STK2100 - Machine learning and statistical methods

Mandatory assignment 1

Rohullah Akbari

2/16/2021

Problem 1

a)

We use the following code to make the data available:

```
library(MASS)
datadir = "http://www.uio.no/studier/emner/matnat/math/STK2100/data/"
nuclear = read.table(paste(datadir,"nuclear.dat",sep=""),header=T)
n = nrow( nuclear )
```

In order to be able to access the data directly by the variables, we use the attach() function. Our interest will be in the "cost" variable and since this variable is always positive, we will model this variable at the log-scale.

```
attach(nuclear)
nuclear["cost"] = log(cost)
names(nuclear)[names(nuclear) == "cost"] <- "logcost"
attach(nuclear)
```

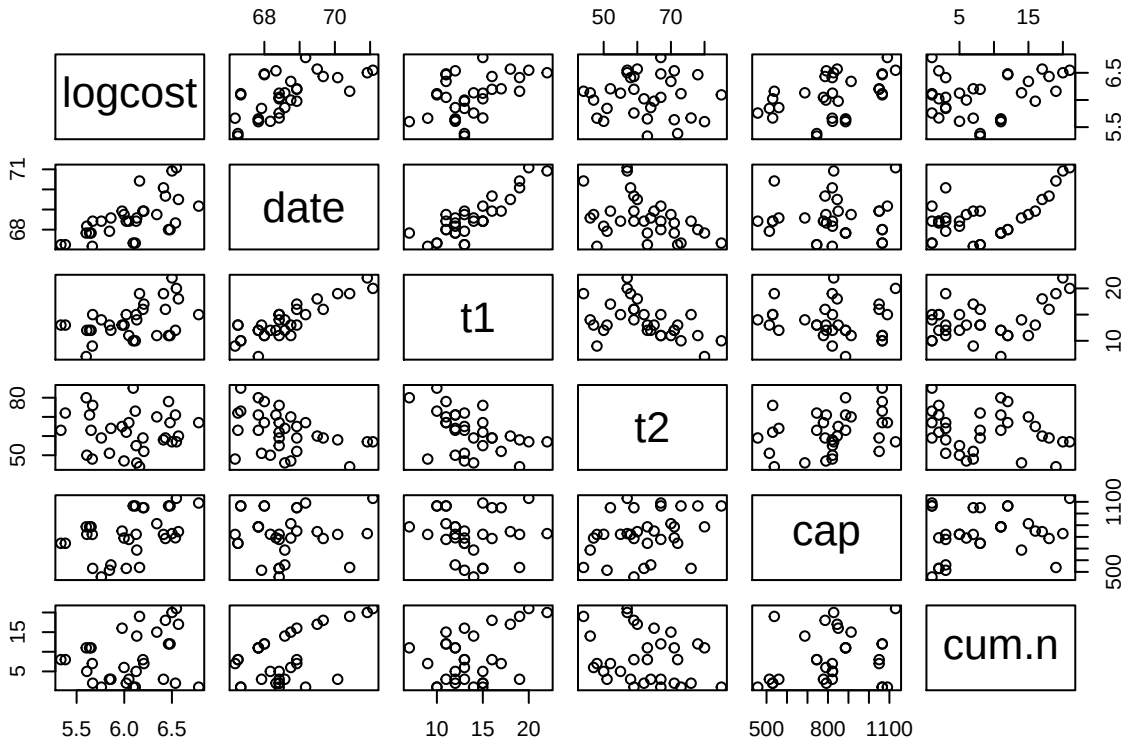
```
## The following objects are masked from nuclear (pos = 3):
##
##     bw, cap, ct, cum.n, date, ne, pr, pt, t1, t2
```

```
head(nuclear)
```

```
##    logcost  date t1 t2  cap pr ne ct bw cum.n pt
## 1 6.131335 68.58 14 46  687  0  1  0  0    14  0
## 2 6.115870 67.33 10 73 1065  0  0  1  0     1  0
## 3 6.094066 67.33 10 85 1065  1  0  1  0     1  0
## 4 6.480535 68.00 11 67 1065  0  1  1  0    12  0
## 5 6.464946 68.00 11 78 1065  1  1  1  0    12  0
## 6 5.844674 67.92 13 51  514  0  1  1  0     3  0
```

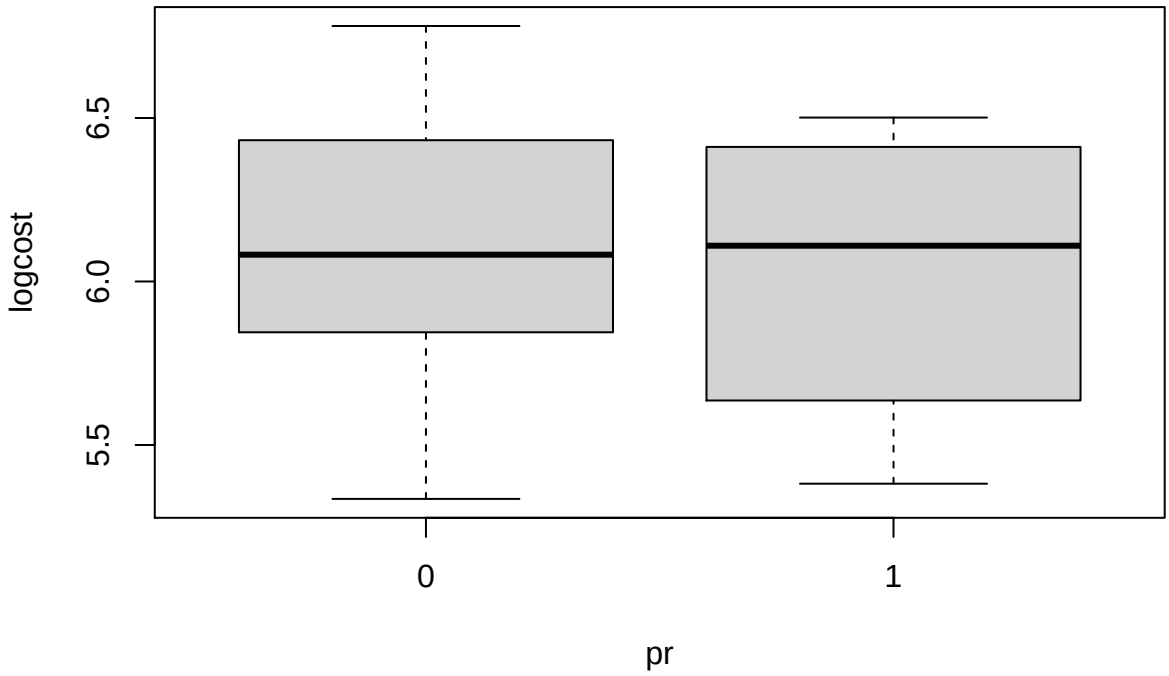
Here we can see that the "pr","ne","ct","bw" and "pt" variables are categorical variables since they consist of only 0's and 1's. Therefore, we plot without them. The matrix-plot of the data:

```
nuclear.plot = subset(nuclear,select = -c(pr,ne,ct,bw,pt))
pairs(nuclear.plot)
```

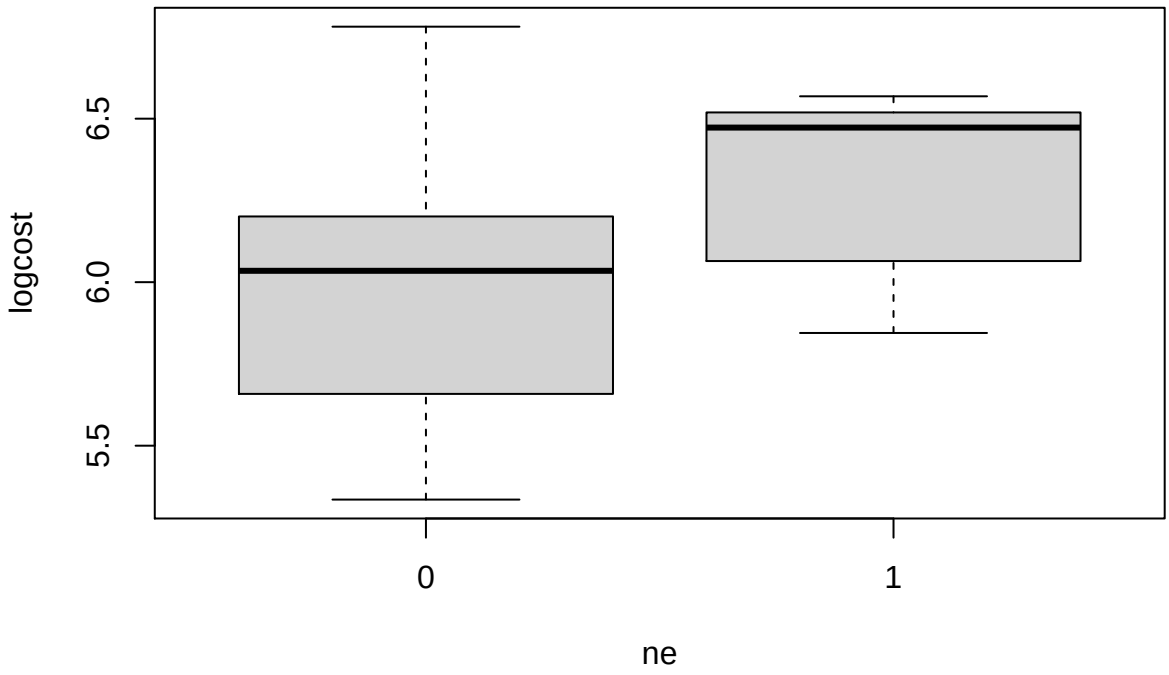


We use boxplot to plot the categorical variables:

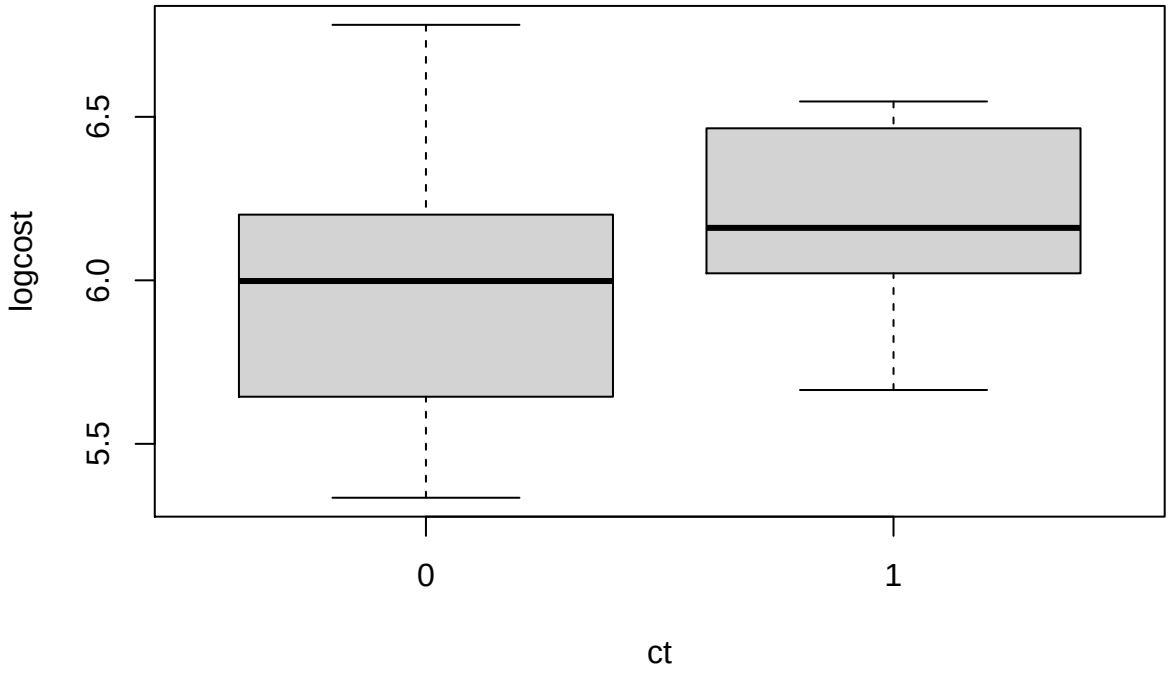
```
boxplot(logcost ~ pr)
```



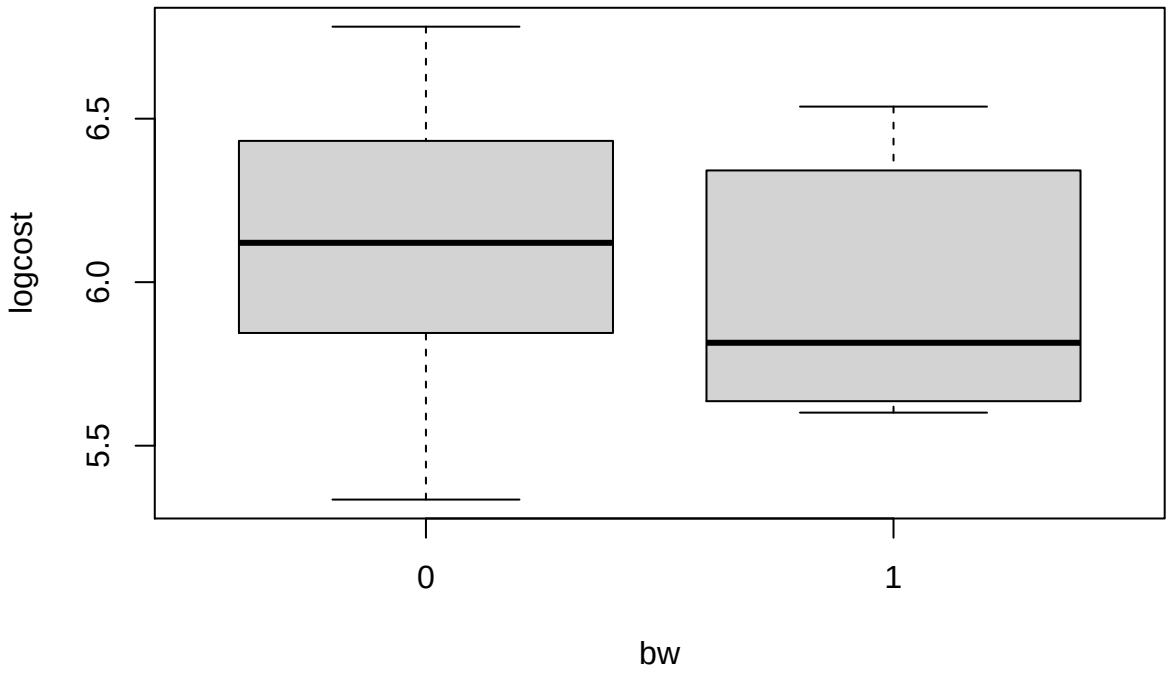

```
boxplot(logcost ~ ne)
```



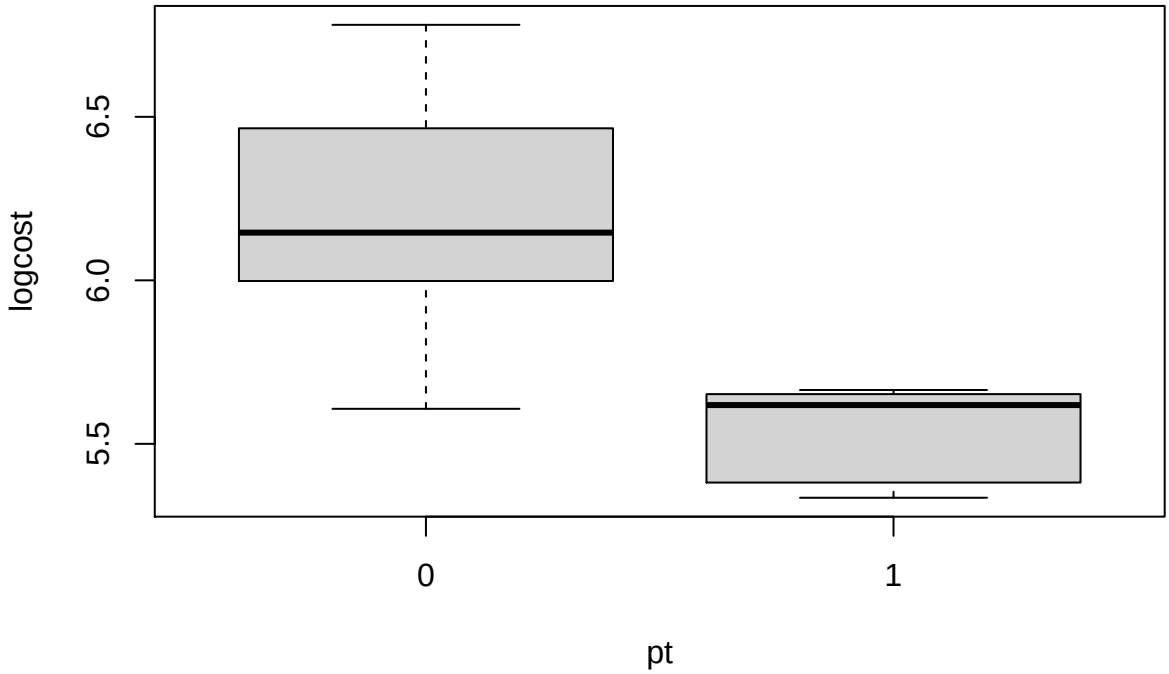

```
boxplot(logcost ~ ct)
```




```
boxplot(logcost ~ bw)
```




```
boxplot(logcost ~ pt)
```



b)

We will first look at a model:

$$Y_i = \beta_0 + \beta_1 x_{i,1} + \mathring{u}\mathring{u}\mathring{u} + \beta_p x_{i,p} + \epsilon_i$$

where Y_i is cost at log-scale for observation i.

The standard assumptions about the noise terms ϵ_i is that

$$\epsilon_1 ... \epsilon_n \sim N(0, \sigma^2)$$

In other words that the noise terms are normal distributed. The noise terms are independent of each other, the expectation value of them is 0 and the variance of them is σ^2, where σ^2 is a positive constant value for all replications.

Fitting this model:

```
fit1 = lm(logcost ~.,data=nuclear)
summary(fit1)
```

```
##
## Call:
## lm(formula = logcost ~ ., data = nuclear)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.284032 -0.081677  0.009502  0.090890  0.266548
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.063e+01  5.710e+00  -1.862  0.07662 .
## date         2.276e-01  8.656e-02   2.629  0.01567 *
## t1           5.252e-03  2.230e-02   0.236  0.81610
## t2           5.606e-03  4.595e-03   1.220  0.23599
## cap          8.837e-04  1.811e-04   4.878 7.99e-05 ***
## pr          -1.081e-01  8.351e-02  -1.295  0.20943
## ne           2.595e-01  7.925e-02   3.274  0.00362 **
## ct           1.155e-01  7.027e-02   1.644  0.11503
## bw           3.680e-02  1.063e-01   0.346  0.73261
## cum.n       -1.203e-02  7.828e-03  -1.536  0.13944
## pt          -2.220e-01  1.304e-01  -1.702  0.10352
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1697 on 21 degrees of freedom
## Multiple R-squared:  0.8635, Adjusted R-squared:  0.7985
## F-statistic: 13.28 on 10 and 21 DF,  p-value: 5.717e-07
```

Here we can see that the p-value of the model is very small ($5.717 \cdot 10^{-07}$), meaning that some of the variables are useful. By looking at the p-value for the individual variables, we can see that variables such as "t1" and "bw" have high p-value. We can conclude that this are not very useful and could remove them from the model.

c)

In general, we do a hypothesis-testing to determine if a variable should be thrown away or not. In this case, we do the following test:

$$H_0 : \beta_j = 0$$

From STK1110, we know that the smaller p-value the more evidence there is in the sample data against the H_0 and vice versa. Therefore, we start by removing the variable with the highest p-value ("t1").

```
fit2 = lm(logcost ~.-t1,data=nuclear)
summary(fit2)
```

```
##
## Call:
## lm(formula = logcost ~ . - t1, data = nuclear)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28898 -0.07856  0.01272  0.08983  0.26537
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.161e+01  3.835e+00  -3.027 0.006187 **
## date         2.431e-01  5.482e-02   4.435 0.000208 ***
## t2           5.451e-03  4.449e-03   1.225 0.233451
## cap          8.778e-04  1.755e-04   5.002 5.25e-05 ***
## pr          -1.035e-01  7.944e-02  -1.303 0.205922
## ne           2.607e-01  7.738e-02   3.368 0.002772 **
## ct           1.142e-01  6.853e-02   1.667 0.109715
## bw           2.622e-02  9.423e-02   0.278 0.783401
## cum.n       -1.220e-02  7.626e-03  -1.599 0.124034
## pt          -2.157e-01  1.249e-01  -1.727 0.098181 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.166 on 22 degrees of freedom
## Multiple R-squared:  0.8631, Adjusted R-squared:  0.8072
## F-statistic: 15.42 on 9 and 22 DF,  p-value: 1.424e-07
```

The R-squared factor has now increased from 0.7985 to 0.8072, meaning that the new model has a little bit better linear relationship with explanatory variables and logcost. In addition to that, the p-value of some variables has also decreased. It makes sense since the variables can behave differently in different models. We can see from matrix-plot above that "date" and "t1" have a strong correlation between them, and therefore by removing the "t1" will make "data" change its p-value drastic.

This method also applies to the general interpretation of a new model. By removing the variable with the highest p-value, will affect the p-value of the other variables who has a strong correlation the removed variable.

d)

The procedure for continuing to remove explanatory variables, until all p-values are less than 0.05, is to remove a variable at the time and check if the given condition is satisfied. Run it until it is satisfied.

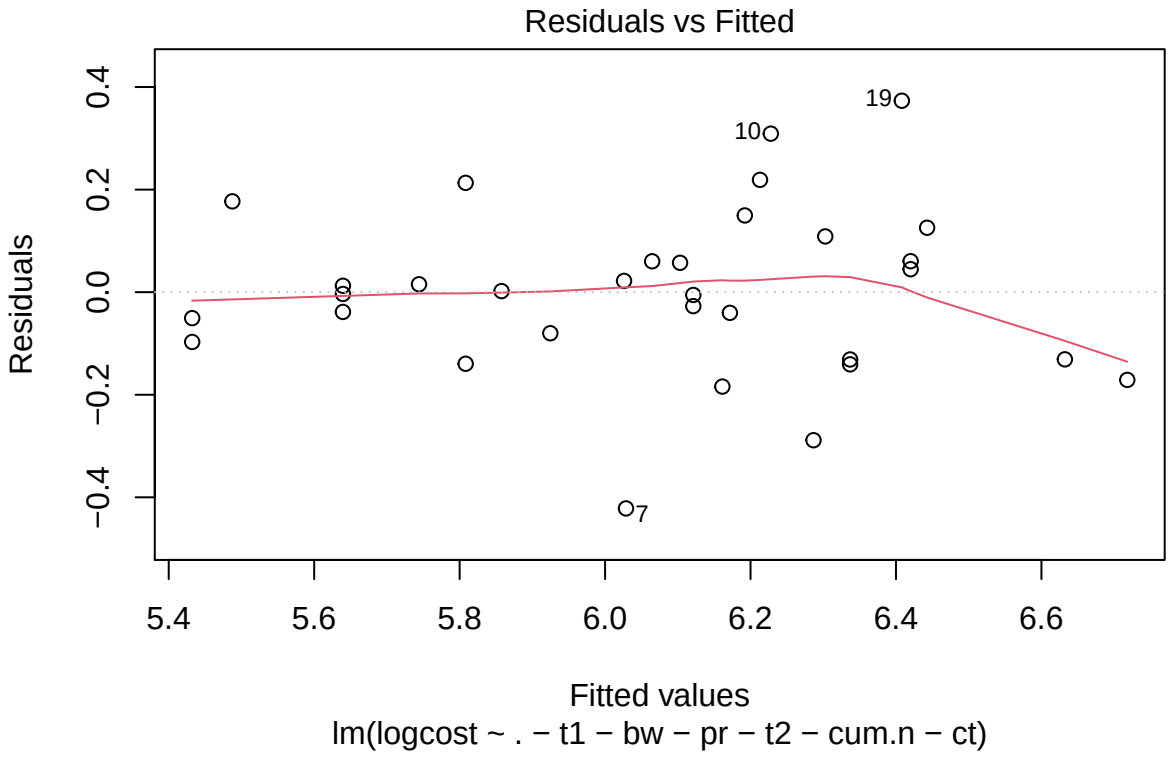
In this task, the procedure is been done manually by using the summary command to check the p-value and remove the variable with highest. By continuing this, we remove first "bw". In the next round we remove "pr", and so on. The final model is:

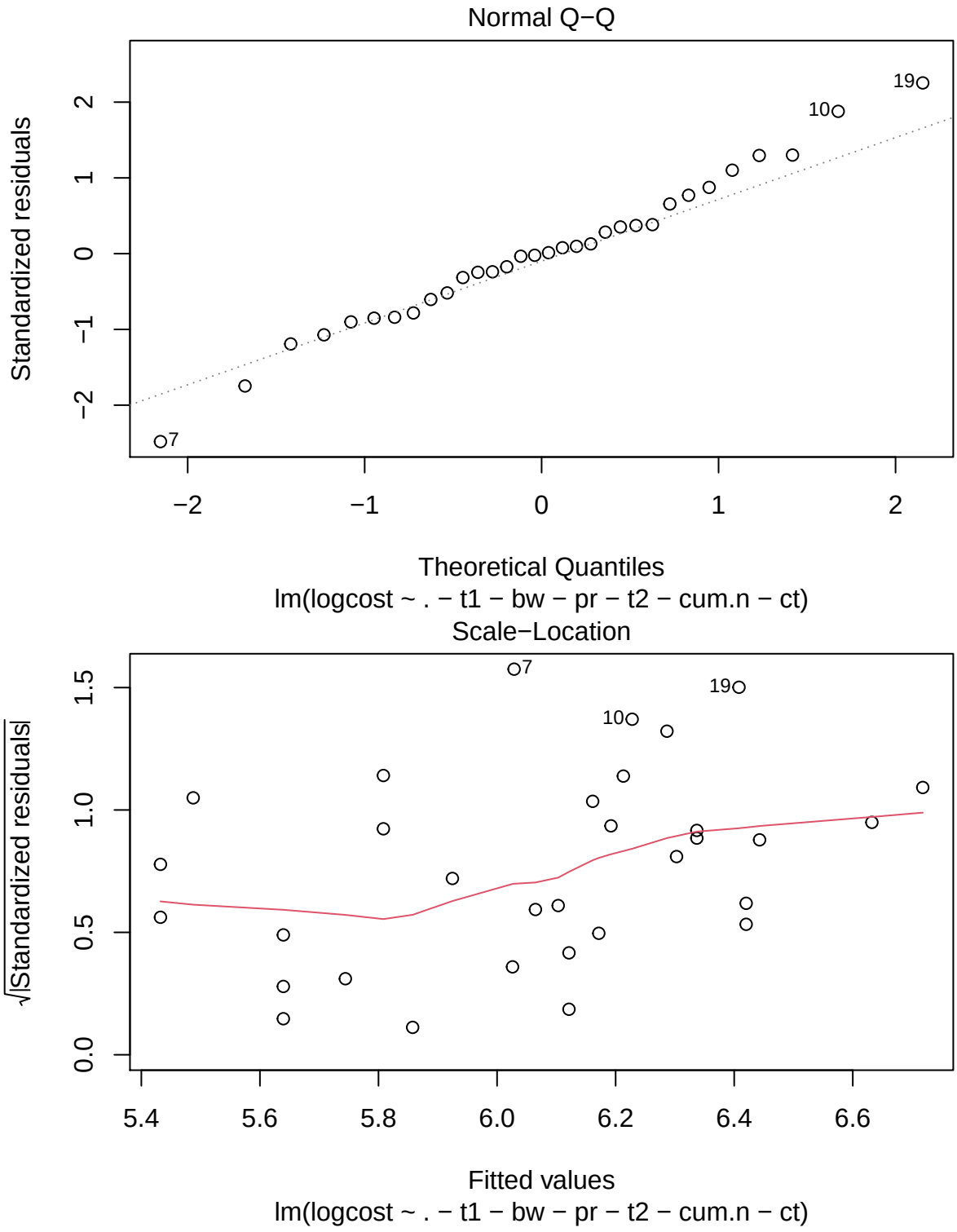
```
fit3 = lm(logcost ~.-t1 -bw -pr-t2 -cum.n -ct,data=nuclear)
summary(fit3)
```

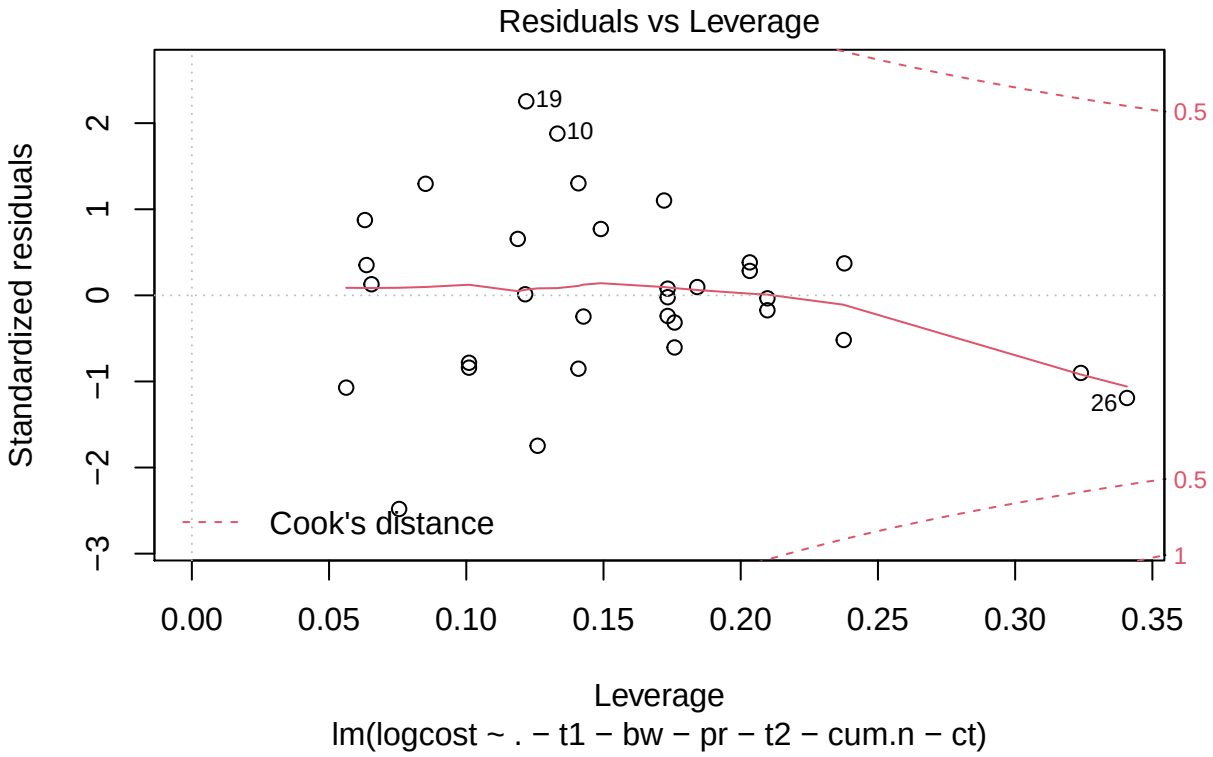
```
##
## Call:
## lm(formula = logcost ~ . - t1 - bw - pr - t2 - cum.n - ct, data = nuclear)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.42160 -0.10554 -0.00070  0.07247  0.37328
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.5035539  2.5022087  -1.800 0.083072 .
## date         0.1439104  0.0363320   3.961 0.000491 ***
```

```
## cap          0.0008783  0.0001677   5.238 1.61e-05 ***
## ne           0.2024364  0.0751953   2.692 0.012042 *
## pt          -0.3964878  0.0963356  -4.116 0.000326 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1767 on 27 degrees of freedom
## Multiple R-squared:  0.8096, Adjusted R-squared:  0.7814
## F-statistic:  28.7 on 4 and 27 DF,  p-value: 2.255e-09
```

```
plot(fit3)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(logcost ~ . − t1 − bw − pr − t2 − cum.n − ct)
Scale−Location
√|Standardized residuals|
Fitted values
lm(logcost ~ . − t1 − bw − pr − t2 − cum.n − ct)



The plots shows the linearity of the new model. There are some points on the plot that are non-linearity, especially the one at the lower and the two at the upper quantile (points nr. 7,10,19). But other than that, the new model follow normal distribution quite good. It is although difficult to say more about the non-linearity because the limited data we have.

e)

The average quadratic error is giving by:

$$MSE = \frac{1}{n}\sum_{i=n}^{n}(y_i - \hat{f}(x_i)) = \frac{1}{n}\sum_{i=n}^{n}(y_i - \hat{y}_i)$$

Here we have to predict the response (yhat), and then calculate the MSE.

```
yHat = predict(fit3,newdata = nuclear)
yi = logcost
MSE = 1/n * sum((yi - yHat)^2)
MSE
```

```
## [1] 0.02635124
```

MSE tells us the error between the predicted value and the actual value. In simple words, we can say the lower value of MSE the better model, and if the value of MSE is equal to 0 then the model is perfect since there is no error. In this case, we got 0.026 which tells that the model is quite "good". However this procedure does not tell us exactly how good the new model is and we do not get an answer for what percentage of the nuclear-data fits with the model.

f)

We start by writing the Gaussian and then the likelihood function:

$$f(y;\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i-\theta)^2}{2\sigma^2}}$$

$$L(\theta) = \prod_{i=1}^{n} f(x;\theta)$$

In order to make the equation easier to evaluate, we take the natural logarithm of it:

$$\log L(\theta) = \log(\prod_{i=1}^{n} f(x;\theta))$$

$$= \sum_{i=1}^{n} -\log(\sqrt{2\pi\sigma^2}) - \frac{(y_i-\theta)^2}{2\sigma^2} = \sum_{i=1}^{n} const - \frac{1}{2}\log\sigma^2 - \frac{(y_i-\theta)^2}{2\sigma^2}$$

$$= const - \frac{n}{2}\log\sigma^2 - n\frac{(y_i-\theta)^2}{2\sigma^2}$$

By declaring an estimate for σ^2 as $\hat{\sigma}^2 = (y_i - \theta)^2$, we get:

$$\log L(\hat{\theta}) = const - \frac{n}{2}\log(\hat{\sigma}^2) - \frac{n}{2}$$

Here n is the amount of variables we have in the model, and therefore we can add the $-n/2$ into the constant-term. We get:

$$\log L(\hat{\theta}) = const - \frac{n}{2}\log(\hat{\sigma}^2)$$

By inserting this expression into the equations for AIC and BIC, we get

$$AIC = -2\log(L(\hat{\theta})) + 2(q+2) = const + n\log(\hat{\sigma}^2) + 2(q+2)$$

$$BIC = -2\log(L(\hat{\theta})) + \log(n)(q+2) = const + n\log(\hat{\sigma}^2) + \log(n)(q+2)$$

g)

We start by AIC:

```
fullModel = lm(logcost ~.,data = nuclear)
AICModel = stepAIC(fullModel,direction = "backward", k =2)
```

```
## Start:  AIC=-105.01
## logcost ~ date + t1 + t2 + cap + pr + ne + ct + bw + cum.n +
##     pt
##
##         Df Sum of Sq     RSS      AIC
## - t1     1   0.00160 0.60603 -106.930
## - bw     1   0.00345 0.60788 -106.832
## <none>               0.60443 -105.014
## - t2     1   0.04284 0.64727 -104.823
## - pr     1   0.04826 0.65269 -104.556
## - cum.n  1   0.06792 0.67235 -103.607
## - ct     1   0.07781 0.68224 -103.140
## - pt     1   0.08337 0.68781 -102.879
## - date   1   0.19899 0.80343  -97.907
## - ne     1   0.30859 0.91302  -93.815
## - cap    1   0.68497 1.28940  -82.770
##
## Step:  AIC=-106.93
```

```
## logcost ~ date + t2 + cap + pr + ne + ct + bw + cum.n + pt
##
##         Df Sum of Sq     RSS      AIC
## - bw     1   0.00213 0.60816 -108.818
## <none>               0.60603 -106.930
## - t2     1   0.04135 0.64738 -106.818
## - pr     1   0.04680 0.65283 -106.550
## - cum.n  1   0.07045 0.67648 -105.411
## - ct     1   0.07654 0.68257 -105.124
## - pt     1   0.08216 0.68819 -104.862
## - ne     1   0.31255 0.91858  -95.621
## - date   1   0.54190 1.14793  -88.489
## - cap    1   0.68916 1.29518  -84.627
##
## Step:  AIC=-108.82
## logcost ~ date + t2 + cap + pr + ne + ct + cum.n + pt
##
##         Df Sum of Sq     RSS      AIC
## <none>               0.60816 -108.818
## - pr     1   0.05738 0.66554 -107.932
## - t2     1   0.06379 0.67195 -107.626
## - cum.n  1   0.06839 0.67656 -107.407
## - ct     1   0.07440 0.68257 -107.124
## - pt     1   0.08066 0.68882 -106.832
## - ne     1   0.31375 0.92192  -97.505
## - date   1   0.54592 1.15408  -90.318
## - cap    1   0.68739 1.29556  -86.617
```

BIC:

```
BICModel = stepAIC(fullModel,direction = "backward", k =log(n))
```

```
## Start:  AIC=-88.89
## logcost ~ date + t1 + t2 + cap + pr + ne + ct + bw + cum.n +
##     pt
##
##         Df Sum of Sq     RSS     AIC
## - t1     1   0.00160 0.60603 -92.273
## - bw     1   0.00345 0.60788 -92.175
## - t2     1   0.04284 0.64727 -90.166
## - pr     1   0.04826 0.65269 -89.899
## - cum.n  1   0.06792 0.67235 -88.949
## <none>               0.60443 -88.891
## - ct     1   0.07781 0.68224 -88.482
## - pt     1   0.08337 0.68781 -88.222
## - date   1   0.19899 0.80343 -83.250
## - ne     1   0.30859 0.91302 -79.158
## - cap    1   0.68497 1.28940 -68.113
##
## Step:  AIC=-92.27
## logcost ~ date + t2 + cap + pr + ne + ct + bw + cum.n + pt
##
##         Df Sum of Sq     RSS     AIC
## - bw     1   0.00213 0.60816 -95.626
## - t2     1   0.04135 0.64738 -93.626
```

```
## - pr     1   0.04680 0.65283 -93.358
## <none>               0.60603 -92.273
## - cum.n  1   0.07045 0.67648 -92.219
## - ct     1   0.07654 0.68257 -91.933
## - pt     1   0.08216 0.68819 -91.670
## - ne     1   0.31255 0.91858 -82.430
## - date   1   0.54190 1.14793 -75.297
## - cap    1   0.68916 1.29518 -71.435
##
## Step:  AIC=-95.63
## logcost ~ date + t2 + cap + pr + ne + ct + cum.n + pt
##
##         Df Sum of Sq     RSS     AIC
## - pr     1   0.05738 0.66554 -96.207
## - t2     1   0.06379 0.67195 -95.900
## - cum.n  1   0.06839 0.67656 -95.681
## <none>               0.60816 -95.626
## - ct     1   0.07440 0.68257 -95.398
## - pt     1   0.08066 0.68882 -95.106
## - ne     1   0.31375 0.92192 -85.779
## - date   1   0.54592 1.15408 -78.592
## - cap    1   0.68739 1.29556 -74.892
##
## Step:  AIC=-96.21
## logcost ~ date + t2 + cap + ne + ct + cum.n + pt
##
##         Df Sum of Sq     RSS     AIC
## - t2     1   0.02447 0.69001 -98.517
## - cum.n  1   0.05351 0.71905 -97.198
## <none>               0.66554 -96.207
## - ct     1   0.10237 0.76791 -95.094
## - pt     1   0.12015 0.78570 -94.361
## - ne     1   0.28784 0.95339 -88.171
## - date   1   0.49109 1.15664 -81.987
## - cap    1   0.68019 1.34573 -77.141
##
## Step:  AIC=-98.52
## logcost ~ date + cap + ne + ct + cum.n + pt
##
##         Df Sum of Sq     RSS     AIC
## - cum.n  1   0.06006 0.75007 -99.312
## <none>               0.69001 -98.517
## - pt     1   0.11719 0.80720 -96.963
## - ct     1   0.12931 0.81932 -96.486
## - ne     1   0.27215 0.96216 -91.343
## - date   1   0.46672 1.15673 -85.450
## - cap    1   0.89456 1.58457 -75.379
##
## Step:  AIC=-99.31
## logcost ~ date + cap + ne + ct + pt
##
##        Df Sum of Sq     RSS     AIC
## <none>              0.75007 -99.312
## - ct    1   0.09317 0.84324 -99.031
```

```
## - ne    1   0.21478 0.96485 -94.720
## - pt    1   0.37487 1.12494 -89.807
## - date  1   0.55668 1.30675 -85.013
## - cap   1   0.83451 1.58458 -78.845
```

We end up with following models:

```
summary(AICModel)
```

```
##
## Call:
## lm(formula = logcost ~ date + t2 + cap + pr + ne + ct + cum.n +
##     pt, data = nuclear)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.290513 -0.082692  0.008663  0.098259  0.260204
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.169e+01  3.748e+00  -3.118 0.004839 **
## date         2.438e-01  5.366e-02   4.544 0.000145 ***
## t2           6.018e-03  3.874e-03   1.553 0.134024
## cap          8.739e-04  1.714e-04   5.099 3.65e-05 ***
## pr          -1.099e-01  7.458e-02  -1.473 0.154275
## ne           2.611e-01  7.580e-02   3.445 0.002206 **
## ct           1.111e-01  6.622e-02   1.677 0.106991
## cum.n       -1.176e-02  7.311e-03  -1.608 0.121414
## pt          -2.071e-01  1.186e-01  -1.747 0.094059 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1626 on 23 degrees of freedom
## Multiple R-squared:  0.8627, Adjusted R-squared:  0.8149
## F-statistic: 18.06 on 8 and 23 DF,  p-value: 3.307e-08
```

```
summary(BICModel)
```

```
##
## Call:
## lm(formula = logcost ~ date + cap + ne + ct + pt, data = nuclear)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36175 -0.08063  0.00584  0.08594  0.42355
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.4058393  2.4567323  -2.200 0.036861 *
## date         0.1563992  0.0356036   4.393 0.000167 ***
## cap          0.0008674  0.0001613   5.378 1.24e-05 ***
## ne           0.1973468  0.0723259   2.729 0.011252 *
## ct           0.1154229  0.0642266   1.797 0.083943 .
## pt          -0.3477717  0.0964752  -3.605 0.001299 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.1698 on 26 degrees of freedom
## Multiple R-squared:  0.8306, Adjusted R-squared:  0.798
## F-statistic:  25.5 on 5 and 26 DF,  p-value: 2.958e-09
```

Both methods start with models with all explanatory variables, but decrease with one variable at a time until it has reached the best solution. The variables that we are left with after both methods has run:

- AIC: date + t2 + cap + pr + ne + ct + cum.n + pt
- BIC: date + cap + ne + ct + pt

h)

The order between AIC and BIC is the same when stepAIC runs the selection. This is due to the penalty term, which is defined as a given limit multiplied by a specific constant. In other words, the penalty term is the same for both AIC and BIC within the same dimension. In addition, both models have the same log-likelihood value, and since order is determined by log-likelihood, it makes sense that both models have the same order.

i)

From task 1d) and 1g) we are left with models containing the following variables:

- Manually: date,cap,ne,pt
- AIC: date + t2 + cap + pr + ne + ct + cum.n + pt
- BIC: date + cap + ne + ct + pt

The order the other variables were removed:

- Manually: -t1 -bw -pr -t2 -cum.n -ct
- AIC: -t1 -bw
- BIC: -t1 -bw -pr -t2 -cum.n

We can see that the "manually selection" has the least amount of variables, while the AIC and BIC has more. That is the case because of the manually selection occur by removing variables one by one and independently of the model as whole, while the AIC and BIC do the opposite. They select and evaluate the model as a whole. This means that even if a variable in AIC or BIC has a non-significant p-value, it is not necessarily removed and thus can lead to the model being improved as a whole. Another interesting finding is the variables and the order of those that were removed from the various models. We can see that the same variables were removed, and they were removed in the same order.

j)

We assume that $Z \sim N(\mu, \sigma^2)$ and will show that $E(e^Z) = exp(\mu + 0.5\sigma^2)$. To do that, we need little calculation. Just to make it easier, we will use moment generating functions. From STK1100, we have that a moment generating function for a stochastic variable is giving by:

$$M_Z(t) = E(e^{tZ})$$

From formula collection in STK1110/STK1100, we have the moment generating function for a normal distribution:

$$M_Y(t) = e^{\mu t} e^{\sigma^2 t^2/2}$$

The expectation value is obtained when $t = 1$ in the $M_Z(t)$ function above:

$$E(e^Z) = M_Z(1) = e^{\mu} e^{\sigma^2/2} = e^{\mu + \sigma^2/2}$$

Which is the same expression as the one we wanted to show. We can use this to argue for $\hat{\eta}_2$ since it can be obtained directly from the expression above. By inserting $\theta = \mu$, we get the estimate:

$$\hat{\eta}_2 = e^{\hat{\theta} + 0.5\sigma^2}$$

In order to obtain $\hat{\eta}_1$ from the expression $E(e^Z)$ we must say that $E(e^Z) = e^{E(Z)} = e^{\theta}$ and that is not correct. Therefore, we can conclude with the $\hat{\eta}_2$ is correct.

k)

The code goes as follow:

- First: defining the dataframe d.new
- Calculating the values for the θs for all three models
- Finding sigma-values for the three models
- Finally: calculate the ηs by the expression given above.

```
d.new = data.frame(date=70.0,t1=13,t2=50,cap=800,pr=1,ne=0,ct=0,bw=1,cum.n=8,pt=1)
thetaManually = predict(fit3,d.new)
thetaAIC = predict(AICModel,d.new)
thetaBIC = predict(BICModel,d.new)

sigmaManually = summary(fit3)$sigma
sigmaAIC = summary(AICModel)$sigma
sigmaBIC = summary(BICModel)$sigma

etaManually = exp(thetaManually + (1/2) * sigmaManually^2 )
etaAIC = exp(thetaAIC + (1/2) * sigmaAIC^2 )
etaBIC = exp(thetaBIC + (1/2) * sigmaBIC^2 )
```

From this we got:

- $\hat{\eta} = 362.10$ for manually selection
- $\hat{\eta} = 396.10$ for AIC
- $\hat{\eta} = 366.02$ for BIC

Problem 2

This script is about splitting the data from nuclear into two different parts, namely training and evaluation set. It starts by selecting a random list by the sample-function. Then two other lists of empty values are defined, RMSE.test1 and RMSE.test2. For this task we use two models, one with all variables and one with no variables. Then we use stepAIC to obtain and predict the models, and then save them in the RMSE.test1 and RMSE.test2. This happens ten times, the stepAIC is done for i = 1,. . . ,10 steps. Note that RMSE.test1 stores test set for log(cost) variable and RMSE.test2 stores for cost variable. At the last, the these are plotted as function of a sequence 0:10.

```
datadir = "http://www.uio.no/studier/emner/matnat/math/STK2100/data/"
nuclear = read.table(paste(datadir,"nuclear.dat",sep=""),header=T)
attach(nuclear)
```

```
## The following objects are masked from nuclear (pos = 3):
##
##     bw, cap, ct, cum.n, date, ne, pr, pt, t1, t2
```

```
## The following objects are masked from nuclear (pos = 4):
##
##     bw, cap, cost, ct, cum.n, date, ne, pr, pt, t1, t2
```

```
n = nrow(nuclear)
ind = sample(1:n,n/2,replace=FALSE)
RMSE.test1 = rep(NA,11)
RMSE.test2 = rep(NA,11)
model_narrow = lm(log(cost) ~ 1, data = nuclear)
```

```
model_wide = lm(log(cost) ~ ., data = nuclear)
for(i in 0:10)
{

 fit = stepAIC(model_narrow, direction="forward", steps=i,data=nuclear[ind,],
               scope=list(lower=model_narrow, upper=model_wide), k = 0)
 pred = predict(fit,nuclear[-ind,])
 RMSE.test1[i+1] = sqrt(mean((log(nuclear$cost)-pred)^2))
 RMSE.test2[i+1] = sqrt(mean((nuclear$cost-exp(pred))^2))

}
```

```
## Start:  AIC=-63.29
## log(cost) ~ 1
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
```

```
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
## + ct     1   0.03383 1.4930  -98.078
```

```
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
##
## Step:  AIC=-108.75
## log(cost) ~ pt + cap + date
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
## + ct     1   0.03383 1.4930  -98.078
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
##
## Step:  AIC=-108.75
```

```
## log(cost) ~ pt + cap + date
##
##         Df Sum of Sq     RSS     AIC
## + ne     1  0.226351 0.84324 -116.36
## + ct     1  0.104740 0.96485 -112.05
## + t2     1  0.024659 1.04493 -109.50
## + pr     1  0.018471 1.05112 -109.31
## + bw     1  0.016965 1.05263 -109.26
## + t1     1  0.006412 1.06318 -108.94
## + cum.n  1  0.001211 1.06838 -108.79
## <none>               1.06959 -108.75
##
## Step:  AIC=-116.36
## log(cost) ~ pt + cap + date + ne
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
```

```
## + ct     1   0.03383 1.4930  -98.078
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
##
## Step:  AIC=-108.75
## log(cost) ~ pt + cap + date
##
##         Df Sum of Sq     RSS     AIC
## + ne     1  0.226351 0.84324 -116.36
## + ct     1  0.104740 0.96485 -112.05
## + t2     1  0.024659 1.04493 -109.50
## + pr     1  0.018471 1.05112 -109.31
## + bw     1  0.016965 1.05263 -109.26
## + t1     1  0.006412 1.06318 -108.94
## + cum.n  1  0.001211 1.06838 -108.79
## <none>               1.06959 -108.75
##
## Step:  AIC=-116.36
## log(cost) ~ pt + cap + date + ne
##
##         Df Sum of Sq     RSS     AIC
## + ct     1  0.093171 0.75007 -120.11
## + t2     1  0.053311 0.78993 -118.45
## + cum.n  1  0.023915 0.81932 -117.28
## + pr     1  0.015679 0.82756 -116.96
## + bw     1  0.014710 0.82853 -116.92
## + t1     1  0.008887 0.83435 -116.70
## <none>               0.84324 -116.36
##
## Step:  AIC=-120.11
## log(cost) ~ pt + cap + date + ne + ct
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
```

```
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
## + ct     1   0.03383 1.4930  -98.078
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
##
## Step:  AIC=-108.75
## log(cost) ~ pt + cap + date
##
##         Df Sum of Sq     RSS     AIC
## + ne     1  0.226351 0.84324 -116.36
## + ct     1  0.104740 0.96485 -112.05
## + t2     1  0.024659 1.04493 -109.50
## + pr     1  0.018471 1.05112 -109.31
## + bw     1  0.016965 1.05263 -109.26
## + t1     1  0.006412 1.06318 -108.94
## + cum.n  1  0.001211 1.06838 -108.79
## <none>               1.06959 -108.75
##
## Step:  AIC=-116.36
## log(cost) ~ pt + cap + date + ne
##
##         Df Sum of Sq     RSS     AIC
## + ct     1  0.093171 0.75007 -120.11
## + t2     1  0.053311 0.78993 -118.45
## + cum.n  1  0.023915 0.81932 -117.28
## + pr     1  0.015679 0.82756 -116.96
## + bw     1  0.014710 0.82853 -116.92
## + t1     1  0.008887 0.83435 -116.70
## <none>               0.84324 -116.36
##
## Step:  AIC=-120.11
## log(cost) ~ pt + cap + date + ne + ct
##
##         Df Sum of Sq     RSS     AIC
```

```
## + cum.n  1  0.060059 0.69001 -122.78
## + t2     1  0.031017 0.71905 -121.46
## + bw     1  0.012904 0.73716 -120.66
## + pr     1  0.008847 0.74122 -120.49
## + t1     1  0.002225 0.74784 -120.20
## <none>              0.75007 -120.11
##
## Step:  AIC=-122.78
## log(cost) ~ pt + cap + date + ne + ct + cum.n
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
## + ct     1   0.03383 1.4930  -98.078
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
```

```
##
## Step:  AIC=-108.75
## log(cost) ~ pt + cap + date
##
##         Df Sum of Sq     RSS     AIC
## + ne     1  0.226351 0.84324 -116.36
## + ct     1  0.104740 0.96485 -112.05
## + t2     1  0.024659 1.04493 -109.50
## + pr     1  0.018471 1.05112 -109.31
## + bw     1  0.016965 1.05263 -109.26
## + t1     1  0.006412 1.06318 -108.94
## + cum.n  1  0.001211 1.06838 -108.79
## <none>               1.06959 -108.75
##
## Step:  AIC=-116.36
## log(cost) ~ pt + cap + date + ne
##
##         Df Sum of Sq     RSS     AIC
## + ct     1  0.093171 0.75007 -120.11
## + t2     1  0.053311 0.78993 -118.45
## + cum.n  1  0.023915 0.81932 -117.28
## + pr     1  0.015679 0.82756 -116.96
## + bw     1  0.014710 0.82853 -116.92
## + t1     1  0.008887 0.83435 -116.70
## <none>               0.84324 -116.36
##
## Step:  AIC=-120.11
## log(cost) ~ pt + cap + date + ne + ct
##
##         Df Sum of Sq     RSS     AIC
## + cum.n  1  0.060059 0.69001 -122.78
## + t2     1  0.031017 0.71905 -121.46
## + bw     1  0.012904 0.73716 -120.66
## + pr     1  0.008847 0.74122 -120.49
## + t1     1  0.002225 0.74784 -120.20
## <none>               0.75007 -120.11
##
## Step:  AIC=-122.78
## log(cost) ~ pt + cap + date + ne + ct + cum.n
##
##        Df Sum of Sq     RSS     AIC
## + bw    1 0.0266829 0.66333 -124.04
## + t2    1 0.0244663 0.66554 -123.93
## + pr    1 0.0180560 0.67195 -123.63
## + t1    1 0.0089848 0.68102 -123.20
## <none>              0.69001 -122.78
##
## Step:  AIC=-124.04
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
```

```
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
## + ct     1   0.03383 1.4930  -98.078
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
##
## Step:  AIC=-108.75
## log(cost) ~ pt + cap + date
##
##         Df Sum of Sq     RSS     AIC
## + ne     1  0.226351 0.84324 -116.36
## + ct     1  0.104740 0.96485 -112.05
## + t2     1  0.024659 1.04493 -109.50
## + pr     1  0.018471 1.05112 -109.31
## + bw     1  0.016965 1.05263 -109.26
## + t1     1  0.006412 1.06318 -108.94
## + cum.n  1  0.001211 1.06838 -108.79
## <none>               1.06959 -108.75
##
```

```
## Step:  AIC=-116.36
## log(cost) ~ pt + cap + date + ne
##
##         Df Sum of Sq     RSS     AIC
## + ct     1  0.093171 0.75007 -120.11
## + t2     1  0.053311 0.78993 -118.45
## + cum.n  1  0.023915 0.81932 -117.28
## + pr     1  0.015679 0.82756 -116.96
## + bw     1  0.014710 0.82853 -116.92
## + t1     1  0.008887 0.83435 -116.70
## <none>               0.84324 -116.36
##
## Step:  AIC=-120.11
## log(cost) ~ pt + cap + date + ne + ct
##
##         Df Sum of Sq     RSS     AIC
## + cum.n  1  0.060059 0.69001 -122.78
## + t2     1  0.031017 0.71905 -121.46
## + bw     1  0.012904 0.73716 -120.66
## + pr     1  0.008847 0.74122 -120.49
## + t1     1  0.002225 0.74784 -120.20
## <none>               0.75007 -120.11
##
## Step:  AIC=-122.78
## log(cost) ~ pt + cap + date + ne + ct + cum.n
##
##        Df Sum of Sq     RSS     AIC
## + bw    1 0.0266829 0.66333 -124.04
## + t2    1 0.0244663 0.66554 -123.93
## + pr    1 0.0180560 0.67195 -123.63
## + t1    1 0.0089848 0.68102 -123.20
## <none>              0.69001 -122.78
##
## Step:  AIC=-124.04
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw
##
##        Df Sum of Sq     RSS     AIC
## + pr    1  0.015945 0.64738 -124.82
## + t2    1  0.010501 0.65283 -124.55
## + t1    1  0.000179 0.66315 -124.05
## <none>              0.66333 -124.04
##
## Step:  AIC=-124.82
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw + pr
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
```

```
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
## + ct     1   0.03383 1.4930  -98.078
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
##
## Step:  AIC=-108.75
## log(cost) ~ pt + cap + date
##
##         Df Sum of Sq     RSS     AIC
## + ne     1  0.226351 0.84324 -116.36
## + ct     1  0.104740 0.96485 -112.05
## + t2     1  0.024659 1.04493 -109.50
## + pr     1  0.018471 1.05112 -109.31
## + bw     1  0.016965 1.05263 -109.26
## + t1     1  0.006412 1.06318 -108.94
## + cum.n  1  0.001211 1.06838 -108.79
## <none>               1.06959 -108.75
##
## Step:  AIC=-116.36
## log(cost) ~ pt + cap + date + ne
##
##         Df Sum of Sq     RSS     AIC
## + ct     1  0.093171 0.75007 -120.11
```

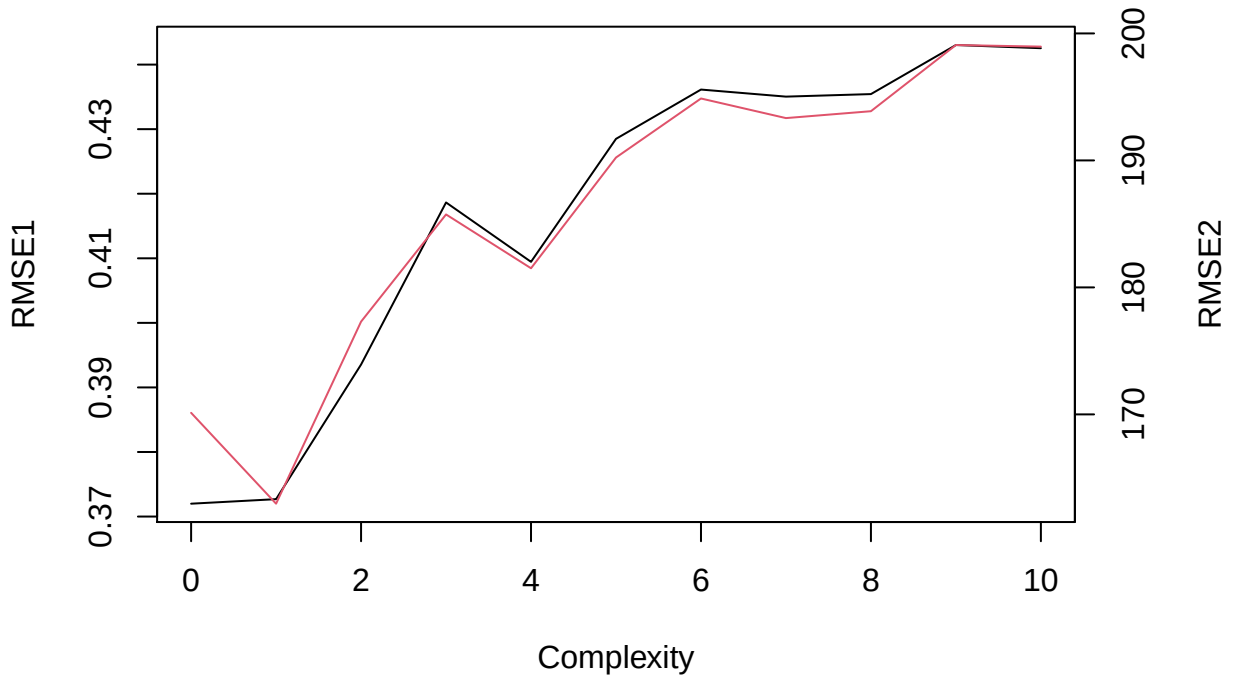
```
## + t2     1  0.053311 0.78993 -118.45
## + cum.n  1  0.023915 0.81932 -117.28
## + pr     1  0.015679 0.82756 -116.96
## + bw     1  0.014710 0.82853 -116.92
## + t1     1  0.008887 0.83435 -116.70
## <none>               0.84324 -116.36
##
## Step:  AIC=-120.11
## log(cost) ~ pt + cap + date + ne + ct
##
##         Df Sum of Sq     RSS     AIC
## + cum.n  1  0.060059 0.69001 -122.78
## + t2     1  0.031017 0.71905 -121.46
## + bw     1  0.012904 0.73716 -120.66
## + pr     1  0.008847 0.74122 -120.49
## + t1     1  0.002225 0.74784 -120.20
## <none>               0.75007 -120.11
##
## Step:  AIC=-122.78
## log(cost) ~ pt + cap + date + ne + ct + cum.n
##
##        Df Sum of Sq     RSS     AIC
## + bw    1 0.0266829 0.66333 -124.04
## + t2    1 0.0244663 0.66554 -123.93
## + pr    1 0.0180560 0.67195 -123.63
## + t1    1 0.0089848 0.68102 -123.20
## <none>              0.69001 -122.78
##
## Step:  AIC=-124.04
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw
##
##        Df Sum of Sq     RSS     AIC
## + pr    1  0.015945 0.64738 -124.82
## + t2    1  0.010501 0.65283 -124.55
## + t1    1  0.000179 0.66315 -124.05
## <none>              0.66333 -124.04
##
## Step:  AIC=-124.82
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw + pr
##
##        Df Sum of Sq     RSS     AIC
## + t2    1  0.041352 0.60603 -126.93
## + t1    1  0.000108 0.64727 -124.82
## <none>              0.64738 -124.82
##
## Step:  AIC=-126.93
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw + pr + t2
##
## Start:  AIC=-63.29
## log(cost) ~ 1
##
##         Df Sum of Sq    RSS     AIC
## + pt     1   2.01272 2.4153 -82.685
## + date   1   1.75252 2.6755 -79.411
```

```
## + t1     1   0.91394 3.5141 -70.686
## + cap    1   0.86932 3.5587 -70.283
## + ne     1   0.65915 3.7689 -68.446
## + cum.n  1   0.32635 4.1017 -65.739
## + ct     1   0.29142 4.1366 -65.467
## + bw     1   0.08878 4.3393 -63.937
## + pr     1   0.05087 4.3772 -63.658
## + t2     1   0.00581 4.4223 -63.331
## <none>               4.4281 -63.289
##
## Step:  AIC=-82.68
## log(cost) ~ pt
##
##         Df Sum of Sq    RSS     AIC
## + cap    1   0.88851 1.5268 -97.361
## + date   1   0.49197 1.9234 -89.973
## + cum.n  1   0.43596 1.9794 -89.054
## + ne     1   0.18965 2.2257 -85.301
## + t1     1   0.18163 2.2337 -85.186
## + bw     1   0.07200 2.3433 -83.653
## + ct     1   0.04550 2.3698 -83.293
## + t2     1   0.03212 2.3832 -83.113
## + pr     1   0.00261 2.4127 -82.719
## <none>               2.4153 -82.685
##
## Step:  AIC=-97.36
## log(cost) ~ pt + cap
##
##         Df Sum of Sq    RSS      AIC
## + date   1   0.45724 1.0696 -108.751
## + t1     1   0.27238 1.2545 -103.649
## + cum.n  1   0.23740 1.2894 -102.769
## + ne     1   0.19360 1.3332 -101.700
## + ct     1   0.03383 1.4930  -98.078
## + bw     1   0.02493 1.5019  -97.888
## + t2     1   0.01585 1.5110  -97.695
## + pr     1   0.01068 1.5162  -97.586
## <none>               1.5268  -97.361
##
## Step:  AIC=-108.75
## log(cost) ~ pt + cap + date
##
##         Df Sum of Sq     RSS     AIC
## + ne     1  0.226351 0.84324 -116.36
## + ct     1  0.104740 0.96485 -112.05
## + t2     1  0.024659 1.04493 -109.50
## + pr     1  0.018471 1.05112 -109.31
## + bw     1  0.016965 1.05263 -109.26
## + t1     1  0.006412 1.06318 -108.94
## + cum.n  1  0.001211 1.06838 -108.79
## <none>               1.06959 -108.75
##
## Step:  AIC=-116.36
## log(cost) ~ pt + cap + date + ne
```

```
##
##         Df Sum of Sq     RSS     AIC
## + ct     1  0.093171 0.75007 -120.11
## + t2     1  0.053311 0.78993 -118.45
## + cum.n  1  0.023915 0.81932 -117.28
## + pr     1  0.015679 0.82756 -116.96
## + bw     1  0.014710 0.82853 -116.92
## + t1     1  0.008887 0.83435 -116.70
## <none>               0.84324 -116.36
##
## Step:  AIC=-120.11
## log(cost) ~ pt + cap + date + ne + ct
##
##         Df Sum of Sq     RSS     AIC
## + cum.n  1  0.060059 0.69001 -122.78
## + t2     1  0.031017 0.71905 -121.46
## + bw     1  0.012904 0.73716 -120.66
## + pr     1  0.008847 0.74122 -120.49
## + t1     1  0.002225 0.74784 -120.20
## <none>               0.75007 -120.11
##
## Step:  AIC=-122.78
## log(cost) ~ pt + cap + date + ne + ct + cum.n
##
##        Df Sum of Sq     RSS     AIC
## + bw    1 0.0266829 0.66333 -124.04
## + t2    1 0.0244663 0.66554 -123.93
## + pr    1 0.0180560 0.67195 -123.63
## + t1    1 0.0089848 0.68102 -123.20
## <none>              0.69001 -122.78
##
## Step:  AIC=-124.04
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw
##
##        Df Sum of Sq     RSS     AIC
## + pr    1  0.015945 0.64738 -124.82
## + t2    1  0.010501 0.65283 -124.55
## + t1    1  0.000179 0.66315 -124.05
## <none>              0.66333 -124.04
##
## Step:  AIC=-124.82
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw + pr
##
##        Df Sum of Sq     RSS     AIC
## + t2    1  0.041352 0.60603 -126.93
## + t1    1  0.000108 0.64727 -124.82
## <none>              0.64738 -124.82
##
## Step:  AIC=-126.93
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw + pr + t2
##
##        Df Sum of Sq     RSS     AIC
## + t1    1 0.0015964 0.60443 -127.01
## <none>              0.60603 -126.93
```

```
##
## Step:  AIC=-127.01
## log(cost) ~ pt + cap + date + ne + ct + cum.n + bw + pr + t2 +
##     t1
```

```
par(mar = c(5, 4, 4, 4) + 0.3)  # Leave space for z axis
plot(0:10,RMSE.test1,xlab="Complexity",ylab="RMSE1",type="l") # first plot
par(new = TRUE)
plot(0:10,RMSE.test2, type = "l", axes = FALSE, bty = "n", xlab = "", ylab = "",col=2)
axis(side=4, at = pretty(range(RMSE.test2)))
mtext("RMSE2", side=4, line=3)
```



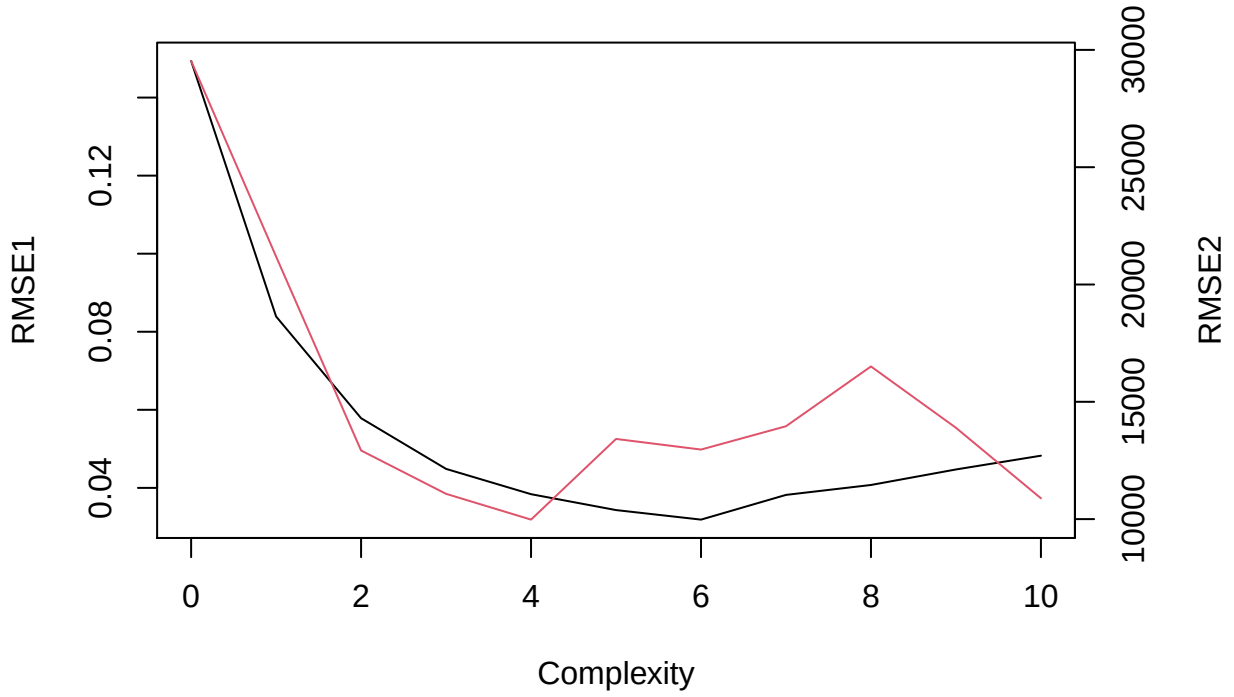
The plot of results are shown above. By running this code multiple times the values of the axis in the plot changes, meaning this approach is not very stable. One possible reason why it is like this may have something with randomness to do. Since the variable *ind* is defined with the sample () function that generates a random list at a given length.

b)

In this exercise, we are using 10-folded cross-validation. Again, like previous task, we are using a loop to fit and find the best model. Then we are storing the result for each step into the $RMSE.cv1$ and $RMSE.cv2$. This procedure is more stable than the previous one, but it is also more computationally sensitive and it uses more different models for each prediction.

```
library(lmvar)
RMSE.cv1 = rep(0,10)
RMSE.cv2 = rep(0,10)
for(i in 0:10)
{
fit = stepAIC(model_narrow, direction="forward", steps=i,k=0,
        scope=list(lower=model_narrow, upper=model_wide),trace=0)
 fit = lm(formula(fit),data=nuclear,x=TRUE,y=TRUE)
 #Note: the k in the command below has a different meaning than k above!!!
 RMSE.cv1[i+1] = cv.lm(fit,k=10)$MSE$mean
```

```
 RMSE.cv2[i+1] = cv.lm(fit,k=10,log=TRUE)$MSE$mean
}
par(mar = c(5, 4, 4, 4) + 0.3)  # Leave space for z axis
plot(0:10,RMSE.cv1,xlab="Complexity",ylab="RMSE1",type="l") # first plot
par(new = TRUE)
plot(0:10,RMSE.cv2, type = "l", axes = FALSE, bty = "n", xlab = "", ylab = "",col=2)
axis(side=4, at = pretty(range(RMSE.cv2)))
mtext("RMSE2", side=4, line=3)
```



c)

In this exercise, we will modify the previous commands to perform LOOCV. To do this we need a large training set as possible. So the idea is to split the data into 32 sets.

Unfortunately, I could not get further than this. My plan was to implement my idea, but could not pursue it.

Problem 3

We will in this exercise look at linear regression with quantitative (categorical) explanatory variables. Assume we have data $(c1, y1), ..., (cn, yn)$ where $ci \in 1, ..., K$. Define for $j = 1, ..., K$

$$x_{i,j} = 1, c_i = j$$
$$x_{i,j} = 0, c_i \neq j$$

a)

We can show that the two models

$$Y_i = \beta_0 + \beta_2 x_{i,2} + ... + \beta_K x_{i,K} + \epsilon_i$$

$$Y_i = \alpha_1 x_{i,1} + ... + \alpha_K x_{i,K} + \epsilon_i$$

are equivalent simply by writing the expressions step by step.

For $c_i = j = 1$ we have

$$Y_i = \beta_0 + \epsilon_i = \alpha_1 + \epsilon_i$$

since every other xs are equal to 0 because $c_i \neq j$.

For $c_i = j = 2$ we have

$$Y_i = \beta_0 + \beta_2 + \epsilon_i = \alpha_2 + \epsilon_i$$

For $c_i = j = 3$ we have

$$Y_i = \beta_0 + \beta_3 + \epsilon_i = \alpha_3 + \epsilon_i$$

We can see a pattern here between βs and αs, and a general relationship can be

$$\beta_0 + \beta_i = \alpha_i$$

for every $i = 1, ..., K$. But not for β_1, since it is not included in the Y_i-function.

b)

The design matrix can be represented as

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1k} \\ x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nk} \end{pmatrix}$$

Then we have:

$$X^T X = \begin{pmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ x_{12} & x_{22} & \cdots & x_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1k} & x_{2k} & \cdots & x_{nk} \end{pmatrix} \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1k} \\ x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nk} \end{pmatrix}$$

Multiplication of matrices is done by multiplying rows of the left matrix with every columns of the right matrix, and end up a new matrix of size (Kxn) containing sum of each row and each columns like

$$\sum_{i=1}^{n} x_{m,i} \cdot x_{m,j}$$

in the new matrix. The $x_{m,i}$ and $x_{m,j}$ can both be equal to 1 if $c_i = j = i$, and if $i \neq j$ and/or $c_i \neq j$ then the $x_{m,i}$ and $x_{m,j}$ are not equal to 1. So there we will end up with a diagonal matrix with diagoanl elements n_j. In other words:

$$\sum_{i=1}^{n} x_{m,i} \cdot x_{m,j} = \begin{cases} 1, i=j \\ \\ 0, else \end{cases}$$

We can do the same for

$$X^T y = \begin{pmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ x_{12} & x_{22} & \cdots & x_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1k} & x_{2k} & \cdots & x_{nk} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

We get almost the same answer as above when we multiply this two, we will end up with a new vector of size (1xn) with elements such as

$$\sum_{i=1}^{n} x_{m,i} y_i$$

As mentioned above, $x_{m,i}$ will be 1 and/or 0 if $c_i = j$ and/or $c_i \neq j$. So for the j-element we get

$$\sum_{i,c_i=j} y_i$$

We use the formula (from formula collection STK1100/STK1110)

$$\hat{\alpha} = (X^T X)^{-1} X^T Y$$

to find least squares estimates for $\alpha_1, ..., \alpha_K$. We get

$$\begin{pmatrix} 1/n_1 & 0 & \cdots & 0 \\ 0 & 1/n_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/n_k \end{pmatrix} \begin{pmatrix} \sum_{i,c_i=1} y_i \\ \sum_{i,c_i=2} y_i \\ \vdots \\ \sum_{i,c_i=k} y_i \end{pmatrix}$$

$$= \begin{pmatrix} \frac{\sum_{i,c_i=1} y_i}{n_1} \\ \frac{\sum_{i,c_i=2} y_i}{n_2} \\ \vdots \\ \frac{\sum_{i,c_i=K} y_i}{n_K} \end{pmatrix}$$

So, the j-th element of the least square estimate is

$$\hat{\alpha} = \frac{1}{n_j} \sum_{i,c_i=j} y_i$$

c)

From task a), we had the following relationship between α and β:

$$\alpha_i = \beta_0 + \beta_i$$

which was true for every $i = 1, 2, ..., k$ but β_0 is always equal to 0. By rearranging the expression we get the least square estimate for

$$\hat{\beta}_i = \alpha_i - \beta_0 = \hat{\alpha}_i - \hat{\alpha}_1$$

d)

We will use same procedure as in task 1), finding relationship through writing out the expression. So for $c_i = j = 1$ we get

$$Y_i = \gamma_0 + \gamma_1 + \epsilon_i$$

Similarly we get for $c_i = 2$ and $c_i = 3$

$$Y_i = \gamma_0 + \gamma_2 + \epsilon_i$$

$$Y_i = \gamma_0 + \gamma_3 + \epsilon_i$$

We can see a pattern here, and it can be generalized by

$$\alpha_i = \gamma_0 + \gamma_i$$

By taking the sum over all explanatory variables, we get:

$$\sum_{i=1}^{K} \alpha_i = \sum_{i=1}^{K} \gamma_0 + \sum_{i=1}^{K} \gamma_i$$

$$= K\gamma_0 + \sum_{i=1}^{K} \gamma_i$$

From the exercise, we had that $\sum_{i=1}^{K} \gamma_i = 0$, and we get an expression for

$$\gamma_0 = \frac{1}{K} \sum_{i=1}^{K} \alpha_i = \overline{\alpha}$$

The γ_j must have the values of

$$\gamma_j = \alpha_j - \gamma_0 = \alpha_j - \overline{\alpha}$$

in order for this model to be equivalent to the previous models. To obtain the same results for βs, one need to insert $\alpha_j = \beta_j + \beta_0$ in the expression above

$$\gamma_j = \beta_j + \beta_0 - \overline{\alpha} = \beta_j + \beta_0 - \frac{1}{K} \sum_{i=1}^{K} \beta_j + \beta_0$$

$$= \beta_j + \beta_0 - \frac{K}{K}\beta_0 - \sum_{i=1}^{K} \beta_j = \beta_j - \frac{1}{K} \sum_{i=1}^{K} \beta_j = \beta_j - \overline{\beta}$$

e)

Using the code provided in the exercise:

```
datadir = "http://www.uio.no/studier/emner/matnat/math/STK2100/data/"
Fe <- read.table(paste(datadir,"fe.txt",sep=""),
                 header=T,sep=",")
fit <- lm(Fe~form,data=Fe)
summary(fit)
```

```
##
## Call:
## lm(formula = Fe ~ form, data = Fe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.113  -2.580  -0.290   2.901  11.279
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  21.5050     1.6202  13.273 7.59e-16 ***
## form          2.8540     0.5916   4.824 2.30e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 4.183 on 38 degrees of freedom
## Multiple R-squared:  0.3798, Adjusted R-squared:  0.3635
## F-statistic: 23.27 on 1 and 38 DF,  p-value: 2.296e-05
```

Here we can see that R is not distinguish between the categorical variables, but we must use the other code

```
Fe$form <- as.factor(Fe$form)
fit1 <- lm(Fe~form-1,data=Fe)
summary(fit1)
```

```
##
## Call:
## lm(formula = Fe ~ form - 1, data = Fe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.340 -1.255 -0.250  1.770 10.360
##
## Coefficients:
##       Estimate Std. Error t value Pr(>|t|)
## form1   26.080      1.251   20.85   <2e-16 ***
## form2   24.690      1.251   19.74   <2e-16 ***
## form3   29.950      1.251   23.95   <2e-16 ***
## form4   33.840      1.251   27.06   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.955 on 36 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9815
## F-statistic: 532.5 on 4 and 36 DF,  p-value: < 2.2e-16
```

Now, R has distinguished between the categorical factors and we see four variable in the regression above. Because of that we get incredibly better result. The predicted model look like the α-model since it has got no intercept.

f)

```
options()$contrasts
```

```
##         unordered           ordered
## "contr.treatment"      "contr.poly"
```

```
options(contrasts=c("contr.treatment","contr.treatment"))
fit2 <- lm(Fe~form,data=Fe)
summary(fit2)
```

```
##
## Call:
## lm(formula = Fe ~ form, data = Fe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.340 -1.255 -0.250  1.770 10.360
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   26.080      1.251  20.852  < 2e-16 ***
## form2         -1.390      1.769  -0.786   0.4371
## form3          3.870      1.769   2.188   0.0352 *
## form4          7.760      1.769   4.387  9.6e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.955 on 36 degrees of freedom
## Multiple R-squared:  0.4748, Adjusted R-squared:  0.431
## F-statistic: 10.85 on 3 and 36 DF,  p-value: 3.199e-05
```

This model looks like the β-model since it has got an intercept and no "β_1" ("form1") variable, but it has β_2, β_3 and β_4.

```
options(contrasts=c("contr.sum","contr.sum"))
options()$contrasts
```

```
## [1] "contr.sum" "contr.sum"
```

```
fit3 <- lm(Fe~form,data=Fe)
summary(fit3)
```

```
##
## Call:
## lm(formula = Fe ~ form, data = Fe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.340 -1.255 -0.250  1.770 10.360
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  28.6400     0.6254  45.798  < 2e-16 ***
## form1        -2.5600     1.0831  -2.363 0.023622 *
## form2        -3.9500     1.0831  -3.647 0.000833 ***
## form3         1.3100     1.0831   1.209 0.234375
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.955 on 36 degrees of freedom
## Multiple R-squared:  0.4748, Adjusted R-squared:  0.431
## F-statistic: 10.85 on 3 and 36 DF,  p-value: 3.199e-05
```

```
-sum(fit3$coefficients[-1])
```

```
## [1] 5.2
```

This model looks like the γ-model since it has got both the intercept and the other explanatory variables.

All of this is summarized in the table below

$$|-|Intercept|form1|form2|form3|form4|$$

$$|\alpha - model|-|26.080|24.690|29.950|33.840|$$

$$|\beta - model|26.080|-|-1.390|3.870|7.760|$$

$$|\gamma - model|28.6400|-2.5600|-3.9500|1.3100|5.2|$$

g)

In order to test the difference between different types of iron, we can do a test with null hypothesis as every variables are equal to zero against alternative hypothesis such as they are not equal to zero. We can write it as

$$H_0 : \beta_j = 0$$

against H_a : at least one $\beta_j = 0$. We perform this test for all three models.

We can use the fitted models to perform this tests. Generally, the smaller p-value a fitted model have the more evidence there is in the sample data against the null hypothesis. By observing the p-value for all three models, we can see that they are extremely small and therefore we can reject the null hypothesis. We can conclude with that there is a certain form of difference between the 4 iron types.

h)

A possible simplification of the model could have been to add together the variables that are almost the same. From the output of model we can see that form1 and form2 have almost equal t-value, and can therefore be merged since they have equal properties and it simplifies the model.