and Y is stored in the density (likelihood) function, we argue that a density modeling approach, such as GMR, is a convincing approach for high dimensional data modeling. We believe this regression-via-density approach is a natural way to build a nonparametric regression model for high dimensional data. We discuss the details of GMR in Chapter 4.

2.3 Kernel Density Estimators

In search of a flexible parametric model for the density function, we first turn to the nonparametric density estimation, a well-established field. The standard reference, Scott (1992, p. 125) is the main source of our brief review on the subject.

Given an iid sample $\{x_i\}_{i=1}^n$ from an unknown density f_X, the kernel density estimator takes the form

$$\widehat{f}_X(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - x_i}{h}).$$

The kernel function K is a symmetric density function. We consider the special case of Gaussian kernel, $K(u) = \phi(u; 0, 1)$. The Gaussian kernel density estimator

$$\widehat{f}_X(x) = \sum_{i=1}^{n} n^{-1}\phi(x; x_i, h^2),$$

can easily be generalized to $X \in \mathbb{R}^p$ as

$$\widehat{f}_X(x) = \sum_{i=1}^{n} n^{-1}\phi(x; x_i, h^2 I_p). \tag{2.8}$$

The estimator (2.8) is a n-component Gaussian mixture density with equal mixing weights n^{-1}. Notice that h is the only parameter in this estimator. The parameter h is called the bandwidth. Notice that if $h \to 0$, $\widehat{f}_X(x)$ becomes a collections of Dirac spikes, i.e., $\widehat{f}_X(x) \to \sum_{i=1}^n n^{-1}\delta(x - x_i)$, where $\delta(t)$ is the Dirac delta function with the properties: $\delta(t) = 0$ for all $t \neq 0$, $\delta(t) = \infty$ as $t \to 0$, and $\int \delta(t)dt = 1$. This fact implies that a density estimation problem is not well-defined and that the MLE does not exist because the likelihood function is unbounded above. We shall revisit the issue of bandwidth selection in Chapter 4.

For now, the most important point is that a kernel density estimator is a special case of the finite Gaussian mixture models. Devroye and Györfi (1985) study the L_1 consistency of the kernel density estimator. They prove that as n increases and h decreases in a way that $h + n^{-1}h^{-p} \to 0$, the kernel density estimator can approximate any unknown density arbitrarily close in L_1-norm. A direct implication is that the consistency of the kernel density estimator ensures that the Gaussian mixture models inherit this asymptotic property. We discuss the details and their implications in Chapter 4.

2.4 A Review of Finite Gaussian Mixtures

The literature on mixture models, especially Gaussian mixture models, is abundant. McLachlan and Peel (2000) provide a comprehensive review of the theory and application of mixture models. Its bibliography is 45-page long. One of the reasons

that Gaussian Mixture models are so popular is that they are parametric models of elegant form, yet they are very flexible in estimating a general density. In the previous section, we showed that a n-component Gaussian mixture is equivalent to the kernel density estimator. Inheriting the asymptotic consistency of the kernel density estimator gives finite Gaussian mixtures the ability to estimate any general density function in $\mathbb{R}^p$. This is exactly why we use them to build our regression procedure.

One natural interpretation of the mixture model is to view each mixture component as a cluster by introducing a latent cluster indicator variable G, where $G \in \{1, 2, \ldots, K\}$. With this latent variable G, the Gaussian mixture model can be formulated as a hierarchical model.

$$\begin{aligned} G &\in \{1, 2, \ldots, K\} \\ \Pr(G = k) &= \pi_k \\ X|G = k &\sim N(\mu_k, \Sigma_k). \end{aligned}$$

Under this hierarchical model, the mixing probability π_k can be viewed as the prior probability of the latent cluster index variable G. We derive the marginal density f_X as

$$f_X(x) = \sum_{j=1}^{K} \Pr(G = j)\, f_{X|G}(x|j) = \sum_{j=1}^{K} \pi_j\, \phi(x; \mu_j, \Sigma_j).$$

Hence the pdf of a K-component Gaussian mixture is of the from

$$f_X(x) = \sum_{j=1}^{K} \pi_j\, \phi(x; \mu_j, \Sigma_j), \tag{2.9}$$

where ϕ is the multivariate Gaussian pdf

$$\phi(x; \mu, \Sigma) = |2\pi\Sigma|^{-1/2} \exp\{-1/2(x-\mu)^T \Sigma^{-1}(x-\mu)\}.$$

From (2.9), we can compute the posterior probability

$$\Pr(G = k|X = x) = \frac{\Pr(G = k, X = x)}{f_X(x)} = \frac{\pi_k\, \phi(x; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j\, \phi(x; \mu_j, \Sigma_j)}. \tag{2.10}$$

This expression of the posterior probability plays an important role in our development and interpretation of Gaussian Mixture Regression procedure. We will revisit (2.10) in Chapter 4.

Observe that the model parameters of (2.9) are $\{K, (\pi_j, \mu_j, \Sigma_j)_{j=1}^K\}$. For $X \in \mathbb{R}^p$, the total degrees of freedom of the parameter for a K-component Gaussian Mixture model is $K \times (1 + p + p \times (p+1)/2) - 1$, with the -1 indicating the natural constraint $\sum_j \pi_j = 1$. In the next section, we discuss the problem of estimating these parameters.

2.5 Fitting Gaussian Mixtures Models

The first important issue in using Gaussian Mixtures is to determine the number of components K. Ideally, we would like to obtain information about K from the data. This turns out to be a nontrivial problem.

The standard Bayesian solution of this problem is to assume that K is a random variable, set up a hierarchical model with a hyperprior on K, and implement a MCMC algorithm to obtain the posterior probability of K. This is the approach of Richardson and Green (1997). Stephens (1997) takes a different approach. He sets up a birth-death process on the components of Gaussian mixtures. Both Bayesian approaches require intensive computation. Richardson and Green apply their method only on the univariate case, Stephens to univariate and bivariate data. It is safe to say that to extend either approach to arbitrary dimensions would require prohibitive amount of computation. For our purpose of developing an EDA tool for regression of high dimensional data, methods that require MCMC computation are not practical.

Frequentists often take a random or educated guess for K and the values of model parameters, then apply the EM algorithm to obtain the MLE. In comparing models of different degrees of freedom, a standard statistic called Bayesian Information Criterion (BIC) is popular for its straightforward interpretation and simple computation. A BIC is defined as

$$BIC = -2 \times \log L(x; \Theta^*) + d \times \log N,$$

where $L(x; \Theta^*)$ is the likelihood function evaluated at the MLE Θ^*; N is the sample size and d the degrees of freedom of the given model. For a Gaussian mixture model, $d = K \times (1 + p + p \times (p+1)/2) - 1 = O(Kp^2)$ as shown in Section 2.4. The problem with BIC is that it still requires the MLE computation for each value of K. The MLE is usually computed using the EM algorithm, which requires good initial values of the parameters. Moreover, the EM algorithm requires good initial values to increase the chance for the convergence to a good maximum. And there is no known way to check if a good maximum is attained other than to rerun the EM algorithm with different initial values.

A more fundamental problem of the MLE computation is that the global maximum of the mixture likelihood does not exist. This is because the likelihood function of the Gaussian mixture models is not bounded. Let the variance of one component go to 0 and the the value of the likelihood function goes to infinity.

The common practice in using Gaussian mixtures is to determine K by trial-and-error. For each K, a user must also provide good initial parameter values to initiate the EM algorithm. As the dimension of the data grows, it quickly becomes an impossible task. Hastie and Tibshirani (1996) implement their Mixture Discriminant Analysis (MDA) using Gaussian mixtures. Their solution is to use the k-means procedure for a given K to generate initial values of model parameters for the EM algorithm. But the selection of K remains to be in the trial-and-error mode.

A more practical approach is in great demand. In practice, we prefer using the

data to help us identify K. A profile of models of every possible K will be a better approach than repetitive trial-and-error. A promising solution to this problem comes from an unexpected direction, which is the subject of the next section.

2.6 The Iterative Pairwise Replacement Algorithm

The iterative pairwise replacement algorithm (IPRA) was first proposed by Scott and Szewczyk (2001). Their main goal was to find a parsimonious representation of a finite Gaussian mixtures density. Their insight is that a Gaussian kernel density estimator can be viewed as an n-component Gaussian mixture model and that the appropriate K-component mixtures should be a parsimonious simplification of the initial kernel density model. By iteratively merging the most similar pairs of components, they reduce the initial model to a finite Gaussian mixture of K components (as K ranges from n to 1). One key feature of IPRA is that it generates a sequence of mixture models indexed by K, where $K = 1, \ldots, n$. By defining a goodness-of-fit measure for each mixture model, IPRA gives us a systematic way to determine the appropriate K. Another important point is that Scott and Szewczyk (2001) propose using the method-of-moments to update the model parameters. This leads to a simple alternative to the cumbersome MLE. The key ingredients of IPRA are the initial kernel model, the order of comparisons, the similarity measure, and the update algorithm of parameter estimates. Scott and Szewczyk (2001) focus their attention on the univariate case. For our purpose, we must extend their algorithm to the multivariate kernel density. Essentially, this multivariate version of IPRA serves as the work horse for GMR. In Chapter 3 we examine the details of IPRA and extend it to multivariate data.

Chapter 3

Multivariate IPRA

In this chapter, we implement a multivariate extension of the Iterative Pairwise Replacement Algorithm (IPRA). We first review the univariate IPRA and partition it into three basic components: the similarity measure, the order of merging, and the update parameter estimates after merging. We then design our multivariate IPRA by necessary extensions for each component.

3.1 The Basic IPRA Components

IPRA was first proposed by Scott and Szewczyk (2001) for the task of simplifying a kernel density model to a more parsimonious model. Their insight was to view the kernel model as an n-component Gaussian mixture. By iteratively merging the most similar pair of components, the IPRA reduces the initial model to a finite Gaussian mixture of K components, K ranges from n to 1. Essentially IPRA generates a sequence of mixture models indexed by K. We call this sequence the IPRA sequence. This IPRA sequence of mixture models plays a key role in our GMR procedures.

The original IPRA starts with the kernel density estimator

$$\widehat{f}_n(x) = \sum_{i=1}^{n} n^{-1} \phi(x; x_i, h^2 I), \tag{3.1}$$

which is a saturated n-component mixture Gaussian model. The IPRA procedure combines the most similar pair of components into one, thus reducing the number of components by 1. Continuing this procedure to reduce the number of components, eventually a sequence of K-component mixture models $\{\widehat{f}_K : K = 1, \ldots, n\}$ emerges, where

$$\widehat{f}_K(x) = \sum_{i=1}^{K} \pi_i \phi(x; \mu_i, \Sigma_i).$$

The original univariate IPRA consists of the following steps:

Step 1. Construct the initial model

$$\widehat{f}(x) = \sum_{i=1}^{n} n^{-1}\phi_i(x),$$

where $\phi_i(x) = \phi(x; x_i, h^2)$.

Step 2. Combine the most similar pair of adjacent components $(w_i\phi_i, w_j\phi_j)$ of parameters (w_i, μ_i, σ_i) and (w_j, μ_j, σ_j) into one component (w, μ, σ). Refit the parameter using the method of moments (MoM) gives:

$$w = w_i + w_j,$$

$$\mu = \widetilde{w}_i\,\mu_i + \widetilde{w}_j\,\mu_j,$$

and

$$\sigma = \widetilde{w}_i\,\sigma_i^2 + \widetilde{w}_j\,\sigma_j^2 + \widetilde{w}_i\,\widetilde{w}_j\,(\mu_i - \mu_j)^2.$$

where $\widetilde{w}_i = w_i/w$, and $\widetilde{w}_j = w_j/w$.

Step 3. Repeat Step 2 until there is only one component left.

The key ingredients of IPRA are the measure of similarity, the order of candidate merging pairs, and the update of model parameters after merging.

In the multivariate situation, because there is no natural ordering among the mixture components, and we must find a way to determine the order of comparing the "consecutive" pairs of mixture components. To implement IPRA for the multivariate Gaussian mixture models, in each step we will have to identify the closest pair for combination. In high dimensions, we will follow the suggestion of Scott and Szewczyk (2001) to define the order of "adjacent" components to be considered by the minimum spanning tree (MST).

The MST consists of $(n-1)$ pairs of mixture components (ϕ_i, ϕ_j). We will then sort the pairs by their Hellinger distances as defined in (3.2). The sorted pairs will provide the order of IPRA trimming. In the beginning the MST will give the same ordering as the Hellinger distance because each component has the same variance and weight to start with. Following is a sketch of our multivariate IPRA:

Step 1. Define a weighted Hellinger metric on a pair of mixtures components:

$$H(w_i, w_j, \phi_i, \phi_j) = \sqrt{w_i\,w_j}\left(1 - \int \sqrt{\phi_i\,\phi_j}\right). \tag{3.2}$$

Step 2. Construct an MST that connect all n components with $(n-1)$ edges, where the distance (similarity) between any two components is computed using (3.2).

Step 3. Rank $(n-1)$ edges by their similarity in ascending order.

Step 4. Merge the pair with the smallest similarity metric in the MST; update the parameter of the new component using the method of moments (MoM). That is,

$$w = w_i + w_j, \tag{3.3}$$

$$\mu = \frac{w_i}{w}\mu_i + \frac{w_j}{w}\mu_j, \tag{3.4}$$

$$\Sigma = \frac{w_i}{w}\Sigma_i + \frac{w_j}{w}\Sigma_j + \frac{w_i w_j}{w^2}(\mu_i - \mu_j)(\mu_i - \mu_j)^T. \tag{3.5}$$

Step 5. Update the length of each remaining edge in the MST, re-rank the edges by their updated lengths, and go to Step 5.

In Step 5, we use the MoM update for the parameters of the newly merged mixture component. The detailed derivations of equations (3.3), (3.4), and (3.5) are given in Appendix B. They are straightforward extensions of the algorithm introduced by Scott and Szewczyk (2001). The detailed discussion of the weighted Hellinger metric is given in the next section. Section 3.3 describes the details of building MST using Prim's algorithm.

3.2 Similarity Measure: Weighted Hellinger Metric

The most important component of IPRA is the similarity metric, i.e., the distance between two mixture components. There are various ways to define the distance between two general density functions. We choose Hellinger distance because it does not depend upon the scale of the data. Although evaluating Hellinger distance between two general density functions is not an easy task, a simple closed form exists when both density functions are Gaussian pdf's. The Hellinger distance between two density functions $f(x)$ and $g(x)$ is defined as

$$H(f,g) = \int \left(\sqrt{f(x)} - \sqrt{g(x)}\right)^2 dx. \tag{3.6}$$

As Scott and Szewczyk (2001) point out, to apply IPRA to Gaussian mixtures, the definition of similarity metric should take the mixing weights into consideration. A natural candidate is to apply (3.6) to the two Gaussian components directly. This gives us

$$H(w_1\,\phi_1, w_2\,\phi_2) = \int \left(\sqrt{w_1\,\phi_1} - \sqrt{w_2\,\phi_2}\right)^2 dx \tag{3.7}$$

$$= w_1 + w_2 - 2\sqrt{w_1\,w_2}\int \sqrt{\phi_1\,\phi_2}\,dx. \tag{3.8}$$

One consequence of the weighted Hellinger metric defined as (3.8) is that the two components of similar weights have a smaller Hellinger distance than the two components of distinctly different weights. For example, given the same pair of pdf's (ϕ_1, ϕ_2) of different mixing weights, say $(.30, .20)$ and $(.49, .01)$, we have $H(.30\,\phi_1, .20\,\phi_2) < H(.49\,\phi_1, .01\,\phi_2)$. It implies that the $(.30, .20)$ pair is more similar than the $(.49, .01)$ pair, and therefore the IPRA procedure will merge the $(.30, .20)$ pair first.

In practice, it is preferable to first merge a much smaller weighted component to a larger one nearby. This preference motivates us to abandon (3.8) and choose the weighted Hellinger metric in the form of (3.9):

$$H(w_1, w_2, \phi_1, \phi_2) = \sqrt{w_1\, w_2}\,\big(1 - 2\int \sqrt{\phi_1\, \phi_2}\, dx\big). \tag{3.9}$$

Here the farther apart the w_1 and w_2 are, the smaller the value of $\sqrt{w_1\, w_2}$, and hence the smaller the value of (3.9). This implies a smaller distance, or higher similarity, between two mixture components of very distinct weights (w_j, w_k) with all other things being equal. Hence, we define the weighted Hellinger metric as (3.9) because it renders two unevenly weighted mixtures components a smaller distance, or higher similarity. The direct consequence is that (3.9) promotes the merging of two unevenly weighted mixture components, which is a desired property for the IPRA procedure.

The integral in (3.9) has a closed form and its computation is straightforward:

$$\int \sqrt{\phi_1 \phi_2} = (2\sqrt{2\pi})^p |\Sigma_1|^{1/4} |\Sigma_2|^{1/4} \phi(0; \mu_1 - \mu_2, 2\Sigma_1 + 2\Sigma_2).$$

The mathematical details are in Appendix A.

3.3 The Order of Merging: Minimum Spanning Tree

In the most naive sense, given n mixtures components, we can compute the pairwise distance between all possible pairs $d(\phi_j, \phi_k)$ and then rank all possible $n(n-1)/2$ pairs in ascending order. This ranked list will specify the order in which the IPRA merges similar pairs sequentially. However, because this scheme requires $O(n^2)$ operations and storage. It is sufficient to construct an MST for the same purpose, which requires only $O(n \log n)$ operations. MST is a well-studied subject in graph theory. It is defined as the undirected graph that connects n points with the minimum cost. For our application, cost is defined as the total length of the $(n-1)$ edges. The main reference for our discussion here is Sedgewick (2002). Sedgewick categorizes the problem of finding the shortest connection of n points as the Euclidean MST. Sedgewick concludes that the worst-case cost of the standard Prim's algorithm with priority-first search is $O(n \cdot \log n)$, which is a sizeable improvement in comparison to the direct computation of a $n \times n$ distance matrix. Prim's algorithm is a straightforward way to grow the MST. At each step, the edge that connects the point that is closest to the tree is added to it. The procedure continues until all points all included in the tree.

First we define the distance between a point x to a set G as

$$d(x, G) = \min_{g \in G} \parallel x - g \parallel .$$

Following is a sketch of Prim's algorithm:

Step 1. Initiate $G = \{x_1\}$, $M = \emptyset$.

Step 2. Find $x_j = \underset{x \notin G}{argmin}\{d(x, G)\}$. Denote $E(x_j, g)$ the edge of minimum length.

Step 3. Update the edge set $M = M \cup \{E(x_j, g)\}$, $G = G \cup \{x_j\}$.

Step 4. Continue Step 2 until $G = \{x_1, \ldots, x_n\}$. The M consists of $(n-1)$ edges is the MST.

We implement Prim's algorithm in MATLAB. Figure 3.1 shows the initial MST generated by our MATLAB code on the Motorcycle Data.

After the initial MST defines the IPRA merging order, the merging operation will alter the pairwise distances among the remaining mixture components. In principle, we may need to recompute the MST must be updated accordingly. In our experience, we found that recomputing MST in not necessary. In our implementation, after each IPRA merging operation, which includes recomputing the pairwise distance between each remaining pair of components in the MST using weighted Hellinger metric (3.9) and re-rank the remaining pairs by the updated distances.

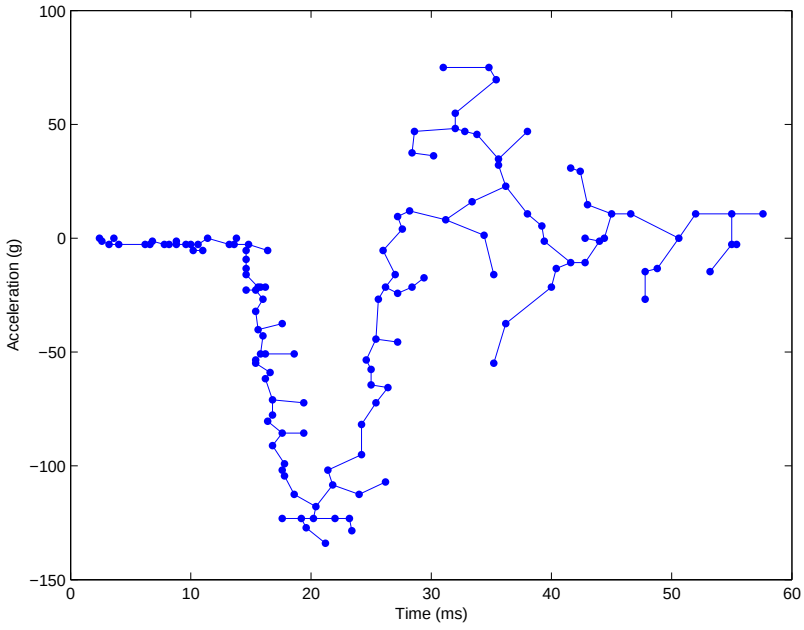

Figure 3.1: Minimum spanning tree of the Motorcycle data

3.4 Method-of-Moments Updates versus MLE

The likelihood and MLE of mixture models are well-studied subjects in statistics. Lindsay (1983) develops a general theory of the mixture likelihood and provides the sufficient conditions for the uniqueness of MLE. However, in the context of this dissertation, our focus is not on the MLE of the underlying Gaussian mixture density models. Instead, we are interested in an efficient approximation of the model parameters that is good enough to provide reasonable grounds for model selection. That is, we need good enough estimates of $(w_j, \mu_j, \Sigma_j)_{j=1}^K$ for any given K so that we can compute goodness-of-fit measures that can help determine the appropriate value of K.

The most important reason we want to avoid MLE in IPRA procedures is that the MLE update of the model parameters after merging requires intensive computation. Strictly speaking, an exact MLE update requires the MLE update of the whole $(K-1)$-component mixtures after merging one pair of the K-component model.

Scott and Szewczyk (2001) propose a reasonable idea of local refitting, which updates only the component after the merging operation and keeps the other components fixed. They investigate the method of moments and L_2E updates of the parameters of the merged component.

We choose the MoM update because it requires simpler computation than the L_2E. For our purpose, the IPRA sequence of Gaussian Mixtures with method-of-moments estimate of the parameters is sufficient. After all, the ultimate goal is to determine the appropriate value of K for the modeling of $m(x)$. And, in our setting, K is not determined by the density estimation itself, but by the performance of the GMR models derived from the IPRA family. Hence, the exact MLE of the model parameters for each K is not necessary for our use of IPRA, which is to help discern appropriate values of K for the GMR models. The task of determining K is deferred to the GMR model selection algorithm. Once K is determined, we can always launch the EM algorithm with the IPRA MoM estimates as the initial values and get the MLE updates of the selected model.

In Chapter 4 we discuss the MoM versus MLE within the context of GMR.

Chapter 4

Gaussian Mixture Regression

This chapter is the core of the thesis. In Section 4.1 we describe the classical result of the joint Gaussian density and its regression function. In Section 4.2 we extend the fundamental result to derive the Gaussian Mixture Regression (GMR) models, which resumes the form $m(x) = \mathrm{E}[Y|X = x] = \sum_{j=1}^{K} w_j(x)\, m_j(x)$, from the K-component Gaussian mixture of the underlying joint density $f_{X,Y}$. Applying IPRA procedure described in Chapter 3 we generate a sequence of Gaussian mixture density models index by K; Consequently, the IPRA/GMR procedure provide data analysts a family of regression model indexed by K. Section 4.3 describes the estimation of the GMR model parameters for any given K.

In Section 4.4 and Section 4.5 we discuss the key parameters that govern the IPRA/GMR procedure: the bandwidth h that governs the IPRA procedure and the number of components K that controls the smoothness of the GMR models.

The Bayesian perspective is described in Section 4.6 and finally in Section 4.7 we investigate its practical and theoretical properties of IPRA/GMR and compare its performance to the standard Multivariate Adaptive Regression Splines (MARS) procedure (Friedman 1991).

4.1 Regression from the Joint Gaussian Density

In this section we derive the exact form of GMR model from a K-component Gaussian mixture joint density model. Our starting point is the simple joint Gaussian density of (X, Y). One classical result of the multivariate Gaussian density is that when partitioning the joint density into $f_{X,Y} = f_{Y|X} f_X$, both $f_{Y|X}$ and f_X are also multivariate Gaussian. Theorem 4.1.1 is a well established result from Mardia et al. (1979, p.63).

Theorem 4.1.1. *If* $\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N_{p+q}(\mu, \Sigma)$, *where* $X_1 \in \mathbb{R}^p$, $X_2 \in \mathbb{R}^q$, *and* $\Sigma =$

$\begin{bmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{bmatrix}$, *then*

$$X_2|X_1 \sim N_q(\mu_2 + \Sigma_{21}\Sigma_{11}^{-1}(X_1 - \mu_1), \Sigma_{22.1}), \tag{4.1}$$

where $\Sigma_{22.1} = \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}$.

Theorem 4.1.1 states that when (X, Y) is a joint Gaussian distribution, the conditional density is also Gaussian and the regression function is a linear function (4.2) whose slope is determined by Σ_X, the variance of X, and Σ_{YX}, the covariance of Y and X:

$$m(x) = \mathrm{E}[Y|X = x] = \mu_Y + \Sigma_{YX}\Sigma_X^{-1}(x - \mu_X), \tag{4.2}$$

$$\sigma^2 = \mathrm{Var}[Y|X = x] = \Sigma_Y - \Sigma_{YX}\Sigma_X^{-1}\Sigma_{XY}, \tag{4.3}$$

and the joint density can be partitioned as

$$\phi(x, y; \mu, \Sigma) = f_{Y|X}(y|x)f_X(x) = \phi(y; m(x), \sigma^2)\,\phi(x; \mu_X, \Sigma_X). \tag{4.4}$$

4.2 Regression Function of the Gaussian Mixtures

Extending the classic results of the joint Gaussian to a finite Gaussian mixtures, it can be shown that the mixture Gaussian model for the joint density f_{XY} inherits the elegant results of (4.2), (4.3), and (4.4) from each Gaussian component of the mixture model. Assume the data follow the the joint density

$$f_{X,Y}(x, y) = \sum_{j=1}^{K} \pi_j \phi(x, y; \mu_j, \Sigma_j), \tag{4.5}$$

where

$$\sum_{j=1}^{K} \pi_j = 1, \quad \mu_j = \begin{bmatrix}\mu_{jX}\\ \mu_{jY}\end{bmatrix}, \quad \Sigma_j = \begin{bmatrix}\Sigma_{jX} & \Sigma_{jXY}\\ \Sigma_{jYX} & \Sigma_{jYY}\end{bmatrix}.$$

Here, $\phi(x, y; \mu, \Sigma)$ is the pdf of the multivariate Gaussian $N_{p+1}(\mu, \Sigma)$. The parameters of model (4.5) include the number of the mixture components K, the prior weights, the means , and the variances of each Gaussian component. That is, $\Theta = (\theta_1, \theta_2, \ldots, \theta_K)$, where $\theta_j = (\pi_j, \mu_j, \Sigma_j)$, with the constraint $\sum_{j=1}^{K} \pi_j = 1$.

Partitioning each Gaussian component ϕ_j according to equation (4.4), the joint density can be written as

$$f_{X,Y}(x, y) = \sum_{j=1}^{K} \pi_j\, \phi(y|x; m_j(x), \sigma_j^2)\, \phi(x; \mu_{jX}, \Sigma_{jX}), \tag{4.6}$$

where

$$m_j(x) = \mu_{jY} + \Sigma_{jYX}\Sigma_{jX}^{-1}(x - \mu_{jX}) \tag{4.7}$$

and

$$\sigma_j^2 = \Sigma_{jYY} - \Sigma_{jYX}\Sigma_{jX}^{-1}\Sigma_{jXY}. \tag{4.8}$$

Note that f_X and $f_{Y|X}$ can be derived directly from (4.6), followed by the regression function (4.12) and the conditional variance (4.13). The marginal density of X is

$$f_X(x) = \int f_{X,Y}(x, y)\, dy = \sum_{j=1}^{K} \pi_j\, \phi(x; \mu_{jX}, \Sigma_{jX}). \tag{4.9}$$

The conditional pdf of $Y|X$ is

$$f_{Y|X}(y|x) = \sum_{j=1}^{K} w_j(x)\, \phi(y; m_j(x), \sigma_j^2), \tag{4.10}$$

with the mixing weight

$$w_j(x) = \frac{\pi_j\, \phi(x; \mu_{jX}, \Sigma_{jX})}{\sum_{j=1}^{K} \pi_j\, \phi(x; \mu_{jX}, \Sigma_{jX})}, \tag{4.11}$$

and the $m_j(x)$, σ_j^2 from (4.7), (4.8).

From (4.10), the regression function is of the form:

$$m(x) = \mathrm{E}[Y|X = x] = \sum_{j=1}^{K} w_j(x)\, m_j(x). \tag{4.12}$$

The conditional variance function is

$$v(x) = \mathrm{Var}[Y|X = x] = \sum_{j=1}^{K} w_j(x)\big(m_j(x)^2 + \sigma_j^2\big) - \big(\sum_{j=1}^{K} w_j(x) m_j(x)\big)^2. \tag{4.13}$$

Equation (4.12) is the core of this thesis. We call the $m(x)$ in (4.12) a Gaussian Mixture Regression (GMR) model of index K, abbreviated as GMR(K) or $m(x; K)$. Notice that the regression function $m(x)$ derived from the joint mixture Gaussian density is of the form of a kernel estimator. However, there is a key difference: the weight function $w_j(x)$ is not determined by local structure of the data but by the components of a global Gaussian mixture model. Therefore GMR is a global parametric model with nonparametric flexibility.

Obviously, the Nadaraya-Watson kernel smoother is a GMR model of index $K = n$, GMR(n), with the prior mixing probability $\pi_j = 1/n$, and the covariance $\Sigma_{jYX} = 0$. At the other end of the spectrum, GMR(1) is approximately the classical linear model.

Hence, the GMR sequence of models covers a spectrum of regression models of varying flexibility, ranging from the nonparametric kernel regression ($K = n$) to the classical linear model ($K = 1$).

Another feature of GMR is that the pointwise variance function $v(x) = \text{Var}[Y|X = x]$ is readily available. The function $v(x)$ provides a second-order summary statistic of the conditional distribution. Finally, unlike the local kernel estimators whose weights are determined solely by local features of the data, the GMR estimator (4.12) offers an elegant composition of the global and the local information of the data.

4.3 Estimating the GMR Model Parameters

The main computing task for fitting GMR is the estimation of the parameters of the Gaussian mixture model for the joint density $f_{X,Y}$. In this section, we explore some theoretical aspects and develop algorithms to estimate the Gaussian mixture model parameters.

The main difficulty in estimating Gaussian mixture models is picking the right number of components. We avoid the issue of picking the right K by an obvious observation: the best mixture for the density f_X is somewhere between the saturated, n-component mixture, $f_X(x) = \sum_{i=1}^{n} n^{-1}\phi(x; x_i, \Sigma)$, and the one-component simple Gaussian $f_X(x) = \phi(x; \bar{x}, S)$. And instead of picking K from the density estimation perspective, we delegate the choice of K to the GMR model selection procedure.

The parameter estimation in the GMR model relies on the MoM estimates employed by IPRA. Although this is not MLE, its computation is fast and straightforward. Because we defer the choice of K to the later stage, it is not necessary to obtain the MLE for every GMR(K). A more reasonable strategy is to use the method of moments estimation for every GMR(K), then apply model selection for picking the best K with respect to the mean squared error or prediction error. Once a good K is determined, we could use the EM algorithm to obtain the MLE fit for the selected GMR(K), using the starting values provided by the MoM fits of GMR(K). The IPRA/MoM fit is a natural choice for an initial value for the EM algorithm. Lindsay and Basak (1993) point out that there is no known simple, effective way to obtain starting values for the MLE. The IPRA/MoM could be used as a procedure to select good K and provide natural initial value with it so that it will be much easier to use the Gaussian mixture models for data analysis.

In our empirical study, we compare the IPRA/method of moments fit of GMR and their MLE update. The result of our study shows that the MLE update usually sharpen the fit, that is, GMR/MoM fit is smoother than GMR/MLE fit. Whether it is beneficial or not is data dependent. We will encounter an example where MLE ameliorates the model fit in Section 4.7.3.2; and another example in Section 7.3.1.1, where the MLE update increases the test error in classification problems. It is a good strategy to always compute the MLE update of the best GMR(K) model selected.

4.4 Initial Kernel Bandwidth Selection

The bandwidth selection is a central issue in both kernel regression and kernel density estimation. We argue that for GMR, the optimal bandwidth with respect to the kernel density estimation is an upper bound for our bandwidth selection problem in (3.1). That is, a suboptimal bandwidth on the overfitting (i.e., under-smoothing) side should serve our purpose well. Our reasoning is that because GMR(1) always underfit (over-smooth) the data, we want to make sure that the initial model GMR(n) is an overfit so that the "right" model is contained in the GMR model sequence.

Scott (1992, p.150) studies the bias and variance of product kernel estimator of the form:

$$\widehat{f}_n(x) = \frac{1}{nh_1 \dots h_d} \sum_{i=1}^{n} \prod_{j=1}^{d} K(\frac{x_i - x_{ij}}{h_j}). \tag{4.14}$$

According to Scott's analysis, when K in (4.14) is Gaussian, the bandwidth that minimizes the Asymptotic Mean Integrated Squared Error (AMISE) is

$$h_j^* = \big(\frac{4}{d+2}\big)^{1/(d+4)} n^{-1/(d+4)} \sigma_j. \tag{4.15}$$

As dimension d varies, the constant in h_j^* ranges over the interval $(.924, 1.059)$, which is tightly around 1. Therefore Scott recommended the simple rule:

$$\widehat{h}_j = \widehat{\sigma}_j \, n^{-1/(d+4)}. \tag{4.16}$$

In practice, Scott's $\widehat{h}_j$ can be used as a guideline for GMR. Since we rescale the data before applying GMR so that $\widehat{\sigma}_j = 1$ for every dimension j, we can use the same h for every dimension and Scott's rule is simplified as

$$h_S = n^{-1/(d+4)}. \tag{4.17}$$

h_S is optimal when the density is product of Gaussian, in other words, a multivariate Gaussian with diagonal covariance matrix. Our target density is Gaussian mixture. Therefore an adjustment is required. We apply a simple adjustment by multiplying h_S with a factor 1/20.

To ensure an overfitting GMR(n), we could pick a bandwidth smaller than Scott's recommendation so that the initial GMR(n) overfits the data. A sensitivity analysis is conducted on the Motorcycle data to demonstrate this concept. The details are in Section 7.2.3.1.

Precisely, an under-smoothed kernel estimate leads to regression prediction that almost interpolates the data points, that is, $\widehat{y}_i = \widehat{m}(x_i) \approx y_i$. On the other hand, an over-smoothed estimate gives predictions closer to the classical linear model (global least-square) fit:

$$\widehat{m}(x_i) \approx \bar{Y} + x_i^T (X^T X)^{-1} X^T Y.$$

A case study using the Motorcycle data (see Section 7.2.3) demonstrates that choosing h less then the optimal h_{CV} will lead to an almost identical final GMR model fit. This suggests that when applying the GMR procedure, we can use a simple rule of thumb to pick a sufficiently small h. One simple option is to use the minimum distance between two consecutive observations in each dimension:

$$h_m = \min_{j,k}\{|X_{k,j} - X_{k,j-1}|\}, \tag{4.18}$$

with each dimension X_k standardized and $x_{k,j} \neq x_{k,j-1}$.

One cautious point in choosing h for GMR is that if h is too small, a singularity may occur and make the GMR numerically unstable. To guard against an h too small, we pick the maximum of (4.18) and Scott's rule (4.17) with adjustment. Our resulting rule of thumb of bandwidth selection is

$$h^* = \max\{h_m, 1/20\, h_S\}. \tag{4.19}$$

In practice, it is prudent to use several different values of h to initiate the IPRA/GMR procedure. It allows us the compare the RMSE profile generated by different h.

4.5 GMR Model Selection

Once the bandwidth h is determined, the IPRA procedure will generate a sequence of GMR models indexed by the number of components K. The next crucial task is the selection of K. In this section we first discuss the difference between selecting K for the underlying density and selecting K for the regression function. Usually the latter is smaller. This is a direct consequence of a theorem we prove in the next section.

Bearing this difference in mind, we will focus on selecting K for the regression function. The standard model selection criterion for this task is the mean squared error (MSE). When there are test data available, we will fit GMR procedures on the training data and then apply them on the test set to estimate the prediction error (PE). In general, PE is a better criterion than MSE. When the sample size is small and computationally feasible, we apply leave-one-out cross-validation technique to estimate the prediction error. We define MSE and leave-one-out cv in Section 4.5.2. We then discuss the meaning of residuals of GMR models. A word of caution is that the usual justification of using MSE as a model selection criterion is that the residuals are iid white noise. This is not true in GMR. We further discuss this topic in Section 4.5.3.

4.5.1 Redundant Component Theorem

The central issue in GMR model selection is the parameter K. We argue that since GMR is designed to be an EDA tool and not to produce the definitive final model,

the goal of GMR model selection is not to pick the definitive K but to pick several reasonable ones. An important point is that the K that is right for the regression is not necessarily the right K for the underlying density, $f_{X,Y}$. In general, the best K for the underlying density estimation is larger than necessary for GMR fit. We present Theorem 4.5.1 to demonstrate this point. Theorem 4.5.1 also suggests the non-uniqueness of the best K. We call it the Redundant Component Theorem.

Theorem 4.5.1 (Redundant Components). *Let (μ_j, Σ_j) and (μ_k, Σ_k) be two distinct components of a GMR model. If they satisfy the conditions that $\mu_{jX} = \mu_{kX}$ and $\Sigma_{jX} = \Sigma_{kX}$; then merging the two components does not affect the overall $m(x)$.*

The proof is straightforward. Recall the weight function of the GMR in (4.11):

$$w_j(x) = \frac{\pi_j \phi(x; \mu_{jX}, \Sigma_{jX})}{\sum_{j=1}^{K} \pi_j \phi(x; \mu_{jX}, \Sigma_{jX})}.$$

One natural interpretation of $w_j(x)$ is that $w_j(x) = Pr(G = j|X = x)$, where G is the latent mixture indicator variable. The assumption $\mu_{jX} = \mu_{kX}$ and $\Sigma_{jX} = \Sigma_{kX}$ implies that for all $X = x$,

$$\frac{w_j(x)}{w_k(x)} = \frac{\pi_j \, \phi(x; \mu_{jX}, \Sigma_{jX})}{\pi_k \, \phi(x; \mu_{kX}, \Sigma_{kX})} = \frac{\pi_j}{\pi_k} = \text{constant}.$$

It is straightforward to show that

$$w_{jk}(x) \, m_{jk}(x) = w_j(x) \, m_j(x) + w_k(x) \, m_k(x), \tag{4.20}$$

where

$$w_{jk}(x) = w_j(x) + w_k(x)$$

,

$$\begin{aligned} m_j(x) &= \mu_{jY} + \Sigma_{jYX} \, \Sigma_{jX}^{-1} \, (x - \mu_{jX}), \\ m_k(x) &= \mu_{kY} + \Sigma_{kYX} \, \Sigma_{kX}^{-1} \, (x - \mu_{kX}). \end{aligned}$$

From the MoM equations (3.3), (3.4), and (3.5), we have the merged parameters:

$$\begin{aligned} w_{jk}(x) &= w_j(x) + w_k(x), \\ \mu_{jk} &= \frac{w_j(x)\mu_j + w_k(x)\mu_k}{w_{jk}(x)}, \\ \Sigma_{jk} &= \frac{w_j \Sigma_j + w_k \Sigma_k}{w_{jk}(x)} + \frac{w_j w_k}{w_{jk}^2}(\mu_j - \mu_k)(\mu_j - \mu_k)^T \\ &= \frac{w_j \Sigma_j + w_k \Sigma_k}{w_{jk}(x)}. \end{aligned}$$

Now we verify (4.20):

$$\begin{aligned} w_{jk}(x)m_{jk}(x) &= w_{jk}(x)\mu_{jkY} + w_{jk}(x)\Sigma_{jkYX}\Sigma_{jX}^{-1}(x-\mu_{jX}) \\ &= w_j(x)\mu_{jY} + w_k(x)\mu_{kY} + (w_j(x)\Sigma_{jYX} + w_k(x)\Sigma_{kYX})\Sigma_{jX}^{-1}(x-\mu_{jX}) \\ &= w_j(x)m_j(x) + w_k(x)m_k(x). \end{aligned}$$

Finally,

$$\begin{aligned} m(x) &= \sum_{i=1}^{K} w_i(x)m_i(x) \\ &= w_j(x)m_j(x) + w_k(x)m_k(x) + \sum_{i\neq j,k} w_i(x)m_i(x) \\ &= w_{jk}(x)m_{jk}(x) + \sum_{i\neq j,k} w_i(x)m_i(x). \end{aligned}$$

Therefore, the MoM merging of $m_j(x)$, $m_k(x)$ within the overall GMR $m(x)$ does not change the value of $m(x)$.

Theorem 4.5.1 states that if there is no difference between two mixture components with respect to their *marginal distribution* in X, then merging the two components will not change the overall regression function $m(X)$. Figure 4.1 demonstrates this fact using an example of a 5-component Gaussian mixture $f_{X,Y}$. The two components located at $x = 6$ have identical marginal density f_X with $\mu = 6.0$, $\sigma^2 = 5.0$. The line that connects the centers of the other 3 components and goes through the middle of the two with identical f_X is the GMR(5) fit. Merging the two components gives the big component with the same f_X and an inflated variance in Y. The inflated σ_Y indicates that the underlying 4-component Gaussian mixture has a terrible fit of the true underlying density. However, the resulting GMR(4) model yields identical regression curve to the one of GMR(5).

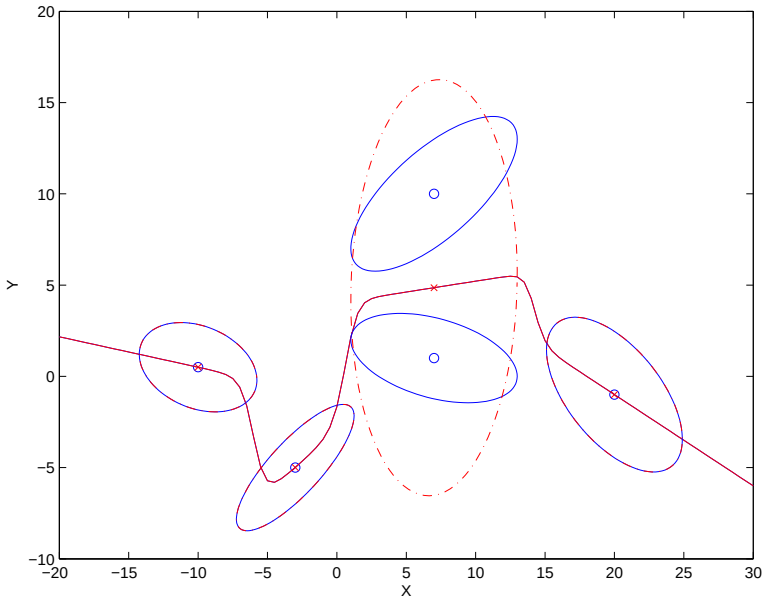

Figure 4.1: An example of redundant components in GMR models

From the density model perspective, if the two components are far apart in Y, then merging the two components diminishes such information. Geometrically, in this example, the conditional distribution $Y|X = 6$ is bimodal and the conditional mean is not a good summary anyway. Merging the two modes does not change the conditional mean. Based on Occam's razor principle, we will favor the merging operation in this situation.

Another direct implication of Theorem 4.5.1 is that GMR models of different K's may have very similar regression curves. A sensible solution is to pick the GMR model with the smallest K among all models of similar MSE and predictive error. We demonstrate applications of GMR in Chapter 7.

4.5.2 Model Selection Criteria

The most common regression model selection criterion is mean squared error (MSE). MSE is defined as

$$\text{MSE}(\widehat{m}) = E[Y - \widehat{m}(X)]^2.$$

In most of the figures and tables we record the root-mean squared error (RMSE), which is the square root of MSE: $\text{RMSE}(\widehat{m}) = \sqrt{\text{MSE}(\widehat{m})}$. In practice, MSE is computed as

$$\text{MSE}(\widehat{m}) = n^{-1} \sum_{i=1}^{n} \left(y_i - \widehat{m}(x_i)\right)^2,$$

for a given model $\widehat{m}$.

Obviously a model that reproduces the original data, $\widehat{m}(x_i) = y_i$, has MSE = 0. This means that the MSE criterion favors models that overfit the data. A more sensible criterion is to evaluate a model based on its performance in predicting future observations. When the data are abundant, a simple practice is to split the data into a training set and a test set, fit the model using the training set, and than evaluate the mean squared error on the test set. A sensible idea known as leave-one-out cross-validation is very popular, especially when the sample size is moderate. One good thing about leave-one-out cv is that it uses $n-1$ data points to "train the model;" thus the leave-one-out cv evaluation of the predictive performance of the procedure is closer to the one using all n data points.

For a data set of size n, the leave-one-out cv procedure fits a model without the j-th observation (x_j, y_j); and then applies the model to evaluate the response y_j. Precisely,

$$\mathrm{cv}(K) = n^{-1} \sum_{j=1}^{n} (y_j - \widehat{m}_{-j}(x_j; K))^2. \tag{4.21}$$

In this thesis we use both test error and leave-one-out cross-validation to select the best GMR and GMC models. Applications of GMR are reported in Chapter 7.

4.5.3 The Residuals of GMR

One important distinction between GMR and other nonparametric methods is that GMR starts with the joint density model instead of the conditional density. A direct consequence is that the residuals of GMR are not homoscedastic. Therefore the usual residuals analysis for model diagnostics is not applicable in GMR. To be more specific, standard nonparametric models assume

$$Y_i = m(X_i) + \varepsilon_i, \tag{4.22}$$

where

$$\varepsilon_i \sim N(0, \sigma^2). \tag{4.23}$$

By nature, model (4.22) is unbiased and homoscedastic, and we can always use residuals for model diagnosis by checking if (4.23), the exact residuals distribution given the model (4.22), is correct.

The distribution of the residuals of GMR is far more complicated than (4.23), so it is difficult to apply residuals for GMR model diagnosis. Precisely, a GMR(K) model can be written as

$$(Y|X = x_i, G = k) \sim N(m(x_i), \sigma_k).$$

Therefore, the distribution of the residuals of GMR, $\varepsilon_i = y_i - m(x_i)$ is

$$(\varepsilon_i|G = k) \sim N(0, \sigma_k).$$

An overall pdf of ε is a K-component Gaussian mixture

$$f_\varepsilon(u) = \sum_{k=1}^{K} w_k \phi(u; 0, \sigma_k),$$

with the mixing weights $w_k = Pr(G = k|X)$.

Because the mixing weights w_k depend on X, the residuals of GMR are not suitable for a global model diagnosis. In practice, we apply the bootstrap method to estimate the bias and variance of the GMR model fits. Since the residuals of the GMR models are not iid, it is not applicable to bootstrap from the residuals.

4.6 The Bayesian Perspectives of GMR

One advantage of a Bayesian approach to GMR is that it offers a way to quantify the finite-sample behavior of the model. From the computational point of view, the Bayesian approach is natural for mixture models because it allows the unobserved cluster indicator Z to play a significant role in the inference, and hence simplifies the MLE part of the model fitting. The literature on the Bayesian approach in mixture models is abundant, especially since the rediscovery of the Markov-Chain Monte-Carlo (MCMC) algorithms. Bayesians finally have a powerful tool to execute the intensive Bayesian computations. Many recent articles are interested in mixture models with an unknown number of components. Richardson and Green (1997) propose a reversible jump MCMC algorithm for finite mixture models with unknown number of components. Under Brian Ripley's supervision, Matthew Stephens used a Markov birth-death process to study the same subject in his Ph.D. thesis:Stephens (1997).

The posterior probability $Pr(K|X, Y)$ is not the focus of this dissertation. We contend that for the purpose of EDA, we do not need a precise estimate of $Pr(K|X, Y)$. Because when the target is the regression function, different values of K can give approximately the same regression function. As Theorem 4.5.1 implies that when the marginal $f_{X|G=j}$ and $f_{X|G=k}$ are very similar, then merging the j-th and the k-th components does not affect the overall $m(X)$. Hence the value of K is not too crucial for GMR.

4.7 Simulation Study of GMR

One main difficulty in the asymptotic analysis of GMR is that the underlying Gaussian mixture density estimation is not well-defined. Because the likelihood function is unbounded, the ordinary MLE does not exist. Therefore it requires regularization to have a unique MLE fit. Since mean squared error (MSE) and predictive error (PE) are the objective functions for GMR model selection, they provide the necessary regularization because both MSE and PE are bounded.

Another aspect of GMR is that different Gaussian mixtures models for the underlying pdf $f_{X,Y}$ can yield very similar regression fits, a direct ramification of the Redundant Component Theorem discussed in Section 4.5.1.This phenomenon implies that the best model for GMR may not be unique, even if the sample size goes to infinity. We observe this in our simulation study. In both theory and practice, the non-uniqueness of the best models is hardly a bad thing. We will demonstrate the non-uniqueness of GMR(K) in our simulation study.

Our intuition is, by forcing GMR(n) to overfit, the best model GMR(K) is included in the GMR model sequence. Conceptually, for every given K, there exists a best model for the data in terms of minimum MSE or PE. One approach to approximate this best model sequence is to fit the MLE of the K-component Gaussian mixture for each $K = 1, \ldots, n$ by the EM algorithm. However, this computation is too expensive and unnecessary. IPRA provides an alternative way to generate a sensible path through the model space. The resulting GMR model sequence provides a good representitive sample of the model space, and consequently, it gives us a systematic way to choose the best K for a given data set. As long as the initial kernel model, GMR(n), overfits, we will have a good chance of finding the appropriate K. We verify this intuition by a simulation study. Any more rigorous proof is a subject of future research.

Following is a sketch of our simulation study of the consistency of GMR:

1. Simulate a training data set and a test set of the same size n from a K-component mixture density $f_{X,Y}$.

2. Fit GMR(k) on the training set, computing the MSE and PE of GMR(k) as $k = 1, \ldots, n$.

3. Repeat steps 1 and 2 m times, computing the summary statistics of MSE and PE.

4. Run the same simulation for several different sample sizes, n.

We can draw several inferences out of the simulation study:

1. Evaluate the effect of sample size on MSE and PE.

2. Verify if GMR(k) can achieve the minimum PE of the true model.

3. Estimation of bias and variance of the GMR fits.

4.7.1 GMR on Null Data

The first example is designed to investigate how GMR perform on the independent noise, that is, when the covariates X contain no information regarding the response

Y, the true regression model should be $y = \bar{y}$. In this situation, a sensible regression procedure should produce global constant fit, the sample mean.

We simulated 30 null data sets, $X_1, \ldots, X_p$, Y, and random error ε, each from the standard Gaussian distribution. We repeated the simulation with $p = 5, 20$ and $n = 100$. It is interesting to observe that for $p = 5$ scenario, as shown in Figure 4.2, a typical RMSE curve of GMR models appears to have a corner around $K = 11$. After that, it goes straight to $K = 1$. Similar pattern is apparent in $p = 20$ scenario (Figure 4.3), but the first corner is around $K = 7$. Other than this anomaly, the RMSE curves in both scenario indicate that the best is model GRM(1). We will see in later analysis that this first corner or shoulder of the RMSE curve usually indicates the overfit adjustment for the overfit caused by a bandwidth too small.

The $p = 20, n = 100$ case in Figure 4.3 is satisfactory, consider the relatively small sample size in the 20-D feature space. It indicates that GMR will generally ignore spurious features. When modeling null data, the GMR profile (RMSE vs K) will indicate no sharp decrease of RMSE in a specific K; therefore suggest the simplest GMR(1) fit. It is still not a constant, but we consider a global linear model is the simplest fit GMR can achieve.

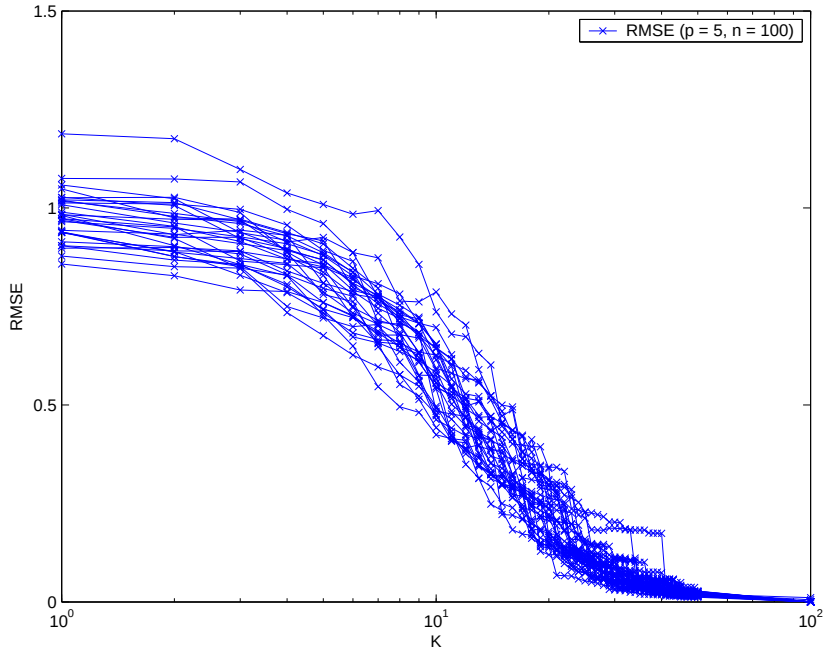

Figure 4.2: RMSE of GMR models for null data ($n = 100$, $p = 5$)

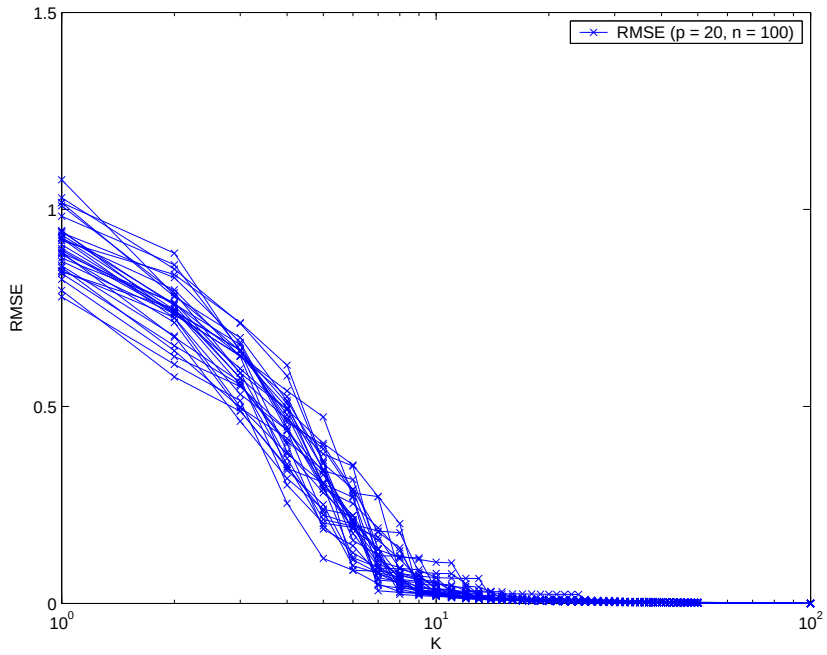

Figure 4.3: RMSE of GMR models for null data ($n = 100$, $p = 20$)

4.7.2 GMR on True GMR Data

The goal of this example is to investigate how well GMR picks up the correct number of components K. Instead of simulating Y from a given function $g(X)$, we simulate the data from a Gaussian mixture pdf of five components, that is, $(X_i, Y_i)|G = k \sim N_{p+1}(\mu_k, \Sigma_k), k = 1, \ldots, 5$.

We simulate 300 training sets, and 300 test sets with $p = 5, n = 300$. The result in Figure 4.4 indicates that the best GMR model is around GMR(3), GMR(4), and GRM(5). A empirical confirmation of the condition predicted by the redundant component theorem.

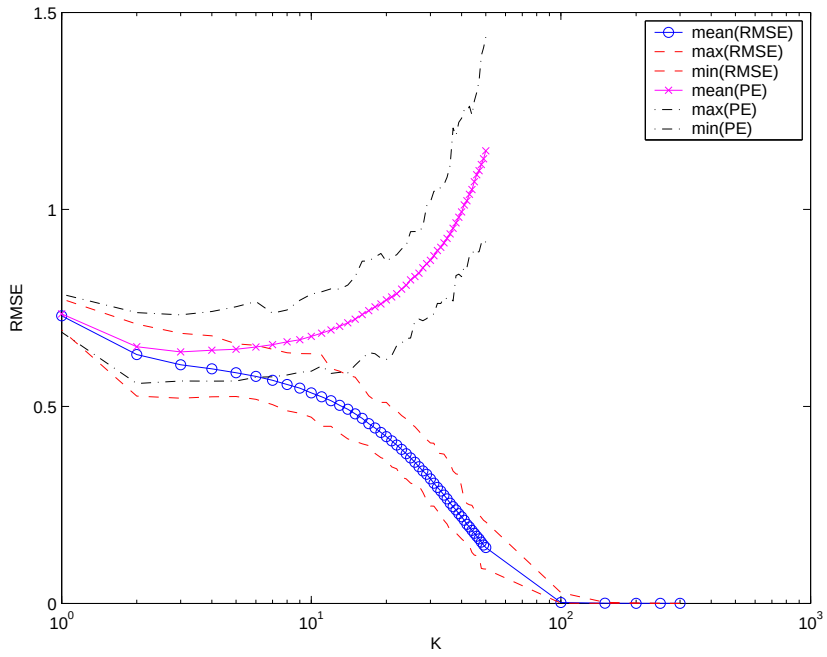

Figure 4.4: RMSE of GMR models of the simulated 5-component Gaussian mixture data ($n = 300$, $p = 5$)

4.7.3 GMR vs MARS

Multivariate Adaptive Regression Splines (MARS) is one of the most versatile non-parametric regression procedures for multivariate data. In this section we apply both GMR and MARS procedures to several examples to compare the two procedures.

4.7.3.1 Tensor Products and Neural Network Data

Hastie, Tibshirani, and Friedman (2001, p.288) apply MARS on three simulated examples. Each one has sample size $n = 100$. The first example is

$$Y = (X_1 - 1)_+ + (X_1 - 1)_+ (X_2 - .8)_+ + 0.12 \cdot \varepsilon, \tag{4.24}$$

where X_1, X_2, and ε are all simulated from the standard Gaussian distribution. The second example is the same as example 1, with extra 18 independent standard Gaussian noise. That makes it total $p = 20$ predictors. They call the first two examples Tensor product $p = 2$ and $p = 20$. These two examples are especially designed in favor of MARS. The third example has a more complicated, neural network style

structure that will be harder for MARS to pick up:

$$\begin{aligned} l_1 &= X_1 + X_2 + X_3 + X_4 + X_5, \\ l_2 &= X_6 - X_7 + X_8 - X_9 + X_{10}, \\ \sigma(t) &= \frac{1}{1+e^{-t}}, \\ Y &= \sigma(l_1) + \sigma(l_2) + 0.12 \cdot \varepsilon. \end{aligned}$$

For each model fit of the simulated data, Hastie et al. (2001) compute the value of MSE and R^2, defined as the following:

$$\begin{aligned} MSE_0 &= n^{-1} \sum_{i=1}^{n} (\bar{y}_i - \mu(x_i))^2, \\ MSE &= n^{-1} \sum_{i=1}^{n} (\widehat{m}(x_i) - \mu(x_i))^2, \\ R^2 &= \frac{MSE_0 - MSE}{MSE_0}, \end{aligned}$$

where $\mu(x)$ is the true mean of Y and $\widehat{m}(x)$ the fitted value. MSE_0 is the mean squared error of the null model $\widehat{m}(x) = \bar{y}$.

We applied the same simulation procedure on MARS. Table 4.1 records the results of 100 simulations of GMR and the corresponding values of MARS from Hastie et al. (2001, p.289). As expected, in the first two examples, MARS perform almost perfectly. With the neural network data, however, MARS is less efficient. Interestingly, GMR performs very well in the neural network data. For the first two Tensor product data sets, GMR is not as good as MARS, yet its performance is satisfactory. One interesting aspect of GMR shown in Table 4.1 is that the extra noise dimensions do hurt GMR. This is because GMR fits the global pdf model. MARS shows great strength in filtering out noise dimensions. In the second scenario, 18 more noise dimensions have no effect on MARS at all. This is because MARS fits the data one dimension at a time.

Among all three examples, GMR shows good balance with comparable performance to MARS in example 1. And an excellent result in the neural network example. We conjecture that GMR can outperform MARS when the true regression function contains smooth nonlinear interactions such as example 3. This conjecture is further reaffirmed in the next example.

4.7.3.2 Interaction Spline Example

Friedman (1991) applies his MARS procedure to a simulated example first investigated by Gu et al. (1989). The original example consists of 300 points in the

Table 4.1: The R^2 values of MARS and GMR on 100 simulated examples

	MARS		GMR	
Scenario	Mean	(S.E.)	Mean	(S.E.)
Tensor product ($p = 2$)	0.97	(.01)	0.93	(.04)
Tensor product ($p = 20$)	0.96	(.01)	0.83	(.07)
Neural Network	0.79	(.01)	0.94	(.01)

unit square $[0,1] \times [0,1]$. That is, (x_{1i}, x_{2i}) are iid samples simulated from uniform $[0,1] \times [0,1]$ distribution. The true regression function is (4.25) *

$$f(x_1, x_2) = \frac{40\, exp\{8[(x_1 - .5)^2 + (x_2 - .5)^2]\}}{exp\{8[(x_1 - .2)^2 + (x_2 - .7)^2]\} + exp\{8[(x_1 - .7)^2 + (x_2 - .2)^2]\}} \quad (4.25)$$

The simulated response y is

$$y_i = f(x_{1i}, x_{2i}) + \varepsilon_i,$$

where $\varepsilon_i, i = 1, \ldots, 300$ are simulated from the standard Gaussian $N(0,1)$. Figure 4.5 shows the true regression surface and the simulated data y_i. The MARS fit in Figure 4.6 is generated by the R function *mars* implemented by Trever Hastie and Rob Tibshirani in their R-package *mda*, which is freely available in http://www.r-project.org.

The GMR fit involves a little more work. We first applied GMR procedure with bandwidths $h = .1$ and $h = .001$. We computed the RMSE of the GMR fits on both the true function values $f(x_{1i}, x_{2i})$ and the observations y_i. The RMSE of the GMR fits are shown in Figure 4.9, which reveals several interesting points. First of all, observe that the RMSE of the clearly overfit GMR path initiated by $h = .001$ merges with the more mild $h = .1$ fits after about $K = 10$. The same pattern also shows in the RMSE for the true function values. Since both MSE curves go flat after $K = 5$, it indicates that the best GMR model fit is GMR(5). In this regard, the bandwidth selection is almost irrelevant. We will revisit the bandwidth selection issue again in our analysis of the Motorcycle data in Section 7.2.3.1 and Figure 7.8.

The second point in Figure 4.9 is that the RMSE with respect to $f(x_1, x_2)$ initiated by $h = .001$ is very close to the RMSE of $h = .1$ on y. And the RMSE by $h = .001$ is almost 0. Both indicate that $h = .001$ is too small. On the other hand, the parallel pattern of the RMSE curves by $h = .1$ suggests that $h = .1$ is an appropriate bandwidth for this simulated data.

Using GMR(5) with MoM parameter estimation, we applied the EM algorithm to obtain the MLE updates. Figure 4.7 and Figure 4.8 are the GMR(5)/MoM and

*We obtain (4.25) directly from Gu et al. (1989). It is worth noticing that the formula in Friedman (1991, p. 50) is printed incorrectly. And the year of Gu et al. (1989) cited by Friedman (1991) is 1990, which is incorrect, too.

	MARS	Interaction Splines	GMR(5)/MoM	GMR(5)/MLE
$\sqrt{E[f-\widehat{y}]^2}$	.6958	.3298	.8230	.3698
$\sqrt{E[y-\widehat{y}]^2}$	1.1532	NA	1.2894	1.0364

Table 4.2: The RMSE of models for the interaction spline example

GMR(5)/MLE fits of the data. In comparison to the MARS fit in Figure 4.6, it is obvious that both GMR(5) models give a smoother fit than the MARS and the interaction spline model fit as shown in Figure 4 of Gu et al. (1989). The actual RMSE of all the models in comparison is in Table 4.2. The numbers show that the interaction spline model has the smallest RMSE on the true $f(x_1, x_2)$, GMR(5)/MLE is at a close second. The RMSE on y shows that the MARS and GMR(5)/MLE are pretty close and GMR(5)/MoM is not far behind. Considering that the standard error of the white noise is 1.0473, all three models provide good fits. Gu et al. (1989) have not reported the RMSE or MSE on y of their interaction spline so it is not given. However, comparing both GMR(5)/MoM and MLE in Figure 4.7 and Figure 4.8 to the MARS fit in Figure 4.6, it is fair to conclude that both GMR models yield smoother fits than MARS in this example. The same conclusion holds in comparing the original MARS fit in Figure 11 of Friedman (1991) and the interaction spline fit in Figure 4 of Gu et al. (1989). Considering both the MSE and the 3-D plots, GMR(5)/MLE is the best model for this 2-D simulated data.

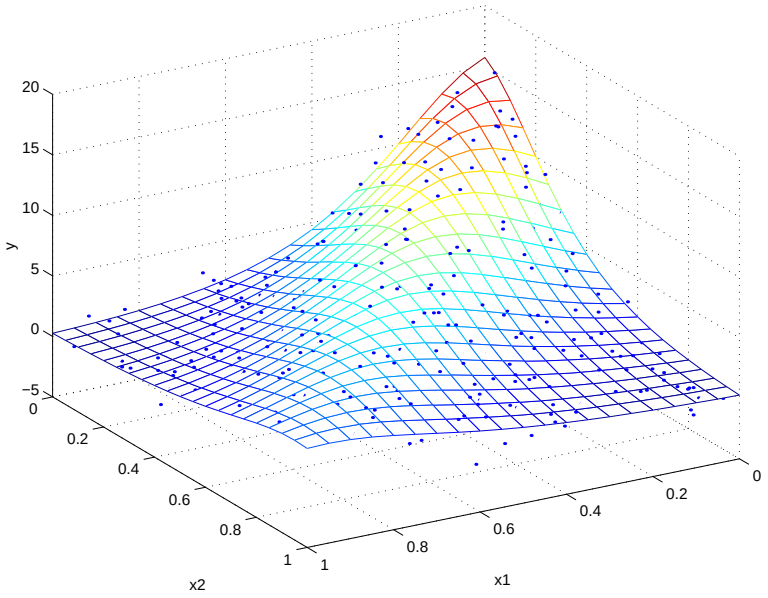

Figure 4.5: The simulated interaction spline example

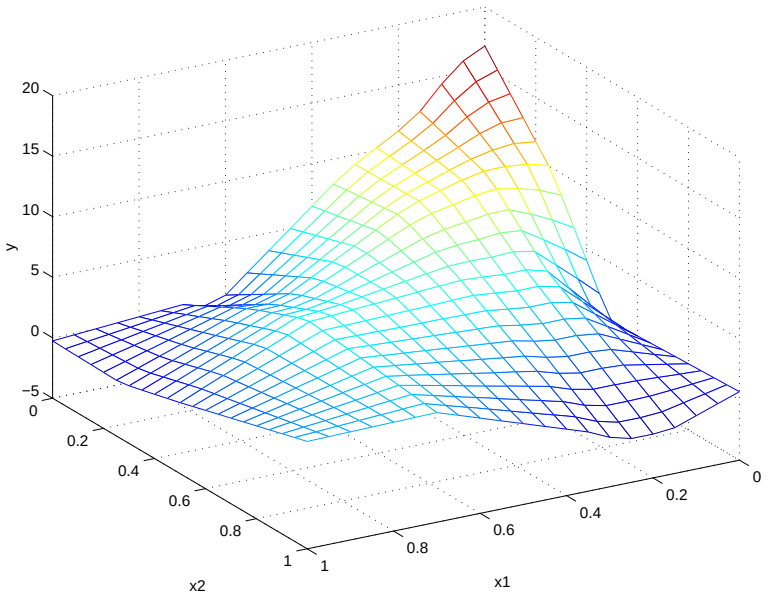

Figure 4.6: The MARS fit of the simulated interaction spline example

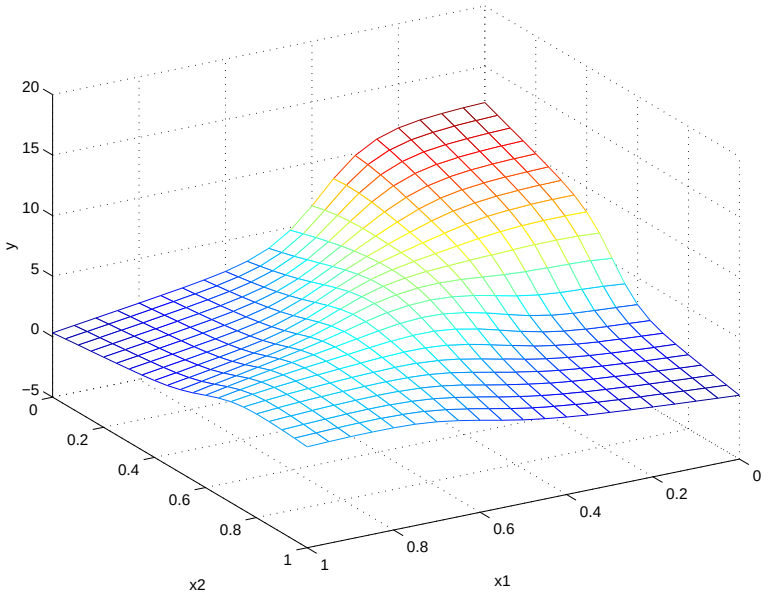

Figure 4.7: The GMR(5)/MoM fit of the interaction spline example

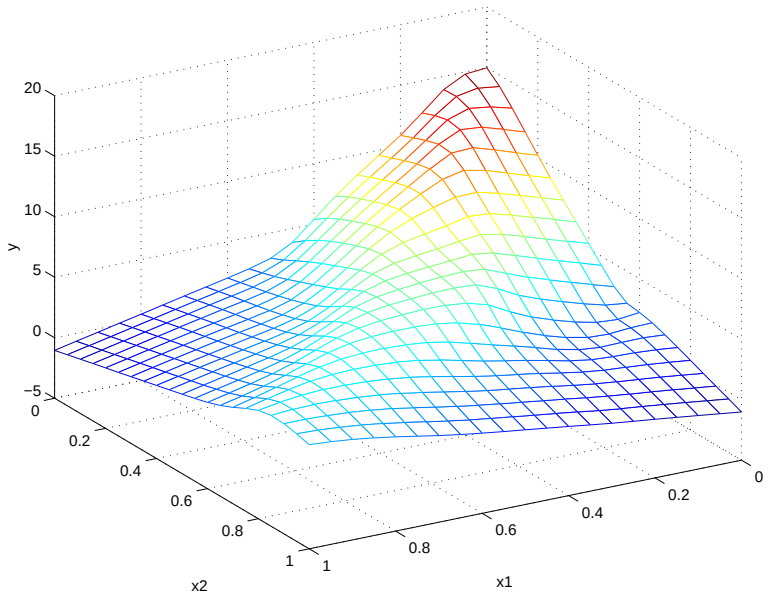

Figure 4.8: The GMR(5)/MLE fit of the interaction spline example

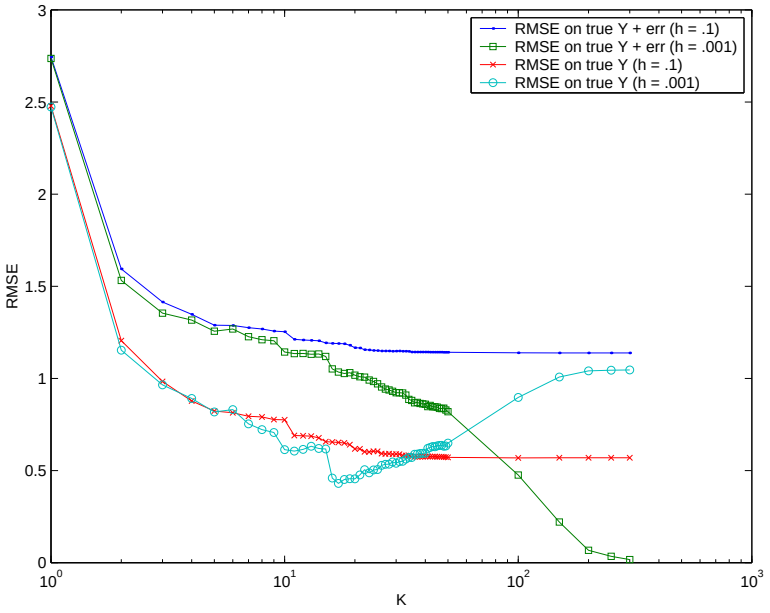

Figure 4.9: The RMSE of GMR on the interaction spline example

4.7.4 Conclusion

We have applied the GMR procedure on simulated examples to explore its properties. In Section 4.7.1 we applied GMR to the null data, that is, when the feature space X contains no information on the response Y. We found that GMR does not pick up any frivolous pattern.

In Section 4.7.2, GMR was put to test on data simulated from a 5-component Gaussian mixture. We found that both the RMSE and PE of GMR suggest the best model is around $K = 3, 4, 5$. This is not surprising because the Redundant Component Theorem implies that a smaller K can provide similar regression fits.

Finally, in Section 4.7.3 we apply GMR and MARS on 4 examples. The finding is very interesting. It appears that when the true regression function consists of a smooth nonlinear term that includes more than two variables, GMR will do a very good job in pick up the pattern. MARS, on the other hand, cannot properly fit such function without either using higher interaction terms or additional summands. The evidence is reflected in the 2-D neural network data and 2-D interaction spline data in Section 4.7.3. On the other hand, MARS is very effective in fitting models with clear patterns along the axes of the feature space. Because MARS fits the data one dimension at a time, it can filter out a large number of noise dimensions. GMR fits the data as a whole; therefore, a large amount of noise dimensions can hurt GMR.

One unique feature of GMR is that it is coordinate invariant. That is, the rotation of coordinate axes of the feature space does not affect the structure of the GMR model. On the other hand, both GAM and MARS are coordinate sensitive. The main reason is because they both fit the data dimension by dimension. It does not affect GMR because GMR is not based on the form of regression function but based on the underlying pdf. In particular, the MSE will be rotated with the data. Hence all IPRA/GMR models remain the "same" under any rotation of the coordinate axes of the feature space. This feature will prove to be very useful in our development of dimensional reduction projection of the data in Chapter 6.

Chapter 5

Gaussian Mixture Classification

5.1 Introduction

In this chapter we extend the GMR procedure to binary responses. That is, the data we consider here are $\{(X_i, Y_i)_{i=1}^n\}$, where $X_i \in \mathbb{R}^p$, $Y_i \in \{0, 1\}$. There are two major approaches to solving classification problems. The first one is to model the class boundary as a function of X. Examples of this approach include classification trees and support vector machines. The second approach is to construct classifiers using the class densities. The classical example is Fisher's linear discriminant analysis (LDA). The main theme of this research is to use density models to construct statistical procedures. Thus we will develop our classifier using density models. In statistical decision theory, the classifier using posterior probabilities $\Pr(Y|X)$ is called the Bayes classifier. It is well-known that the Bayes classifier, which minimizes the expected zero-one loss, is of the form:

$$\widehat{Y}(x) = \underset{g \in \{0,1\}}{argmin}\{1 - \Pr(Y = g|X = x)\}. \tag{5.1}$$

The error rate of the Bayes classifier is called Bayes risk or Bayes rate. For binary Y, (5.1) can be expressed in terms of posterior odds

$$r(x) = \frac{\Pr(Y = 1|X = x)}{\Pr(Y = 0|X = x)}. \tag{5.2}$$

That is,

$$\widehat{Y}(x) = I\{r(x) > 1\}. \tag{5.3}$$

Because of the optimal property of Bayes classifier, the posterior odds $r(x)$ is the target of most classification procedures. One classical example is the logistic regression model

$$\log r(x) = x^T\beta + \varepsilon, \tag{5.4}$$

which models $r(x)$ as a function of x directly.

In this chapter we focus on another approach to the estimation of $r(x)$. Apply Bayes Theorem to (5.2) we derive

$$r(x) = \frac{\Pr(Y = 1|X = x)f_X(x)}{\Pr(Y = 0|X = x)f_X(x)} = \frac{f_{X|Y}(x|1)\Pr(Y = 1)}{f_{X|Y}(x|0)\Pr(Y = 0)}. \tag{5.5}$$

Equation (5.5) suggests a natural algorithm: obtain an estimate for the density of each class, then take the ratio. That is,

$$\widehat{r}(x) = \frac{\widehat{f}_1(x)\pi_1}{\widehat{f}_0(x)\pi_0},$$

where $\widehat{f}_g$ is the model for $f_{X|Y=g}$ and π_g denotes the prior probability $\Pr(Y = g)$, $g = 0, 1$.

Fisher's linear discriminant analysis (LDA) is the classical procedure using the recipe suggested by (5.5). LDA assumes each class a Gaussian distribution with identical covariance matrix. That is, $f_0(x) = \phi(x; \mu_0, \Sigma)$ and $f_1(x) = \phi(x; \mu_1, \Sigma)$.

Hastie and Tibshirani (1996) propose mixture discriminant analysis (MDA). MDA is a natural extension of LDA. It models each class density as a Gaussian mixture. In their implementation, a user will specify the number of components for each density (K_0, K_1). MDA uses the EM algorithm to fit the MLE with the initial values generated by the K-means procedure.

In practice, specifying (K_0, K_1) is not an easy task; hence we would like to find a systematic way of selecting appropriate values of (K_0, K_1). This is the motivation of the GMC procedure we develop in the next section.

5.2 The GMC Procedure

In this section we formally develop the classifier suggested by (5.5). The Gaussian Mixture Classification (GMC) procedure consists of the following steps:

- Construct the initial kernel density estimator for each class $g = 0\,, 1$:

$$\widehat{f}_g(x; n_d, h) = \sum_{j=1}^{n_g} n_g^{-1}\phi(x; x_{gi}, h^2 I). \tag{5.6}$$

- Apply IPRA on $\widehat{f}_g(x; n_g)$ to generate a sequence of models for each class, $\{\widehat{f}_g(x, K_g) : K_g = n_g, \cdots, 1\}$, where

$$\widehat{f}_g(x, K_g) = \sum_{j=1}^{K_g} \pi_{gj}\phi(x, \mu_{gj}, \Sigma_{gj}). \tag{5.7}$$

- Construct the classifier $\widehat{Y}(x; K_0, K_1)$:

$$\widehat{Y}(x; K_0, K_1) = I\{\widehat{r}(x; K_1, K_0) > 1\}, \tag{5.8}$$

where

$$\widehat{r}(x; K_1, K_0) = \frac{\pi_1 \widehat{f}_1(x, K_1)}{\pi_0 \widehat{f}_0(x, K_0)}, \tag{5.9}$$

and $\pi_0 = n_0/(n_0 + n_1)$, $\pi_1 = n_1/(n_0 + n_1)$.

We view GMC as an alternative implementation of Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani 1996). In practice, one of the most difficult problems in using MDA is to provide as input the number of mixtures for each class. The standard MDA package in R sets the default to be $(K_0, K_1) = (3, 3)$. The second problem is to provide the initial parameters values for the EM algorithm. GMC does not force users to pick (K_0, K_1) nor initial guesses of the parameters. Instead, GMC selects from all possible combinations of (K_0, K_1). We discuss the model selection aspect of GMC in Section 5.3.

Another perspective is to view (5.9) as a nonparametric logistic regression procedure. That is,

$$\log r(x) = \log \frac{\pi_1}{\pi_0} + \log \widehat{f}_1(x, K_1) - \log \widehat{f}_0(x, K_0). \tag{5.10}$$

The logistic regression model perspective indicates the connection between the two main approaches to classification problems we mentioned earlier: model the class boundary and model the class densities. Since the classification error depends on the class boundary $\{x : \log r(x) = 0\}$, it is sensible to merge mixture components which are away from the boundary. This observation indicates that (K_0, K_1) that is good for class densities estimation may be too large for classification.

5.3 GMC Model Selection

The GMC procedure on binary responses produces a family of models which are indexed by two integer parameters (K_0, K_1). We know that $K_0 = 1, \ldots, n_0$ represents the number of mixture components for the Gaussian mixture density model of density $f_0(x)$ and $K_1 = 1, \ldots, n_1$ is the number of mixtures for density $f_1(x)$, where n_0 and n_1 are the sample sizes of the subsets corresponding to $Y_i = 0$ and $Y_i = 1$.

The model selection criterion of the GMC is straightforward. We will select the best models which yeild the minimum test error, or the misclassification rate on the test set. The test error is defined as

$$\mathrm{Err}(K_0, K_1) = n_t^{-1} \sum_{i=1}^{n_t} I\{y_i \neq \widehat{Y}(x_i; K_0, K_1)\}, \tag{5.11}$$

where $\widehat{Y}(x; K_0, K_1)$ is defined as in (5.8) and applied to the test data set $\{(x_i, y_i)\}_{i=1}^{n_t}$.

Applying IPRA separately to the initial models $\widehat{f}_1(x; n_1, h)$ and $\widehat{f}_0(x; n_0, h)$ produces two sequences of density models: $\{\widehat{f}_1(x; K_1)\}$ and $\{\widehat{f}_0(x; K_0)\}$. We compute the misclassification error of each model on the test data set. The result is a 3-D graph of (K_0, K_1) vs $\text{Err}(K_0, K_1)$. In practice, there can be more than one model that yields the minimum misclassification error. We will pick the model that gives the minimum error with the fewest total number of components $K_0 + K_1$.

5.4 Parameters Estimation: MoM vs MLE

Recall that IPRA uses the method-of-moments estimates of the model parameters, which are not MLE. In real applications, we may compute the MLE for the selected model using the EM algorithm. In such situations, the IPRA fit of the model parameters provides natural initial values for the EM algorithm.

In Chapter 7 we compare the MoM and MLE GMR fits of Motorcycle data and the MoM and MLE fits of GMC on the Zip Code data. The general conclusion is that MoM/GMR and MoM/GMC are good enough to help select the appropriate value of K in GMR and (K_0, K_1) in GMC. Once the K's are determined, the MoM parameter values are natural initial values for EM algorithm to obtain the MLE fit. Hastie and Tibshirani (1996) propose using the K-means procedure to generate sensible initial values for the EM algorithm. However, the users of their MDA procedure still have to pick the number of components (K_0, K_1). Their default selection is $(3, 3)$. We propose running GMC to obtain several reasonable models $\text{GMC}(K_0, K_1)$ and then run MDA. We apply this strategy in several data sets for classification. The details are in Chapter 7.

Chapter 6

Optimal Subspace Projection

In this chapter we describe the property of GMR of linearly projected data and two basic algorithms for the construction of the optimal subspace projection. The objective of a subspace projection algorithm is to construct a projection matrix $P \in \mathbb{R}^{p\times q}$ such that it projects the data X, $X \in \mathbb{R}^p$ onto $\mathbb{R}^q$, a subspace of lower dimension, while still retains the information of Y of a given model GMR(K).

There is an obvious trade-off: as the subspace dimension q decreases, the performance of the GMR model deteriorates. Bear this fact in mind, the optimal subspace here means the best subspace of a given dimension q. In practice, the subspace projection can be viewed as a model-based dimension reduction technique that serves as a companion to GMR.

In Section 6.1 we will show that the GMR is coordinate invariant. That means the structure of the GMR models for the original data does not change after any linear projection of X. This feature of GMR allows us to implement straightforward algorithms to construct optimal subspace projection for the data.

We describe two algorithms for constructing the optimal subspace projection: the forward algorithm in Section 6.2 and the backward algorithm in Section 6.3. Finally, in Section 6.4 we describe similar two algorithms for the GMC models.

Every algorithm described in this chapter has the same structure. It is to construct the subspace projection by solving a least squares problem iteratively. In other words, all four algorithms are characterized by the least squares problems they solve.

6.1 GMR of the Projected Data

One key feature of GMR is that its underlying Gaussian mixture density model is invariant under any linear projection of the data. In other words, the projection of the data does not change the structure of the GMR models. Therefore, there is no need to refit the GMR models of the projected data. In this section we derive the exact formula of GMR of the projected data.

Let B be a linear projection matrix $B : \mathbb{R}^p \mapsto \mathbb{R}^r$, $B \in \mathbb{R}^{p\times r}$, with $r < p$. Let Z

be the projected covariate such that $Z = B^T X$, we have $Z \in \mathbb{R}^q$ given $X \in \mathbb{R}^p$. The GMR model of Z can easily be derived from the GMR model of X based on a well-known property of Gaussian random variable: a linear projected Gaussian random variable is also a Gaussian random variable. Mardia et al. (1979, p. 62) state this properties as the following theorem.

Theorem 6.1.1. *If* $X \sim N(\mu_{_X}, \Sigma_{_X})$, *then*

$$Z = B^T X \sim N(\mu_{_Z}, \Sigma_{_Z}),$$

with parameters

$$\mu_{_Z} = B^T \mu_{_X}, \quad \Sigma_{_Z} = B^T \Sigma_{_X} B. \tag{6.1}$$

Theorem 6.1.1 is the foundation of the dimension reduction projection proposed by this thesis.

Recall from Chapter 4 that the GMR model is

$$m(x) = \mathrm{E}(Y|X = x) = \sum_{j=1}^{K} w_j(x)\, m_j(x), \tag{6.2}$$

where $m_j(x) = \mathrm{E}(Y|X = x, G = j)$ with G the latent class variable. The conditional density is $(X, Y|G = j) \sim N(\mu_j, \Sigma_j)$, where

$$\mu_j = \begin{bmatrix} \mu_{jX} \\ \mu_{jY} \end{bmatrix}, \quad \Sigma_j = \begin{bmatrix} \Sigma_{jX} & \Sigma_{jXY} \\ \Sigma_{jYX} & \Sigma_{jY} \end{bmatrix}.$$

The projected data can be written as

$$\begin{bmatrix} Z \\ Y \end{bmatrix} = \begin{bmatrix} B^T X \\ Y \end{bmatrix} = \begin{bmatrix} B & 0 \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} X \\ Y \end{bmatrix}.$$

Using Theorem 6.1.1, we derive the joint density of $(Z, Y|G = j)$ as $N(\mu_{jZY}, \Sigma_{jZY})$, where

$$\mu_{jZY} = \begin{bmatrix} B & 0 \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} \mu_{jX} \\ \mu_{jY} \end{bmatrix} = \begin{bmatrix} B\mu_{jX} \\ \mu_{jY} \end{bmatrix}$$

and

$$\Sigma_{jZY} = \begin{bmatrix} B & 0 \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} \Sigma_{jX} & \Sigma_{jXY} \\ \Sigma_{jYX} & \Sigma_{jY} \end{bmatrix} \begin{bmatrix} B & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} B^T \Sigma_{jX} B & B^T \Sigma_{jYX} \\ \Sigma_{jYX} B & \Sigma_{jY} \end{bmatrix}.$$

The matrix computation verifies that to obtain the GMR $\widehat{m}(z) = \mathrm{E}(Y|Z = z)$ from the original $\widehat{m}(x) = \mathrm{E}(Y|X = x)$ is straightforward. First, the joint density of Z, Y is

$$f_{Z,Y}(z, y) = \sum_{j=1}^{K} \pi_j \phi(z, y; \mu_{jZY}, \Sigma_{jZY}), \tag{6.3}$$

where

$$\mu_{jZY} = \widetilde{B}^T \mu_j, \quad \Sigma_{jZY} = \widetilde{B}^T \Sigma_j \widetilde{B},$$

with $\widetilde{B} = \begin{bmatrix} B & 0 \\ 0 & 1 \end{bmatrix}$. Therefore, the GMR(K) model for the projected data, $\widehat{m}(z) = \mathrm{E}[Y|Z=z]$ follows from (6.3), is

$$f_{Y|Z}(y,z) = \sum_{j=1}^{K} w_{zj}(z)\, \phi(z; m_{zj}(z), \sigma_{zj}^2), \tag{6.4}$$

where

$$\begin{aligned} m_{zj}(z) &= \mathrm{E}[Y|Z=z, G=j] = \mu_{j_Y} + \Sigma_{j_{YZ}}\, \Sigma_{jZ}^{-1}\, (z - \mu_{jZ}), \\ \sigma_{zj}^2 &= \mathrm{V}[Y|Z=z, G=j] = \Sigma_{j_Y} - \Sigma_{j_{YZ}}\, \Sigma_{jZ}^{-1}\, \Sigma_{jZY}, \end{aligned}$$

and

$$w_{zj}(z) = Pr(G=j|Z=z) = \frac{\pi_j \phi(z; \mu_{jZ}, \Sigma_{jZ})}{\sum_{j=1}^{K} \pi_j \phi(x; \mu_{jZ}, \Sigma_{jZ})}.$$

Thus GMR(K) of the projected data is

$$m(z) = \mathrm{E}[Y|Z=z] = \sum_{j=1}^{K} w_{zj}(z)\, m_{zj}(z). \tag{6.5}$$

Equation (6.5) implies that a GMR(K) model of the original data X will not change for the projected data $Z = B^T X$. That is, there is no need to refit the GMR model for Z. This feature of GMR allows straightforward implementation of subspace projection algorithms, which is the subject of the next two sections.

6.2 Forward Projection Algorithm

The idea of the forward projection algorithm is to construct the optimal subspace projection by adding the best 1-D projection iteratively. Start with the original feature space $\mathbb{R}^p$, the forward algorithm searches for the best 1-D projection such that the resulting GMR model $\widehat{m}(z), z = \beta^T x, z \in \mathbb{R}$ has the minimum MSE. Once the best 1-D projection from $\mathbb{R}^p$ to $\mathbb{R}$ is identified, we project the data onto the subspace of $\mathbb{R}^{p-1}$ that is orthogonal to β and search the best 1-D projection in $\mathbb{R}^{p-1}$. Thus we have a sequential algorithm. In each iteration, the forward projection algorithm finds the best 1-D projection that contains the most information of X for Y. Thus, the forward selection procedure consists of solving a sequence of least squares problems.

Specifically, the forward projection algorithm consists of the following steps:

- Step 1: Set $z_i = x_i, \quad z_i \in \mathbb{R}^k, \quad k = p.$
- Step 2: Find the 1-D projection $\beta_k : \mathbb{R}^k \mapsto \mathbb{R}$ such that

$$\beta_k = \underset{\beta:\mathbb{R}^k \mapsto \mathbb{R}}{argmin} \sum_{i=1}^{n} \big(y_i - \widehat{m}(\beta^T z_i)\big)^2. \tag{6.6}$$

- Step 3: Project data onto $\{\beta_k\}^\perp \subseteq R^{k-1}$

$$z_i \leftarrow \Gamma_{k-1}\, z_i, \quad \Gamma_{k-1} = I - \beta_k \beta_k^T.$$

- Step 4: Compute SVD of Γ_{k-1}:

$$\Gamma_{k-1} = U_{k-1} D_{k-1} U_{k-1}^T = \sum_{j=1}^{k-1} d_j u_j u_j^T, \quad u_j \perp \beta_k.$$

- Step 5: Let $k \leftarrow (k-1)$, go to Step 2 until $k = 1$.

The forward projection $P_q : \mathbb{R}^p \mapsto \mathbb{R}^q$ is

$$P_q = \Gamma_{p-1}\, \Gamma_{p-2} \cdots \Gamma_q. \tag{6.7}$$

6.3 Backward Projection Algorithm

The idea of the backward projection algorithm is the opposite of the forward algorithm. It takes a conservative approach by eliminating the 1-D direction that contains the least information of Y. This is equivalent to search for the best subspace of one less dimension. The implementation of the backward algorithm is almost identical to the forward algorithm except the formulation of the least squares problem. The backward algorithm consists of solving a sequence of least squares problems, each one formulated as

$$\beta_k = \underset{\beta:\mathbb{R}^k \mapsto \mathbb{R}}{argmin} \sum_{i=1}^{n} \big(y_i - \widehat{m}(\Gamma_{k-1} z_i)\big)^2, \tag{6.8}$$

such that

$$\Gamma_{k-1} = (I - \beta_k \beta_k^T).$$

(6.8) identifies the 1-D projection that contains the least information. Another interpretation is that (6.8) identifies the $\mathbb{R}^{p-1}$ subspace that has the minimum loss of information, measured by MSE. Similar to the forward algorithm, we project the data onto a subspace of dimension $p-1$ then solve another problem (6.8) in $\mathbb{R}^{p-1}$.

Naturally, (6.8) leads to a sequential algorithm for the linear dimension reduction mapping, the optimum subspace in $\mathbb{R}^q$ is

$$B_q = \Gamma_{p-1}\,\Gamma_{p-2}\cdots\Gamma_q. \tag{6.9}$$

Obviously backward algorithm is a conservative one which builds upon the notion that it is safer to eliminate the least important direction than to pick the most important direction.

6.4 Forward and Backward Algorithms for GMC

The structure of the forward and backward algorithms for a given GMC model is the same as described in Section 6.2. Every algorithm developed in this chapter is characterized by the least square problems it solves sequentially.

For a given GMC model GMC(K_0, K_1), the forward algorithm constructs the subspace projection by solving the problem:

$$\beta_k = \underset{\beta:\mathbb{R}^k\mapsto\mathbb{R}}{argmin} \sum_{i=1}^{n_t} I\{y_i \neq \widehat{Y}(\beta^T z_i; K_0, K_1)\}. \tag{6.10}$$

Similarly, the backward algorithm for the GMC model solves the problem:

$$\beta_k = \underset{\beta:\mathbb{R}^k\mapsto\mathbb{R}}{argmin} \sum_{i=1}^{n_t} I\{y_i \neq \widehat{Y}(\Gamma_{k-1} z_i; K_0, K_1)\}, \tag{6.11}$$

where $\Gamma_{k-1} = I - \beta\,\beta^T$.

The GMC model is build upon the ratio of two Gaussian mixture densities. We know that the Gaussian mixture is invariant under linear projection, therefore, the GMC of projected data can be evaluated by direct application of Theorem 6.1.1.

Computationally, each algorithm consists of a sequence of least squares problems that can be solved by Newton's method. As the dimension of the optimum subspace R^q decreases, the MSE for GMR and test error for GMC increase.

In practice, we first apply GMR to the data, select the best GMR model, then apply both the forward and backward algorithms to the given GMR model. Each algorithm generates a profile of MSE for each subspace nested in R^p. This MSE profile is the guideline for selecting the appropriate subspaces. Similar practice on the GMC procedure, given the best GMC model, a forward and backward algorithms will provide the profile of test error under subspace of various dimensions. We demonstrate the application of the forward and backward algorithms in Chapter 7.

Chapter 7

Applications

In this Chapter we discuss some applications of the GMR and GMC procedures to real and simulated data sets. The GMR and GMC procedures and the supporting routines IPRA and MST are implemented in MATLAB.

7.1 A Note on Random and Fixed Designs

One important point in applying GMR is that GMR is derived under the assumption that both the X and Y data are random samples. For fixed design data, the values of X are pre-specified by the experimenters, and they cannot be used to estimate the true underlying distribution of X. Therefore, it makes no sense to model the data based on the joint density function $f_{X,Y}$. However, we contend that although the true joint density of $f_{X,Y}$ is not estimable from the fixed design data, it does make sense to define the "designed distribution." That is,

$$f_X(x) = \sum_{i=1}^{n} n^{-1} \phi(x; x_i, h^2 I_p). \tag{7.1}$$

The interpretation is that for fixed design data, with equal probability, $1/n$, X can take a value of the knot x_i with a small fluctuation controlled by h.

Since the goal of regression analysis is to make inferences about the conditional distribution $Y|X$, we use the marginal part of X in the fixed design as a vehicle to connect every individual observation (X_i, Y_i). In other words, in the case of fixed design data, the marginal f_X refers to the pattern of the design, characterized by f_X in (7.1), not to the true underlying distribution of X.

7.2 Applications of GMR

7.2.1 Simulated 3-component Bivariate Gaussian Data

We first investigate a GMR application with an exploratory analysis of a simple bivariate data set. The data set consists of 150 observations simulated from a 3-component Gaussian mixture: $\{(x_i, y_i) : x_i \in \mathbb{R}, y_i \in \mathbb{R}, i = 1, 2, \ldots, 150\}$ where the simulated sample sizes are $(n_1, n_2, n_3) = (78, 12, 60)$. The sample means and variances are

$$\widehat{\mu}_1 = (-3.21, 3.44)^T, \quad \widehat{\mu}_2 = (7.43, 6.88)^T, \quad \widehat{\mu}_3 = (.02, -5.04)^T$$
$$\widehat{\Sigma}_1 = \begin{bmatrix} 3.53 & 0.07 \\ 0.07 & 8.79 \end{bmatrix}, \quad \widehat{\Sigma}_2 = \begin{bmatrix} 2.16 & 2.50 \\ 2.50 & 5.65 \end{bmatrix}, \quad \widehat{\Sigma}_3 = \begin{bmatrix} 8.58 & -0.16 \\ -0.16 & 2.83 \end{bmatrix}$$

Figure 7.1 displays the RMSE and PE profiles of the GMR models. Both curves indicate the correct model GMR(3). Figure 7.2 shows the true GMR(3) regression function (the red line), the global 5th-degree polynomial fit (the green line), and GMR(3)/MoM fit (the blue line).

We observe that in Figure 7.2 the GMR(3)/MoM fit is very close to the true regression curve. The 5th degree polynomial fit is able to capture the general pattern but it somewhat underfits the valley around $(3, -5)$ and overfits the peak around $(-5, 2)$. This example demonstrates that the GMR procedure is capable of capturing the correct regression curve when the true underlying density is a Gaussian mixture at least in low dimension and small K situation.

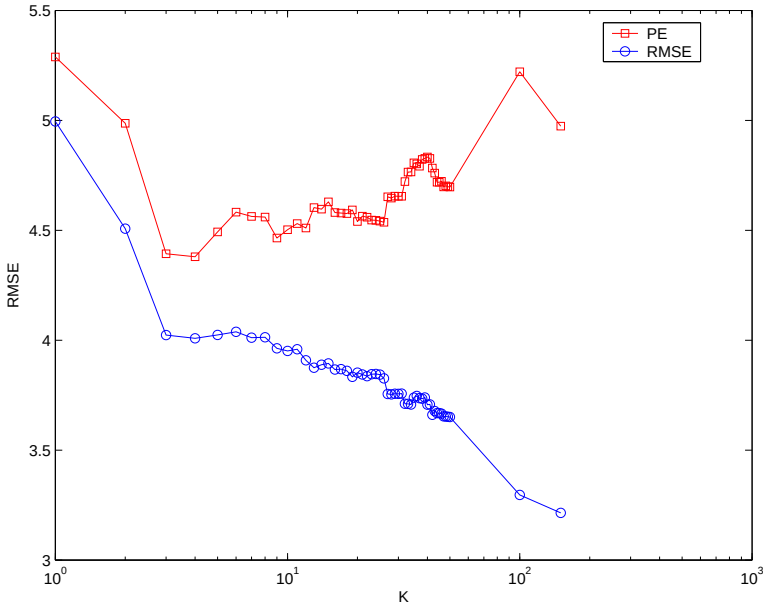

Figure 7.1: RMSE and PE of the GMR of the simulated data from a 3-component Gaussian mixture density ($n = 150$).

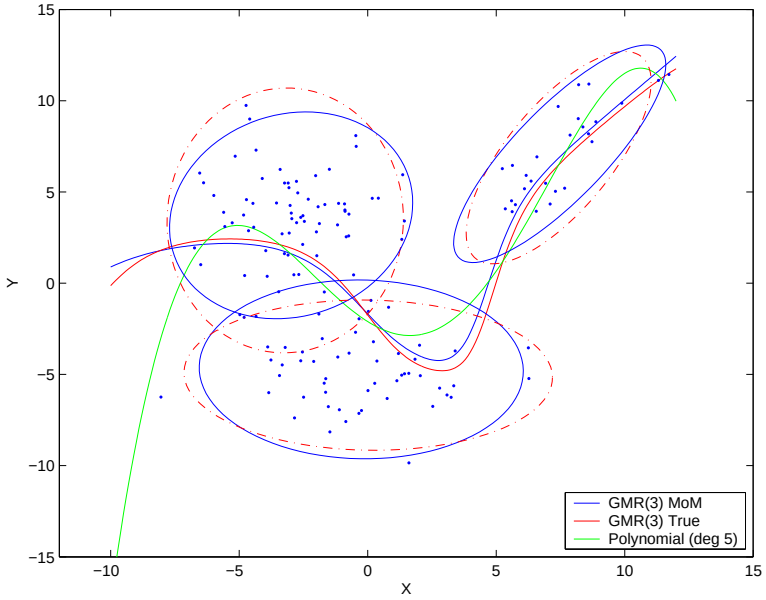

Figure 7.2: True GMR model, GMR(5)MoM, and 5-th degree polynomial regression of the simulated data from a 3-component Gaussian mixture density ($n = 150$).

7.2.2 Simulated GAM Data: Subspace Projections

In this study, we simulate data from a GAM model and throw in extra dimensions of random noise in order to investigate how well GMR picks up meaningful dimensions and to see if the forward and backward projections can identify the correct subspace and direction.

We simulate several data sets: $(X_i, Y_i)_{i=1}^n$, $X_i \in \mathbb{R}^p$, $Y_i \in \mathbb{R}$; sample sizes $n = 200, 300, 400$; dimensions $p = 5, 10$; $X^{(j)} \sim U(-4, 4)$; and $Z_1 = 1/3\beta_1^T X$; $Z_2 = 1/5\beta_2^T X$;

$$\begin{aligned} \beta_1 &= (1,0,2,0,0)^T/\sqrt{5} = (.4472, 0, .8944, 0, 0)^T, \\ \beta_2 &= (2,0,-1,2,0)^T/3 = (.6667, 0, -.3333, .6667, 0)^T, \\ Y &= \sin(5Z_1/\pi) + 3\cos(5Z_2/\pi) + \varepsilon, \end{aligned}$$

where $\varepsilon \sim N(0, .3^2)$. We vary both the sample size and the dimension of X. The true regression function is a function of variable 1, 3, and 4, the rest of the dimensions are random noise. The goal of this simulation study is not to find the best K for the

	$p=5$		$p=10$		
n	K	RMSE	K	RMSE	True sd
200	22	.256	20	.272	.273
300	22	.256	32	.267	.277
400	28	.237	34	.286	.283

Table 7.1: The best GMR models of the simulated data

Best 3-D			Best 2-D		Best 1-D
-.0016	.9161	.3865	.1241	.7208	-.4625
.0493	.1215	-.0814	-.0545	.1424	-.0226
-.9537	.1174	-.2754	.8996	.2070	-.8852
-.2919	-.3509	.8749	.4115	-.6371	-.0450
-.0525	-.0954	.0512	.0542	-.1073	.0055

Table 7.2: The backward projection for $(n, p) = (200, 5)$

GMR fit but to see if the GMR model can capture the true regression function and to see if our dimension reduction procedure can identify the correct subspace where the true $m(X)$ resides. Table 7.1 records the best GMR(K) fit for the simulated data of varying sample sizes and dimensions. The values of K are relatively high. The RMSE (square root of the mean squared error) are fairly close to the true standard deviation of the simulated data.

The forward projection result for $n = 200$, $p = 5$ data is:

$$\widehat{\beta}_F = \begin{bmatrix} .4317 & -.6640 & -.4191 & -.0780 \\ .0416 & .0185 & .2954 & .8555 \\ .8992 & .3598 & .1493 & -.0046 \\ .0577 & -.6552 & .5220 & .0954 \\ .0064 & .0068 & .6651 & -.5030 \end{bmatrix}$$

We first ran GMR on the simulated data and used the RMSE to selection the proper K. The circled points in Figure 7.3 are GMR(9), GMR(18), and GMR(28) for $p = 5$; GMR(9), GMR(25), and GMR(34) for $p = 10$. We applied forward and backward projection algorithms on GMR(28) and GMR(34) for $p = 5, 10$.

For the simulated data set of size 400, $X \in \mathbb{R}^{10}$, the overall IPRA/GMR procedure takes about 60 minutes of CPU time. The forward projection takes 21 minutes and the backward projection takes 32 minutes for GMR(9) on a Sun workstation SPARC 1000. As shown in Table 7.3, the run times of both forward and backward projections are roughly linear in K.

Compared to the true subspace projection, the forward projection does a good job picking up the true (β_1, β_2) in the first two columns. Backward projection, on the other hand, does not identify the exact projection until the dimension is correct. Nevertheless, both algorithms identify the right subspace, as shown in Figure 7.4. In

p	K	9	18	28
5	forward	4.0	5.8	10.4
	backward	4.3	12.2	15.4
p	K	9	25	34
10	forward	21.3	77.3	86.4
	backward	32.2	96.1	129.4

Table 7.3: Run time (CPU minutes) for projection procedures on different GMR(K) ($n = 400$)

this example, the optimal subspace is in $\mathbb{R}^3$. The MSE and RMSE are computed as

$$\begin{aligned} \text{MSE}(B) &= n^{-1} \sum_{i=1}^{n} \left(y_i - \widehat{m}(B^T x_i) \right)^2, \\ \text{RMSE}(B) &= \sqrt{\text{MSE}(B)} \end{aligned}$$

MSE is the standard goodness-of-fit measure. To make it comparable to the scale of the data (X, Y), we report root-mean squared error (RMSE) in all our figures and tables.

We expect the first two dimensions to pick up most of the variation of the data. Hence the big drop off indicates that all the informative dimensions are included. In Figure 7.4 we observe that when $p = 5$, the backward projection picks up the correct dimension more cleanly, while the forward projection seems to favor one dimension higher. When there are more noise dimensions ($p = 10$), the forward projection has a sharper drop at dimension 2. Also in this case, the forward projection is able to capture the correction projections one by one. Computationally, the forward projection is more straightforward and much less expensive than the backward projection.

Figure 7.5 shows the RMSE of the original GMR family on $\mathbb{R}^5$ and $\mathbb{R}^{10}$ and the RMSE on the projected subspaces identified by the forward projection. One interesting feature in Figure 7.5 is that the GMR models on the subspace $\mathbb{R}^2$ requires much fewer mixture components to capture the regression curve. The direct consequence is that the best GMR models on the subspace have much smaller K.

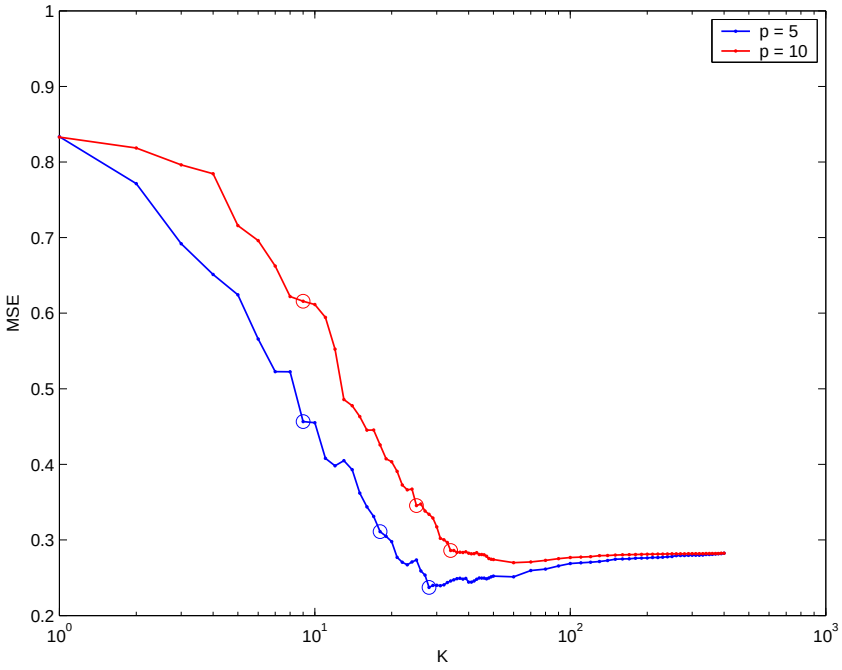

Figure 7.3: RMSE of GMR models for simulated data (n = 400). The red line is the RMSE curve of GMR models for the $\mathbb{R}^5$ data. The blue line is for the $\mathbb{R}^{10}$ data. The circles on the red line are GMR(9,18,28). The cricles on the blue line are GMR(9,25,34).

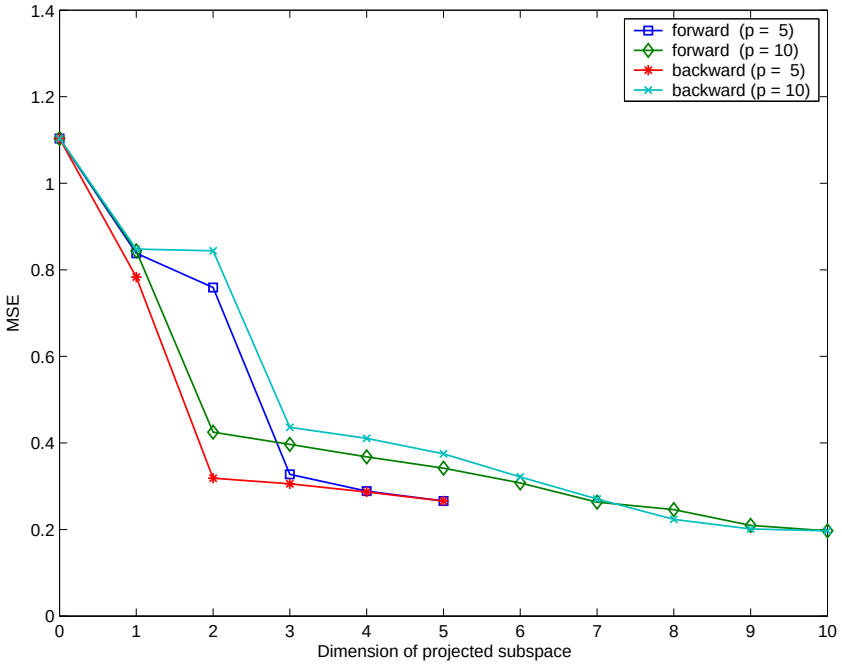

Figure 7.4: RMSE of GMR(28) and GMR(34) on projected subspaces (n = 400)

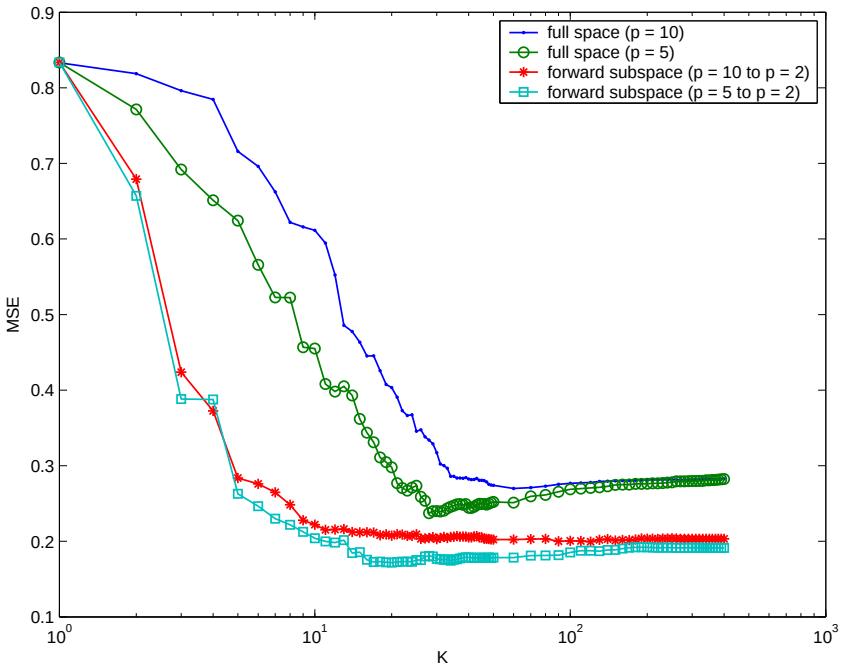

Figure 7.5: RMSE of GMR models on projected space (n = 400)

There are several conclusions we draw from this simulation study:

- Different starting GMR(K) models result in very similar Forward subspace projections. This is not as clear with backward projection. The reason is that the backward procedure is focusing on eliminating frivolous dimensions; hence, it usually results in a different basis for the same remaining subspace orthogonal to the 1-D noise dimension. However, backward and forward projections should be complementary. It is conceivable that when the true subspace dimension is greater than the noise dimension, the backward projection should have some advantage over the forward projection algorithm. The same may hold if the feature space has many highly correlated variables.

- Computationally, it may be wise to run forward procedure on a simpler GMR model, that is, GMR(K) with smaller K, to identify the subspace, then re-run IPRA/GMR directly in the projected subspace. It is not clear if this will work for backward procedure. We tested this idea and indeed the forward projection is able to pick up the correct 2-D projections with a relatively smaller GMR(K) model. We executed this strategy for $n = 400$. The results are summarized

in Appendix D. As shown in Table 7.3, the saving in computing time is more apparent, especially when most of the dimensions are noise ($p = 10$).

7.2.3 Motorcycle Data

The famous Motorcycle data was first compiled by Schmidt, Mattern, and Schüler (1981). Silverman (1985) used this data set to demonstrate scatterplot smoothing with splines. Since then, the Motorcycle data set has been one of the standard examples for any nonparametric smoother. It has 133 observations of X: milliseconds(ms) and the impact Y measured as the acceleration of a human head X (ms) after the impact in a simulated collision experiment. The first important feature of this data set is that it is not a random design. That is, the covariate X is not a random variable. The second feature is that there are 94 distinct values of X among the 133 observations.

We demonstrate the GMR analysis of the Motorcycle data and a detailed discussion of two practical issues: bandwidth selection and GMR model selection.

7.2.3.1 Bandwidth and Model Selections

As discussed in Section 4.4, the practical selection of the bandwidth for GMR is $h \approx min_j|x_j, x_{j-1}|$. The results in Figure 7.6 show that h affects GMR(n) the most. The mse for GMR(n) increases as h increases. This implies a smoother GMR(n). As the IPRA starts to merge similar components and reduce the number of mixtures, the effect of h diminishes. We believe that since GMR(1) should be underfit, the most important point concerning h is to have it sufficiently small so that GMR(n) overfits. This will ensure that the GMR family contains the properly fit models in between the two extreme members GMR(1) and GMR(n). We use $h = .56$ for the Motorcycle data. As Scott's rule give $1/20\, n^{-1/6}\sqrt{\widehat{\sigma}_x\, \widehat{\sigma}_y} = .56.$, while the minimum distance between two distinct data points x_j and x_{j-1} is 0.20.

Figure 7.7 shows how the bandwidth h affects the initial GMR fits. It is apparent that both $h = 1$ and $h = 2$ yield over-smoothed initial GMR fits.

Figure 7.8 indicates two interesting features of the RMSE of GMR family. The first feature is that the bandwidth effect is most prominent in the initial GMR fit. As IPRA starts to merge mixture components, the bandwidth effect diminishes. As expected, the RMSE of GMR(1) is identical for all values of h. The second feature is that h does not alter the pattern of RMSE curves. All of them remain almost constant before the RMSE shoots up after GMR(6). Notice that the three RMSE curves corresponding to $h = 0.1$, 0.2, and 1.0 all merge together soon after a few merges. Obviously we can say that any choice of $h < 1.0$ will have very similar RMSE curves to that of $h = 1.0$. This suggests that $h = 1.0$ is an upper bound for "good enough" bandwidth. Even for $h = 2.0$, which is clearly too large a bandwidth, its RMSE curve still shows a similar pattern, hence, also suggests the same best model GMR(6). The lesson here is that if the bandwidth is not too large, the GMR

Bandwidth (h)	RMSE (GMR(133))	RMSE (GMR(6))
0.10	13.70	22.34
0.20	16.38	22.39
0.50	19.62	22.31
1.00	21.37	23.02
1.50	22.96	25.13
2.00	24.56	25.78
2.20	25.22	26.15
2.50	26.19	26.91
3.00	27.78	28.27

Table 7.4: Bandwidth vs RMSE of GMR models on the Motorcycle data

procedure will provide approximately the same profile RMSE curve, and therefore, will not adversely affect the model selection. In this example, all values of h lead to GMR(6) as the best model. Both the RMSE and the leave-one-out cv in Figure 7.9 indicate that GMR(6) is the best model for the Motorcycle data.

The GMR family is indexed by K, as shown in Figure 7.10. K can be viewed as a tuning parameter for the smoothness or flexibility of the GMR fits. The key point is to choose a h smaller enough so that it allows K to take over the role of calibration. Of course, it does not mean that we should pick $h = 0$. For data of finite sample size, $h = 0$ leads to a Dirac spike and the singularity is bad news to both theory and practice.

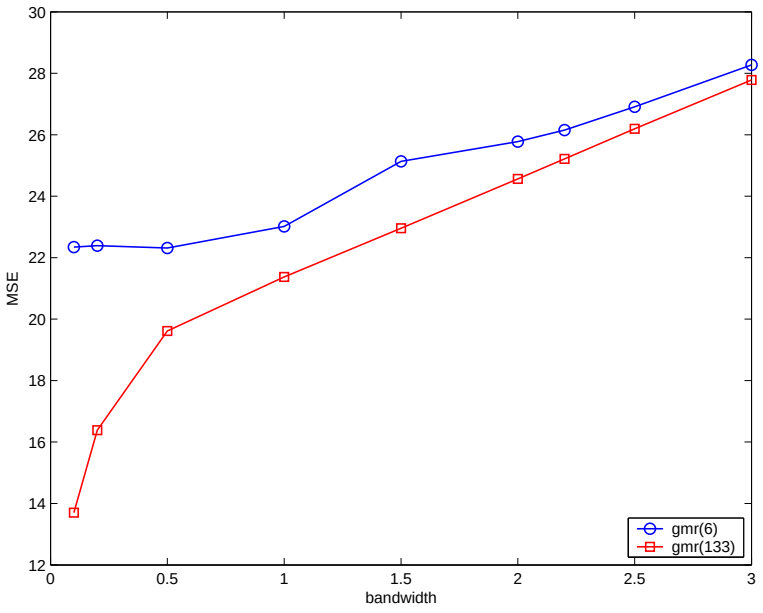

Figure 7.6: Bandwidths vs RMSE of GMR models on the Motorcycle data

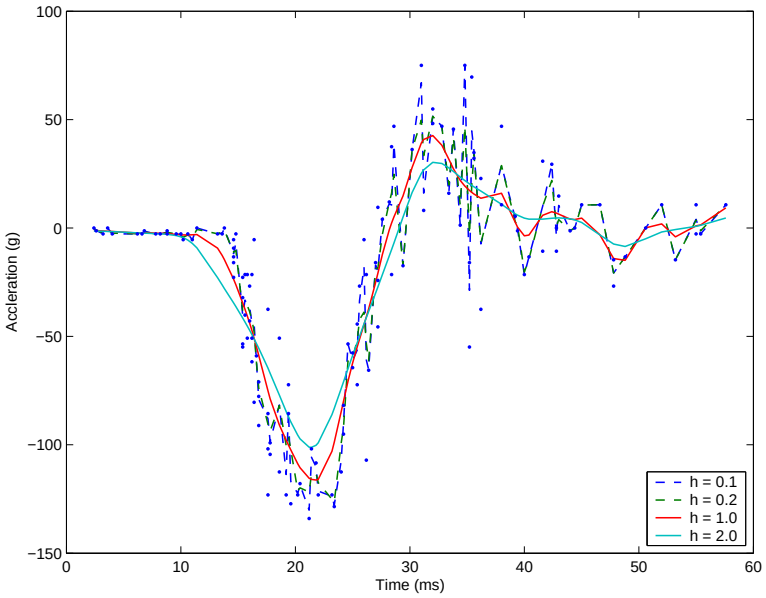

Figure 7.7: GMR(133) of various bandwidths on the Motorcycle Data

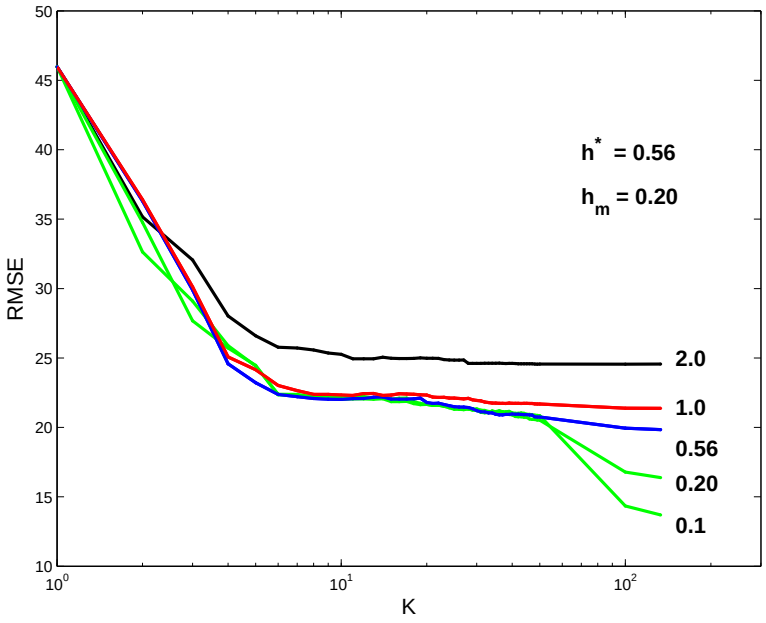

Figure 7.8: The bandwidth effect on RMSE of the Motorcycle data

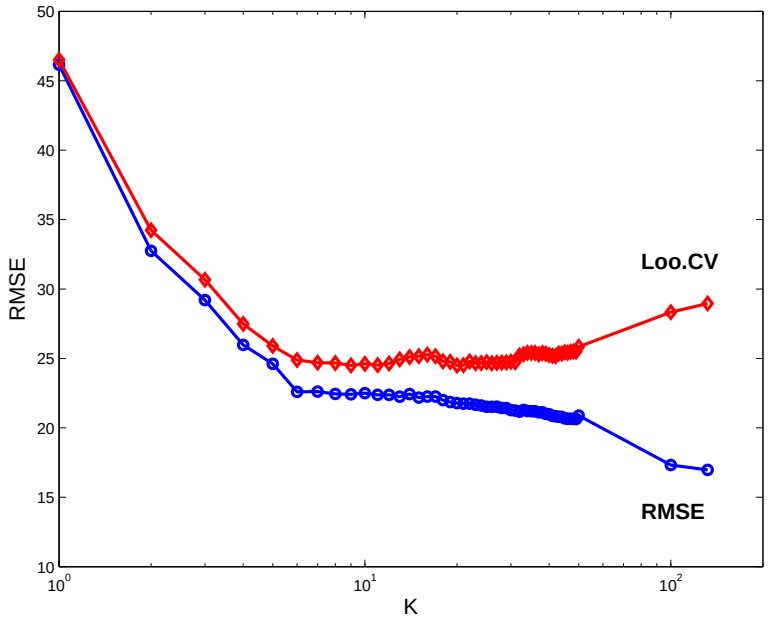

Figure 7.9: The leave-one-out cv of GMR on the Motorcycle data

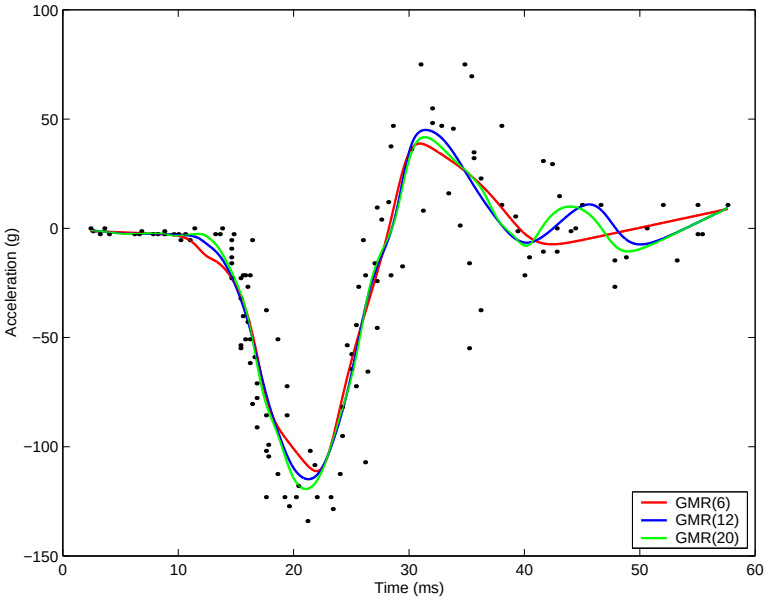

Figure 7.10: The GMR fits of the Motorcycle data: $K = 6, 12, 20$

7.2.3.2 Bootstrap Analysis of GMR

In this section we use the bootstrap method to approximate the variation of the GMR procedure. We focus on the GMR(6) model with $h = 0.2$. Unlike the standard regression model, the residuals of GMR model are not homoscedastic. Therefore it is invalid to bootstrap from the raw residuals. A viable choice is to bootstrap the raw data. We draw 200 bootstrap data sets of the same size as the original data ($n = 133$). Starting with the initial kernel bandwidth $h = .2$, we obtained 200 GMR sequences. We plot the 200 GMR curves in Figure 7.11. The solid red curve is the original $\widehat{m}(x; 6)$ fit. The solid blue curve is the pointwise bootstrap mean and the dashed blue lines the pointwise standard error times 1.96. They indicate the bootstrap estimates of the expected $m(x)$ and the confidence band of the GMR estimator.

Figure 7.12 presents a contrast of two different types of variations of GMR. One is that the GMR $\widehat{m}(x)$ and $\sqrt{\widehat{v}(x)}$ are the summary statistics of the density $\widehat{f}_{Y|X=x}$. This implies that $\sqrt{\widehat{v}(x)}$ measures the predictive band of GMR fit. The bootstrap distribution measures the variation of the GMR algorithm itself, that is, the mean and variance of the bootstrap distribution approximate the expected value and confidence band of the estimator $\widehat{m}(x; 6)$. As expected, in Figure 7.12, we observe a tighter confidence band than a prediction band. The difference between $\widehat{m}(x; 6)$ and the mean fit is the bootstrap estimate of the bias of GMR(6), which indicates that the bias of GMR(6) is mild. And as K decreases, the variance goes down and the bias goes up.

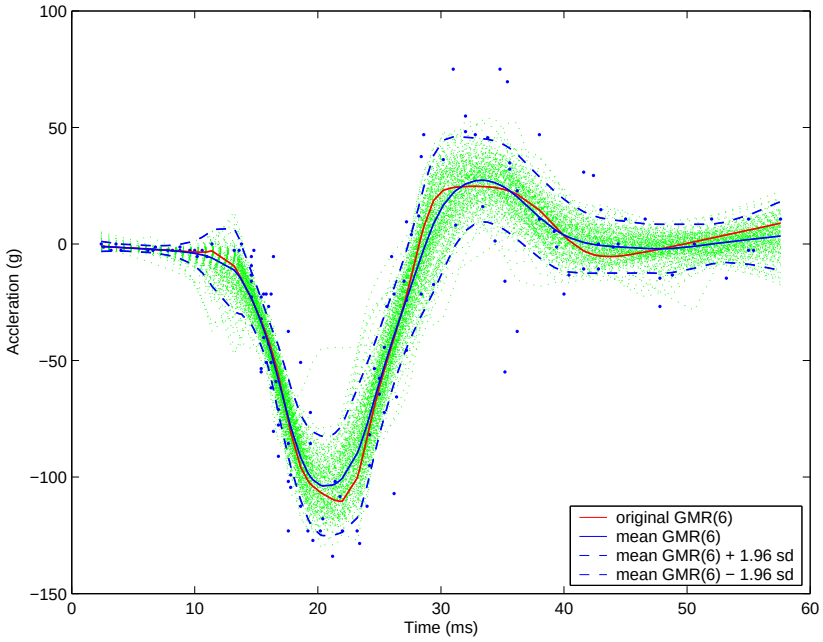

Figure 7.11: The bootstrap sample curves of GMR(6)

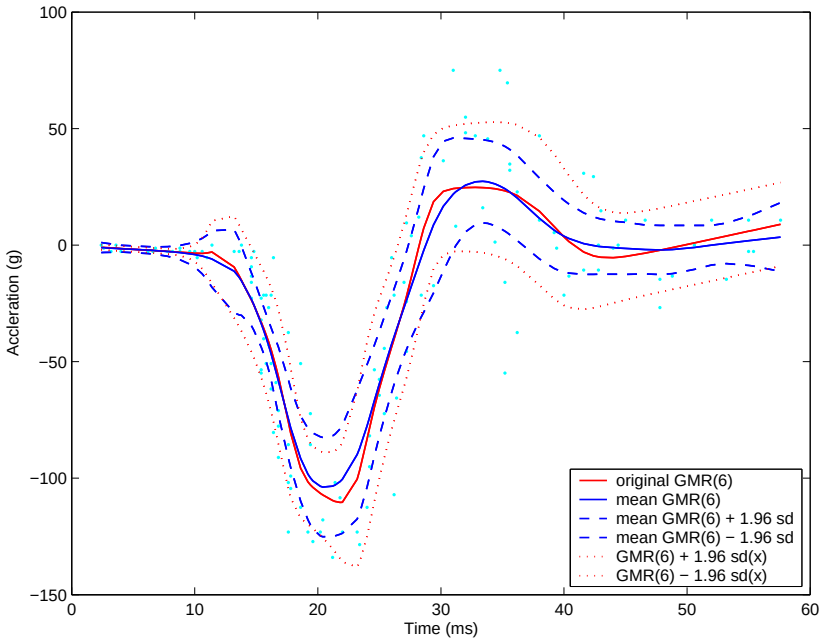

Figure 7.12: The confidence band of GMR(6)

7.2.3.3 Model parameters: MoM vs MLE

Another practical issue of GMR is the estimation of the model parameters. As described in Chapter 3, the parameter values are estimated using the method of moments. We contend that for practical use of GMR for EDA purposes, it is not necessary or desirable to obtain the MLE, especially for high dimensional data. Part of the reason is that the likelihood function in high dimensional space is flat, so that there are many local maximums, especially when $K > K^*$. For univariate data such as the Motorcycle data, it is informative to compare the MoM values and the MLE. For this purpose, the MoM fit serves as a natural choice for the initial values. We applied the EM algorithm using the MoM fit as the initial guess and the results are plotted in Figure 7.14. One interesting point in Figure 7.14 is that the MLE fit and MoM fit of the cluster means $\widehat{\mu}_k$ are almost identical, but the MLE cluster variance is tighter than the MoM estimate. This gives the MLE fit sharper edges than the MoM fit. In this case, we contend that the MLE fit is less smooth and consequently overfits the corners around $(15, 0)$ and $(22, -130)$. The bump around $(18, -90)$ is the result of bumpy transition between two neighboring mixture components. Overall, however, the MLE and MoM fits are very similar. It suggests that for EDA purpose, the MoM estimate is usually sufficient.

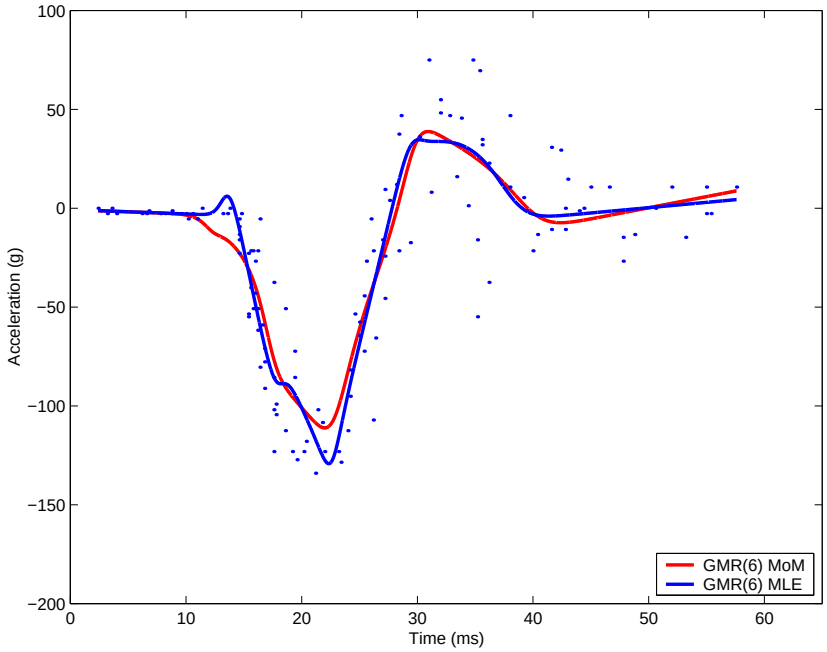

Figure 7.13: The GMR(6)/MoM and GMR(6)/MLE fits of the Motorcycle data

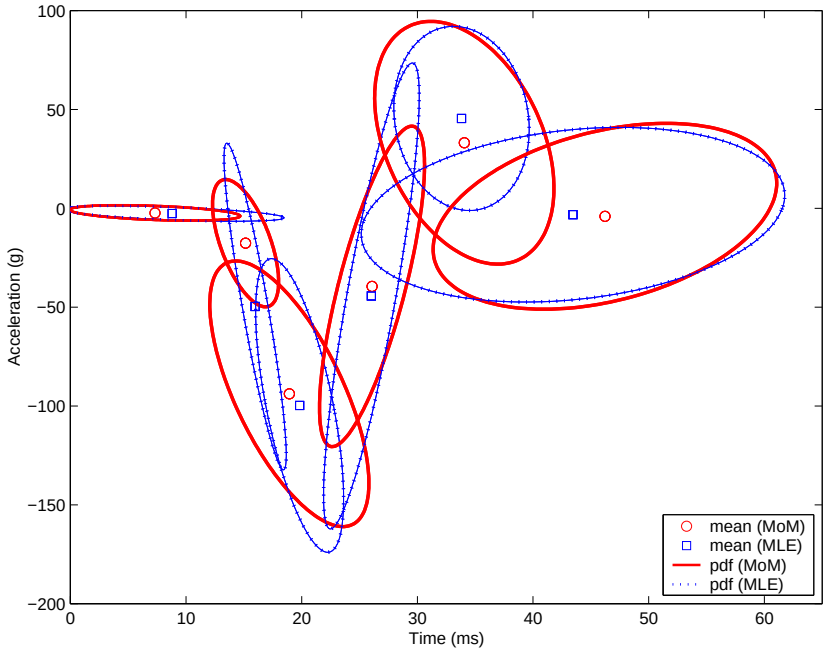

Figure 7.14: The contour plots of the underlying pdf of GMR(6)/MoM and GMR(6)/MLE

7.3 Applications of GMC

7.3.1 Handwritten Zip Code Data

The handwritten Zip Code data were collected by the U.S. Postal Service by automatic scanning from envelope. Each handwritten digit is stored as a 16×16 8-bit grayscale image. LeCun (1989) first analyzed the Zip code data using neural networks. The 256 pixel values were the inputs to the neural network classifier. A good introduction of the neural network analysis on the Zip code data can be found in Hastie et al. (2001, p. 362). We downloaded the standard version of the data from the National Institute of Standards and Technology (NIST) website (http://www.nist.gov/srd/).

We apply the GMC procedure on three binary classification problems: digit 0 vs 3, digit 0 vs 6, and digit 1 vs 7. First, we fit GMC models on the training set, picking the best model based on the test error. Then we apply forward and backward projection to the best model to identify optimal subspaces for classification. Finally we update the model parameters to MLE and compare the model performance of MLE and MoM.

7.3.1.1 Digit 0 vs 3

In this section we apply GMC to classify digit 0 versus digit 3 (encoded as $Y = 0$ and $Y = 1$). The training set has size 1852 (1194 0's and 658 3's). The test set is of size 525 with 359 0's and 166 3's. We first apply SVD on the training set to obtain the principal components. We chose the span of the first 10 principal components as the derived feature space for the GMC application. As derived in Chapter 5, GMC classifier is

$$\widehat{r}_{10}(z; K_0, K_1) = \frac{\widehat{f}_{X|Y=1}(x)\pi_1}{\widehat{f}_{X|Y=0}(x)\pi_0},$$

where

$$\begin{aligned} \widehat{f}_{X|Y=1}(x) &= \sum_{j=1}^{K_1} \pi_{1j}\, \phi_{1j}(x), \\ \widehat{f}_{X|Y=0}(x) &= \sum_{j=1}^{K_0} \pi_{0j}\, \phi_{0j}(x). \end{aligned}$$

From the training set, we use the simple priors for Y. That is, $\pi_0 = 1194/1852 = .645$ and $\pi_1 = 658/1852 = .355$. The class densities $\widehat{f}_{X|Y=0}$ and $\widehat{f}_{X|Y=1}$ are Gaussian mixtures of components K_0 and K_1. (K_0, K_1) is the key model parameter for GMC procedure. We use the misclassification rate as the first order criterion. Since the misclassification rate is the ratio of two counts, most likely there will be ties among

several models, so we use the log-likelihood as the tie-breaker:

$$\log L(K_0, K_1; X, Y) = \sum_{i=1}^{n} y_i \log \widehat{f}_{X|Y=1}(x_i) + (1 - y_i) \log \widehat{f}_{X|Y=0}(x_i). \tag{7.2}$$

Figure 7.15 gives us a general landscape of the GMC model space by plotting test error of GMC(K_0, K_1) as a function of (K_0, K_1). The flat surface implies that there are many GMC models perform equally well. Figure 7.16 shows the log likelihood surface of the GMC models. One interesting feature of log-likelihood shown in Figure 7.16 is that the log-likelihood is not a good criterion for selecting the best model, but it is good at eliminating bad models. Obviously log-likelihood suggests the best model has small K_0 and K_1. But around the maximum area is so flat that it is not possible to pick the best model using log-likelihood. Another issue is that the numerical values of the log-likelihood function is around 10^{-35} range. It is not wise to read too much into it.

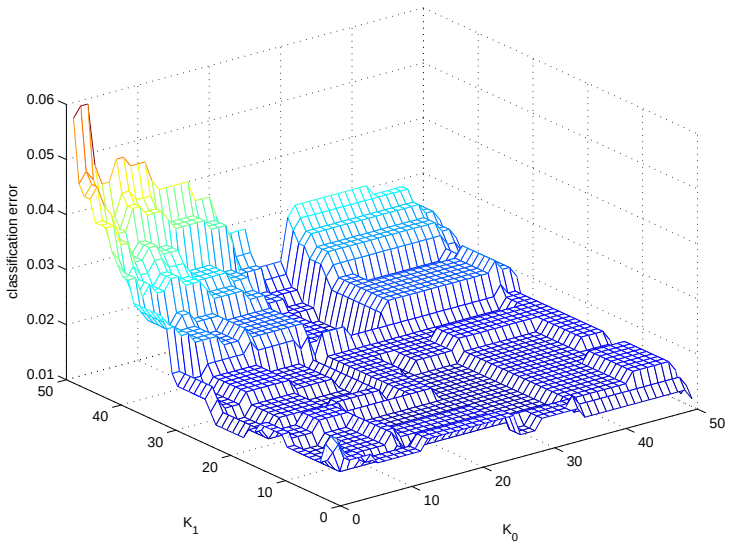

Figure 7.15: Test error of GMC for the Zip code data (0 vs 3)

Model		Misclassification Rate		Log-likelihood	
K_0	K_1	train	test	train	test
1	5	.0032	.0152	-50.1808	-35.2938
1	9	.0032	.0152	-44.7897	-36.2305
13	17	.0027	.0114	-36.6950	-62.3365
21	21	.0022	.0114	-39.0221	-84.5934
21	31	.0016	.0114	-27.0304	-111.4212
50	1	.0011	.0114	-2.1292	-135.7826
50	3	.0005	.0114	-2.8124	-129.8174

Table 7.5: Top GMC classifiers for the Zip code data (digit 0 vs 3)

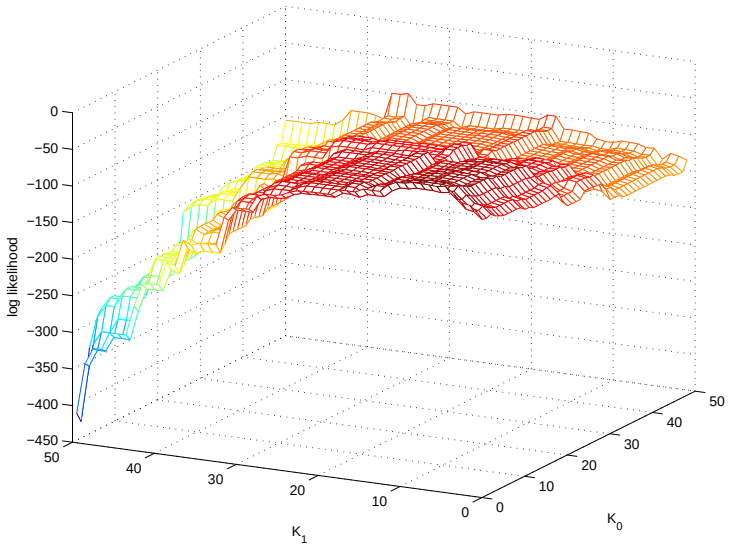

Figure 7.16: Log-likelihood of GMC for the Zip code data (0 vs 3).

Table 7.5 records top candidates of the best GMC model. There are 3 models worth further consideration (Table 7.5): GMC(1, 5) with the maximum log-likelihood value on the test data; GMC(1, 9) with the maximum "total likelihood" (the sum of log-likelihood values on training set and test set); and GMC(13, 17) with the smallest degrees of freedom ($K_0 + K_1$) among the models of minimum test error (.0114).

We use the IPRA fitted values as the initial values for EM algorithm and fit the MLE for these top three models. The results are in Table 7.6. Notice that the log-likelihood values of MoM and MLE for GMC(1,5) and GMC(1,9) are very similar. The highest discrepancy among the three models is in GMC(13,17). The reason is that the MoM and MLE are identical for the single Gaussian density, which is the case in $K_0 = 1$ for GMC(1,5) and GMC(1,9). For model selection purpose, MoM

Model		Log-likelihood		Misclassification Rate			
		Train	Test	Train		Test	
GMC(1, 5)	MoM	-50.1808	-35.2938	.0032	(6/1852)	.0152	(8/525)
	MLE	-52.7466	-24.4877	.0038	(7/1852)	.0152	(8/525)
MDA(1, 5)		-73.6808	-50.0312	.0097	(18/1582)	.0305	(16/525)
GMC(1, 9)	MoM	-44.7897	-36.2305	.0032	(6/1852)	.0152	(8/525)
	MLE	-41.7348	-34.1453	.0027	(5/1852)	.0190	(10/525)
MDA(1, 9)		-71.5722	-53.6600	.0097	(5/1852)	.0267	(14/525)
GMC(13, 17)	MoM	-36.6950	-62.3365	.0027	(5/1852)	.0114	(6/525)
	MLE	-8.6762	-89.0349	.0005	(1/1852)	.0210	(11/525)
MDA(13, 17)		-61.4147	-45.7965	.0038	(7/1852)	.0114	(6/525)

Table 7.6: Top GMC/MLE and MoM for the Zip code data (digit 0 vs 3)

favors GMC(13,17). MLE favors GMC(1,5). In this example, GMC(1,5) has a slight edge over GMC(13,17) for it has a much smaller degrees of freedom.

Table 7.6 also records the corresponding results from MDA. (We use the R-package mda implemented by Hastie and Tibshirani.) We find that GMC/MoM and GMC/MLE both perform better than MDA except in GMC(13,17); where MDA pulls a draw.

It is interesting to observe that GMC(50,1) and GMC(50,3) both have much better training set log-likelihood values over all other models, yet their test set log-likelihood values are the worst. This is a strong indication of overfitting that the misclassification rate fails to capture.

7.3.1.2 Dimension Reduction for GMC

Finally we apply forward and backward projection algorithms on GMC(13,17) and the results in Figure 7.17 show that the forward projection suggest a subspace of dimension 5.

For the purpose of visualization, Figure 7.18 and Figure 7.20 show the best 2-D feature spaces identified by forward and backward projections. Both forward and backward 2-D feature spaces are good. The most interesting contrast is that the first dimension in forward projected space already does a great job at classifying the two digits (0 vs 3). The backward projected space works well as a whole, but is marginally not as good as the first dimension of the forward-projected space. This is the direct consequence of the greedy nature in our forward projection algorithm. The backward projection, on the other hand, may has a better chance to grasp the best 2-D subspace as a whole.

It is certainly an interesting topic for further research to develop a data-driven strategy which combines the forward and backward algorithms.

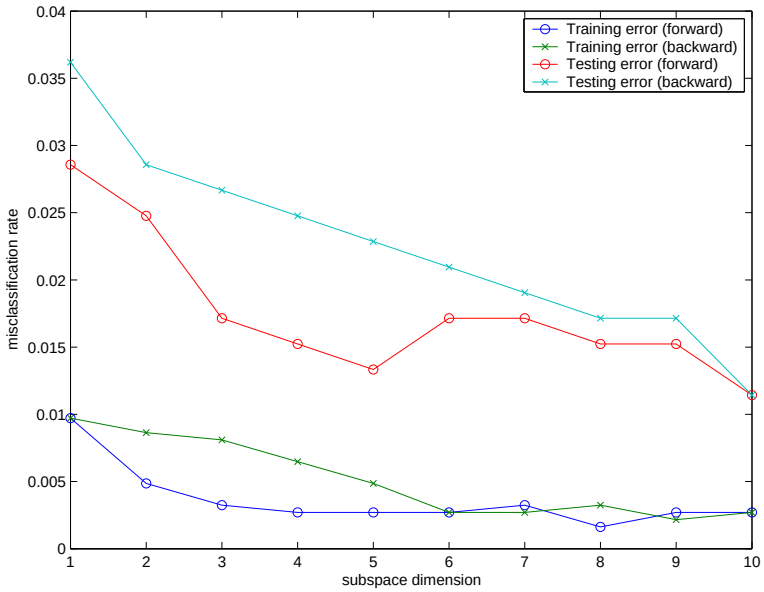

Figure 7.17: The error rates in subspaces of GMC(13, 17) for the Zip code data (digit 0 vs 3)

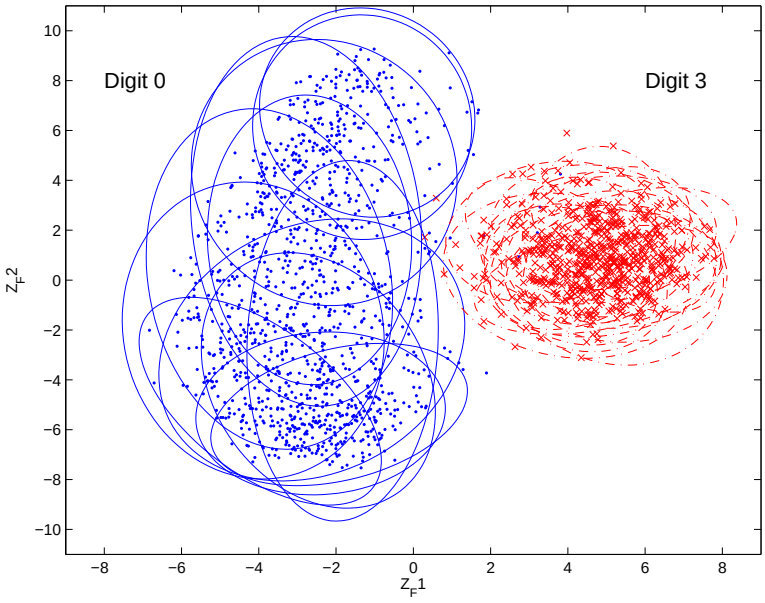

Figure 7.18: The contour plot of GMC(13, 17) on the forward 2-D space with training data (zip code data (digit 0 vs 3)

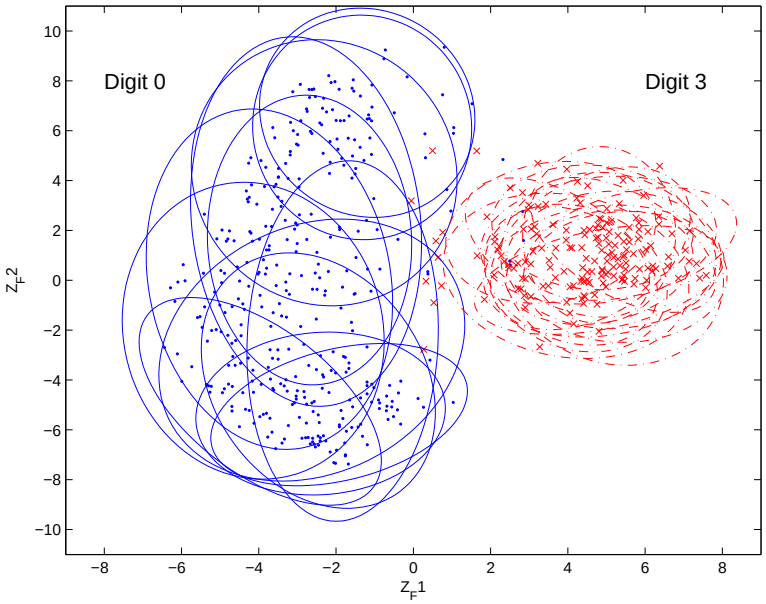

Figure 7.19: The contour plot of GMC(13, 17) on the forward 2-D space with test data (zip code data (digit 0 vs 3))

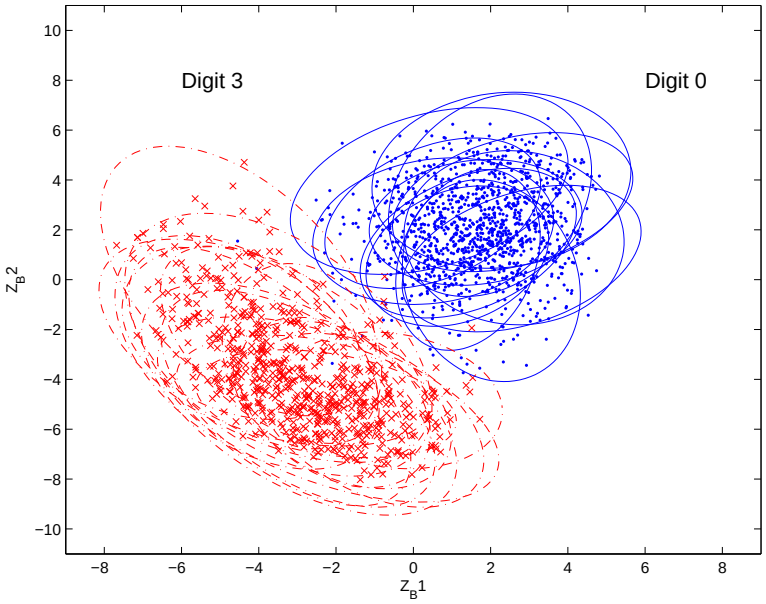

Figure 7.20: The contour plot of GMC(13, 17) on the backward 2-D space with training data (zip code data with digit 0 vs 3)

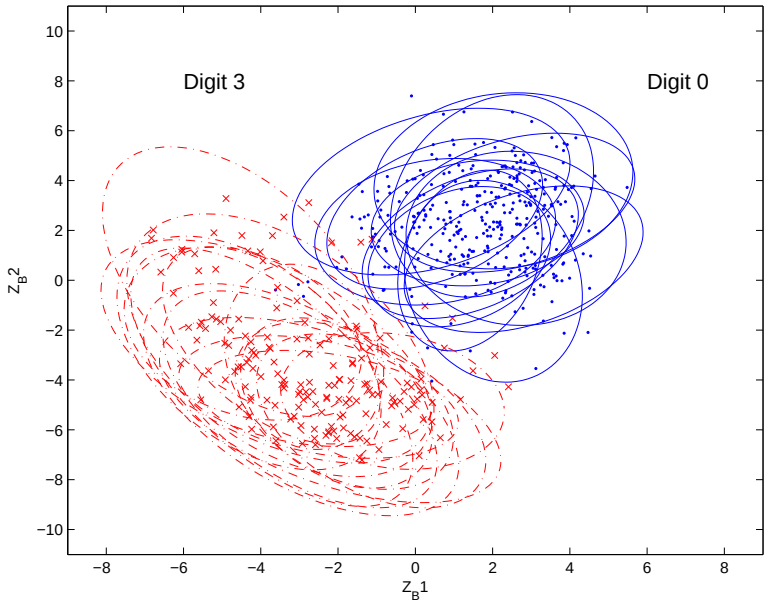

Figure 7.21: The contour plot of GMC(13, 17) on the backward 2-D space with test data (zip code data with digit 0 vs 3)

7.3.1.3 Digit 0 vs 6

In this section we analyze Zip code data digit 0 vs 6. The training set consist of 1585 observations (1194 0's, 664 6's). The test set consists of 529 observations (359 0's, 170 6's).

Similar to the previous analysis on digit 0 vs 3, we use the first 10 principal components of the data for the GMC application. Table 7.8 records top GMC models. One unusual feature is that some top GMC models have smaller test error than training error. The reason is the anomaly of the data. We will observe it later in Figure 7.27 and Figure 7.27.

Among the top models in Table 7.8, a clear winner is GMC(3, 7), which has the smallest degrees of freedom $(K_0 + K_1)$ and the largest log-likelihood value. We will base further analysis on GMC(3,7).

The comparison of forward and backward projections in this example is intriguing. Figure 7.24 (Figure 7.25 for a closer look) tells an interesting story. Notice that the 1-D and 2-D subspaces identified by backward projection both give high test errors relative to the 1-D and 2-D forward projected subspaces. However, from 3-D and beyond, backward project projection excel over forward projection. The test errors of backward projection in Figure 7.24 show that backward projection does identify good 1-D projections, rather it identifies good subspace as a whole. Individual 1-D projections from backward projection are usually not as good as the ones identified

	Log-likelihood		Misclassification Rate			
Model	Train	Test	Train		Test	
GMC(3,7) MoM	-45.6930	-32.1166	.0065	(12/1858)	.0057	(3/529)
GMC(3,7) MLE	-23.3606	-31.7216	.0038	(7/1858)	.0095	(5/529)
MDA(3,7)	-96.2153	-51.6038	.0151	(28/1858)	.0208	(11/529)

Table 7.7: GMC(3,7) vs MDA(3,7) for the Zip code data (digit 0 vs 6)

Model		Misclassification Rate		Log-likelihood	
K_0	K_1	Train	Test	Train	Test
3	7	.0065	.0057	-45.6930	-32.1166
3	8	.0070	.0057	-46.9049	-32.1195
4	7	.0059	.0057	-41.5073	-32.1393
4	8	.0065	.0057	-42.7198	-32.1417
4	9	.0054	.0057	-38.0118	-32.6595
5	7	.0054	.0057	-38.5791	-32.5150
5	8	.0059	.0057	-39.7881	-32.5164
5	9	.0048	.0057	-34.6373	-32.8922
6	7	.0048	.0057	-39.1865	-32.6413
6	8	.0054	.0057	-40.3974	-32.6427
6	9	.0043	.0057	-34.7917	-33.2824

Table 7.8: Top GMC classifiers for the Zip code data (digit 0 vs 6)

by forward algorithm. In practice, forward and backward are complementary to each other.

Finally, using the IPRA/MoM estimates of GMC(3,7) as the initial values for EM algorithm, we obtain the MLE fit of GMC(3,7). Table 7.7 shows an interesting result: the MLE GMC(3,7) has a better training performance over the MoM GMC(3,7). In fact, the MLE training error is almost half the training error of MoM fit. However, the MLE GMC's test error is worse. The lesson to learn here is that the MLE do fit the training data better because the individual covariance matrix is tighter. But the consequence for this tighter fit on the training set is a worse test error.

The puzzling result in Table 7.7 is that the MDA(3,7) performs much poorer than both GMC(3,7)/MLE and GMC(3,7)/MoM in both training and test errors. One possible explanation, hinted by the much lower log-likelihood values, is that MDA converge to a local maximum instead of the MLE.

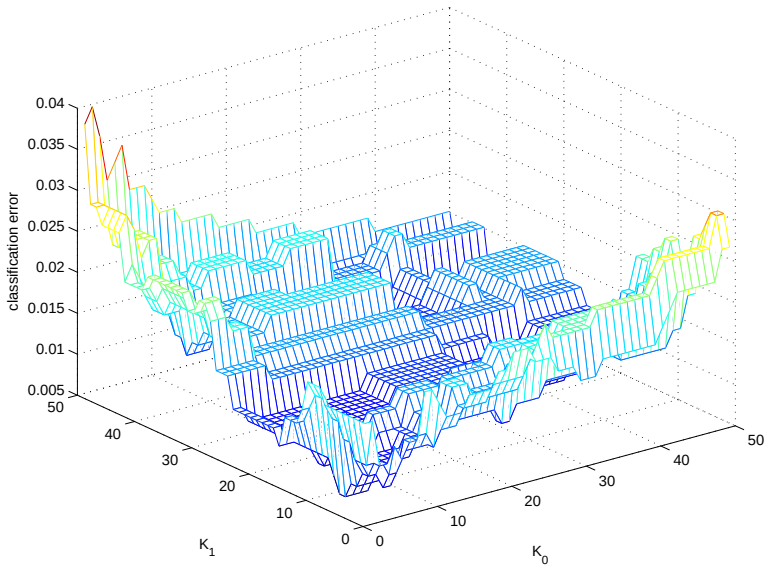

Figure 7.22: Test error of GMC for the Zip code data (digit 0 vs 6)

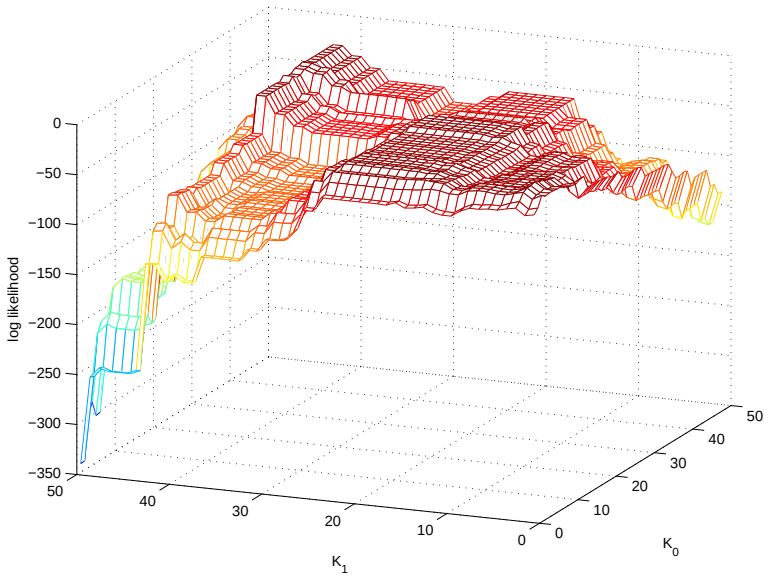

Figure 7.23: Log-likelihood of GMC for the Zip code data (digits 0 vs 6)

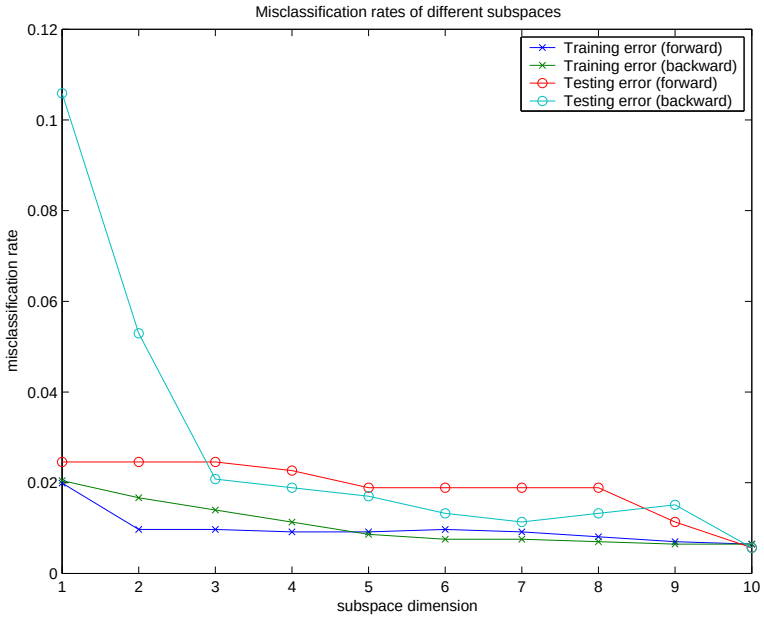

Figure 7.24: The error rates in subspaces of GMC(3, 7) for the Zip code data (digit 0 vs 6)

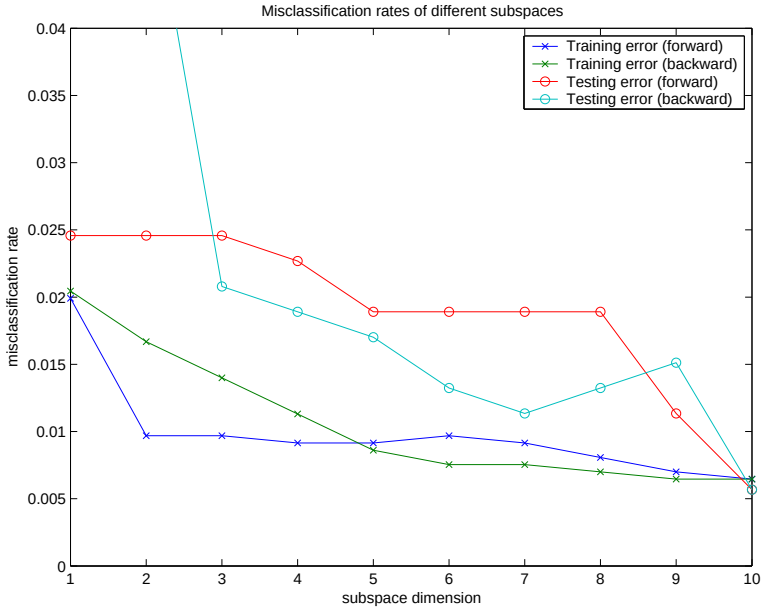

Figure 7.25: The error rates in subspaces of GMC(3, 7) for the Zip code data (digit 0 vs 6): a closer look

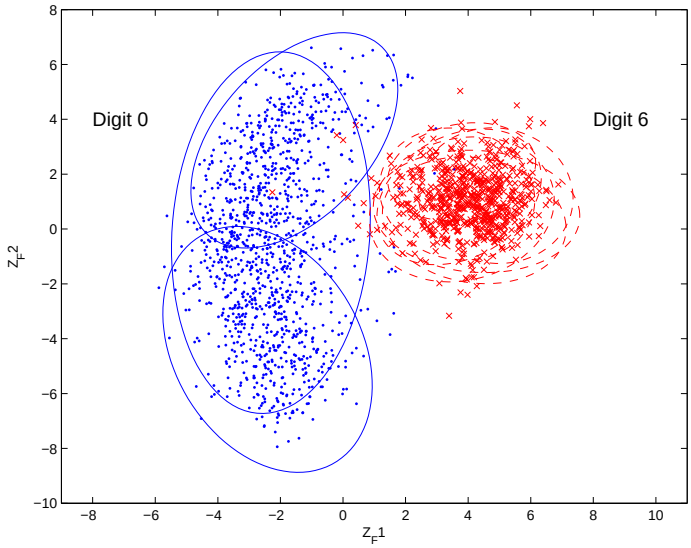

Figure 7.26: GMC(3, 7) contours on the forward 2-D space with training data (zip code data, digit 0 vs 6)

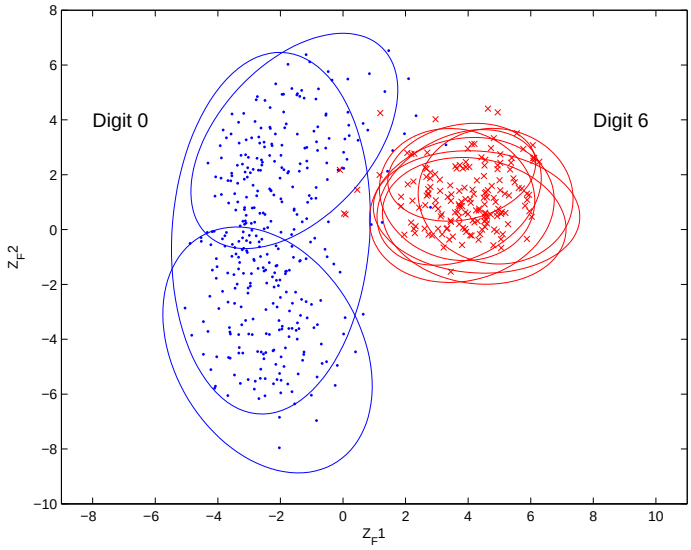

Figure 7.27: GMC(3, 7) contour on the forward 2-D space with test data (zip code data, digit 0 vs 6)

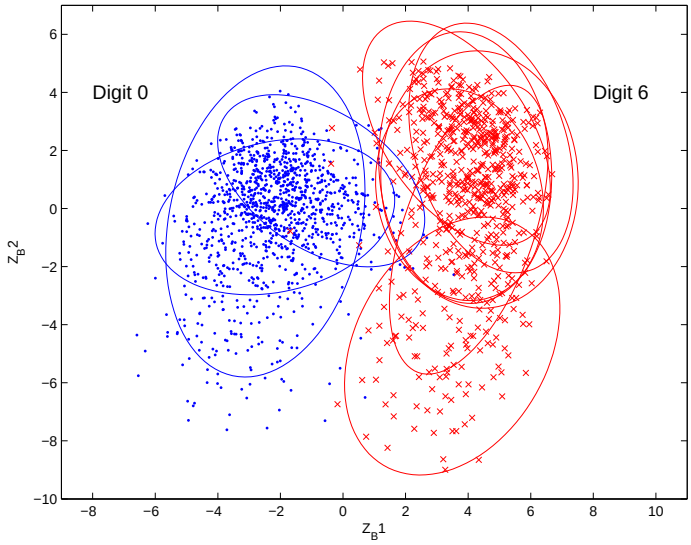

Figure 7.28: GMC(3, 7) contours on the backward 2-D space with training data (zip code data, digit 0 vs 6)

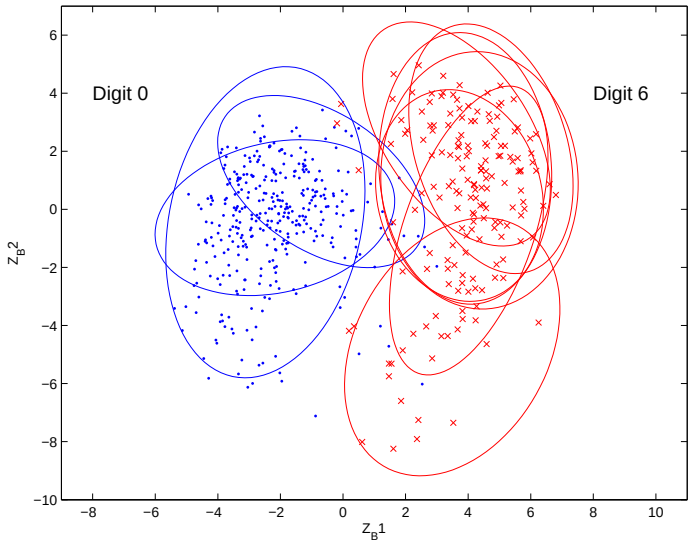

Figure 7.29: GMC(3, 7) contour on the backward 2-D space with test data (zip code data, digit 0 vs 6)

7.3.1.4 Digit 1 vs 7

Compared to the previous examples, digit 1 and 7 are much more difficult to classify. The size of the training sets are also smaller, with total of 1650 observations (1005 digit 1's, 645 digit 7's). The test set is of size 411 (264 digit 1's, 147 digit 7's).

Model		Misclassification Rate		Log-likelihood	
K_1	K_7	Train	Test	Train	Test
1	26–34	0.0018	0.0122	-51.3263	-87.9604
1	35	0.0012	0.0122	-46.0794	-87.9604
1	36	0.0012	0.0122	-46.0794	-114.6557
8	47	0	0.0122	-0.0000	-248.8695
9	47	0	0.0122	-0.0000	-222.6896
10	47	0	0.0122	-0.0000	-222.6702

Table 7.9: Top GMC classifiers for the Zip code data (digit 1 vs 7)

We ran the same procedure as in previous examples. First, we standardize each dimension of the original $X \in \mathbb{R}^{256}$, then use SVD to get the principal components. We fit GMC procedure on the space of the top 10 principal components. The top GMC classifiers are in Table 7.9. The clear winner is GMC(1,26) with minimum test error, .0122, and the smallest degrees of freedom $K_1 + K_7$. Apply forward and backward projection algorithms on GMC(1,26), we obtain the optimal sub-feature spaces. In Figure 7.30 we see that the forward projected 1-D subspace is as good as 7-D subspace. Backward projection shows interesting anomaly that the test errors on 4-D and 5-D subspaces are higher than the lower subspaces. This seems to suggest that a subspace beyond 3-D, the added noise dimensions can be detrimental to the classifier.

We plot the projected training set and GMC(1,26) of the 2-D forwarded subspace in Figure 7.31 and the projected test set in Figure 7.32. The first 1-D forward projection can provide almost perfect separation and the 2nd dimension does not offer much help. The counterparts of backward projection results are in Figure 7.33 and Figure 7.34. It is a simple ration away from the 2-D forward projection subspace.

The lesson we learn here is the possible strategy to rerun GMC on projected data to reduce K_0 and K_1. The MDA(1,26) in this example gives misclassification rate .0012 on the training set and .0194 on the test set. In comparison to GMC(1,26)'s .0018 and .0122, MDA is slightly overfit the training set and therefore gives a bit higher test error. In practice GMC and MDA both do well in this case.

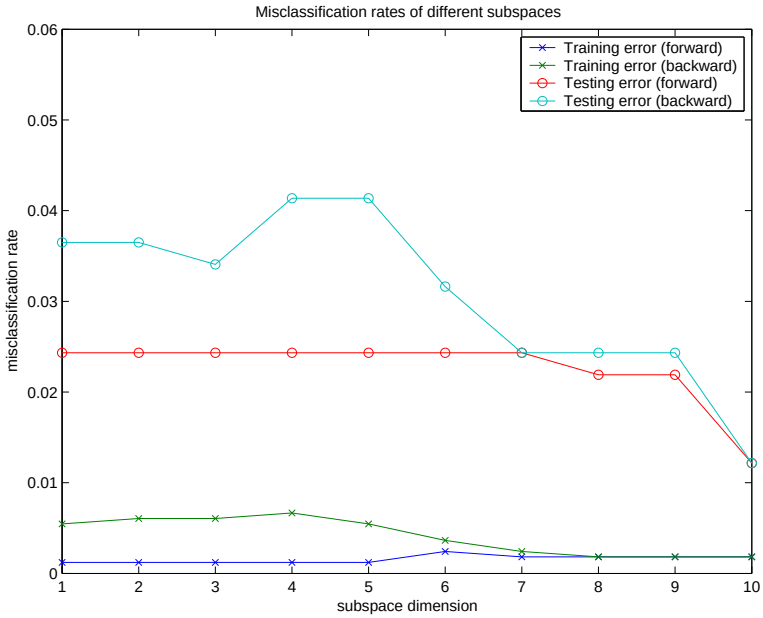

Figure 7.30: The error rates in subspaces of GMC$(1, 26)$ for the Zip code data (digit 1 vs 7)

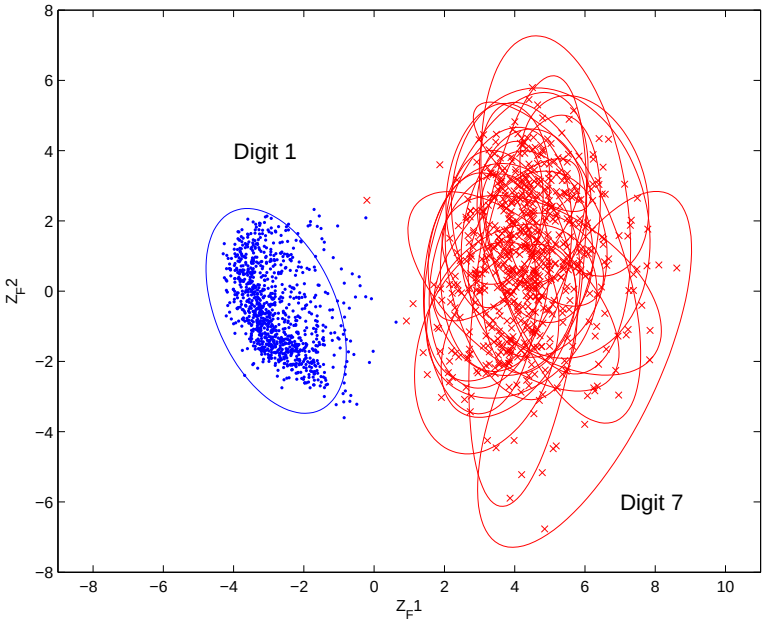

Figure 7.31: GMC$(1, 26)$ contours on the forward 2-D space with training data (zip code data, digit 1 vs 7)

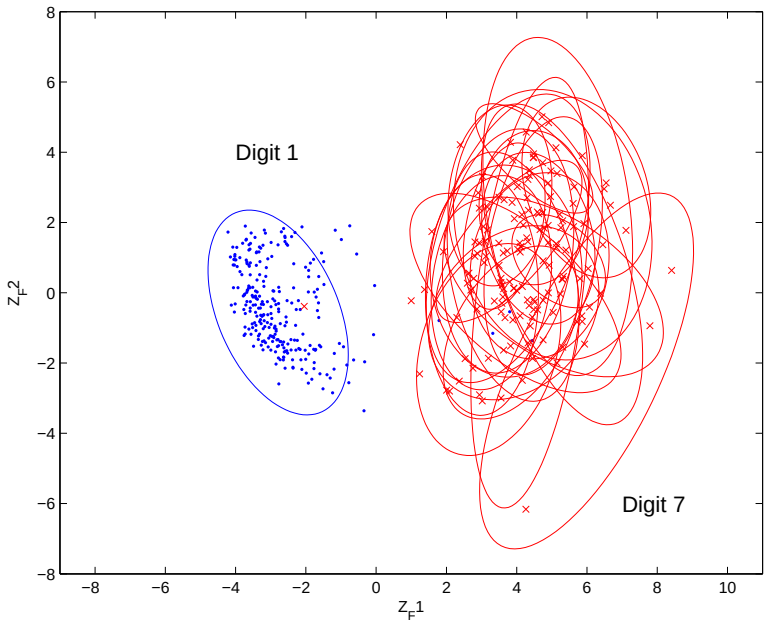

Figure 7.32: GMC(1, 26) contour on the forward 2-D space with test data (zip code data, digit 1 vs 7)

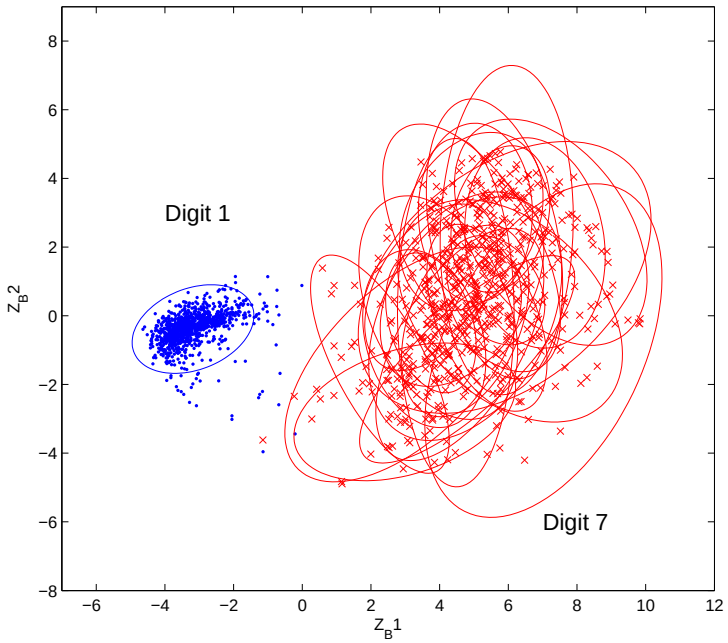

Figure 7.33: GMC(1, 26) contours on the backward 2-D space with training data (zip code data, digit 1 vs 7)

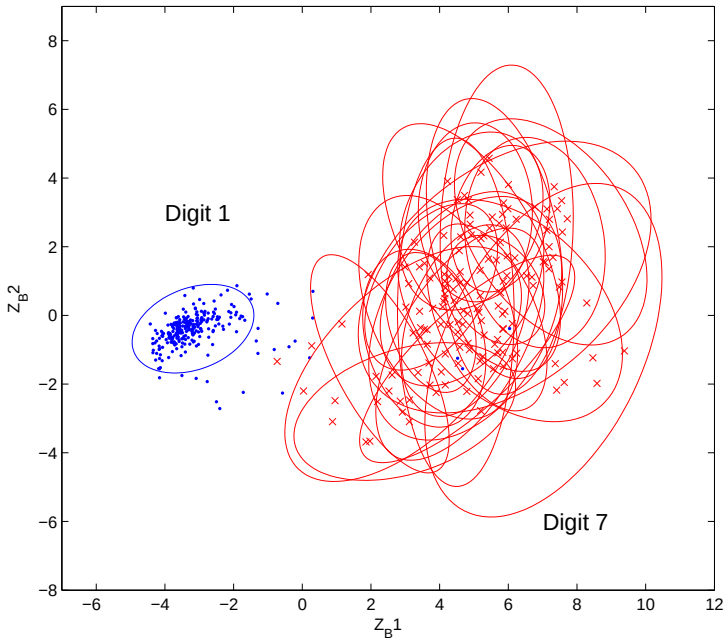

Figure 7.34: GMC$(1, 26)$ contours on the backward 2-D space with test data (zip code data, digit 1 vs 7)

7.3.2 Conclusions of GMC analysis on Zip Code Data

After applying the GMC procedure on the classification between three different pairs of zip code digits, we are pleased to find that in all three examples, GMC is either comparable or better performance than MDA. In practice, it is a good idea to use the GMC procedure at least for the initial analysis. As shown in all three examples, GMC can outperform MDA, and at the very least, help determine the appropriate of the parameter (K_0, K_1).

The empirical results in Table 7.6 and Table 7.7 show that the MLE update does increase the log-likelihood values and improve the training performance. But the MLE models have larger test errors than their MoM counterparts. This phenomenon indicates that the GMC/MLE is somewhat overfitting the training set, which leads to higher test error. Hence we conclude that for classification problems, MLE is generally unnecessary, sometimes even detrimental in the sense that GMC/MLE overfits the training data.

The misclassification rate is the standard criterion for classifier selection. But it is sometimes too crude to differentiate among the best models. We think the log-likelihood is a good tie-breaker.

Both forward and backward projection algorithms are very useful for classification problems. As Fisher pointed out in his LDA, for two-class problems if we assume each

class pdf is Gaussian, then we need only a 1-D feature space for an effective classifier. In our case we assume each class pdf is Gaussian mixtures, so we may need more then 1-D, but certainly no need for a 10-dimension feature space. In our experience of the Zip Code analysis, a 2-D space is the most we need. In practice, it may be a good strategy to fit GMC on the original data, selection the best GMC based on the test error. Then apply forward and backward projection algorithms to identify optimal subspaces. Intuitively, the original GMC model will have larger (K_0, K_1) then necessary for the projected data. Therefore it should be a good idea to rerun GMC procedure on the projected data. A recursive GMC procedure can be very interesting and beneficial.

7.3.3 Affymatrix ALL/AML Data

In this section we lay our hands on the famous ALL/AML microarray data set. Golub et al. (1999) first published their analysis of this data set. Since then, the ALL/AML data have been made publicly available and have become one of the benchmark data sets for new microarray data analysis procedures.

The ALL/AML data consist of samples from two kinds of leukemia: acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML). The authors's goal was to use the gene expression profile to distinguish ALL from AML, which is critical for selecting the effective treatment. The gene expression profile of each sample was collected by Affymatrix chip record of the cDNA expression level of 6,817 genes, with 7,129 probes in each chip (some genes have multiple reads).

Golub's initial data set consisted of 38 bone marrow samples, 27 ALL and 11 AML. They developed a 50-gene classifier based on the 38 samples. They applied this predictor to another data set of 34 more samples. They predicted 29 out of 34 correctly. That is equivalent to a 14.7% prediction error. As is common for microarray data, the training error is 0.

We obtained the data from the website (http://www.broad.mit.edu/cgi-bin/ cancer/publications/), there is one more sample in the test set, thus we have the same training set of size $n = 38$ and a test set of size $n = 35$ such that (X_i, Y_i), $X_i \in \mathbb{R}^{7129}$, $Y_i \in \{0, 1\}$. Our goal is to construct a classifier using Gaussian Mixture Classification (GMC) procedure and compare our results to Golub et al. (1999).

We first normalize the data by standardizing the gene expressions level across the samples. It is a common practice in microarray analysis to calibrate the individual array in order to eliminate the non-biological effect. Dudoit and Yang (2003) gives a comprehensive introduction to the general issues in cDNA Microarray Data analysis. After across-array normalization, the next step is to reduce the dimension of the data. At this moment, our software cannot handle $p = 7129$ straight up, so we first reduce the dimension to a manageable range.

We compute and rank the genes by their two-sample t-score t_j, $j = 1, \cdots, 7129$:

$$t_j = \frac{\overline{X}_{j|1} - \overline{X}_{j|0}}{\sqrt{S_p^2(1/n_0 + 1/n_1)}},$$

where

$$S_p^2 = \frac{(n_0 - 1)S^2[X_{j|0}] + (n_1 - 1)S^2[X_{j|1}]}{n_0 + n_1 - 2}.$$

There are 151 genes having $|t_j| > 4$. Our next step is to run the principal component analysis on these 151 genes. The principal components give up a hierarchy of nested feature spaces. We apply GMC procedure on the space spanned by the first p principal components with $p = 2, 3, 4, 5, 10$ and the initial bandwidth $h = .01$. The run time is less than 20 seconds (on Sun SPARC workstation). The results are summarized in Table 7.10. GMC procedure performs well. It is interesting to note that the test error starts high at 8.57%, then it goes down to 5.71% and finally goes up again at $p = 10$. This trend suggests that the dimension of the feature space is less than 5. 10 is obviously too high. In this case, we only need a GMC(2,1) on the first 3 principal components to give us a nice 5.71% test error. That is almost 1/3 the test error reported in Golub et al. (1999). We do not report the training error because all the models have perfect classification on the training set.

Table 7.11 records the test errors of the corresponding MDA models applied to the same data sets. The results show that GMC is comparable to MDA. One obvious advantage of GMC over MDA is that GMC does not require the guess work on (K_0, K_1).

Figure 7.35 shows the GMC(2,2) classifier for the ALL/AML data on the space of the first two principal components. One interesting feature of ALL/AML data is that the ALL samples are much tightly distributed on this 2-D principal component space. AML samples spread all over the place. This means biologically, ALL samples are more homogeneous while the individual variation among AML samples is large. Fortunately this AML variation does not overshadow the location differences between the ALL and AML groups. From Figure 7.35 it is obvious that $K_{ALL} = 1$ is good enough to classify the two groups. And K_{AML} seems irrelevant for the classification. This picture is consistent with the results in Table 7.10, where K_{AML} varies among models of the same minimum test error. This is an empirical evidence that the best (K_0, K_1) for GMC model performance may not be the best for the underlying class pdf's.

Dimension (p)	Minimum Test Error	GMC(K_{ALL}, K_{AML})
2	.0857	(2, 1)
3	.0571	(2, 1–3), (3, 1–3), (26–27, 4)
4	.0571	(1–3, 1)
5	.0571	(1, 1), (2, 1)
10	.0857	(1, 2–11)

Table 7.10: Top GMC Classifiers for the ALL/AML data

Dimension (p)	(K_{ALL}, K_{AML})	Test Error (MDA)	Test Error (GMC)
2	(2, 1)	.2000	.0857
3	(2, 1)	.0571	.0571
4	(1, 1)	.1429	.0571
4	(2, 1)	.0571	.0571
5	(1, 1)	.0285	.0571
5	(2, 1)	.0571	.0571
10	(1, 1)	.1143	.0857
10	(1, 2)	.0857	.0857

Table 7.11: Top GMC and MDA classifiers for the ALL/AML data

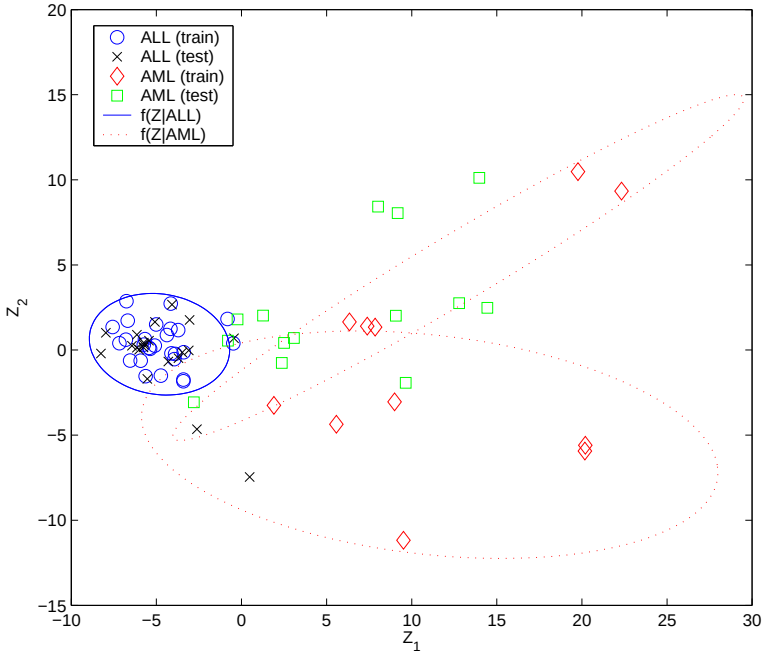

Figure 7.35: The contour plot of GMC(1, 2) for the ALL/AML Data

7.3.4 Wisconsin Breast Cancer Data

Wisconsin Breast Cancer Data were first analyzed by Street et al. (1993). Mangasarian et al. (1995) propose a linear programming approach for the problem. The training data set consists of 10 covariates with 569 observations. The goal is to predict whether the tumor is cancerour or benign using the 10 features of the cells:

1. radius (mean of distances from center to points on the perimeter)
2. texture (standard deviation of gray-scale values)
3. perimeter
4. area
5. smoothness (local variation in radius lengths)
6. compactness ($perimeter^2/(area - 1)$)
7. concavity (severity of concave portions of the contour)
8. concave points (number of concave portions of the contour)
9. symmetry
10. fractal dimension ("coastline approximation" - 1)

A test set of size 100 (60 negative; 40 positive) is used to estimate the predictive error and misclassification rate. We first apply GMR to the data, pretending the binary response Y as continuous. We then design the GMR classifier as

$$\widehat{y}(x) = I(\widehat{m}(x) > cut). \tag{7.3}$$

There are two options to pick the cut value. One is to 1/2, the other is $\bar{y}$. They correspond to equal prior probability $\pi_1 = 1/2$ and $\pi_1 = n_1/(n_0 + n_1)$. We try both classifiers and the results are very similar. Figure 7.36 shows that the difference between the two choices are very small.

Figure 7.37 records the RMSE profile of GMR models on both training set and test set. GMR(4) has the minimum predictive error on test set. In Figure 7.36, GMR(4) and GMR(5) both have minimum misclassification rate of 5% on the test set using $cut = 0.5$. The classifier using $cut = \bar{y} = .37$ gives test error rate 8% and 6% for GMR(4) and GMR(5). Because y is actually a binary variable, the misclassification rates on the test set should be the most reasonable criterion. The informative prior cut $cut = .37$ is closer to the bayes classifier; therefore, we pick GMR(5) as the best

	Fitted Y		0 (Benign)		1 (Cancerous)		
Train	True Y	0	286	(.610)	11	(.024)	297
		1	8	(.017)	164	(.350)	172
Test	True Y	0	55	(.55)	5	(.05)	60
		1	1	(.01)	39	(.39)	40

Table 7.12: The GMR(5) fit of the Wisconsin breast cancer data

model for the data set. Table 7.12 records the details of "GMR(5) classifier" using the informative prior cut.

The overall training error of the GMR fit is .041 with test error .06. The results of GMR analysis are in Table 7.12.

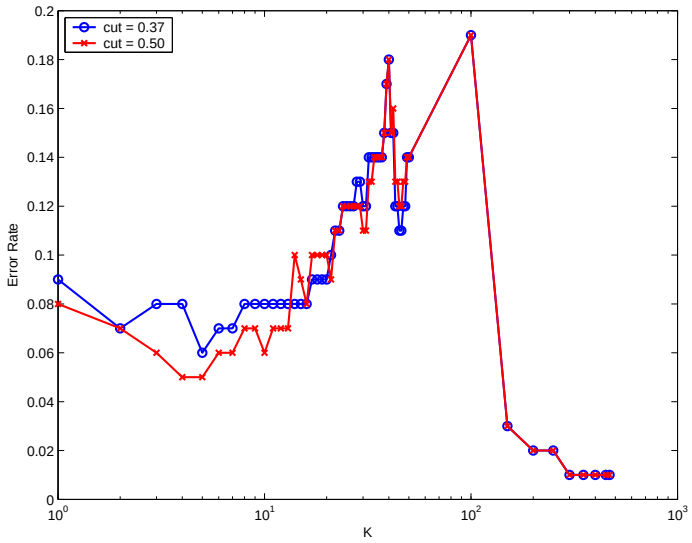

Figure 7.36: Misclassification rates of GMR of the Wisconsin Data

model		misclassification rate		log-likelihood	
K_0	K_1	train	test	train	test
4	2	.0320	.0800	-50.2449	-30.3972
5	2	.0341	.0800	-48.5124	-30.8421
6	2	.0299	.0800	-44.0597	-29.5315
7	2	.0299	.0800	-43.9520	-29.5296
8	2	.0299	.0800	-42.1869	-27.4682

Table 7.13: Top GMC classifiers for the Wisconsin data

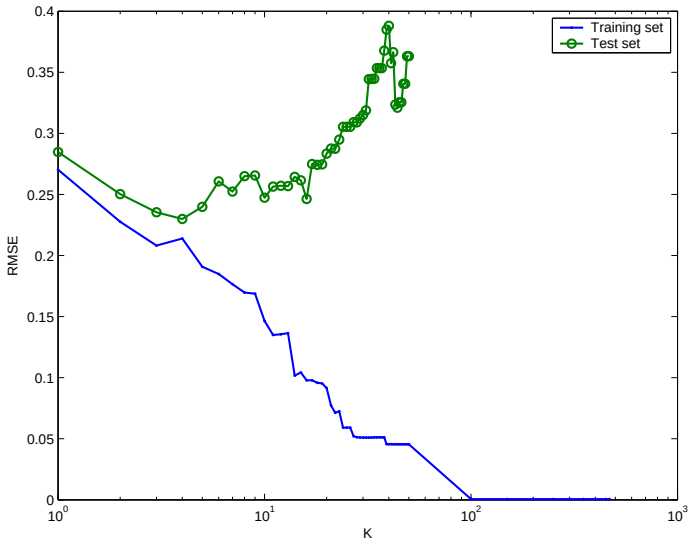

Figure 7.37: RMSE and PE of GMR of the Wisconsin Data

We apply GMC for the same data set. Table 7.13 displays the top GMC models in terms of minimum test error. An obvious choice is GMC(4,2). Table 7.14 lists four models: GMR(5) chosen from the GMR procedure, GMC(4,2) with MoM parameters, GMC(4,2) with MLE parameters, and MDA(4,2). Interestingly, GMR(5) gives the minimum test error of 6%. The MLE update of GMC(4,2) improves the training error but has no effect on the test error. Once again, it indicates that the MLE update tightens the model fit of the training data but it does not improve the test error. Finally, the best GMR and GMC models all give smaller test errors than does MDA.

		misclassification rate		log-likelihood	
model		train	test	train	test
GMR(5)	MoM	.0420	.0600		
GMC(4,2)	MoM	.0320	.0800	-50.2449	-30.3972
GMC(4,2)	MLE	.0235	.0800	-36.0696	-49.8813
MDA(4,2)		.0405	.1000	-52.4627	-23.2221

Table 7.14: The GMR(5), GMC(4,2) and MDA for the Wisconsin data

Chapter 8

Conclusions

This chapter summarizes the key findings and discusses several problems for future research that can further enhance the GMR and GMC procedures.

8.1 Key Findings

The immediate goal of this research is to construct new statistical procedures that can overcome the curse of dimensionality. We have accomplished this goal by showing that both GMR and GMC offer different modeling techniques that give results comparable to the popular MARS and MDA procedures for multivariate data. The objective here is not to replace MARS and MDA but to offer worthy alternatives. As Breiman (1991, p.87) pointed out:

> ...for creative data analysis, the desideratum is to get as many different views as possible of what may be going on.

GMR and GMC do offer different views from the standard procedures such as MARS and MDA, and therefore, worthy additions to a data analyst's tool box.

GMR and GMC overcome the curse of dimensionality by employing a global parametric model of the underlying joint density. By using the Gaussian mixture density for this task, GMR and GMC inherit the versatility of the Gaussian mixture density to capture nonlinear patterns in the data.

One unique contribution of this research is a new way to harness the power of the Gaussian mixture. The key component of GMR and GMC is the IPRA procedure. The beauty of IPRA is its simplicity in generating a sequence of Gaussian mixture densities that goes through the complex model space of the Gaussian mixture with one representative for each number of components, K. IPRA achieves simplicity by applying method of moments for the estimation of model parameters, thereby avoiding

the heavy computation required by MLE. We treat IPRA as a simple "sampling technique" that enables the construction of the GMR model family indexed by K and of the GMC family, indexed by (K_0, K_1). The corresponding profiles of goodness-of-fit thus provide systematic ways to select the appropriate number of components.

Finally, a unique feature of GMR and GMC, inherited from the Gaussian mixture density, is their invariance under linear transformation. This feature allows us to implement straightforward model-based dimensional reduction algorithms for GMR and GMC.

8.2 Future Research

IPRA/GMR and IPRA/GMC are new procedures based on the fundamental principle that all statistical information is contained in the joint density function. They demonstrate a new template for building global models for high dimensional data via density estimation. We believe this is fertile ground for intriguing research. To name a few possibilities:

Computation Efficiency: We have demonstrated that in theory GMR/GMC is applicable to data of any dimension. In our case study in Chapter 7, we did not apply GMR/GMC to feature space of dimension higher than 25 because it is unnecessary to go beyond that for our examples. Another reason is that as the dimension grows, the computing time increases exponentially, thereby making GMR/GMC not practical for extra high dimensional data. One promising approach to improve the computing efficiency is to take advantage of the parallel nature of the IPRA merging operations. Future implementation of GMR should explore the new specialized linear algebra tools in MATLAB to speed up the computation.

Early Dimension Reduction: A weakness of GMR is its susceptibility to a large number of noisy dimensions because GMR fits the data as a whole instead of approximating the regression function one dimension at a time. The key to ameliorate the GMR performance under noisy data is to have an effective dimensional reduction technique. In Chapter 6 we developed the model-based forward and backward algorithm for GMR and GMC. One promising solution to improve GMR performance under noisy data is to incorporate these dimension reduction techniques early in the IPRA/GMR procedure to eliminate noisy dimensions.

Mixture of Generalized Linear Models: In practice, it is often to have both categorical and continuous features in one data set. So far, GMR is limited to continuous features only. To extend GMR procedure to categorical features and responses, we may explore a mixture of Gaussian and non-Gaussian components for the joint density. This line of research should lead to a full-blown extension of GMR to generalized linear models such as Poisson regression.

Gaussian Mixture Quantile Regression: GMR can be viewed as a summary statistic of a Gaussian mixture density. It is natural to derive other summary statis-

tics, such as the conditional median and the conditional mode. In the most general form, a Gaussian mixture quantile regression will be a useful EDA tool.

Bayesian GMR: Gaussian mixture is a popular object for Bayesian inference. Under a Bayesian framework, the model interpretation is straightforward and the model uncertainty can be evaluated using MCMC technology. Following is a road map of a Bayesian analysis for GMR:

- Set up conjugate prior distributions for Θ: the Dirichlet prior for the mixing probabilities $\Pi = (\pi_1, \cdots, \pi_K)$, multivariate Gaussian prior for the mean $\mu_j \sim N(0, I_p)$, and inverse-Gamma prior for the variance $\Sigma_j \sim IG$.

- Set up the full likelihood for the hierarchical model described in Chapter 2: The inclusion of the latent cluster indicator variable in the full likelihood will simplify the mixture model fit, and therefore simplify the GMR derivation.

- Under the full model, write down the explicit form of the posterior mode of each parameter.

- Set up a MCMC framework for simulation. For any given value of Θ, it is not difficult to evaluate the full likelihood $L(X, Y; \Theta)$, hence it is not hard to evaluate the acceptance probability required by the Metropolis-Hasting algorithm.

- Apply the Metropolis-Hasting algorithm to generate samples from the appropriate stationary distribution $\Theta|\text{Data}$. We can then obtain samples of GMR estimator $m(x; \Theta)|\text{Data}$. The summary statistics of these sample regression curves will provide the uncertain measure of the finite-sample GMR curve.

Appendix A

Hellinger Metric

The inner product of two Gaussian density functions has an elegent form (A.1). They are very useful in our exploration of the theoretical properties of finite Gaussian mixtures (Wand and Jones 1995).

$$\langle \phi_{\mu_1,\Sigma_1}, \phi_{\mu_2,\Sigma_2} \rangle = \phi(0; \mu_1 - \mu_2, \Sigma_1 + \Sigma_2) \tag{A.1}$$

where $\langle f, g \rangle = \int f\, g$ and ϕ is the Gaussian pdf for $x \in \mathbb{R}^p$:

$$\phi(x; \mu, \Sigma) = |2\pi\Sigma|^{-1/2} \exp\{-1/2(x-\mu)^T \Sigma^{-1}(x-\mu)\}$$

That is,

$$\int \phi(x; \mu_1, \Sigma_1)\, \phi(x; \mu_2, \Sigma_2) dx = \phi(0; \mu_1 - \mu_2, \Sigma_1 + \Sigma_2) \tag{A.2}$$

To simplify the notations, we define the integral

$$I(\mu_1, \mu_2, \Sigma_1, \Sigma_2) = \int \exp\{-\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)\} \exp\{-\frac{1}{2}(x-\mu_2)^T \Sigma_2^{-1}(x-\mu_2)\}.$$

Since we have

$$\int \phi_1\, \phi_2 = |2\pi\Sigma_1|^{-1/2} |2\pi\Sigma_2|^{-1/2} I(\mu_1, \mu_2, \Sigma_1, \Sigma_2),$$

(A.2) implies that

$$I(\mu_1, \mu_2, \Sigma_1, \Sigma_2) = |2\pi\Sigma_1|^{1/2} |2\pi\Sigma_2|^{1/2} \phi(0; \mu_1 - \mu_2, \Sigma_1 + \Sigma_2). \tag{A.3}$$

Let $\phi_j = \phi(x; \mu_j, \Sigma_j)$ and observe that

$$\begin{aligned}
\sqrt{\phi_j} &= \left(|2\pi\Sigma_j|^{-1/2} \exp\{-1/2(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)\} \right)^{1/2} \\
&= |2\pi\Sigma_j|^{-1/4} \exp\{-1/4(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)\} \\
&= |2\pi\Sigma_j|^{-1/4} \exp\{-1/2(x-\mu_j)^T (2\Sigma_j)^{-1}(x-\mu_j)\},
\end{aligned}$$

we obtain the closed form for another integral

$$\begin{aligned}\int \sqrt{\phi_1\phi_2} &= |2\pi\Sigma_1|^{-1/4}|2\pi\Sigma_2|^{-1/4} I(\mu_1,\mu_2,2\Sigma_1,2\Sigma_2) \\ &= |2\pi\Sigma_1|^{-1/4}|2\pi\Sigma_2|^{-1/4}|2\pi 2\Sigma_1|^{1/2}|2\pi 2\Sigma_2|^{1/2}\phi(0;\mu_1-\mu_2,2\Sigma_1+2\Sigma_2) \\ &= (2\sqrt{2\pi})^p|\Sigma_1|^{1/4}|\Sigma_2|^{1/4}\phi(0;\mu_1-\mu_2,2\Sigma_1+2\Sigma_2).\end{aligned}$$

Applying this result to obtain the closed form of the Hellinger metric

$$\begin{aligned}H(\phi_1,\phi_2) &= \int (\sqrt{\phi_1}-\sqrt{\phi_2})^2 \\ &= \int \phi_1+\phi_2-2\sqrt{\phi_1\phi_2} \\ &= 2-2\int\sqrt{\phi_1\phi_2} \\ &= 2-2(2\sqrt{2\pi})^p|\Sigma_1|^{1/4}|\Sigma_2|^{1/4}\phi(0;\mu_1-\mu_2,2\Sigma_1+2\Sigma_2).\end{aligned}$$

Hence, the direct Hellinger metric between two mixture components is

$$\begin{aligned}H(w_1\phi_1,w_2\phi_2) &= \int (\sqrt{w_1\phi_1}-\sqrt{w_2\phi_2})^2 \\ &= \int w_1\phi_1+w_2\phi_2-2\sqrt{w_1w_2}\sqrt{\phi_1\phi_2} \\ &= w_1+w_2-2\sqrt{w_1w_2}\int\sqrt{\phi_1\phi_2}.\end{aligned}$$

Appendix B

Method of Moments Estimation

In this section we derive the method of moments estimations of a 2-component Gaussian mixture models employed in multivariate IPRA (Section 3.1). Following is the problem setting: Given the pdf of X

$$f_X(x) = w\,\phi(x;\mu_1,\Sigma_1) + \widetilde{w}\,\phi(x;\mu_2,\Sigma_2),$$

where $\widetilde{w} = 1 - w$, compute the mean and variance of X.

First, derive $\mathrm{E}X$ and $\mathrm{E}[XX^T]$:

$$\begin{aligned}
\mathrm{E}X &= \int x\, f_X(x)\, dx \\
&= \int x\, w\, \phi(x;\mu_1,\Sigma_1) + x\, \widetilde{w}\, \phi(x;\mu_2,\Sigma_2)\, dx \\
&= w \int x\, \phi(x;\mu_1,\Sigma_1)\, dx + \widetilde{w} \int x\, \phi(x;\mu_2,\Sigma_2)\, dx \\
&= w\,\mu_1 + \widetilde{w}\,\mu_2
\end{aligned}$$

$$\begin{aligned}
\mathrm{E}[XX^T] &= \int xx^T f_X(x) dx \\
&= \int xx^T w\phi(x;\mu_1,\Sigma_1) + xx^T \widetilde{w}\phi(x;\mu_2,\Sigma_2) dx \\
&= w \int xx^T \phi(x;\mu_1,\Sigma_1) dx + \widetilde{w} \int xx^T \phi(x;\mu_2,\Sigma_2) dx \\
&= w(\Sigma_1 + \mu_1\mu_1^T) + \widetilde{w}(\Sigma_2 + \mu_2\mu_2^T)
\end{aligned}$$

$$
\begin{aligned}
\mathrm{Var}(X) &= \mathrm{E}\{(X-\mathrm{E}X)(X-\mathrm{E}X)^T\} \\
&= \mathrm{E}[XX^T] - \mathrm{E}X\mathrm{E}X^T \\
&= w\,(\Sigma_1+\mu_1\mu_1^T) + \widetilde{w}\,(\Sigma_2+\mu_2\mu_2^T) - (w\mu_1+\widetilde{w}\mu_2)(w\mu_1+\widetilde{w}\mu_2)^T \\
&= w\,\Sigma_1 + \widetilde{w}\,\Sigma_2 + w\,\widetilde{w}\,(\mu_1\mu_1^T+\mu_2\mu_2^T-\mu_1\mu_2^T-\mu_2\mu_1^T) \\
&= w\,\Sigma_1 + \widetilde{w}\,\Sigma_2 + w\,\widetilde{w}\,(\mu_1-\mu_2)(\mu_1-\mu_2)^T
\end{aligned}
$$

Appendix C

EM Algorithm for Gaussian Mixture Models

The landmark article that introduces the EM algorithm is Dempster, Laird, and Rubin (1977). EM algorithm is an iterative method for finding the local maximum of the likelihood function. There are several reasons for its popularity. It is conceptually elegant and in many cases the implementation is straightforward. In addition, the EM algorithm always converges monotonically. On the other hand, EM algorithm also suffers some disadvantages. The most severe one is that the EM algorithm results depend on the initial value. Also, the convergence rate can be very slow. Hence, the key for the success of EM algorithm is a good initial guess.

EM algorithm is the standard algorithm for computing the MLE of the Gaussian mixture models. Given the number of component K, the EM algorithm for Gaussian Mixtures is straightforward.

The natural way to formulate a K-component mixture model is to introduce a latent variable G, the indicator of the mixture components. The mixture model can then be formulated as a hierarchical model:

$$G_i \in \{1, 2, \cdots, K\}, \tag{C.1}$$

$$Pr(G_i = k) = \pi_k, \tag{C.2}$$

and

$$X_i | G_i = k \sim N(\mu_k, \Sigma_k). \tag{C.3}$$

The parameter of the model is $\Theta = (\theta_1, \theta_2, \cdots, \theta_K)$, where $\theta_k = (\pi_k, \mu_k, \Sigma_k)$. For data $X \in R^p$, the dimension of Θ is $K(p + p(p+1)/2 + 1) - 1$. As K and p increase, the dimension of Θ grows quickly. However, in principle, the EM algorithm for the mixture models is straightforward. The key fact is that given the value of the latent

variable G, the mixture model reduces to a simple Gaussian. The MLE for a single Gaussian is just the sample mean and sample variance. Therefore, the EM algorithm for the Gaussian Mixture Models is straightforward:

1. Take the initial value $\Theta^{(0)} = (\pi^{(0)}, \mu^{(0)}, \Sigma^{(0)})$

2. E-step: compute the mixture weights for each X_i, which is

$$w_{ik} = Pr(G_i = k|X_i, \Theta^{(0)}) = \frac{\pi_k^{(0)}\phi(X_i; \mu_k^{(0)}, \Sigma_k^{(0)})}{\sum_{k=1}^{K} \pi_k^{(0)}\phi(X_i; \mu_k^{(0)}, \Sigma_k^{(0)})}. \tag{C.4}$$

3. M-step: compute the weighted mean, variance, and the new mixing weight:

$$\mu_k^{(1)} = \frac{\sum_{i=1}^{n} w_{ik} X_i}{\sum_{i=1}^{n} w_{ik}}, \tag{C.5}$$

$$\Sigma_k^{(1)} = \frac{\sum_{i=1}^{n} w_{ik}(X_i - \mu_k^{(1)})(X_i - \mu_k^{(1)})^T}{\sum_{i=1}^{n} w_{ik}}, \tag{C.6}$$

and

$$\pi_k^{(1)} = \sum_{i=1}^{n} w_{ik}/n. \tag{C.7}$$

4. Iterate step 2 and 3 until convergence.

One of the most critical issues in using the EM algorithm is the choice of the initial value $\Theta^{(0)}$. The EM algorithm is guaranteed to converge to a local maximum, so the initial value will determine if EM algorithm converges to a good local maximum. Since the likelihood of the Gaussian mixture is unbounded above, the global MLE does not exist. We apply IPRA to the starting kernel model to obtain good approximation of the K-component mixture model. Obviously this IPRA approximation is not the MLE. But it should be a good initial value for the EM algorithm.

An alternative for the MLE is to maximize the log-likelihood function directly.

$$\log L(\Theta; X) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k \, \phi(X_i; \theta_k) \tag{C.8}$$

The IPRA approximation of the mixture models provides good initial values for a general numerical maximization package as well.

Appendix D

Some Results in the Simulation Study

There are some detailed numerical results in the simulation study of GMR in Section 7.2.2.

Table D.1: Root mean squared errors of projected GMR models

dim	forward			dim	backward		
n	200	300	400		200	300	400
K	9	18	28		9	18	28
0	1.1921	1.1168	1.1033	5	0.1938	0.2609	0.2658
1	0.8519	0.8269	0.8382	4	0.2382	0.2766	0.2866
2	0.2998	0.3280	0.7592	3	0.2715	0.3016	0.3057
3	0.2634	0.3139	0.3274	2	0.7541	0.3099	0.3185
4	0.2352	0.2952	0.2886	1	0.8530	0.7678	0.7834
5	0.1938	0.2609	0.2658	0	1.1921	1.1168	1.1033
K	9	25	34		9	25	34
0	1.1921	1.1168	1.1033	10	0.1407	0.0882	0.1969
1	0.8465	0.8224	0.8435	9	0.1423	0.0902	0.2013
2	0.3369	0.5190	0.4248	8	0.1463	0.1178	0.2234
3	0.2958	0.3740	0.3966	7	0.1623	0.1643	0.2706
4	0.2666	0.3333	0.3680	6	0.2188	0.2595	0.3215
5	0.2542	0.2835	0.3419	5	0.2694	0.3505	0.3747
6	0.2131	0.2167	0.3077	4	0.2925	0.4381	0.4107
7	0.1941	0.1617	0.2631	3	0.3605	0.4917	0.4360
8	0.1717	0.1193	0.2459	2	0.8503	0.5292	0.8439
9	0.1541	0.1006	0.2096	1	0.8820	0.8662	0.8478
10	0.1407	0.0882	0.1969	0	1.1921	1.1168	1.1033

Table D.2: Forward projections ($n = 400$, $p = 5$)

	1st-D	2nd-D	3rd-D	4th-D
GMR(9)	0.4165	0.6572	-0.6161	-0.0805
	0.0568	0.0735	-0.0749	0.8183
	0.9016	-0.3245	0.2696	0.0479
	0.0223	0.6763	0.7362	0.0090
	0.0994	-0.0040	0.0139	-0.5670
GMR(18)	0.4554	-0.6035	0.6508	0.0561
	0.0811	-0.0024	0.0362	-0.4563
	0.8818	0.2782	-0.3683	0.0961
	0.0383	0.7471	0.6629	0.0242
	0.0841	0.0136	0.0007	-0.8825
GMR(28)	0.4596	-0.5744	0.6658	0.1124
	0.0869	-0.0540	0.0434	-0.4707
	0.8770	0.2789	-0.3804	0.0918
	0.0357	0.7670	0.6404	-0.0183
	0.1042	-0.0317	0.0089	-0.8701

Table D.3: Backward projections ($n = 400$, $p = 5$)

	best 3-D			best 2-D		best 1-D
GMR(9)	0.3820	0.0706	-0.9038	-0.4290	-0.6058	0.3962
	0.1999	-0.0653	-0.0891	0.0238	-0.2232	-0.0357
	-0.2125	0.9674	-0.0401	-0.9015	0.3095	0.9167
	-0.8591	-0.2310	-0.4163	0.0435	0.6753	-0.0076
	-0.1757	0.0383	0.0178	-0.0279	0.1770	0.0373
GMR(18)	0.0000	0.7930	-0.0000	-0.0218	0.7773	0.2738
	-0.0870	0.1766	-0.7787	-0.0245	0.0189	0.0293
	-0.9092	0.2139	0.2233	0.9249	0.2539	-0.7918
	0.4006	0.5373	0.2412	-0.3787	0.5744	0.5452
	-0.0722	0.0746	-0.5343	-0.0031	-0.0327	-0.0077
GMR(28)	0.6969	0.0219	0.4535	-0.7765	0.1202	0.5966
	-0.1021	0.1513	0.4420	-0.0038	-0.0063	0.0065
	0.0973	-0.9461	0.1920	-0.0701	0.9589	-0.4456
	0.6861	0.1834	-0.2649	-0.6261	-0.2567	0.6674
	0.1538	-0.2188	-0.7014	0.0125	-0.0103	-0.0052

Table D.4: Forward projections ($n = 400$, $p = 10$)

	1st-D	2nd-D	3rd-D	4th-D	5th-D
GMR(9)	0.3688	0.6576	0.4807	0.0167	-0.3921
	0.0597	0.0979	-0.2246	-0.2293	-0.3789
	0.9204	-0.2549	-0.1619	0.0536	0.1992
	-0.0252	0.6980	-0.4706	0.0753	0.5017
	0.1019	0.0295	-0.4136	-0.0716	-0.1797
	-0.0313	-0.0118	-0.0768	0.3271	0.1192
	-0.0052	-0.0269	0.2736	0.5158	0.1727
	0.0157	0.0132	0.0565	-0.6878	0.1588
	-0.0252	0.0322	-0.2766	0.2651	-0.4321
	0.0175	0.0533	0.3732	-0.1310	0.3508
GMR(25)	0.4575	0.6344	-0.4151	-0.0860	-0.1332
	0.0652	0.0039	-0.1122	-0.6905	0.0279
	0.8785	-0.3531	0.2431	0.0482	0.1120
	0.0341	0.6840	0.4801	0.1086	0.1775
	0.0948	-0.0305	-0.2876	0.1382	0.0582
	-0.0269	-0.0360	-0.2010	-0.1750	0.6269
	-0.0495	0.0268	-0.2036	-0.0728	0.6330
	0.0328	-0.0179	-0.5927	0.2123	-0.2237
	-0.0159	-0.0119	0.0933	-0.6264	-0.2941
	-0.0005	0.0392	0.0308	-0.0964	0.0610
GMR(34)	0.4589	0.6127	0.1317	-0.0810	-0.1253
	0.0759	0.0048	0.0901	0.3199	0.3781
	0.8730	-0.3570	0.0303	0.0234	0.0648
	0.0453	0.7043	-0.1095	0.0703	0.1506
	0.1121	-0.0111	-0.3011	0.3382	-0.3226
	-0.0348	-0.0103	0.2529	0.1609	0.7524
	-0.0735	0.0052	0.6449	0.4460	-0.3660
	0.0163	-0.0273	0.0704	-0.3417	0.0400
	0.0006	-0.0079	-0.6195	0.2980	0.0932
	-0.0083	0.0053	-0.0496	0.5839	-0.0103

Table D.5: Backward projections ($n = 400$, $p = 10$)

	best 3-D			best 2-D		best 1-D
GMR(9)	-0.2900	-0.4647	0.5402	0.7103	0.2881	-0.3738
	-0.2902	-0.0041	-0.0232	0.1237	0.0107	-0.0899
	-0.2840	0.5666	0.6842	0.5667	-0.7407	-0.9055
	-0.1739	-0.6663	0.1233	0.3565	0.6019	0.0985
	-0.1914	-0.0062	0.0705	0.1541	-0.0145	-0.1293
	0.1385	0.0055	0.0384	-0.0372	-0.0164	0.0187
	0.6174	-0.1070	0.4025	0.0606	-0.0144	-0.0563
	-0.4194	0.0695	-0.1712	0.0449	-0.0169	-0.0456
	0.3140	-0.0082	0.1588	-0.0192	-0.0383	-0.0090
	-0.0960	-0.0513	-0.0378	0.0284	0.0601	0.0154
GMR(25)	0.8618	0.2452	0.1190	-0.2486	-0.7150	-0.3077
	-0.0376	0.0032	0.0221	-0.0190	0.0280	0.0331
	-0.1789	0.4351	0.8294	-0.9152	0.0679	0.7141
	0.2955	-0.6799	0.4418	0.0978	-0.6314	-0.5032
	0.1094	0.4399	-0.0810	-0.2203	0.0811	0.2162
	-0.0176	0.1906	-0.0839	-0.0580	0.1073	0.1157
	0.1915	-0.0764	0.0280	0.0276	-0.2053	-0.1605
	-0.1332	-0.2205	0.1010	0.0642	0.0112	-0.0391
	-0.0951	0.0410	-0.0029	-0.0241	0.0998	0.0858
	0.2465	0.0485	-0.2787	0.1825	-0.1201	-0.2153
GMR(34)	-0.8366	-0.2914	-0.0948	0.7124	0.1181	-0.6458
	-0.3698	0.2405	-0.0350	0.0821	-0.1833	0.0413
	0.0648	0.4175	-0.7384	0.0976	-0.8321	0.4100
	0.0667	-0.7220	-0.4800	0.6656	0.1067	-0.6012
	0.0491	-0.1083	-0.1003	0.0925	-0.0054	-0.0716
	0.2868	-0.1426	-0.0913	-0.0277	0.0237	0.0085
	0.1496	-0.1822	0.0346	0.0086	0.1451	-0.0922
	-0.0549	0.0483	0.1989	-0.1045	0.1193	0.0144
	-0.1817	0.1613	0.0938	-0.0457	-0.0341	0.0570
	-0.1085	0.2609	-0.3824	0.1045	-0.4600	0.1858

Table D.6: Forward projections ($n = 200$, $p = 10$)

1st-D	2nd-D	3rd-D	4th-D	5th-D	6th-D	7th-D
0.3786	-0.6856	-0.3964	0.1145	-0.1074	-0.0751	-0.1810
0.0098	0.0380	0.4299	0.3209	0.2361	0.2050	0.3201
0.9100	0.3215	0.0690	-0.0166	0.0441	0.1415	0.0997
0.0372	-0.6470	0.4429	-0.1148	0.1345	0.1839	0.2974
-0.0491	0.0083	0.1113	-0.5005	-0.4522	0.6541	-0.1046
-0.0476	0.0177	-0.0442	-0.3323	0.6322	-0.0232	0.0148
0.0081	-0.0511	0.2454	0.4499	0.1787	0.2666	-0.5015
-0.1359	0.0421	-0.5286	0.3258	0.2165	0.6257	0.0959
0.0484	-0.0420	-0.2450	-0.4093	0.4246	0.0854	0.0670
0.0386	0.0030	0.2078	-0.1830	0.2224	0.0162	-0.7001

Table D.7: Backward projections ($n = 200$, $p = 10$)

best 3-D			best 2-D		best 1-D
0.1622	0.7231	0.2710	0.0343	0.4431	-0.4438
-0.1803	0.2054	-0.4481	-0.4896	0.1660	-0.1543
-0.8601	0.3064	0.2138	0.1086	0.8542	-0.8565
0.4231	0.5077	0.0359	-0.1228	0.0622	-0.0593
-0.0748	0.1926	-0.6442	-0.6722	0.0403	-0.0243
0.0466	-0.1909	0.4125	0.4513	-0.0712	0.0604
0.0345	0.0037	-0.1427	-0.1368	-0.0532	0.0564
-0.0054	-0.0484	0.0959	0.1061	-0.0081	0.0056
0.0351	-0.0642	-0.2375	-0.2060	-0.1212	0.1260
-0.1106	0.0574	0.0874	0.0654	0.1355	-0.1370

Bibliography

Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University Press.

Breiman, L. (1991). Discussion of multivariate adaptive regression splines. *Annals of Statistics 19*(1), 82–91.

Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone (1984). *Classification and Regression Tree*. Pacific Grove, California: Wadsworth & Brooks.

Casella, G. and R. L. Berger (1990). *Statistical Inference*. Belmont, California: Duxbury Press.

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

Devroye, L. and L. Györfi (1985). *Nonparametric Density Estimation: The L_1 View*. New York: John Wiley & Sons.

Dudoit, S. and J. Y. H. Yang (2003). Bioconductor R packages for exploratory analysis and normalization of cdna microarray data. In G. Parmiginai, E. S. Garrett, R. A. Irizarry, and S. L. Zeger (Eds.), *The Analysis of Gene Expression Data*. New York: Springer-Verlag.

Efron, B. and R. J. Tibshirani (1993). *An Introduction to the Bootstrap*. New York: Chapman and Hall/CRC.

Figueiredo, M. (2000). On gaussian radial basis function approximations: interpretation, extensions, and learning strategies. In *Proceedings of the International Conference on Pattern Recognition - ICPR'2000*, Volume 2, Barcelona, Spain, pp. 618–621.

Figueiredo, M., J. M. N. Leitão, and A. K. Jain (1999). On fitting mixture models. In E. Hancock and M. Pellilo (Eds.), *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 54–69. Springer-Verlag.

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

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

Golub, T. R., D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander (1999). Molecular classification of cancer: Class discovery and class prediction by eene expression monitoring. *Science 286*, 531–537.

Gu, C., D. M. Bates, Z. Chen, and G. Wahba (1989). The computation of generalized cross-validation functions through householder tridiagonalization with applications to the fitting of interaction spline models. *SIAM Journal of Matrix Analysis and Applications 10*(4), 457–480.

Härdle, W. (1990). *Applied Nonparametric Regression*. Cambridge, United Kingdom: Cambridge University Press.

Hastie, T. and R. Tibshirani (1990). *Generalized Additive Models*. New York: Chapman and Hall.

Hastie, T. and R. Tibshirani (1996). Discriminant analysis by gaussian mixtures. *Journal of Royal Statistical Society B 58*(1), 155–176.

Hastie, T., R. Tibshirani, and J. Friedman (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.

LeCun, Y. (1989). Generalization and network design strategies. In *Technical Report CRG-TR-89-4*. Department of Computer Science, University of Toronto.

Lindsay, B. (1983). The geometry of mixture likelihoods: A general theory. *Annals of Statistics 11*(1), 86–94.

Lindsay, B. and P. Basak (1993). Multivariate normal mixtures: A fast consistent method of moments. *Journal of the American Statistical Association 88*(422), 468–476.

Loader, C. (1999). *Local Regression and Likelihood*. New York: Springer-Verlag.

Mangasarian, O. L., W. Street, and W. Wolberg (1995). Breast cancer diagnosis and prognosis via linear programming. *Operations Research 43*(4), 570–577.

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

McLachlan, G. and D. Peel (2000). *Finite Mixture Models*. New York: Wiley.

Nadaraya, E. A. (1964). On estimating regression. *Theory of Probability and Its Application 10*, 186–190.

O'Sullivan, F. (1991). Discussion of multivariate adaptive regression splines. *Annals of Statistics 19*(1), 99–102.

Richardson, S. and P. J. Green (1997). On bayesian analysis of mixtures with an unknown number of components. *Journal of Royal Statistical Society B 59*(4), 731–792.

Schmerling, S. and J. Peil (1985). Verfahren der lokalen approximation zur nichtparametrischen schätzung unbekannter stetiger funktionen aus meßdaten. *Gegenbaur Morphologisches Jahrbuch Leipzig 131*, 367–381.

Schmerling, S. and J. Peil (1986). Improvement of the method of kernel estimation by local polynomial approximation of the empirical distribution function and its application to empirical regression. *Gegenbaur Morphologisches Jahrbuch Leipzig 132*, 29–35.

Schmidt, G., R. Mattern, and F. Schüler (1981). Biomechanical investigation to determine physical and traumatological differentiation criteria for the maximum load capacity of head and vertebral column with and without protective helmet under effects of impact. In *EEC Research Program on Biomechanics of Impacts. Final Report Phase III, Project 65*, West Germany. Institue für Rechtmedizin, Universität Heidelberg.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization*. New York: Wiley.

Scott, D. W. and W. F. Szewczyk (2001). From kernels to mixtures. *Technometrics 43*(3), 323–335.

Sedgewick, R. (2002). *Algorithms in C* (3rd ed.). New York: Addison-Wesley.

Silverman, B. W. (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting (with discussion). *Journal of Royal Statistical Society B 47*, 1–52.

Stephens, M. (1997). *Bayesian Methods for Mixtures of Normal Distributions*. Ph. D. thesis, Magdalen College, Oxford University.

Street, W., W. Wolberg, and O. Mangasarian (1993). Nuclear feature extraction for breast tumor diagnosis. In *IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology*, Volume 1905, San Jose, CA, pp. 861–870.

Vapnik, V. N. (2000). *The Nature of Statistical Learning Theory* (2nd ed.). New York: Springer-Verlag.

Wand, M. P. and M. C. Jones (1995). *Kernel Smoothing*. London, UK: Chapman & Hall.

Watson, G. S. (1964). Smooth regression analysis. *Sankhyā 26*, 359–372.