Modelling with Discrete Mixtures of P-Splines

Daniel Bruce

April 28, 2009

MATH 4905 - Honours Project
Supervised by Dr. Jason D. Nielsen

1 Introduction

This paper introduces a method for pulling out multiple signals from a data set. Consider Figure 1 as a motivating example for our approach. We notice that a scatterplot of the data demonstrates a linearly increasing trend but also a periodic sine-like trend.

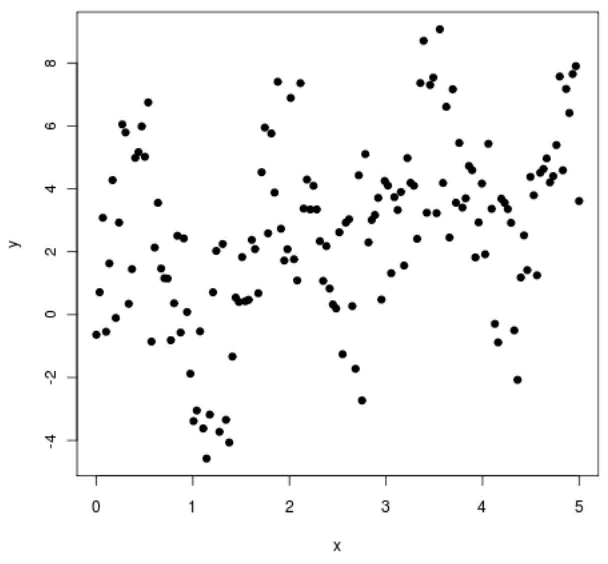

Figure 1: The scatterplot indicates two distinct signals.

We postulate that these signals are a result of our population being partitioned into two or more distinct classes. If we could determine the class membership for each data point, our problem of distinguishing the signals would be much simpler. A useful model for classification is a finite mixture of multivariate normal distributions, and so in chapter 2 we present a short background chapter on the details of the multivariate normal distribution. In chapter 3, we review discrete mixture modelling and discuss an application of the EM-algorithm for fitting these models. Once we have solved our classification problem, we would like to model the signals using nonparametric regression analysis. Chapter 4 discusses nonparametric regression and in particular the use of smoothing splines in scatter plot smoothing. We present an EM-type algorithm for modelling the signals using P-splines in chapter 5 and present a simulation demonstrating our results in chapter 6. Chapter 7 describes our conclusions and ideas for further studies.

2 The Multivariate Normal Distribution

This chapter summarizes a few known results concerning the multivariate normal distribution. For further reading, see Hogg, McKean and Craig (2005).

Let $\mathbf{X} = (X_1, X_2, \ldots, X_p)^T$ be a random vector. Then $\mathbf{X}$ follows a multivariate normal distribution if it has the following pdf,

$$f_{\mathbf{X}}(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{p}{2}} |\mathbf{\Sigma}|^{\frac{1}{2}}} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}, \text{ for } \mathbf{x} \in \mathbb{R}^n.$$

where $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_p)^T \in \mathbb{R}^p$, $\mathbf{\Sigma}$ is a $p \times p$ symmetric positive definite matrix, and $|\mathbf{\Sigma}|$ denotes the determinant of the matrix $\mathbf{\Sigma}$. For convenience, if $\mathbf{X}$ comes from a multivariate normal distribution, then we will write:

$$\mathbf{X} \sim \mathrm{N}_p(\boldsymbol{\mu}, \mathbf{\Sigma}),$$

where

$$\mathrm{E}[X_i] = \mu_i, \text{ for } i = 1, 2, \ldots, p$$

and

$$\mathrm{Cov}(X_i, X_j) = \sigma_{ij} \text{ for } i = 1, 2, \ldots, p, \ j = 1, 2, \ldots, p$$

where σ_{ij} denotes the i, j^{th} element of $\mathbf{\Sigma}$.

Note that the multivariate normal distribution is simply an extension of the univariate case. To see this, let $\mathbf{Y} \sim \mathrm{N}_1(\boldsymbol{\mu}, \mathbf{\Sigma})$. By definition, then $\mathbf{Y}$

has the pdf

$$\begin{aligned} f_{\mathbf{Y}}(\mathbf{y}) &= \frac{1}{(2\pi)^{\frac{p}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2} (\mathbf{y}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{y}-\boldsymbol{\mu})\right\} \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{\frac{-(y-\mu)^2}{2\sigma^2}\right\} \end{aligned}$$

where y, μ and σ^2 denote the single elements in $\mathbf{y}$,$\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ respectively. Note that this is the pdf of a univariate normal random variable, with mean μ and variance σ^2.

2.1 Properties of the Multivariate Normal Distribution

We present some useful properties of the multivariate normal distribution. The proof and derivation of these results can be found in Appendix A.

Let $\mathbf{X} \sim \mathrm{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then:

1. The moment generating function $M_{\mathbf{X}}(\mathbf{t})$ of $\mathbf{X}$ is

$$M_{\mathbf{X}}(\mathbf{t}) = \exp\left\{\frac{1}{2}\mathbf{t}^T\boldsymbol{\Sigma}\mathbf{t} + \mathbf{t}^T\boldsymbol{\mu}\right\}, \forall \mathbf{t} \in \mathbb{R}^p$$

2. Let A be an $m \times p$ matrix, $b \in \mathbb{R}^m$ and $\mathbf{Y} = \mathbf{A}\mathbf{X} + \mathbf{b}$. Then $\mathbf{Y} \sim \mathrm{N}_m(\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T)$.

3. Let $X_1 \sim \mathrm{N}(\mu_1, \sigma_1^2)$ and $X_2 \sim \mathrm{N}(\mu_2, \sigma_2^2)$. Then X_1 and X_2 are independent if and only if $\mathrm{Cov}(X_1, X_2) = 0$.

3 Discrete Mixtures of Multivariate Normal Distributions

Suppose we have a sample of multivariate data $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ from a population $\mathcal{G}$ that we suspect can be partitioned into K mutually exclusive subsets, i.e.

$$\mathcal{G}_1 \cup \mathcal{G}_2 \cup \ldots \cup \mathcal{G}_K = \mathcal{G} \text{ and } \mathcal{G}_i \cap \mathcal{G}_j = \emptyset, \forall i \neq j, \text{ for } i, j = 1, 2, \ldots, K$$

A natural choice for modelling our data is to assume that the $\mathbf{x}_i$ are distributed as a discrete mixture of multivariate normal random variables. That is, we assume that within each subset $\mathcal{G}_i$, the population follows a p-variate normal distribution with mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$.

Let $\mathbf{X}$ be a random vector with pdf,

$$f_{\mathbf{X}}(\mathbf{x}; \boldsymbol{\theta}) = \sum_{i=1}^{K} \pi_i f_i(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \ \mathbf{x} \in \mathbb{R}^p, \ 0 < \pi_i < 1$$

where $\boldsymbol{\theta} = [\pi_1, \ldots, \pi_K, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_K]^T$, f_i is the pdf of a p-variate multinomial distribution with mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ and $\sum_{i=1}^{K} \pi_i = 1$. Then we say that $\mathbf{X}$ has a distribution from a discrete mixture of multivariate normals.

We will use maximum likelihood theory to find estimators for the parameters. Note that the likelihood function for a sample $\mathbf{x}_1, \ldots, \mathbf{x}_n$ is

$$L(\mathbf{x}; \boldsymbol{\theta}) = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

and the log-likelihood is:

$$\log L(\mathbf{x}; \boldsymbol{\theta}) = \sum_{i=1}^{n} \log \left[\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right]$$

Although it is possible to maximize the log-likelihood with respect to all the parameters directly, it is a difficult optimization problem. We will simplify the required optimization by breaking it into simpler problems that we will solve iteratively to obtain a solution. This process is referred to as the EM-algorithm (McLachlan, 2000).

3.1 The EM Algorithm

Recall that we assumed our population can be partitioned into K distinct subsets. Suppose that for each data point $\mathbf{X}_i$ we knew exactly from which of the subsets of $\mathcal{G}$ that $\mathbf{X}_i$ belonged. Then we could simply divide our sample into K groups, and perform maximum likelihood estimation on each of the K subsets of our sample where we assume that each subset has a p-variate normal distribution. Let us define the matrix $\mathbf{G} = \{g_{ik}\}_{n\times K}$ such that

$$g_{ik} = \begin{cases} 1 & , \text{ if the } i^{th} \text{ sample point comes from subset } k \text{ of the population} \\ 0 & , \text{ otherwise} \end{cases}$$

for $i = 1, 2, \ldots, n$, and $k = 1, 2, \ldots, K$.

If we knew the group memberships, then our likelihood function simplifies to

$$L(\mathbf{x};\boldsymbol{\theta}) = \prod_{i=1}^{n}\prod_{k=1}^{K}(f_k(\mathbf{x}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))^{g_{ik}}$$

where g_{ik} denotes the sample values for the group memberships. The log-likelihood is then

$$\begin{aligned} \log L(\mathbf{x};\boldsymbol{\theta}) &= \log\left[\prod_{i=1}^{n}\prod_{k=1}^{K}(f_k(\mathbf{x_i};\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))^{g_{ik}}\right] \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K} g_{ik}\log\left[f_k(\mathbf{x_i};\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right]. \end{aligned}$$

Now we can maximize the log-likelihood with respect to $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$, for $k = 1, 2, \ldots, K$. Note that:

$$\frac{\partial}{\partial\boldsymbol{\mu}_k}\log L(\mathbf{x};\boldsymbol{\theta}) = \sum_{i=1}^{n}\frac{\partial}{\partial\boldsymbol{\mu}_k} g_{ik}\log(f_k(\mathbf{x}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))$$

since only the k^{th} term in the mixture expression contains $\boldsymbol{\mu}_k$. Therefore, setting this derivative equal to zero we get

$$\begin{aligned}
\sum_{i=1}^{n} \frac{\partial}{\partial \boldsymbol{\mu}_k} g_{ik} \log(f_k(\mathbf{x_i}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)) &= 0 \\
\sum_{i=1}^{n} g_{ik} \left[-\frac{1}{2} \left(-2\boldsymbol{\Sigma}^{-1} \left(\mathbf{x}_i - \boldsymbol{\mu}_k \right) \right) \right] &= 0 \\
\boldsymbol{\Sigma}^{-1} \sum_{i=1}^{n} g_{ik} \left(\mathbf{x}_i - \boldsymbol{\mu}_k \right) &= 0 \\
\sum_{i=1}^{n} g_{ik} \mathbf{x}_i &= \boldsymbol{\mu}_k \sum_{i=1}^{n} g_{ik}
\end{aligned}$$

which gives

$$\hat{\boldsymbol{\mu}}_k = \frac{\sum_{i=1}^{n} g_{ik} \mathbf{x}_i}{\sum_{i=1}^{n} g_{ik}} \tag{1}$$

as the maximum likelihood estimator of $\boldsymbol{\mu}_k$; if the group memberships g_{ik} were known. Similarly, if we let $\mathbf{V}_k = \boldsymbol{\Sigma}_k^{-1}$ and note that

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{V}_k} \left[\frac{1}{2} \sum_{i=1}^{n} g_{ik} \log \left(|\mathbf{V}_k| \right) \right] &= \frac{1}{2} \sum_{i=1}^{n} g_{ik} \left(\mathbf{V}_k^{-1} \right)^T \\
&= \frac{1}{2} \mathbf{V}_k^{-1} \sum_{i=1}^{n} g_{ik}
\end{aligned}$$

we get

$$\frac{\partial}{\partial \mathbf{V}_k} \log L(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{2} \mathbf{V}_k^{-1} \sum_{i=1}^{n} g_{ik} - \frac{1}{2} \sum_{i=1}^{n} g_{ik} \left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k \right) \left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k \right)^T$$

Note that $\mathbf{V}_k^{-1} = \boldsymbol{\Sigma}_k$, thus setting the derivative equal to zero, we get

$$\hat{\boldsymbol{\Sigma}}_k = \frac{\sum_{i=1}^{n} g_{ik} \left(\mathbf{x}_i - \boldsymbol{\mu}_k \right) \left(\mathbf{x}_i - \boldsymbol{\mu}_k \right)^T}{\sum_{i=1}^{n} g_{ik}} \tag{2}$$

as the maximum likelihood estimator for $\mathbf{\Sigma}_k$.

We do not know the true group memberships but a reasonable assumption is that the rows of $\mathbf{G}$ are iid from a multinomial distribution, since each row contains a single 1 and $K-1$ zeroes. Let $\mathbf{G}_i$ denote the i^{th} row of $\mathbf{G}$, for $i = 1, 2, \ldots, n$. Then if we assume that

$$\mathbf{G}_i \sim \text{Multinomial}(1, \boldsymbol{\pi}), \text{ where } 0 < \pi_k < 1, \text{ for } k = 1, 2, \ldots, K$$

where $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_K]^T$ and $\sum_{j=1}^{K} \pi_k = 1$. The likelihood function can be written as

$$L(\mathbf{x}; \boldsymbol{\theta}) = \prod_{i=1}^{n} \left(\prod_{k=1}^{K} (f_k(\mathbf{x}_i; \boldsymbol{\mu}_k, \mathbf{\Sigma}_k))^{g_{ik}} \right) f(\mathbf{G}_i).$$

We can maximize the log-likelihood function for $\boldsymbol{\pi}$ as

$$\begin{aligned}
\frac{\partial}{\partial \pi_k} \log L(\mathbf{x}; \boldsymbol{\theta}) &= \frac{\partial}{\partial \pi_k} \left[\sum_{i=1}^{n} \sum_{k=1}^{K} \log\left(f(g_{ik})\right) \right] \\
&= \frac{\partial}{\partial \pi_k} \left[\sum_{i=1}^{n} \log\left(\pi_1^{g_{i1}} \pi_2^{g_{i2}} \ldots \pi_K^{g_{iK}}\right) \right] \\
&= \frac{\partial}{\partial \pi_k} \left[\sum_{i=1}^{n} \log\left(\pi_k^{g_{ik}} (1-\pi_k)^{1-g_{ik}}\right) \right] \\
&= \frac{\partial}{\partial \pi_k} \left[\sum_{i=1}^{n} g_{ik} \log(\pi_k) + (1-g_{ik}) \log(1-\pi_k) \right] \\
&= \sum_{i=1}^{n} \frac{g_{ik}}{\pi_k} - \frac{1-g_{ik}}{1-\pi_k}
\end{aligned}$$

Set this derivative equal to zero and solve to get

$$\hat{\pi}_k = \frac{1}{n} \sum_{i=1}^{n} g_{ik} \tag{3}$$

the maximum likelihood estimator for π_k, $k = 1, 2, \ldots, K$.

We now have estimators for $\boldsymbol{\pi}, \boldsymbol{\mu}_k, \mathbf{\Sigma}_k$ under the assumption we actually knew the group memberships. Since we do not know the group memberships, we will estimate them with their expectation given our sample data. Note that

$$\begin{aligned} \mathrm{E}\left[g_{ik} \mid \mathbf{x}\right] &= 0 \cdot \mathrm{P}(g_{ik} = 0 \mid \mathbf{x}) + 1 \cdot \mathrm{P}(g_{ik} = 1 \mid \mathbf{x}) \\ &= \mathrm{P}(g_{ik} = 1 \mid \mathbf{x}) \end{aligned}$$

so by Bayes' Rule and the definition of conditional probability we have

$$\mathrm{P}(g_{ik} = 1 \mid \mathbf{x}, \boldsymbol{\theta}) = \frac{f_{\mathbf{x}|g_{ik}}\left(\mathbf{x}_i \mid g_{ik} = 1\right) \cdot \mathrm{P}\left(g_{ik} = 1\right)}{f_{\mathbf{x}}(\mathbf{x}_i)}. \tag{4}$$

Since we assumed that the group memberships followed a multinomial distribution, then $\mathrm{P}(G_{ik} = 1) = \pi_k$. Furthermore, the denominator of (4) is simply the pdf of our discrete mixture model. Finally, note that

$$f_{\mathbf{x}|g_{ik}}\left(\mathbf{x}_i \mid g_{ik} = 1\right) = f_k(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

since when $g_{ik} = 1$, we know that $\mathbf{x}_i$ comes from the k^{th} subgroup of the population. Thus we can compute the expected value of the group memberships as

$$\mathrm{E}\left[g_{ik} \mid \mathbf{x}\right] = \frac{\pi_k f_k(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j f_j(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{5}$$

We can use (1), (2), (3), and (5) to construct the following iterative algorithm:

1. Initialize $\tilde{\mathbf{G}}$.

2. For $k = 1, \ldots, K$, set

$$\begin{aligned} \hat{\pi}_k &= \frac{1}{n} \sum_{i=1}^{n} \tilde{g}_{ik} \\ \hat{\boldsymbol{\mu}}_k &= \frac{\sum_{i=1}^{n} \tilde{g}_{ik} \mathbf{x}_i}{\sum_{i=1}^{n} \tilde{g}_{ik}} \\ \hat{\boldsymbol{\Sigma}}_k &= \frac{\sum_{i=1}^{n} \tilde{g}_{ik} \left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k\right) \left(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k\right)^T}{\sum_{i=1}^{n} \tilde{g}_{ik}} \end{aligned}$$

3. Update $\tilde{\mathbf{G}}$. For each $i = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, K$, set

$$\tilde{g}_{ik} = \frac{\hat{\pi}_k f_k(\mathbf{x}_i; \hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k)}{\sum_{j=1}^{K} \hat{\pi}_j f_j(\mathbf{x}_i; \hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j)}$$

4. Repeat steps 2 and 3 until the change in the log-likelihood between iterations is less than some $\epsilon > 0$.

The above iterative algorithm is known as the EM algorithm, since there are two major steps. The "E" step where we compute the expected value of the data missing from our likelihood (in our case the group membership), and the "M" step where we maximize our likelihood using the estimated missing data. It can be shown (Hogg, 2005) that the likelihood increases after each iteration. Therefore, we can conclude that the EM algorithm will converge to some local maximum of the log-likelihood function. We can be fairly confident that if we select initial values for our group membership that are sufficiently close to the true group membership, our algorithm will converge to the true global maximum of the log-likelihood function.

4 Nonparametric Regression Analysis and Spline Smoothing

This chapter reviews well-established concepts relating to nonparametric regression analysis and in particular the application of smoothing splines. The topics discussed are a compendium of facts from Hastie, Tibshirani, and Friedman (2001), Eubank (1999), Ramsay and Silverman (1997) and Eilers and Marx (1996).

4.1 Regression Analysis

In regression analysis, we generally believe there is a relation between a response variable Y and a number of recorded variables $\mathbf{x} = (x_1, x_2, \ldots, x_k)$. After obtaining n samples of data $(Y_i, \mathbf{x}_i)$ $i = 1, 2, \ldots, n$, we propose the following model:

$$Y_i = f(x_i) + \epsilon(x_i),\ i = 1, 2, \ldots, n$$

where f is an unknown function and $\epsilon(X_i)$ is a random error term.

Suppose we assume $\epsilon(x_i)$ to be identically and independently distributed with mean 0 and variance σ^2 for each x_i, which for notational convience will be denoted simply as ϵ_i. Then we can see that,

$$\begin{aligned} \mathrm{E}\left[Y_i \mid x_i\right] &= \mathrm{E}\left[f(x_i) + \epsilon_i \mid x_i\right] \\ &= f(x_i). \end{aligned}$$

In parametric regression, f has the form

$$f = f(x; \theta_1, \theta_2, \ldots, \theta_k) = f(x; \boldsymbol{\theta})$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)^T$, and we typically seek the value for $\boldsymbol{\theta}$ which minimizes the *residual sum of squares*:

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(Y_i - f(X_i, \boldsymbol{\theta})\right)^2 .$$

In the nonparametric regression case, we only restrict f to a family of functions that display a particular property, such as continuity, concavity, monotonicity, etc. One approach to modelling f is to assume that it belongs to a vector space spanned by a set of basis functions. That is, we can represent f in such a space as,

$$f(x) = \sum_{i=1}^{k} a_i \phi_i(x)$$

where $\{\phi_1(x), \phi_2(x), \ldots, \phi_k(x)\}$ are the k basis functions that span the linear space in question.

4.2 Basis-Expansion Regression

Suppose we take the following class of functions:

$$P_d = \left\{ f \mid f(x) = \sum_{i=0}^{d} \psi_i x^i, \psi_i \in \mathbb{R}, x \in \mathbb{R}, d \geq 0 \right\}$$

ie, the class of polynomials of degree d. Then P_d is clearly a linear space spanned by:

$$\{1, x, x^2, \ldots, x^d\}$$

since any polynomial of degree d can be expressed as a linear combination of these basis functions.

Suppose we let $f \in P_d$, so that f can be written as,

$$f(x) = \sum_{i=0}^{d} \psi_i x^i, x \in \mathbb{R}.$$

Since f is linear in the basis coefficients $\psi_i's$, then we can estimate

$$\hat{f}(x) = \sum_{i=0}^{d} \hat{\psi}_i x^i$$

by least squares, where $\hat{\psi}_i, i = 0, 1, \ldots, d$ are the values of ψ_i which minimize the RSS. For a sample $x_1, x_2, \ldots, x_n$, $\hat{\boldsymbol{\psi}} = \left[\begin{array}{ccc} \hat{\psi}_0 & \ldots & \hat{\psi}_d \end{array} \right]^T$ can be obatined with multiple linear regression, since f is linear in the parameters $\boldsymbol{\psi}$. If we let

$$\mathbf{X} = \left[\begin{array}{ccccc} 1 & x_1 & x_1^2 & \ldots & x_1^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^d \end{array} \right]$$

be the design matrix and we define $\mathbf{f} = \left[\begin{array}{ccc} f(x_1) & \ldots & f(x_n) \end{array} \right]^T$, then in matrix notation, we can write the RSS as:

$$\begin{aligned} RSS &= (\mathbf{Y} - \mathbf{f})^T (\mathbf{Y} - \mathbf{f}) \\ &= (\mathbf{Y} - \mathbf{X}\boldsymbol{\psi})^T (\mathbf{Y} - \mathbf{X}\boldsymbol{\psi}) \end{aligned}$$

where $\mathbf{Y} = \begin{bmatrix} Y_1 & \dots & Y_n \end{bmatrix}^T$.

To minimize the RSS with respect to $\boldsymbol{\psi}$ we take the derivative and set it equal to zero:

$$\begin{aligned} \frac{\partial RSS}{\partial \boldsymbol{\psi}} &= -2\mathbf{X}^T\left(\mathbf{Y} - \mathbf{X}\boldsymbol{\psi}\right) \\ &= -2\mathbf{X}^T\mathbf{Y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\psi} \\ \Rightarrow \mathbf{X}^T\mathbf{X}\boldsymbol{\psi} &= \mathbf{X}^T\mathbf{Y}. \end{aligned}$$

This gives us the least squares estimate for $\boldsymbol{\psi}$ as

$$\hat{\boldsymbol{\psi}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{Y} \tag{6}$$

so that

$$\begin{aligned} \hat{\mathbf{f}} &= \mathbf{X}\hat{\boldsymbol{\psi}} \\ &= \mathbf{X}\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{Y} \\ &= \mathbf{H}\mathbf{Y} \end{aligned}$$

where $\mathbf{H} = \mathbf{X}\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T$. $\mathbf{H}$ is called the projection matrix (or hat matrix) since $\mathbf{H}$ depends only on $\mathbf{X}$ and projects the observed $\mathbf{Y}$ to the fitted $\hat{\mathbf{Y}} = \hat{\mathbf{f}}$.

We have shown that for any function $f(x)$ we can fit a linear combination of the polynomial basis functions to the data by least-squares. The problem we have is selecting the value of d. If we choose d to be small, then the fit will not be very flexible and our fitted function may miss important features in the data. If we select d too large, we begin to interpolate the data and start to ignore the random error term, thus overfitting our data. Furthermore, the polynomial space may not be flexible enough to fit the entire class of functions we wish to model. In particular, when our function changes from being “wobbly” to being “flat” within our interval, polynomials will perform poorly at fitting such a function. In Figure 2, the black line denotes the true function $f(x)$. The points have been generated as normal random variables with

$$Y_i = f(x_i) + \epsilon_i, i = 1, 2, \ldots, n$$

where $\epsilon_i \sim \mathrm{N}(0, 1/16)$. The blue and red lines denote best-fitting polynomials of degree 3 and 12 respectively. Note that while the polynomials may perform well over certain intervals, neither fitted polynomial is able to model the general trend in the data over the entire interval.

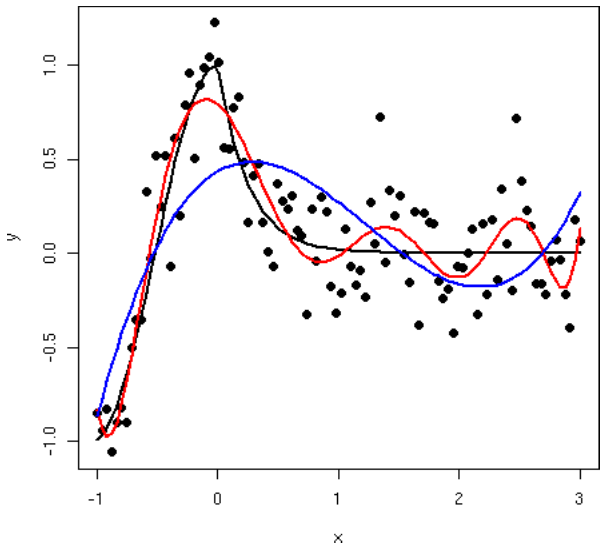

Figure 2: Least squares polynomial regression.

To remedy the problem we have with the polynomial space, we note that our solution to the multiple linear regression problem does not depend on a particular set of basis functions. We can define a more general vector space spanned by the basis functions

$$\{u_1(x), u_2(x), \ldots, u_K(x)\}$$

so that again our f can be written as

$$f(x) = \sum_{k=1}^{K} \psi_k u_k(x).$$

For a sample $x_1, \ldots, x_n$, the design matrix is

$$\mathbf{X} = \begin{bmatrix} u_1(x_1) & u_2(x_1) & \ldots & u_k(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ u_1(x_n) & u_2(x_n) & \ldots & u_k(x_n) \end{bmatrix}$$

and our solution for $\hat{\boldsymbol{\psi}}$ remains the same as in (6).

We will now propose a linear space that will give us a much more flexible family of functions.

4.3 Regression Splines

Suppose we select k points within the interval and divide our interval into $k+1$ sub-intervals, fitting our data piece-wise, using polynomials. Let $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_k)^T$ be the set of points from which we divide our interval. In the spline literature, these break-points are called knots (DeBoor, 1978). Suppose our interval is defined on the closed set of real points $[a, b]$, then if $a = \gamma_0 < \gamma_1 < \ldots < \gamma_k < \gamma_{k+1} = b$ and d is some positive integer, then the *spline space* is defined as:

$$S_{d,\boldsymbol{\gamma}} = \{f \in C_{d-1}(a,b) \mid f(x) \in P_d \text{ for } x \in [\gamma_i, \gamma_{i+1}), i = 0, 1, \ldots, k\}$$

where $C_{d-1}(a, b)$ is the set of functions with the $(d-1)^{th}$ continuous derivative over the interval $[a, b]$.

Note that to construct a basis for the spline space, we must select d and a set of knots $\boldsymbol{\gamma}$. Since continuity over the whole interval is required, a simple polynomial of degree d fitted over each sub interval will be insufficient since these polynomials will be discontinuous at the knots. We can remedy this problem by requiring the derivatives to match at the γ_i's. One possible basis for $S_{d,\boldsymbol{\gamma}}$ is the truncated power basis, defined as

$$\left\{1, x, x^2, \ldots, x^d, (x-\gamma_1)_+^d, \ldots, (x-\gamma_k)_+^d\right\}$$

where

$$(x-\gamma)_+ = \begin{cases} x-\gamma & ,x \geq \gamma \\ 0 & \text{, otherwise} \end{cases} .$$

Once we have selected d and the location and number of knots, we can then fit a *regression spline* to the data in an identical manner as multiple linear regression; provided that the $d+k+1$ basis functions spanning the spline space is less than n.

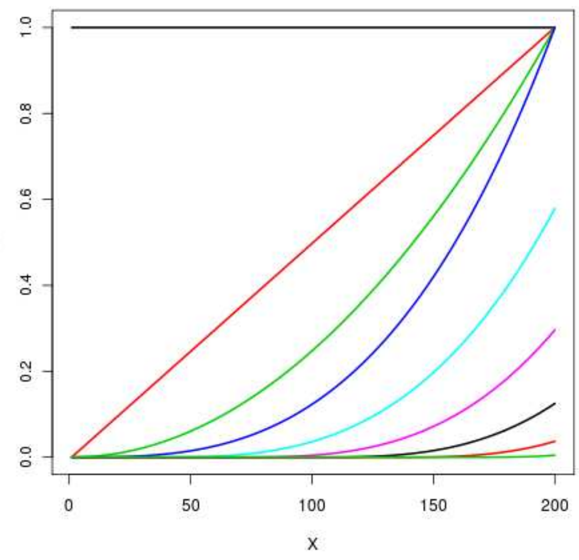

Figure 3: Cubic truncated power basis functions with 5 knots.

When using regression splines, we typically set $d = 3$, since this gives us a smooth function with continuous 2^{nd} derivatives. Figure 3 contains a plot of cubic truncated power basis functions with 5 knots, equally spaced over the interval of interest. Note as any two knots move closer together, their associated basis functions move closer to becoming linearly dependent. In such a sceniario, the truncated power basis will have poor numerical properties. We will discuss a more numerically stable basis for $S_{d,\gamma}$ in section 4.7.

While regression splines are flexible in being able to fit a large class of functions, the problem of selecting the number of knots and their location is difficult and has a strong impact on the resulting estimator. Much like fitting a polynomial globally, by selecting too many knots, we risk over fitting the data, and too few knots may not be flexible enough. The blue curve in Figure 4 is a regression spline fit with a single knot at 0.5, while the red curve is a regression spline with 9 equally spaced knots over the interval. Note that our estimator is highly dependent on the knot selection. Since selecting the number of knots and their position is very challenging, a different approach is required.

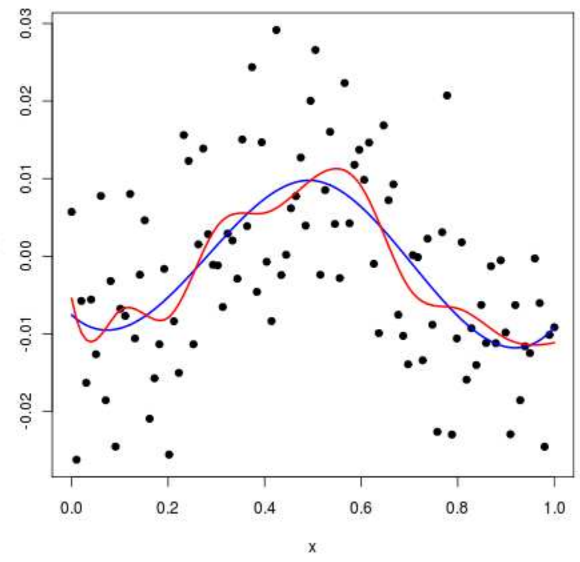

Figure 4: Regression splines fitted with differing knots.

4.4 Smoothing Splines

Instead of selecting the position and number of knots, suppose we take our basis-expansion method and place a knot at each data point. To compensate for over-fitting the data, we add a penalty on the 2^{nd} derivative, and minimize the *penalized residual sum of squares*:

$$\mathrm{PRSS}(\lambda) = \sum_{i=1}^{n} (Y_i - f(X_i))^2 + \lambda \int_a^b [f''(t)]^2 \, dt$$

That is, for a given value of the *smoothing parameter* λ, we balance a trade-off between fitting the data and fidelity to our smoothness assumption in the function. Some things to note:

- Although we are typically only interested in penalizing the 2^{nd} derivative of f, we can penalize any higher order derivative as well (for example, if we believe f lives in a family of smoother functions).
- As $\lambda \to 0$, then we will be fitting an interpolating spline function to the data, since we have placed a knot at each data point.
- As $\lambda \to \infty$, then we will be fitting a simple line to the data (linear regression), since we are forcing $f''(x) \to 0$.

Remarkably, a closed-form finite-dimensional solution exists to minimizing the PRSS(λ) (Wahba, 1990). The solution is a *natural cubic spline* with knots at each data point $x_i, i = 1, 2, \ldots, n$. The space for a natural cubic spline is spanned by n functions, defined as:

$$N_1(x) = 1, N_2(x) = x, N_{k+2}(x) = d_k(x) - d_{n-1}(x)$$

where

$$d_k(x) = \frac{(x - \gamma_k)_+^3 - (x - \gamma_n)_+^3}{\gamma_n - \gamma_k}.$$

So if we assume that f is a member of this space, we can express f as

$$f(x) = \sum_{i=1}^{n} \psi_i N_i(x)$$

where $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_K)$ are the coefficients of the natural cubic spline expansion for $x_1, \ldots, x_n$. We can construct a design matrix

$$\mathbf{N} = \begin{bmatrix} N_1(x_1) & \ldots & N_n(x_1) \\ \vdots & \ddots & \vdots \\ N_1(x_n) & \ldots & N_n(x_n) \end{bmatrix}$$

and $\mathbf{P} = \{p_{ij}\}$ a $n \times n$ penalty matrix where

$$p_{ij} = \int_a^b N_i''(x) N_j''(x) dx, \ i = 1, \ldots, n, \ j = 1, \ldots, n.$$

In matrix notation we can then express the penalized sum of squares as:

$$\begin{aligned} \text{PRSS}(\lambda) &= \sum_{i=1}^{n} (Y_i - f(X_i))^2 + \lambda \int_a^b [f''(t)]^2 \, dt \\ &= (\mathbf{Y} - \mathbf{N}\boldsymbol{\psi})^T (\mathbf{Y} - \mathbf{N}\boldsymbol{\psi}) + \lambda \boldsymbol{\psi}^T \mathbf{P} \boldsymbol{\psi}. \end{aligned}$$

An estimator for $\boldsymbol{\psi}$ can then be found by taking the derivative of PRSS(λ) with respect to $\boldsymbol{\psi}$:

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\psi}} \text{PRSS}(\lambda) &= -2\mathbf{N}^T (\mathbf{Y} - \mathbf{N}\boldsymbol{\psi}) + 2\lambda \mathbf{P}\boldsymbol{\psi} \\ &= -2 \left(\mathbf{N}^T \mathbf{Y} - \mathbf{N}^T \mathbf{N} \boldsymbol{\psi} - \lambda \mathbf{P} \boldsymbol{\psi} \right) \\ &= -2 \left(\mathbf{N}^T \mathbf{Y} - \left(\mathbf{N}^T \mathbf{N} + \lambda \mathbf{P} \right) \boldsymbol{\psi} \right) \end{aligned}$$

and setting this equal to zero:

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\psi}} \text{PRSS}(\lambda) &= 0 \\ -2 \left(\mathbf{N}^T \mathbf{Y} - \left(\mathbf{N}^T \mathbf{N} + \lambda \mathbf{P} \right) \boldsymbol{\psi} \right) &= 0 \\ \left(\mathbf{N}^T \mathbf{N} + \lambda \mathbf{P} \right) \boldsymbol{\psi} &= \mathbf{N}^T \mathbf{Y} \\ \boldsymbol{\psi} &= \left(\mathbf{N}^T \mathbf{N} + \lambda \mathbf{P} \right)^{-1} \mathbf{N}^T \mathbf{Y}. \end{aligned}$$

Thus we can see that for fixed λ, $\hat{\boldsymbol{\psi}} = \left(\mathbf{N}^T\mathbf{N} + \lambda\mathbf{P}\right)^{-1}\mathbf{N}^T\mathbf{Y}$ is the solution to minimizing the penalized residual sum of squares. Our estimator is then given by

$$\begin{aligned}\hat{\mathbf{f}} &= \mathbf{N}\hat{\boldsymbol{\psi}} \\ &= \mathbf{N}\left(\mathbf{N}^T\mathbf{N} + \lambda\mathbf{P}\right)^{-1}\mathbf{N}^T\mathbf{Y} \\ &= \mathbf{S}(\lambda)\mathbf{Y}\end{aligned}$$

where $\mathbf{S}(\lambda) = \mathbf{N}\left(\mathbf{N}^T\mathbf{N} + \lambda\mathbf{P}\right)^{-1}\mathbf{N}^T$ is called the *smoothing matrix* or *smoothing operator*.

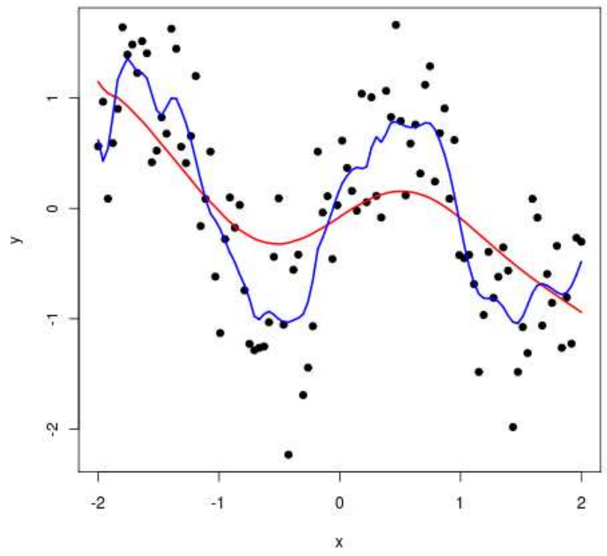

Figure 5: Smoothing Splines with two different smoothing parameters.

Figure 5 displays two smoothing splines with different smoothing parameters. For the red curve, $\lambda = 1$ gives a very smooth function but fails to detect the oscillation in the data. The blue curve has $\lambda = 10^{-3}$ and may be overfitting the data, and so selecting λ will be important for a good fit. We will discuss this further in section 4.5.

The smoothing operator plays an important role in nonparametric regression and has the following properties:

- $\mathbf{S}(\lambda)$ depends only on the parameter λ, since the x's are assumed fixed and known.
- $\mathbf{S}(\lambda)$ is symmetric and positive semi-definite.
- Recall that for multiple linear regression our estimated function was of a similar form, with the hat matrix $\mathbf{H}$ replacing the smoothing operator $\mathbf{S}(\lambda)$. Unlike the hat matrix in linear regression, the smoothing matrix is not idempotent, but rather:

  $$\mathbf{S}(\lambda)\mathbf{S}(\lambda) = \mathbf{S}(\lambda) + \mathbf{R}$$

  where $\mathbf{R}$ is a positive semi-definite matrix. This results in the smoothing/shrinkage effect that $\mathbf{S}(\lambda)$ performs on the $\mathbf{Y}$'s.
- Note that since $\mathbf{H}$ is idempotent,

  $$\begin{aligned} \text{trace}(\mathbf{H}) &= \text{trace}\left(\mathbf{X}\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\right) \\ &= \text{trace}\left(\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\left(\mathbf{X}^T\mathbf{X}\right)\right) \\ &= \text{trace}\left(\mathbf{I}_{K\times K}\right) \\ &= K. \end{aligned}$$

  Thus, the trace of the hat matrix is the number of parameters or degrees of freedom in multiple linear regression. In the smoothing spline context, we define the *effective degrees of freedom* to be

  $$\text{df}(\lambda) = \text{trace}(\mathbf{S}(\lambda))$$

  where $\text{df}(\lambda)$ need not be integer-valued.

4.5 Selecting the Smoothing Parameter

Suppose $\hat{f}(x)$ is an estimator of the true function $f(x)$. Then we can define the *loss function* as

$$L(f(x), \hat{f}(x)) = (f(x) - \hat{f}(x))^2$$

the squared error loss function. The loss function measures the inaccuracy of our fitted function to the true f at any particular point x. Define the *risk function* as:

$$\begin{aligned}
R(f(x), \hat{f}(x)) &= \mathrm{E}\left[L(f(x), \hat{f}(x))\right] \\
&= \mathrm{E}\left[(f(x) - \hat{f}(x))^2\right] \\
&= \mathrm{E}\left[\left(f(x) - \mathrm{E}\left[\hat{f}(x)\right] + \mathrm{E}\left[\hat{f}(x)\right] - \hat{f}(x)\right)^2\right] \\
&= \mathrm{E}\left[\left(\left(f(x) - \mathrm{E}\left[\hat{f}(x)\right]\right) + \left(\hat{f}(x) - \mathrm{E}\left[\hat{f}(x)\right]\right)\right)^2\right] \\
&= \mathrm{E}\left[\left(f(x) - \mathrm{E}\left[\hat{f}(x)\right]\right)^2\right] + \mathrm{E}\left[\left(\hat{f}(x) - \mathrm{E}\left[\hat{f}(x)\right]\right)^2\right] \\
&= \left(\mathrm{bias}(\hat{f}(x))\right)^2 + \mathrm{Var}\left(\hat{f}(x)\right)
\end{aligned}$$

Note, that by estimating $f(x)$ with $\hat{f}(x)$ at any particular x, our risk consists of the sum of two non-negative terms, namely the squared bias and the variance of the estimator $\hat{f}(x)$ at x. We wish to find an estimator $\hat{f}$ of f which attempts to minimize $R(f(x), \hat{f}(x))$ over all x. To do this, we will use the *integrated risk function*

$$R(f, \hat{f}) = \int_x R(f(x), \hat{f}(x))dx.$$

For a sample $x_1, \ldots, x_n$, a natural estimator for $R(f, \hat{f})$ is

$$\hat{R}(f, \hat{f}) = \frac{1}{n}\sum_{i=1}^{n} R(f(x_i), \hat{f}(x_i)).$$

We wish to find the optimal value for λ which minimizes the integrated risk function. Unfortunately, we typically do not know the true function $f(x)$, thus computing the bias is impossible.

Since we cannot compute the bias (due to the true f being unknown), a good statistic for estimating the integrated risk function is the *leave-one-out cross-validation* statistic:

$$\mathrm{CV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{f}^{(-i)}(X_i)\right)^2$$

where $\hat{f}^{(-i)}$ denotes the estimator of f with the data point (Y_i, x_i) removed. Note that it can be shown that we can equivalently express $\mathrm{CV}(\lambda)$ as

$$\begin{aligned}\mathrm{CV}(\lambda) &= \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{f}^{(-i)}(X_i)\right)^2 \\ &= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{f}(X_i)}{1 - S_{ii}(\lambda)}\right)^2 \qquad (7)\end{aligned}$$

where $S_{ii}(\lambda)$ is the i^{th} diagonal element of the smoothing matrix $\mathbf{S}(\lambda)$. Using (7) is preferred since it does not require fitting the smoothing spline to the data n times; only once which greatly improves computational efficiency.

We can replace the diagonal elements of $\mathbf{S}(\lambda)$ with the average $\mathrm{df}(\lambda)/n$. We call this statistic the *generalized cross-validation* statistic and is computed as:

$$\begin{aligned}\mathrm{GCV}(\lambda) &= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}(x_i)}{1 - \frac{\mathrm{df}(\lambda)}{n}}\right)^2 \\ &= \frac{n}{(n - \mathrm{df}(\lambda))^2}\sum_{i=1}^{n}\left(y_i - \hat{f}(x_i)\right)^2\end{aligned}$$

Figure 6 shows the same data as Figure 5, but the green curve is a smoothing spline for which we select λ by minimizing $\mathrm{GCV}(\lambda)$. Note that the "optimal" smoothing parameter gives a much better estimated function than

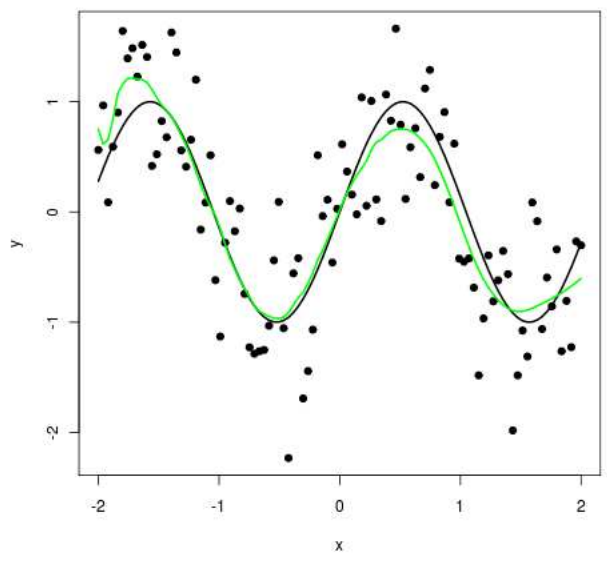

Figure 6: Smoothing Splines with the "optimal" value for λ selected using GCV.

those we selected for Figure 5. The black curve denotes the true function from which we generated the data.

We can find an "optimal" value for λ by either minimizing GCV(λ) or CV(λ). Note that in the generalized cross-validation statistic we need only the effective degrees of freedom (ie the trace of the smoother matrix) and not the individual diagonal elements of the smoothing matrix. In the next section, we will change our basis to an orthogonal basis, which will help demonstrate the effects of the smoothing matrix, as well as give us improved computational efficiency.

4.6 Orthogonalization of the Spline Space Basis

We will try to further examine what is happening when we apply the smoothing matrix $\mathbf{S}(\lambda)$ to our sample data. Recall that $\hat{\mathbf{f}}$ can be expressed as:

$$\begin{aligned} \hat{\mathbf{f}} &= \mathbf{S}(\lambda)\mathbf{Y} \\ &= \mathbf{N}\left(\mathbf{N}^T\mathbf{N} + \lambda\mathbf{P}\right)^{-1}\mathbf{N}^T\mathbf{Y} \end{aligned} \tag{8}$$

Suppose we set $\mathbf{N}^T\mathbf{N} = \mathbf{R}^T\mathbf{R}$ where $\mathbf{R}$ is an upper-right triangular matrix obtained by a Choleski decomposition. Furthermore, let:

$$\mathbf{G} = \mathbf{R}^{-1^T}\mathbf{P}\mathbf{R}^{-1} = \mathbf{E}\mathbf{D}\mathbf{E}^T$$

where $\mathbf{D}$ is a diagonal matrix with entries d_i, $i = 1, 2, \ldots, n$, and the columns of $\mathbf{E}$ are the eigenvectors of $\mathbf{G}$, obtained by a spectral decomposition. Then we can see that

$$\mathbf{P} = \mathbf{R}^T\mathbf{E}\mathbf{D}\mathbf{E}^T\mathbf{R}$$

and hence can express (8) as

$$\begin{aligned}
\hat{\mathbf{f}} &= \mathbf{N}\left(\mathbf{N}^T\mathbf{N}+\lambda\mathbf{P}\right)^{-1}\mathbf{N}^T\mathbf{Y}\\
&= \mathbf{N}\left(\mathbf{R}^T\mathbf{R}+\lambda\mathbf{R}^T\mathbf{E}\mathbf{D}\mathbf{E}^T\mathbf{R}\right)^{-1}\mathbf{N}^T\mathbf{Y}\\
&= \mathbf{N}\left(\mathbf{R}^T\mathbf{E}\mathbf{E}^T\mathbf{R}+\lambda\mathbf{R}^T\mathbf{E}\mathbf{D}\mathbf{E}^T\mathbf{R}\right)^{-1}\mathbf{N}^T\mathbf{Y}\\
&= \mathbf{N}\left(\mathbf{R}^T\mathbf{E}\left(\mathbf{I}+\lambda\mathbf{D}\right)\mathbf{E}^T\mathbf{R}\right)^{-1}\mathbf{N}^T\mathbf{Y}\\
&= \mathbf{N}\left(\mathbf{E}^T\mathbf{R}\right)^{-1}\left(\mathbf{I}+\lambda\mathbf{D}\right)^{-1}\left(\mathbf{R}^T\mathbf{E}\right)^{-1}\mathbf{N}^T\mathbf{Y}\\
&= \left(\mathbf{N}\mathbf{R}^{-1}\mathbf{E}\right)\left(\mathbf{I}+\lambda\mathbf{D}\right)^{-1}\left(\mathbf{N}\mathbf{R}^{-1}\mathbf{E}\right)^T\mathbf{Y}\\
&= \mathbf{U}\left(\mathbf{I}+\lambda\mathbf{D}\right)^{-1}\mathbf{U}^T\mathbf{Y}
\end{aligned}$$

where $\mathbf{U}=\mathbf{N}\mathbf{R}^{-1}\mathbf{E}$. Note that we have re-expressed our smoothing matrix as

$$\begin{aligned}
\mathbf{S}(\lambda) &= \mathbf{N}\left(\mathbf{N}^T\mathbf{N}+\lambda\mathbf{P}\right)^{-1}\mathbf{N}^T\\
&= \mathbf{U}\left(\mathbf{I}+\lambda\mathbf{D}\right)^{-1}\mathbf{U}^T
\end{aligned}$$

We can interpret the columns of $\mathbf{U}$ as a new set of basis functions. This basis is known as the Demmler-Reinsch basis. Furthermore, note that these basis functions are orthogonal, since

$$\begin{aligned}
\mathbf{U}^T\mathbf{U} &= \left(\mathbf{N}\mathbf{R}^{-1}\mathbf{E}\right)^T\left(\mathbf{N}\mathbf{R}^{-1}\mathbf{E}\right)\\
&= \mathbf{E}^T\mathbf{R}^{-T}\mathbf{N}^T\mathbf{N}\mathbf{R}^{-1}\mathbf{E}\\
&= \mathbf{E}^T\mathbf{R}^{-T}\mathbf{R}^T\mathbf{R}\mathbf{R}^{-1}\mathbf{E}\\
&= \mathbf{E}^T\mathbf{E}\\
&= \mathbf{I}
\end{aligned}$$

Therefore, $\mathbf{S}(\lambda)=\mathbf{U}\left(\mathbf{I}+\lambda\mathbf{D}\right)^{-1}\mathbf{U}^T$ is an eigen-decomposition of $\mathbf{S}(\lambda)$ where the eigen-values are

$$e_i=\frac{1}{1+\lambda d_i},\ i=1,2,\ldots,n$$

and the eigenvectors are the associated columns of $\mathbf{U}$.

This decomposition is useful for improving the computational efficiency of a spline fitting algorithm since $(\mathbf{I} + \lambda\mathbf{D})$ is a diagonal matrix and thus finding its inverse is trivial compared to the inverse of $\left(\mathbf{N}^T\mathbf{N} + \lambda\mathbf{P}\right)$. Also, recall that we select the smoothing parameter λ by minimizing the GCV(λ) statistic with respect to λ. Any optimization algorithm for minimizing GCV(λ) will need to evaluate this inverse repeatedly, and so the Demmler-Reinsch basis allows us to select our smoothing parameter in fewer computational steps/operations.

The effective degrees of freedom also becomes easier to obtain, since

$$\begin{aligned}
\text{df}(\lambda) &= \text{trace}(\mathbf{S}(\lambda)) \\
&= \text{trace}\left(\mathbf{U}\left(\mathbf{I} + \lambda\mathbf{D}\right)^{-1}\mathbf{U}^T\right) \\
&= \text{trace}\left(\left(\mathbf{I} + \lambda\mathbf{D}\right)^{-1}\mathbf{U}^T\mathbf{U}\right) \\
&= \text{trace}\left(\left(\mathbf{I} + \lambda\mathbf{D}\right)^{-1}\right) \\
&= \sum_{i=1}^{n} \frac{1}{1 + \lambda d_i} = \sum_{i=1}^{n} e_i.
\end{aligned}$$

The Demmler-Reinsch basis also demonstrates the effect of the smoothing matrix on the sample data. It can be shown (see Hastie, Tibshirani, and Friedman (2001)) that $d_1 = d_2 = 0(e_1 = e_2 = 1)$, that is the intercept and linear term of the basis expansion are not penalized by λ. Furthermore, if the eigenvalues are sorted such that

$$e_1 \geq e_2 \geq \ldots \geq e_n \text{ (or } d_1 \leq d_2 \leq \ldots \leq d_n)$$

then the corresponding basis functions have increasing complexity (i.e. are orthogonal polynomials of increasing order). We can interpret this as the smoothing spline giving higher penalties to the functions with more "wobbliness", resulting in the smoothing effect. So as λ increases, the more "wobbly" basis functions drop out of the expansion first.

4.7 Further Computational Considerations

Recall that we proposed a trunctuated power basis of degree d to be used as the basis for our regression splines, due to their simplicity and evident smoothness at the knot points. As discussed in section 4.3, as any two knots become close to each other, the truncated power basis suffers from multicollinearity. A more attractive basis is the *B-spline basis*, constructed using divided differences of $(x-\gamma)_+^d$. The i^{th} B-spline basis function of degree d has the following form:

$$B_{i,d}(x) = (-1)^{d+1}(\gamma_{i+d+1} - \gamma_i)\left[\gamma_i, \ldots, \gamma_{i+d+1}; (x-\gamma)_+^d\right]$$

where $\left[\gamma_i, \ldots, \gamma_{i+d+1}; (x-\gamma)_+^d\right]$ is the $d+1$ order divided difference of $(x-\gamma)_+^d$ with respect to the knots $\gamma_i, \ldots, \gamma_{i+d+1}$. Suppose our data is recorded over the interval $[a, b]$. Then we define the new set of knots $\boldsymbol{\gamma}'$ by augmenting $\boldsymbol{\gamma}$ as follows:

$$\gamma'_i = \begin{cases} a & , \ i < 1 \\ \gamma_i & , \ 1 \le i \le K \\ b & , \ i > K \end{cases}$$

for $i = -d, -d+1, \ldots, -1, 0, 1, \ldots, K, K+1, \ldots, K+d$. We can then use a recursive formulation for computing our B-splines functions (DeBoor, 1978). Let

$$B_{i,0}(x) = \begin{cases} 1 & , \ \text{if } x \in [\gamma_i, \gamma_{i+1}) \\ 0 & , \ \text{otherwise} \end{cases}$$

then for $d > 0$, we have:

$$B_{i,d}(x) = \frac{x-\gamma_i}{\gamma_{i+d}-\gamma_i} B_{i,d-1}(x) + \frac{\gamma_{i+d+1}-x}{\gamma_{i+d+1}-\gamma_{i+1}} B_{i+1,d-1}(x).$$

Some notes concerning B-splines:

- The matrix $\mathbf{B} = \{B_{ij}\}$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, K+d+1$ where

$$B_{ij} = B_{j,d}(x_i)$$

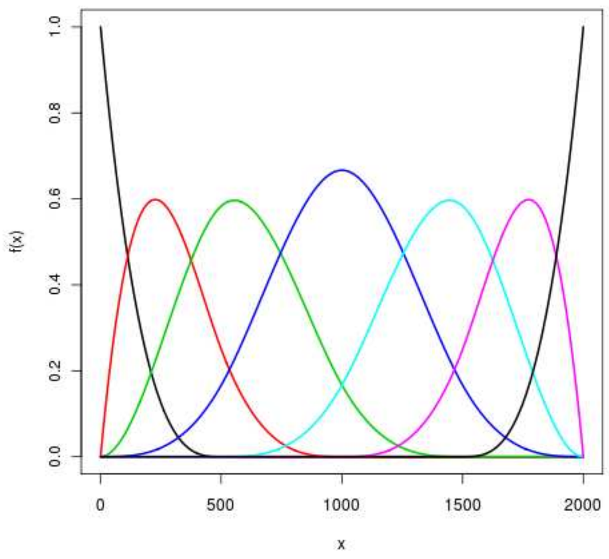

Figure 7: B-spline basis functions with 5 equally-spaced knots.

for any degree d is banded since at any particular $x \in [a, b]$, at most $d + 1$ basis functions will be non-zero. The banded structure of the design matrix gives B-splines the appealing numerical stability and computational efficiency for a large number of knots.

- Regression splines using the B-spline basis although numberically stable, still required the selection and placement of the knots.

We would like to take advantage of the nice numerical properties of B-splines, but would also like to avoid selecting the position and number of knots. Marx and Eiler (1996) proposed that we can use a B-spline basis with a large number of evenly-spaced knots and introduce a smoothness penalty on the coefficients of adjacent B-splines. Recall that the smoothness penalty is

$$\lambda \int_a^b f''(t)^2 dt$$

where a and b are the end points of our interval of interest. Suppose we use cubic B-splines and penalize the 2^{nd} derivative of the B-spline basis functions. Then we can regularize our fit with the smoothing penalty

$$\lambda \int_a^b \left[\sum_{i=1}^{K} \psi_i B''_{i,3}(t) \right]^2 dt. \tag{9}$$

where $\psi_1, \ldots, \psi_K$ are the spline coefficients. Marx and Eiler propose to approximate (9) as

$$\lambda \sum_{j=3}^{n} (\Delta^2 \psi_j)^2$$

where Δ^2 is the 2^{nd} order difference operator. Essentially, we are using a 2^{nd} order divided difference to approximate the 2^{nd} derivatives of the B-spline basis functions, and we sum over the coefficients to approximate the integral.

Our penalized residual sum of squares then becomes

$$\text{PRSS}(\lambda) = (\mathbf{Y} - \mathbf{B}\boldsymbol{\psi})^T (\mathbf{Y} - \mathbf{B}\boldsymbol{\psi}) + \lambda \boldsymbol{\psi}^T \mathbf{D}_2^T \mathbf{D}_2 \boldsymbol{\psi}$$

where $\mathbf{B}$ is a design matrix of B-spline basis functions and $\mathbf{D}_2$ is the matrix representation of 2^{nd} order divided differences. If we let $\mathbf{P} = \mathbf{D}_2^T \mathbf{D}_2$, then we can write the penalized RSS as:

$$\text{PRSS}(\lambda) = (\mathbf{Y} - \mathbf{B}\boldsymbol{\psi})^T (\mathbf{Y} - \mathbf{B}\boldsymbol{\psi}) + \lambda \boldsymbol{\psi}^T \mathbf{P} \boldsymbol{\psi}$$

which has a similar form as the penalized RSS for the smoothing splines, using the natural cubic spline basis. Thus, we will find the minimum at

$$\hat{\boldsymbol{\psi}} = \left(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{P} \right)^{-1} \mathbf{B}^T \mathbf{Y}$$

and our fitted function is then

$$\hat{f}(x) = \sum_{i=1}^{K} \hat{\psi}_i B_i(x).$$

Figure 8 demonstrates the similarity in scatterplot smoothing between natural cubic splines and P-splines. The blue curve denotes a natural cubic spline fit, and the red curve is a P-spline. The smoothing parameter was selected by optimizing the GCV criterion in both cases.

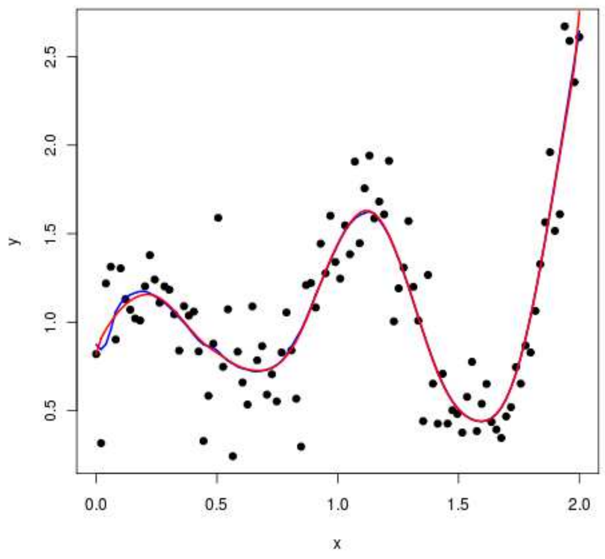

Figure 8: Comparison of P-splines to natural cubic splines.

5 Identifying Signals with P-Splines in Scatterplot Smoothing

In this chapter, we will apply the background theory from the previous chapters in order to develop an EM-algorithm for identifying and modelling signals in scatterplots. Let $Y_i, i = 1, 2, \ldots, n$ be a random sample from a population which can be partitioned into K mutually exclusive subsets. That is

$$\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \ldots \mathcal{G}_K \text{ and } \mathcal{G}_j \cap \mathcal{G}_k = \emptyset, j \neq k.$$

We assume that the model for each sub-population is given by,

$$Y_i = f_j(x_i) + \epsilon_{ji}, j = 1, \dots, K$$

where $\epsilon_{ji} \sim \mathrm{N}(0, \sigma_j^2)$ for $i = 1, \dots, n_j$ and $n = \sum_{j=1}^{K} n_j$ and are independently distributed. We model $f_j(x)$ as a P-spline, where

$$f_j = \mathbf{B}\boldsymbol{\psi}_j$$

and $\mathbf{f}_j = [f(x_1), \dots, f(x_n)]^T$, $\mathbf{B}$ is the design matrix of B-spline basis functions, and $\boldsymbol{\psi}_j = (\psi_{j1}, \psi_{j2}, \dots, \psi_{jK})^T$ are the spline coefficients, and $\boldsymbol{\psi}_j$ are assumed to be i.i.d.

$$\boldsymbol{\psi}_j \sim \mathrm{N}_m\left(\mathbf{0}, \frac{1}{\rho}\mathbf{D}^-\right)$$

where $\mathbf{D}$ is the penalty matrix used in P-spline smoothing and $\mathbf{D}^-$ denotes the Moore-Penrose inverse of $\mathbf{D}$.

Let $\mathbf{G} = \{g_{ij}\}_{n \times K}$ be the matrix of group memberships where

$$g_{ij} = \begin{cases} 1 & , \text{ if the } i^{th} \text{ sample point comes from group } j \\ 0 & , \text{ otherwise} \end{cases}$$

If we knew the group memberships, then our likelihood becomes

$$L = \prod_{i=1}^{n}\prod_{j=1}^{K}\left(f_{Y|\boldsymbol{\psi}_j}(y_i|\boldsymbol{\psi}_j)f(\boldsymbol{\psi}_j)\right)^{g_{ij}}$$

where g_{ij} is the sample group membership. The log-likelihood is

$$\begin{aligned}\log L &= \sum_{i=1}^{n}\sum_{j=1}^{K} g_{ij}\log\left(f_{Y|\boldsymbol{\psi}_j}(y_i|\boldsymbol{\psi}_j)\right) + g_{ij}\log\left(f(\boldsymbol{\psi}_j)\right) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{K} g_{ij}\left(\log\left[\left(\frac{1}{2\pi\sigma_j^2}\right)^{\frac{1}{2}}\right] - \frac{1}{2\sigma_j^2}\left(y_i - f_j(x_i)\right)^2\right) + \\ &\quad \sum_{i=1}^{n}\sum_{j=1}^{K} g_{ij}\left(\log\left[\frac{|\rho D|^{\frac{1}{2}}}{(2\pi)^{\frac{m}{2}}}\right] - \frac{\rho}{2}\boldsymbol{\psi}^T D\boldsymbol{\psi}\right)\end{aligned}$$

If we let $\rho = \frac{\lambda}{\sigma_j^2 \sum_{i=1}^n g_{ij}}$, ie ignoring the constant terms that do not depend on $\boldsymbol{\psi}_j$, then we get

$$\begin{aligned}\log L &= \sum_{i=1}^n \sum_{j=1}^K g_{ij} \left(-\frac{1}{2\sigma_j^2} (y_i - f_j(x_i))^2 \right) + \sum_{i=1}^n \sum_{j=1}^K g_{ij} \left(-\frac{\lambda}{2\sigma_j^2 \sum_{i=1}^n g_{ij}} \boldsymbol{\psi}_j^T D \boldsymbol{\psi}_j \right) \\ &= \sum_{j=1}^K \left(\frac{-1}{2\sigma_j^2} \right) \left[\left(\mathbf{Y} - \mathbf{B}\boldsymbol{\psi}_j\right)^T \mathbf{G}_j \left(\mathbf{Y} - \mathbf{B}\boldsymbol{\psi}_j\right) + \lambda_j \boldsymbol{\psi}_j^T D \boldsymbol{\psi}_j \right]\end{aligned}$$

where $\mathbf{G}_j$ is a diagonal matrix with the j^{th} column of $\mathbf{G}$ along the diagonal. Note that if we fix λ_j and maximize the log-likelihood in terms of $\boldsymbol{\psi}_j$, this is the same as minimizing a penalized residual sum of squares that has been weighted by the group memberships. That is, we are minimizing

$$\text{PRSS}(\lambda_j) = \left(\mathbf{Y} - \mathbf{B}\boldsymbol{\psi}_j\right)^T \mathbf{G}_j \left(\mathbf{Y} - \mathbf{B}\boldsymbol{\psi}_j\right) + \lambda_j \boldsymbol{\psi}_j^T D \boldsymbol{\psi}_j$$

with respect to $\boldsymbol{\psi}_j$. By taking the derivative of $PRSS(\lambda_j)$ and setting it equal to zero, we get the minimum at

$$\hat{\boldsymbol{\psi}}_j = \left(\mathbf{B}^T \mathbf{G}_j \mathbf{B} + \lambda_j \mathbf{D}\right)^{-1} \mathbf{B}^T \mathbf{G}_j \mathbf{Y} \tag{10}$$

We can construct an EM-algorithm to maximize the log-likelihood in the same manner as we did for fitting discrete mixture models.

1. Initialize $\tilde{\mathbf{G}}$.

2. For $j = 1, \ldots, K$, set

$$\hat{\pi}_j = \frac{1}{n} \sum_{i=1}^n \tilde{g}_{ij}$$

$$\hat{\boldsymbol{\psi}}_j = \left(\mathbf{B}^T \tilde{\mathbf{G}}_j \mathbf{B} + \lambda_j \mathbf{D}\right)^{-1} \mathbf{B}^T \tilde{\mathbf{G}}_j \mathbf{Y}$$

$$\hat{\sigma}_j^2 = \frac{\sum_{i=1}^n \tilde{g}_{ij} \left(y_i - \hat{f}_j(x_i)\right)^2}{\sum_{i=1}^n \tilde{g}_{ij}}$$

   where $\hat{f}_j(x_i)$ is the i^{th} element of $\mathbf{B}\hat{\boldsymbol{\psi}}_j$.

3. Update $\tilde{\mathbf{G}}$. For each $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, K$, set

$$\tilde{g}_{ij} = \frac{\hat{\pi}_j \left(f_{Y|\boldsymbol{\psi}_j}(y_i; \hat{\boldsymbol{\psi}}_j) \right) f(\hat{\boldsymbol{\psi}}_j)}{\sum_{l=1}^{K} \hat{\pi}_l \left(f_{Y|\boldsymbol{\psi}_l}(y_i; \hat{\boldsymbol{\psi}}_l) \right) f(\hat{\boldsymbol{\psi}}_l)}$$

4. Repeat steps 2 and 3 until convergence. If we compute the log-likelihood after step 3, we can claim convergence when the change in the log-likelihood is less than some $\epsilon > 0$.

Figure 9 shows the results of the above algorithm on our motivating example from chapter 1. For fixed smoothing parameters λ_1, and λ_2 we initialized the algorithm at random group memberships 200 times, and selected the fit with the highest likelihood as the best fit.

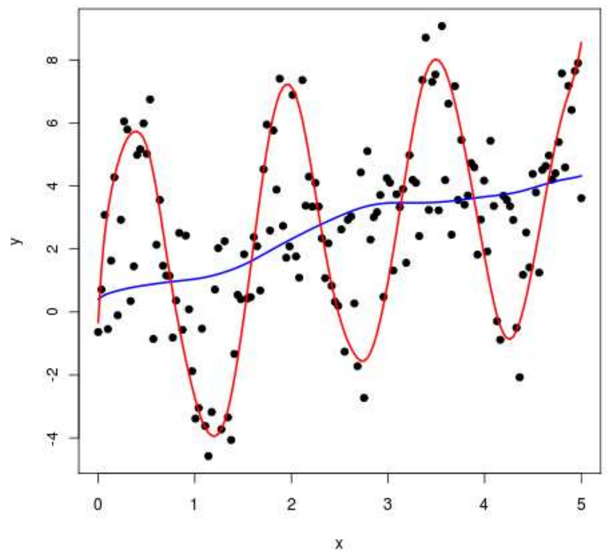

Figure 9: The two signals identified in the motivating example.

6 Simulation Results

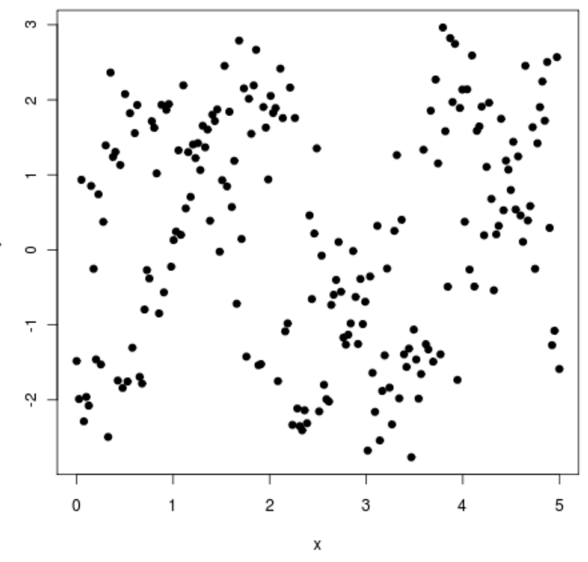

Figure 10: One data set used in our simulation.

We have run a small simulation to demonstrate the usefulness of the algorithm. We generated 2000 data sets from the true signals

$$f_1(x) = 2\sin(2x) \text{ and } f_2(x) = 2\sin(2(x-1))$$

where each data point is generated with probability 0.5 from one of these two signals as

$$Y_i = f(x_i) + \epsilon_i, i = 1, \ldots, n$$

with $\epsilon_i \sim \mathrm{N}(0, 0.5)$ and the x_i are 200 equally spaced points on [0,5]. Figure 10 is an example of one such simulated data sets. Our focus is on the fact the two underlying signals are not immediately evident from the scatterplot. Our goal here is to see how well the proposed method will be able to extract these two signals.

Since we are generating the data ourselves, we know the true group memberships, thus we initialize the group membership matrix in the algorithm to the true group membership. This improves the time required for convergence of the algorithm and ensures that we will obtain the true global maximum of the likelihood. In a real data set, the group membership would be unknown, thus we would have to apply the random restarts approach as we used in the fit found in Figure 9.

We fit each of the 2000 data sets using the EM-algorithm with fixed values for λ_1 and λ_2. Furthermore let $\hat{f}_1^*(x_i)$ and $\hat{f}_2^*(x_i)$ be the average of the 2000 estimates, for $i = 1, \ldots, n$. Figure 11 has a plot of the true functions f_1 and f_2 (in black) overlayed with $\hat{f}_1^*(x_i)$ and $\hat{f}_2^*(x_i)$ (in red). Note that on average our method is able to accurately extract the two underlying signals.

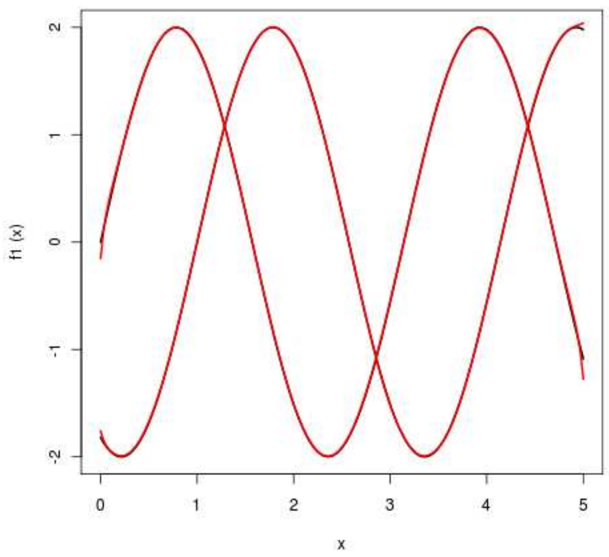

Figure 11: The mean of the 2000 estimators plotted over the true functions.

Since we know the true functions, we are also able to compute the point-

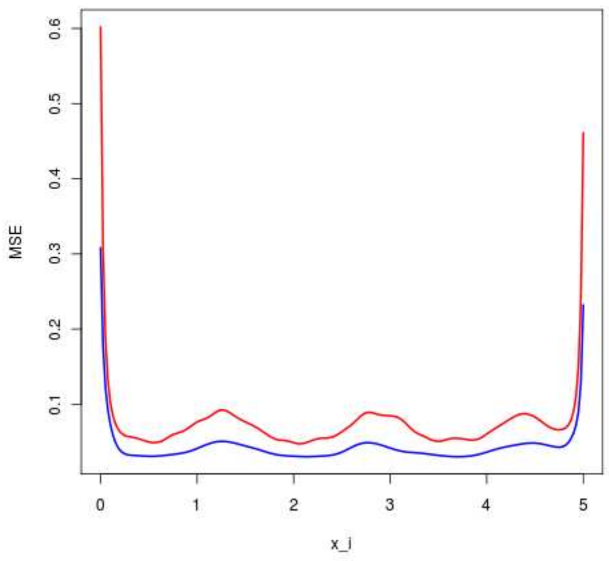

Figure 12: The point-wise mean square error of the 2000 estimators.

wise squared error of our estimators. For each of the 2000 data sets, we compute the point-wise squared difference between the estimator $\hat{f}(x)$ and the true function $f(x)$. Figure 12 has the average point-wise squared difference (over the 2000 data sets) plotted against the values of x_i. The blue and red curves denote the point-wise squared error of $\hat{f}_1(x)$ and $\hat{f}_2(x)$ respectively. Note that the squared error is close to zero except at the end points. This indicates that on average our method will find estimators fairly close to the true signals.

7 Conclusions and Further Study

Our simulation demonstrates that on average our algorithm is capable of picking up the true signals in the data. Even without knowing the group membership (as would typically be the case), using random restarts we are able to eventually find the estimators which maximum the log-likelihood. Developing a more intelligent method of initializing the group memberships would be a focus of future research.

Also, it is important to note that in our simulation we used a fixed value for the smoothing parameters. We originally attempted to use a generalized cross-validation statistic to select the smoothing parameters, but ran into computational difficulties with the application. Further work would go into achieving this goal of data driven selection of the smoothing parameter.

Another limitation to our current simulation is that we assumed exactly two signals from the data. Although this was reasonable for our data set (since we generated the data from exactly two signals), some data sets may require more than two signals. It would be reasonable to extend our algorithm to fit the data to 2,3,4,. . . signals and select the most suitable fit using an information criterion such as AIC.

Appendix A

Properties of the Multivariate Normal Distribution

To derive the moment generating function, let us start with $\mathbf{Z} = (Z_1, \ldots, Z_p)^T$ where $Z_i \sim \mathrm{N}(0, 1)$, for $i = 1, 2, \ldots, p$. Furthermore, suppose that Z_i and Z_j are independently distributed for all $i \neq j$. Then we can write the pdf of $\mathbf{Z}$

(the joint pdf of $Z_1, \ldots, Z_p$) as:

$$\begin{aligned} f_{\mathbf{Z}}(\mathbf{z}) &= \prod_{i=1}^{p} \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{1}{2}z_i^2\right\} \\ &= \left(\frac{1}{2\pi}\right)^{n/2} \exp\left\{-\frac{1}{2}\sum_{i=1}^{p} z_i^2\right\} \\ &= \left(\frac{1}{2\pi}\right)^{n/2} \exp\left\{-\frac{1}{2}\mathbf{z}^T\mathbf{z}\right\}. \end{aligned}$$

Then the moment generating function of $\mathbf{Z}$ is

$$\begin{aligned} M_{\mathbf{z}}(\mathbf{t}) &= \mathrm{E}\left[\exp\left\{\mathbf{t}^T\mathbf{Z}\right\}\right] \\ &= \mathrm{E}\left[\prod_{i=1}^{p} \exp\left\{t_i, Z_i\right\}\right] \\ &= \prod_{i=1}^{p} \mathrm{E}\left[\exp\left\{t_i Z_i\right\}\right]. \end{aligned}$$

Recall that the mgf of the $\mathrm{N}(0,1)$ distribution is $\exp\{t^2/2\}$ so therefore

$$\begin{aligned} M_{Z_i}(t_i) &= \mathrm{E}\left[\exp\left\{t_i Z_i\right\}\right] \\ &= \exp\left\{\frac{t_i^2}{2}\right\} \end{aligned}$$

and hence

$$\begin{aligned} M_{\mathbf{z}}(\mathbf{t}) &= \prod_{i=1}^{n} \exp\left\{\frac{t_i^2}{2}\right\} \\ &= \exp\left\{\frac{1}{2}\sum_{i=1}^{n} t_i^2\right\} \\ &= \exp\left\{\frac{1}{2}\mathbf{t}^T\mathbf{t}\right\}. \end{aligned}$$

Note that $\mathbf{Z} \sim \mathrm{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu} = \mathbf{0}$ (a $p \times 1$ vector of zeroes) and $\boldsymbol{\Sigma} = \mathbf{I}_{p\times p}$ (the $p \times p$ identity matrix).

In the more general case, let $\mathbf{\Sigma}$ be any $p \times p$ symmetric, positive semi-definite matrix. Then we can decompose $\mathbf{\Sigma}$ as

$$\mathbf{\Sigma} = \mathbf{\Gamma}^T \mathbf{\Lambda} \mathbf{\Gamma}$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_p)$ are the non-negative eigenvalues of $\mathbf{\Sigma}$ and the columns of $\mathbf{\Gamma}$ the corresponding orthogonal eigenvectors of $\mathbf{\Sigma}$. That is

$$\mathbf{\Gamma}^T \mathbf{\Gamma} = \mathbf{\Gamma} \mathbf{\Gamma}^T = \mathbf{I}_{p \times p}.$$

Since $\lambda_i \geq 0$ for $i = 1, 2, \ldots_p$, then we can write:

$$\mathbf{\Lambda} = \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Lambda}^{\frac{1}{2}}$$

where the i^{th} diagonal of $\mathbf{\Lambda}^{\frac{1}{2}}$ is $\sqrt{\lambda_i}$.

Furthermore, since $\mathbf{\Gamma} \mathbf{\Gamma}^T = \mathbf{I}_{p \times p}$, then we can decompose $\mathbf{\Sigma}$ as

$$\begin{aligned}
\mathbf{\Sigma} &= \mathbf{\Gamma}^T \mathbf{\Lambda} \mathbf{\Gamma} \\
&= \mathbf{\Gamma}^T \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Gamma} \\
&= \mathbf{\Gamma}^T \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Gamma} \mathbf{\Gamma}^T \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Gamma} \\
&= \left[\mathbf{\Gamma}^T \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Gamma} \right] \left[\mathbf{\Gamma}^T \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Gamma} \right] \\
&= \mathbf{\Sigma}^{\frac{1}{2}} \mathbf{\Sigma}^{\frac{1}{2}}
\end{aligned}$$

where we define $\mathbf{\Sigma}^{\frac{1}{2}} = \mathbf{\Gamma}^T \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Gamma}$. Note that clearly $\mathbf{\Sigma}^{\frac{1}{2}}$ is positive semi-definite as well and that

$$\left(\mathbf{\Sigma}^{\frac{1}{2}} \right)^{-1} = \mathbf{\Gamma}^T \mathbf{\Lambda}^{-\frac{1}{2}} \mathbf{\Gamma}$$

where $\mathbf{\Lambda}^{-\frac{1}{2}} = \text{diag}((1/\sqrt{\lambda_1}), \ldots, (1/\sqrt{\lambda_p}))$ since

$$\begin{aligned}
\mathbf{\Sigma}^{\frac{1}{2}} \left(\mathbf{\Sigma}^{\frac{1}{2}} \right)^{-1} &= \mathbf{\Gamma}^T \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Gamma} \mathbf{\Gamma}^T \mathbf{\Lambda}^{-\frac{1}{2}} \mathbf{\Gamma} \\
&= \mathbf{\Gamma}^T \left(\mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Lambda}^{-\frac{1}{2}} \right) \mathbf{\Gamma}^T \\
&= \mathbf{I}_{p \times p}.
\end{aligned}$$

Finally, we can define the random vector $\mathbf{X}$ as

$$\mathbf{X} = \mathbf{\Sigma}^{\frac{1}{2}}\mathbf{Z} + \boldsymbol{\mu}$$

where $\mathbf{\Sigma}$ is any $p \times p$ symmetric positive semi-definite matrix and $\boldsymbol{\mu} \in \mathbf{R}^p$

Then we see that

$$\begin{aligned}
\mathrm{E}\left[X_i\right] &= \mathrm{E}\left[\sum_{j=1}^{p}\left\{\mathbf{\Sigma}^{\frac{1}{2}}\right\}_{ij} Z_j + \mu_i\right] \\
&= \sum_{j=1}^{p}\left\{\mathbf{\Sigma}^{\frac{1}{2}}\right\}_{ij} \mathrm{E}\left[Z_j\right] + \mu_i \\
&= \mu_i \text{ (since } \mathrm{E}\left[Z_j\right] = 0)
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{Cov}(X_i, X_j) &= \mathrm{Cov}\left(\sum_{k=1}^{p}\left\{\mathbf{\Sigma}^{\frac{1}{2}}\right\}_{ik} Z_k + \mu_i, \sum_{k=1}^{p}\left\{\mathbf{\Sigma}^{\frac{1}{2}}\right\}_{jk} Z_k + \mu_j\right) \\
&= \left\{\mathbf{\Sigma}^{\frac{1}{2}}\mathbf{\Sigma}^{\frac{1}{2}}\mathrm{Cov}\left(\mathbf{Z}\right)\right\}_{ij}.
\end{aligned}$$

Recall that $\mathrm{Cov}(\mathbf{Z}) = \mathbf{I}_{p\times p}$, so:

$$\begin{aligned}
\mathrm{Cov}(X_i, X_j) &= \left\{\mathbf{\Sigma}^{\frac{1}{2}}\mathbf{\Sigma}^{\frac{1}{2}}\mathrm{Cov}\left(\mathbf{Z}\right)\right\}_{ij} \\
&= \left\{\mathbf{\Sigma}\mathbf{I}_{p\times p}\right\}_{ij} \\
&= \left\{\mathbf{\Sigma}\right\}_{ij}
\end{aligned}$$

and thus

$$\mathrm{E}\left[\mathbf{X}\right] = \boldsymbol{\mu}, \text{ and } \mathrm{Cov}(\mathbf{X}) = \mathbf{\Sigma}.$$

Note we can now derive the moment generating function of $\mathbf{X}$ as

$$\begin{aligned}
M_{\mathbf{X}}(\mathbf{t}) &= \mathrm{E}\left[\exp\left\{\mathbf{t}^T\mathbf{X}\right\}\right] \\
&= \mathrm{E}\left[\exp\left\{\mathbf{t}^T\boldsymbol{\Sigma}^{\frac{1}{2}}\mathbf{Z}+\mathbf{t}^T\boldsymbol{\mu}\right\}\right] \\
&= \exp\left\{\mathbf{t}^T\boldsymbol{\mu}\right\}\mathrm{E}\left[\exp\left\{\mathbf{t}^T\boldsymbol{\Sigma}^{\frac{1}{2}}\mathbf{Z}\right\}\right] \\
&= \exp\left\{\mathbf{t}^T\boldsymbol{\mu}\right\}\mathrm{E}\left[\exp\left\{\left(\boldsymbol{\Sigma}^{\frac{1}{2}}\mathbf{t}\right)^T\mathbf{Z}\right\}\right] \\
&= \exp\left\{\mathbf{t}^T\boldsymbol{\mu}\right\}M_{\mathbf{Z}}\left(\boldsymbol{\Sigma}^{\frac{1}{2}}\mathbf{t}\right) \\
&= \exp\left\{\mathbf{t}^T\boldsymbol{\mu}\right\}\exp\left\{\frac{1}{2}\left(\boldsymbol{\Sigma}^{\frac{1}{2}}\mathbf{t}\right)^T\left(\boldsymbol{\Sigma}^{\frac{1}{2}}\mathbf{t}\right)\right\} \\
&= \exp\left\{\mathbf{t}^T\boldsymbol{\mu}\right\}\exp\left\{\frac{1}{2}\mathbf{t}^T\boldsymbol{\Sigma}\mathbf{t}\right\} \\
&= \exp\left\{\frac{1}{2}\mathbf{t}^T\boldsymbol{\Sigma}\mathbf{t}+\mathbf{t}^T\boldsymbol{\mu}\right\}.
\end{aligned}$$

Proof of Property 2

Proof. We derive the moment generating function of $\mathbf{Y}$ as

$$\begin{aligned}
M_{\mathbf{Y}}(\mathbf{t}) &= \mathrm{E}\left[\exp\left\{\mathbf{t}^T\mathbf{Y}\right\}\right] \\
&= \mathrm{E}\left[\exp\left\{\mathbf{t}^T\mathbf{A}\mathbf{X}+\mathbf{t}^T\mathbf{b}\right\}\right] \\
&= \mathrm{E}\left[\exp\left\{\left(\mathbf{A}^T\mathbf{t}\right)^T\mathbf{X}\right\}\right]\exp\left\{\mathbf{t}^T\mathbf{b}\right\} \\
&= M_{\mathbf{X}}\left(\mathbf{A}^T\mathbf{t}\right)\exp\left\{\mathbf{t}^T\mathbf{b}\right\} \\
&= \exp\left\{\frac{1}{2}\left(\mathbf{A}^T\mathbf{t}\right)^T\boldsymbol{\Sigma}\left(\mathbf{A}^T\mathbf{t}\right)+\left(\mathbf{A}^T\mathbf{t}\right)^T\boldsymbol{\mu}\right\}\exp\left\{\mathbf{t}^T\mathbf{b}\right\} \\
&= \exp\left\{\frac{1}{2}\mathbf{t}^T\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T\mathbf{t}+\mathbf{t}^T\left(\mathbf{A}\boldsymbol{\mu}+\mathbf{b}\right)\right\}
\end{aligned}$$

which we note is the mgf of an $\mathrm{N}_m\left(\mathbf{A}\boldsymbol{\mu}+\mathbf{b},\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T\right)$ distribution. □

Proof of Property 3

Proof. Suppose X_1 and X_2 are independent. Then by definition of covariance we have:

$$\begin{aligned}
\mathrm{Cov}(X_1, X_2) &= \mathrm{E}\left[(X_1 - \mathrm{E}[X_1])(X_2 - \mathrm{E}[X_2])\right] \\
&= \mathrm{E}\left[(X_1 - \mu_1)(X_2 - \mu_2)\right] \\
&= \mathrm{E}[X_1X_2] - \mu_1\mathrm{E}[X_2] - \mathrm{E}[X_2]\mu_1 + \mu_1\mu_2 \\
&= \mathrm{E}[X_1X_2] - \mu_1\mu_2
\end{aligned}$$

but due to the independence of X_1 and X_2 we get $\mathrm{E}[X_1X_2] = \mu_1\mu_2$ and thus we see that

$$\mathrm{Cov}(X_1, X_2) = \mathrm{E}[X_1X_2] - \mu_1\mu_2 = 0.$$

Now suppose $\mathrm{Cov}(X_1, X_2) = 0$ and we let $\boldsymbol{\mu} = (\mu_1, \mu_2)^T$ and

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathrm{Var}(X_1) & \mathrm{Cov}(X_1, X_2) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}(X_2) \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}.$$

Now if we let $\mathbf{X} = (X_1, X_2)^T$, then clearly $\mathbf{X} \sim \mathrm{N}_2(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We can then write the mgf of $\mathbf{X}$ as

$$M_{\mathbf{X}}(\mathbf{t}) = \exp\left\{\frac{1}{2}\mathbf{t}^T\boldsymbol{\Sigma}\mathbf{t} + \mathbf{t}^T\boldsymbol{\mu}\right\}.$$

If we let $\mathbf{t} = (t_1, t_2)^T$ then:

$$\begin{aligned}
M_{\mathbf{X}}(t_1, t_2) &= \exp\left\{\frac{1}{2}\left(\sigma_1^2t_1^2 + \sigma_2^2t_2^2 + 2\mathrm{Cov}(X_1, X_2)(t_1t_2)\right) + (t_1\mu_1 + t_2\mu_2)\right\} \\
&= \exp\left\{\frac{1}{2}\left(\sigma_1^2t_1^2 + \sigma_2^2t_2^2\right) + (t_1\mu_1 + t_2\mu_2)\right\} \text{ (since } \mathrm{Cov}(X_1, X_2) = 0) \\
&= \exp\left\{\frac{1}{2}\sigma_1^2t_1^2 + t_1\mu_1\right\}\exp\left\{\frac{1}{2}\sigma_2^2t_2^2 + t_2\mu_2\right\} \\
&= M_{X_1}(t_1)M_{X_2}(t_2).
\end{aligned}$$

Since we have written the joint mgf of X_1 and X_2 as the product of their individual mgf's, then we can conclude X_1 and X_2 are independent. □

This result can easily be extended to higher dimensions as well, although we will omit the proof. Let $\mathbf{X_1} \sim \mathrm{N}_{p1}\left(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1\right), \mathbf{X_2} \sim \mathrm{N}_{p2}\left(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2\right)$. Let $p = p_1 + p_2$, $\mathbf{X} = \left(\mathbf{X_1}^T, \mathbf{X_2}^T\right)^T$, $\boldsymbol{\mu} = \left(\boldsymbol{\mu}_1^T, \boldsymbol{\mu}_2^T\right)^T$ and

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_2 \end{bmatrix}$$

then $\mathbf{X} \sim \mathrm{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\mathbf{X_1}$ and $\mathbf{X_2}$ are independently distributed.

Appendix B

Below is the R function which performs the EM-algorithm developed in chapter 5.

```
fit.spline.em<-function(x,y,Z,lambda1=1,lambda2=1) {
  n<-length(y)
  knots<-knots<-seq(min(x),max(x),length=as.integer(n/3)+2)
  knots<-knots[2:(length(knots)-1)]
  B<-bs(x,degree=3,intercept=TRUE,knots=knots)
  D<-diff(diff(diag(length(knots)+4)))
  D<-t(D)%*%D
  # if the group membership is null we initialize
  # to random values
  if(is.null(Z)) {
    Z<-matrix(0,ncol=2,nrow=n)
    Z[,1]<-runif(n,0,1)
    Z[,2]<-1-Z[,1]
  }

  p1<-0
  p2<-0
```

```
psi1<-NULL
psi2<-NULL
sigma1<-0
sigma2<-0

max<-500
cll<-0
for(iters in 1:max) {
  sumZ<-apply(Z,2,sum)
  # compute MLE's for p1 and p2
  p1<-sumZ[1]/n
  p2<-1-p1

  Z1diag<-diag(Z[,1],nrow=n,ncol=n)
  Z2diag<-diag(Z[,2],nrow=n,ncol=n)

  # compute the splines
  Q<-solve(t(B)%*%Z1diag%*%B + lambda1*D,t(B)%*%Z1diag)
  psi1<-Q%*%y
  S1<-B%*%Q
  fhat1<-S1%*%y
  RSS1<-(y-fhat1)^2
  sigma1<-sum(Z[,1]*RSS1)/sumZ[1]

  Q<-solve(t(B)%*%Z2diag%*%B + lambda2*D,t(B)%*%Z2diag)
  psi2<-Q%*%y
  S2<-B%*%Q
  fhat2<-S2%*%y
  RSS2<-(y-fhat2)^2
```

```
sigma2<-sum(Z[,2]*RSS2)/sumZ[2]

nll<-0
for(i in 1:n) {
  nll<-nll + log(p1*dnorm(y[i],mean=fhat1[i],sd=sqrt(sigma1)) +
             p2*dnorm(y[i],mean=fhat2[i],sd=sqrt(sigma2)))
}

if(is.infinite(nll) || is.nan(nll)) {
  print("ll was no good")
  return(NULL)
}

# check if the change in the log-likelihood is small
# enough to declare convergence
if(cll != 0 && abs(nll - cll) < 1e-8) {
  out<-NULL
  out$fhat1<-fhat1
  out$psi1<-psi1
  out$lambda1<-lambda1
  out$fhat2<-fhat2
  out$psi2<-psi2
  out$lambda2<-lambda2
  out$Z<-Z
  out$ll<-nll
  return(out)
}
cll<-nll
```

```
    # update the group memberships
    for(i in 1:n) {
      Z[i,1]<-p1*dnorm(y[i],mean=fhat1[i],sd=sqrt(sigma1))
      Z[i,2]<-p2*dnorm(y[i],mean=fhat2[i],sd=sqrt(sigma2))
      c<-max(log(Z[i,1]),log(Z[i,2]))
      Z[i,1]<-exp(log(Z[i,1]) - c)
      Z[i,2]<-exp(log(Z[i,2]) - c)
      Z[i,1]<-Z[i,1] / (Z[i,1]+Z[i,2])
    }
    Z[,2]<-1-Z[,1]
  }
  print("Failed to converge")
  return(NULL)
}
```

References

[1] De Boor, Carl. *A Practical Guide to Splines*. New York: Springer-Verlag. 1978.

[2] Eilers, Paul H. C., and Brian D. Marx. *Flexible smoothing with B-splines and penalties. Statistical Science* 11 (1996): 89-121.

[3] Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning; Data Mining, Inference and Prediction.* New York: Springer. 2001.

[4] Hogg, Robert V., Joseph W. McKean, and Allen T. Craig. *Introduction to Mathematical Statistics.* 6^{th} ed. Upper Saddle River, NJ: Pearson Prentice Hall. 2005.

[5] McLachlan, Robert, David Peel. *Finite Mixture Models.* New York: Wiley. 2000.

[6] Ramsay, J.O., B.W. Silverman. *Functional Data Analysis.* New York: Springer-Verlag. 1997.

[7] Wahba, Grace. *Spline Models for Observational Data.* Philadelphia, Pennsylvania: Society for Industrial and Applied Mathematics. 1990.