Solution to Series 2

1. Note that this is not the only possible solution. In real data analytic problems, as opposed to mathematics, it is usually not clear what the "truth" or the "correct solution" is.

The scatterplot matrix indicates skewed distributions for the variables `Pop, HC, NOx` and `SO2` (see Fig.1). Therefore, these variables are transformed to their logarithms (indicated by first letter `l`). Note that the linear regression model does not assume a normal distribution for the predictors, but a skewed distribution and outliers often result in regression solutions that are largely determined by very few points.

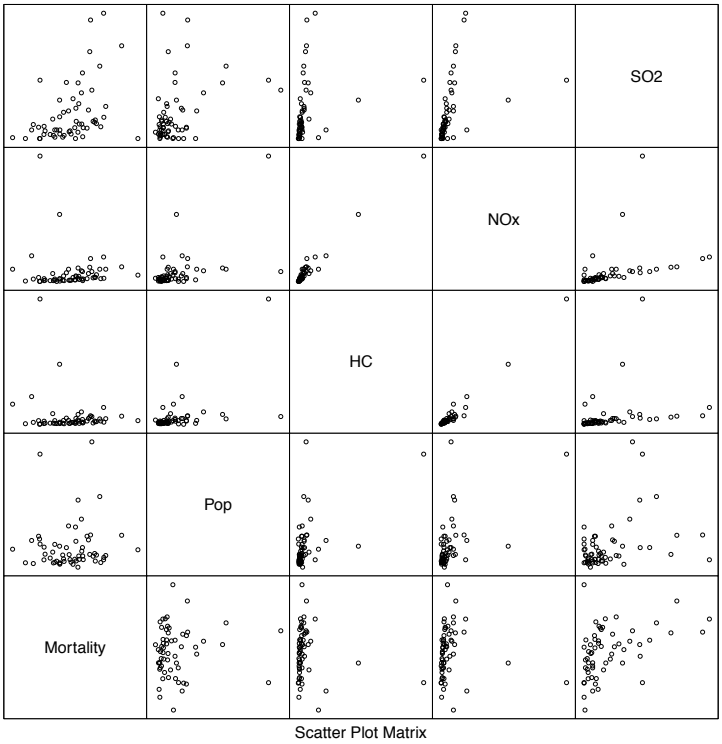

Figure 1: Pairs plot of the variables Mortality, Pop, HC, NOx and SO2

The scatterplot matrix for the transformed data looks better (see Fig. 2); the effects of the predictors on `Mortality` look more or less linear (as far at it can be assessed). A linear appearance for every single variable is not necessary in multiple regression, but systematic deviations should be a reason for suspicion.

```
> mortality[,"Pop"] <- log(mortality[,"Pop"])
> mortality[,"HC"]  <- log(mortality[,"HC"])
> mortality[,"NOx"] <- log(mortality[,"NOx"])
> mortality[,"SO2"] <- log(mortality[,"SO2"])
```

We fit the full model containing all predictors. The summary statistic of this model reveals that only `JanTemp, Rain, NonWhite` and `NOx` have a significant influence on the response.

```
> mortal.full <- lm(Mortality ~., data=mortality)
> summary(mortal.full)
```

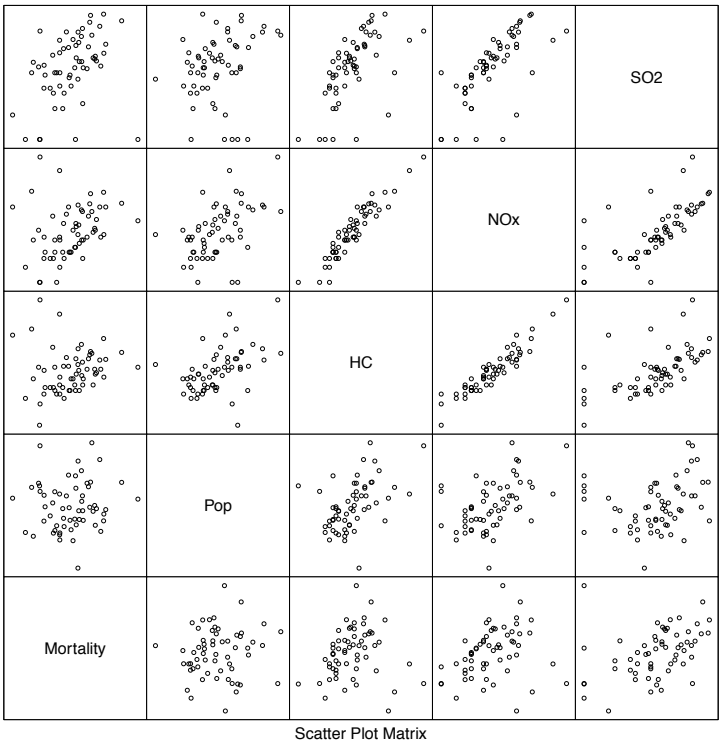

Figure 2: Pairs plot of the transformed variables

```
Call:
lm(formula = Mortality ~ ., data = mortality)

Residuals:
    Min      1Q  Median      3Q     Max
-68.893 -20.704   0.586  24.129  74.604

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.297e+03  2.934e+02   4.422 6.32e-05 ***
JanTemp     -2.368e+00  8.851e-01  -2.676   0.0104 *
JulyTemp    -1.752e+00  2.031e+00  -0.863   0.3931
RelHum       3.420e-01  1.059e+00   0.323   0.7482
Rain         1.493e+00  5.898e-01   2.532   0.0150 *
Educ        -1.000e+01  9.087e+00  -1.101   0.2771
Dens         4.525e-03  4.223e-03   1.072   0.2897
NonWhite     5.152e+00  1.002e+00   5.143 6.01e-06 ***
WhiteCollar -1.883e+00  1.198e+00  -1.572   0.1232
Pop          4.391e+00  7.714e+00   0.569   0.5721
House       -4.574e+01  3.939e+01  -1.161   0.2518
Income      -6.892e-04  1.334e-03  -0.516   0.6081
HC          -2.204e+01  1.523e+01  -1.447   0.1550
NOx          3.397e+01  1.425e+01   2.384   0.0215 *
SO2         -3.687e+00  7.359e+00  -0.501   0.6189
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 34.48 on 44 degrees of freedom
Multiple R-squared:  0.7685,        Adjusted R-squared:  0.6949
F-statistic: 10.43 on 14 and 44 DF,  p-value: 8.793e-10
```

We analyze the model assumptions by making use of residual diagnostics (see Fig. 3). The Tukey-Anscombe looks fine. It seems that the linear model is valid and that the variance of the random errors is constant. The normal QQ-plot indicates that the assumption of Gaussian errors and response, respectively, is reasonable.

```
> par(mfrow=c(1,2))
> plot(fitted(mortal.full),resid(mortal.full),xlab = "Fitted Values",
      ylab = "Residuals", main = "Tukey-Anscombe")
> abline(h = 0, lty = 2)
> qqnorm(resid(mortal.full), xlab = "Standard Normal Quantiles",
       ylab = "Empirical Quantiles")
> qqline(resid(mortal.full))
```

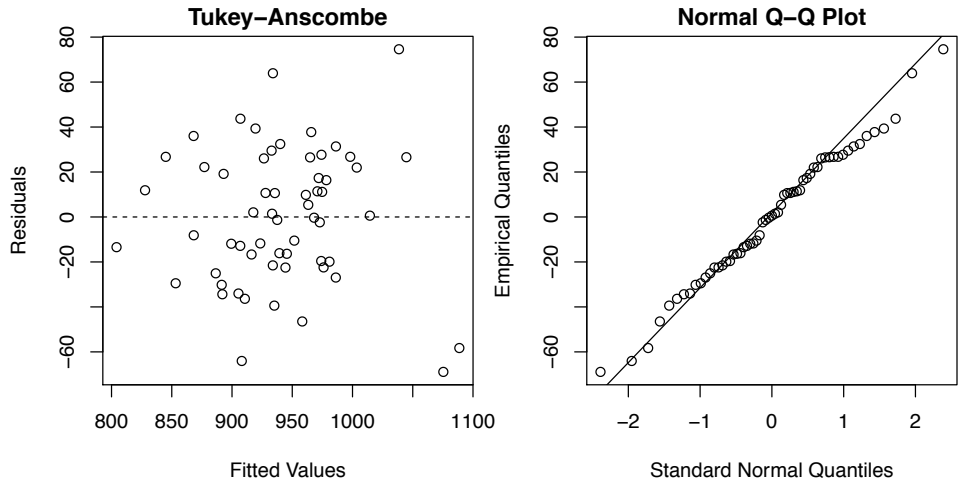

Figure 3: Residual diagnostics; left: Tukey-Anscombe Plot, right: Normal Q-Q Plot

In order to find an adequate model that is useful for prediction we perform a stepwise variable selection. First, we do a backward elimination and then a forward selection. Both are based on the AIC.

Backward elimination starts from the full model and eliminates one single variable at a time. It stops when the AIC converges. Here, our final model has following predictors: `JanTemp`, `Rain`, `Educ`, `NonWhite`, `WhiteCollar` and `NOx`.

```
> mortal.bw <- step(mortal.full, dir="backward")

Start:  AIC=430.46
Mortality ~ JanTemp + JulyTemp + RelHum + Rain + Educ + Dens +
    NonWhite + WhiteCollar + Pop + House + Income + HC + NOx +
    SO2

              Df Sum of Sq   RSS    AIC
- RelHum       1     124.1 52436 428.60
- SO2          1     298.4 52611 428.80
- Income       1     317.1 52629 428.82
- Pop          1     385.2 52698 428.89
- JulyTemp     1     884.5 53197 429.45
- Dens         1    1365.3 53678 429.98
- Educ         1    1440.1 53752 430.06
- House        1    1603.0 53915 430.24


...
```

```
Mortality ~ JanTemp + Rain + Educ + NonWhite + WhiteCollar +
    NOx

              Df Sum of Sq    RSS    AIC
<none>                      59143 421.70
- WhiteCollar  1      2194  61337 421.85
- Educ         1      2621  61764 422.26
- Rain         1     13263  72406 431.64
- JanTemp      1     17593  76736 435.07
- NOx          1     20607  79750 437.34
- NonWhite     1     48049 107192 454.79
```

Next we perform a forward selection. In this case, we start from the empty model having only an intercept. In each step a new predictor is added to the model. Again, the procedure stops when the difference in AIC between to steps is small.

```
> mortal.empty <- lm(Mortality ~ 1, data = mortality)
> mortal.fw    <- step(mortal.empty, dir="forward", data=mortality,
                 scope = list(upper=mortal.full,lower=mortal.empty))

Start:  AIC=488.79
Mortality ~ 1

              Df Sum of Sq    RSS    AIC
+ NonWhite     1     94473 131520 458.85
+ Educ         1     58340 167652 473.17
+ Rain         1     42393 183599 478.54
+ SO2          1     34675 191318 480.97
+ House        1     30608 195385 482.21
+ JulyTemp     1     23407 202586 484.34
+ WhiteCollar  1     18920 207072 485.63
+ Income       1     18138 207855 485.86
+ NOx          1     17696 208296 485.98


...


Mortality ~ NonWhite + Educ + SO2 + JanTemp + Rain + NOx + WhiteCollar +
    House

           Df Sum of Sq   RSS    AIC
<none>                  57067 423.59
+ HC        1   1635.68 55431 423.88
+ Dens      1   1251.25 55816 424.28
+ Pop       1    424.15 56643 425.15
+ RelHum    1    184.44 56882 425.40
+ JulyTemp  1    105.41 56961 425.48
+ Income    1     71.05 56996 425.52
```

Forward selection methods ends with a different model:
`Mortality~NonWhite+Educ+lSO2+JanTemp+Rain+lNOx+WhiteCollar+House`.
Forward selection is a greedy algorithm. In each step it adds variables and never deletes them, although an addition of a new variable may render one or more of the already added variables non-significant.

Finally, we perform an all-subsets regression. Therefore, we have to load the `leaps` package. A nice C_p vs p plot can be produce with the function `p.regsubsets` that can be sourced from the given url.

```
> library(leaps)
> mortal.alls  <- regsubsets(Mortality ~ .,data = mortality,nvmax=9)
> source("ftp://stat.ethz.ch/Teaching/maechler/CompStat/cp-plot.R")
> p.regsubsets(mortal.alls,cex=0.8,cex.main=.8)
```

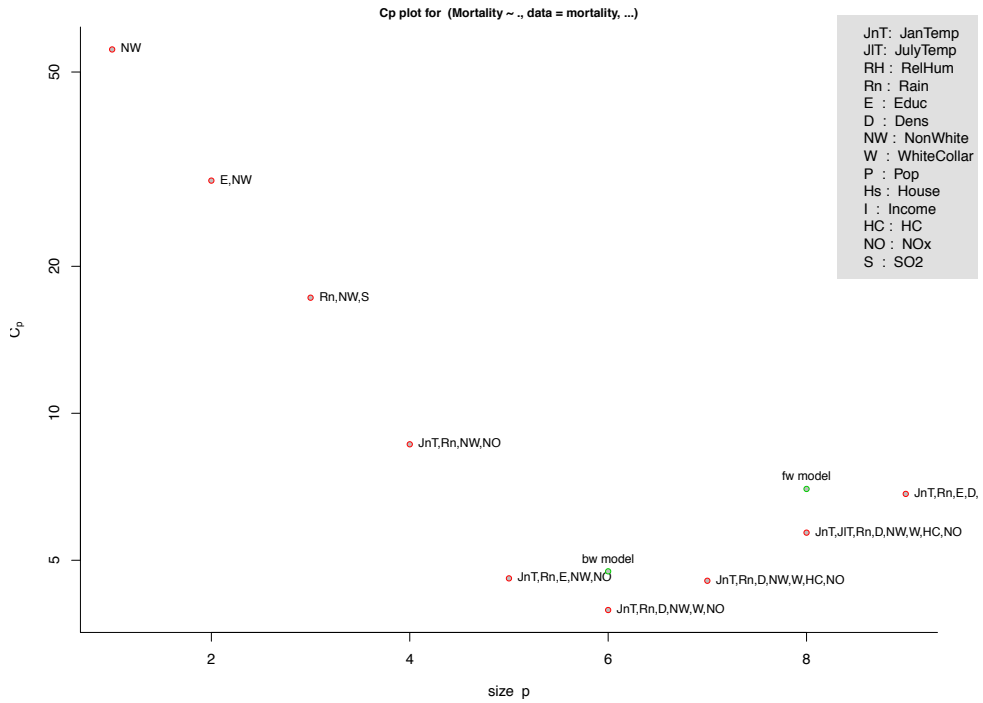

Figure 4: C_p Plot for different p after all-subsets regression

The model with smallest C_p is given for $p = 6$ (see Fig.4). It only differs in one (non-significant) variable from the optimal model of backward selection. The C_p values for the models of backward and forward selection are indicated by green points in the plot.

2.

a) The bandwidth should not be too large so that the left peak coming from the normal density with smaller variance can be resolved. Otherwise, the two modes become mixed and are no longer discriminable. E.g., 0.2 seems to be a reasonable choice.

b)
```
> rmix <- function(n = 100){
    data <- numeric(n)
    for(i in 1:n){
      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)
    }
    data
 }
```

Good R programming would define `rmix` without a `for`-loop, e.g.

```
> rmix <- function(n){
     data <- ifelse(runif(n, min = 0, max = 1) < 0.2,
                    rnorm(n, mean = 0, sd = sqrt(0.01)),
                    rnorm(n, mean = 2, sd = 1))
 }
```

Set up simulation:

```
> for (kernel in c("gaussian","epanechnikov")){
     set.seed(79)
     nrep <- 200
     bandwidths <- list(0.02, 0.1, 0.3, 0.6, 1, 1.5, "sj")
     qualities <- matrix(ncol = length(bandwidths), nrow = nrep)
     xpts <- seq(-1,5,0.1)[-1]
     dmix <- 0.2 * dnorm(xpts, mean = 0, sd = sqrt(0.01)) +
      0.8 * dnorm(xpts, mean = 2, sd = 1)

     for(i in 1:nrep){
        data <- rmix(100)
        for(j in seq_along(bandwidths)){
           ke <- density(data, bw = bandwidths[[j]], kernel = kernel,
                     n = 61, from = -1, to = 5)
           qualities[i,j] <- mean((ke$y[-1] - dmix)^2)
        }
      }
     if (kernel == "gaussian"){
         results.gaussian <- apply(qualities, 2, mean)
     }else{
         results.epanechnikov <- apply(qualities, 2, mean)
     }
 }
> results.gaussian

[1] 0.021799945 0.005548904 0.009923188 0.013693071
[5] 0.015566336 0.017634817 0.008489099

> results.epanechnikov
```

```
[1] 0.020789165 0.005585357 0.011370607 0.014733763
[5] 0.016086500 0.018196732 0.009727836
```

For both kernels a small bandwidth $h = 0.1$ leads to the best results. For the majority of the bandwidths the Gaussian kernel leads to slightly better results than the Epanechnikov kernel (the Epanechnikov kernel is only asymptotically optimal).

However the structure of the normal mixture distribution presumably favours a **local bandwidth selection**.