Exercise 7

1. The data-frame `parboot.dat` contains simulated data from the following model:

$$y = 8 \cdot x + 4 \cdot \cos(14 \cdot x) + \epsilon_i, \quad i \in 1, \dots, 70,$$

where $x \in \{\frac{j}{70}, j = 1, \dots, 70\}$ and $\epsilon_i \sim P$ iid. for an unknown distribution P.

In this exercise we want to compare confidence-intervals for nonparametric-regression which are generated by 3 different techniques, that are:

- hat-matrix approach (as in exercise 3)
- parametric bootstrap with assumption $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$
- model-based bootstrap with no assumptions about the errors.

To do this, fit a smoothing-spline (automatic choice of degrees of freedom) to the `parboot`-data and compute confidence-intervals at selected locations. Those locations are:

```
x.pre <- seq(5,62,by=3)/70
```

Plot the data, the spline-fit, the original curve and and all confidence intervals at the selected locations into the same plot and comment on the results.

R-Hints: The data is located at `http://stat.ethz.ch/Teaching/Datasets/parboot.dat`. Use $B = 2000$ bootstrap-samples in each case. For the hat-matrix approach you need to compute the hat-matrix for smooth.spline for the given data. This can again be done by smoothing unit vectors as in exercise 3. Use the same degrees of freedom for fit and hat-matrix-generation. `smooth.spline` automatically calculates the degrees of freedom. For the parametric bootstrap approach you need an estimate for the error variance σ^2. You can use the same estimate as in hat-matrix-theory, that is

$$\hat{\sigma}^2 = \sum_{i=1}^{n} \frac{(Y_i - \hat{m}(x_i))^2}{n - df}.$$

As a hint for the interpretation you could check the Gaussian assumption that the parametrical bootstrap-technique makes by looking at the normal-plot (`qqnorm`) for the residuals.

2. **a) Quadratic Discriminant Analysis (QDA)**
Assume the normal model $X|Y = j \sim \mathcal{N}_p(\mu_j, \Sigma_j)$, $\mathbb{P}[Y = j] = p_j$, $\sum_{j=0}^{J-1} p_j = 1$.
Show that (6.2) and (6.4) lead to

$$\hat{\delta}_j^{QDA}(x) = -\log(\det(\hat{\Sigma}_j))/2 - (x - \hat{\mu}_j)^\intercal \hat{\Sigma}_j^{-1}(x - \hat{\mu}_j)/2 + \log(\hat{p}_j).$$

b) Linear Discriminant Analysis (LDA)
Use the result from a) and replace $\hat{\Sigma}_j$ by $\hat{\Sigma}$ to get

$$\begin{aligned} \hat{\delta}_j^{LDA}(x) &= x^\intercal \hat{\Sigma}^{-1}\hat{\mu}_j - \hat{\mu}_j{}^\intercal \hat{\Sigma}^{-1}\hat{\mu}_j/2 + \log(\hat{p}_j) \\ &= (x - \hat{\mu}_j/2)^\intercal \hat{\Sigma}^{-1}\hat{\mu}_j + \log(\hat{p}_j). \end{aligned} \quad (1)$$

c) The LDA decision function can be written as (see (1) above)

$$\hat{\delta}_j(x) = x^\intercal b_j + c_j,$$

where $b_j \in \mathbb{R}^p$ and $c_j \in \mathbb{R}$. Assume that we only have two classes ($j = 0, 1$). Use the equation above to characterize the decision boundary.

d) Small Simulation
Use the R-code below to get an idea about how LDA works. Change the covariance matrix and mean vectors if you like.
Manually calculate (see c)) the boundary between group 1 and 2. Add your solution to the plot with `abline()`.
Hint:
If `A <- fit$scaling` it holds (in the case of $p+1$ groups in $\mathbb{R}^p$) that $\hat{\Sigma}^{-1} = AA^\intercal$. The means and prior probabilites can also be found in the `lda`-object.
R-Code:

```
library(mvtnorm) ## Needed for rmvnorm
library(MASS) ## Needed for lda/qda
## prediction plot code
predplot <- function(object, x, main = "", len = 200, ...)
{
    xp <- seq(min(x[,1]), max(x[,1]), length=len)
    yp <- seq(min(x[,2]), max(x[,2]), length=len)
    grid <- expand.grid(xp, yp)
    colnames(grid) <- colnames(x)[-3]
    Z <- predict(object, grid, ...)
    zp <- as.numeric(Z$class)
    zp <- Z$post[,3] - pmax(Z$post[,2], Z$post[,1])
    plot(x[,1], x[,2], col = x[,3], pch = x[,3], main = main)
    contour(xp, yp, matrix(zp, len),
            add = T, levels = 0, drawlabels = FALSE)
    zp <- Z$post[,1] - pmax(Z$post[,2], Z$post[,3])
    contour(xp, yp, matrix(zp, len),
            add = T, levels = 0, drawlabels = FALSE)
}
## Covariance Matrix
sigma <- cbind(c(0.5, 0.3), c(0.3, 0.5))
## Mean vectors
mu1 <- c(3, 1.5)
mu2 <- c(4, 4)
mu3 <- c(8.5, 2)
m <- matrix(0, nrow = 300, ncol = 3)
## Grouping vector
m[,3] <- rep(1:3, each = 100)
## Simulate data
```

```
m[1:100,1:2] <- rmvnorm(n = 100, mean = mu1, sigma = sigma)
m[101:200,1:2] <- rmvnorm(n = 100, mean = mu2, sigma = sigma)
m[201:300,1:2] <- rmvnorm(n = 100, mean = mu3, sigma = sigma)
m <- data.frame(m)
## Perform LDA
fit <- lda(x = m[,1:2], grouping = m[,3])
## Plot the decision boundaries
predplot(fit, m)
```

Preliminary discussion: Friday, April 25, 2008. **Deadline:** Friday, May 2, 2008.