Series 2

1. In a study on the contribution of air pollution to mortality, General Motors collected data from 60 US Standard Metropolitan Statistical Areas (SMSAs). The dependent variable is the age adjusted mortality. The data includes variables measuring demographic characteristics of the cities, variables measuring climate characteristics and variables recording the pollution potential.

Mortality	Age Adjusted mortality
JanTemp	Mean January temperature (degrees Fahrenheit)
JulyTemp	Mean July temperature (degrees Fahrenheit)
RelHum	Relative Humidity
Rain	Annual rainfall (inches)
Educ	Median education
Dens	Population density
NonWhite	Percentage of non whites
WhiteCollar	Percentage of white collar workers
Pop	Population
House	Average population per household
Income	Median income
HC	Hydrocarbon pollution potential
NOx	Nitrous Oxide pollution potential
SO2	Sulfur Dioxide pollution potential

Try to find a good linear model that represents the data. Therefore, take the following steps:

1. Inspect the data using scatterplots of the response **Mortality** versus the predictors and produce diagnostic plots such as, for instance, residual plots. Which (if any) of these plots are useful in order to assess the validity of a model? Why? Should we perform any transformation?

Remark: It is good practice to log-transform data that are very right-skewed even though none of the model assumptions are violated. This and other techniques had been developed by Tukey and are known as “first aid” transformations. The reasons behind these transformations are a little bit elaborated and involve the diagonal entries of the matrix P of the subsection “Geometrical Interpretation” of the script as well as the “Residual vs Leverage” plot produced by the R-function “lm”.

2. Perform the transformations suggested in the R-hints. Now, perform a stepwise variable selection. Do forward selection and backward elimination lead to the same model? Which variables have a significant influence on the **Mortality**-level and in which direction? Compare these results to an all-subsets regression using Mallows-C_p.

R-hints:

```
> ## Reading the dataset
> url <- "http://stat.ethz.ch/Teaching/Datasets/mortality.csv"
> mortality <- read.csv(url,header = TRUE)
> mortality <- mortality[,-1]
> ##Create pairs plot using the splom() function of the ``lattice" package
> library(lattice)
> splom(~mortality,pscales=0,cex=0.5)
> ## Note that the plots of the response Mortality versus the predictors
> ## are shown in the last row.
> ## Transform the data
> mortality_old <- mortality ## It is always a good idea to keep the original data.
> mortality[,"logPop"] <- log(mortality[,"Pop"])
> mortality[,"logHC"]  <- log(mortality[,"HC"])
> mortality[,"logNOx"] <- log(mortality[,"NOx"])
> mortality[,"logSO2"] <- log(mortality[,"SO2"])
> col_num <- which(names(mortality) %in% c("Pop","HC","NOx","SO2"))
> mortality <- mortality[,-col_num]

> ## Fit the full model
> mortal.full <- lm(Mortality ~. , data=mortality)

> ## Fit the empty model. This is not very useful in itself, but is required
> ## as a starting model for stepwise forward variable selection
> mortal.empty <- lm(Mortality ~ 1, data = mortality)

> ## Backward elimination, starting from the full model
> mortal.bw    <- step(mortal.full, direction = "backward")
> ## Forward selection, starting from the empty model
> mortal.fw    <- step(mortal.empty, direction = "forward",
                     scope = list(upper=mortal.full,lower=mortal.empty))

> ## Loading the package for all-subsets regression
> library(leaps)
```

```
## All subsets model choice, compare to the stepwise methods
mortal.alls  <- regsubsets(...)

## Load function to produce a nice figure of C_p versus p
source("ftp://stat.ethz.ch/Teaching/maechler/CompStat/cp-plot.R")
p.regsubsets(mortal.alls)
```

2. **a)** The artificial dataset shown in Figure 1 is generated from a mixture of two normal distributions (see part b)). Imagine you wouldn't know anything about the distribution and you'd want to estimate the density by use of a kernel density estimator with Gaussian kernel. Guess a good bandwidth.

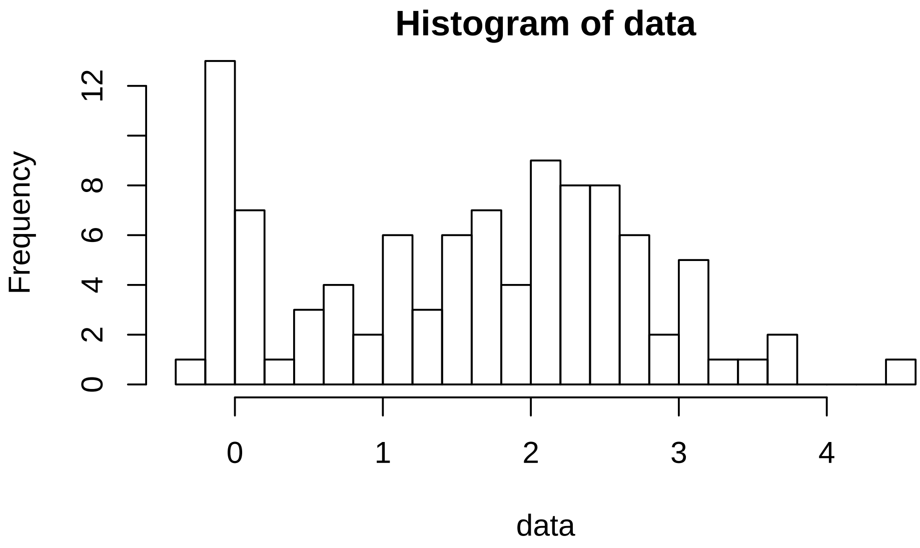

Figure 1: Artificial data from mixture of normal distributions

b) Carry out a simulation with `R` to evaluate the quality of bandwidths for kernel density estimation for a mixture of normal distributions where a point is generated by a $\mathcal{N}(0, 0.01)$-distribution with probability 0.2 and by a $\mathcal{N}(2, 1)$-distribution with probability 0.8. Repeat the following code 200 times for multiple bandwidth values: $h = 2^k$ where $k \in \{-5, -4.8, \ldots, 1\}$ as well as for a plug-in estimate of the optimal bandwidth (see below):

R-hints:

```
> ## You can create a list for the bandwidth values using the following command:
> bandwidths <- c(as.list(2^seq(-5,1,0.2)),"sj")
> ## Elements of a list can be accessed by using double square brackets.
> bandwidths[[5]]
```

1. Generate a dataset of 100 points from the mixture distribution from above:

```
> set.seed(1)
> data <- numeric(100)
> for(i in 1:100){
   p <- runif(1, min = 0, max = 1)
   if (p < 0.2)
     data[i] <- rnorm(1, mean = 0, sd = sqrt(0.01))
   else
     data[i] <- rnorm(1, mean = 2, sd = 1)
 }
```

2. Compute the kernel density estimator with the function `density`. Consult `help(density)` to understand the syntax.

```
> ke <- density(data, bw = 0.1, n = 61, from = -1, to = 5)
```

An estimated plug-in bandwidth is obtained by using `bw ="sj"`.

3. In the lecture, we saw that the goodness of a kernel estimate can be measured using the integrated mean squared error (IMSE). In practice, the IMSE can be approximated by summing the squared difference between the kernel estimator and the true density values (up to a constant factor).

```
> ## Compute the true density at the given datapoints
> dmix <- 0.2 * dnorm(ke$x, mean = 0, sd = sqrt(0.01)) +
     0.8 * dnorm(ke$x, mean = 2, sd = 1)
> ## Take the mean of the squared differences
> quality <- mean((ke$y - dmix)^2)
```

Note that the commands given above are suitable for a single execution, but not necessarily for the full simulation. For example, you will need a vector of the qualities of all simulation runs. You may define a matrix for the qualities to store the results for all different bandwidths. Consider `help(matrix), help(apply)`.

Compute and compare the averaged quality of the kernel estimators for the different bandwidths.

c) (Optional) Repeat the whole project with the Epanechnikov kernel, which can be obtained by specifying `kernel="epanechnikov"` in `density`.

d) Did you guess a good bandwidth?

Preliminary discussion: Friday, March 11.

Deadline: Friday, March 18.