AdvStDaAn, Worksheet, Week 5

Michael Lappert

24 April, 2022

Contents

Exercise 1

Question 1 a) and b)

How do we come to this solution?

Exercise 2

```
path <- file.path('Datasets', 'Dial-a-ride.dat')
df <- read.table(path, header=TRUE)

summary(df)
```

Dataset loading and sanity check:

```
##       POP                AR                RDR               HR       
##  Min.   :  3025   Min.   :  2.300   Min.   :  56.0   Min.   : 4.00  
##  1st Qu.: 13241   1st Qu.:  4.375   1st Qu.: 202.8   1st Qu.:12.00  
##  Median : 24108   Median :  6.450   Median : 272.5   Median :12.00  
##  Mean   : 28113   Mean   : 30.993   Mean   : 415.7   Mean   :12.96  
##  3rd Qu.: 31712   3rd Qu.: 10.775   3rd Qu.: 392.5   3rd Qu.:14.50  
##  Max.   :102711   Max.   :568.000   Max.   :3400.0   Max.   :24.00  
##        VH               F               IND        
##  Min.   : 2.000   Min.   :0.0100   Min.   :0.0000  
##  1st Qu.: 3.250   1st Qu.:0.3500   1st Qu.:0.0000  
##  Median : 4.500   Median :0.5000   Median :0.0000  
##  Mean   : 6.074   Mean   :0.4404   Mean   :0.4444  
##  3rd Qu.: 6.750   3rd Qu.:0.5000   3rd Qu.:1.0000  
##  Max.   :22.000   Max.   :1.0000   Max.   :1.0000
```

```
str(df)
```

```
## 'data.frame':    54 obs. of  7 variables:
##  $ POP: num  100000 8872 17338 26170 60000 ...
##  $ AR : num  13.6 2.3 4.3 4.6 17 7 3.9 6.5 10.9 6.4 ...
##  $ RDR: int  2718 250 350 186 600 420 249 350 925 514 ...
##  $ HR : num  18.5 12 12 12 12 12 12 13 24 24 ...
##  $ VH : int  22 3 2 4 14 5 2 8 19 12 ...
##  $ F  : num  0.25 0.35 0.6 0.5 0.5 0.5 0.5 0.25 0.3 0.6 ...
##  $ IND: int  1 0 1 0 0 1 1 0 0 0 ...
```

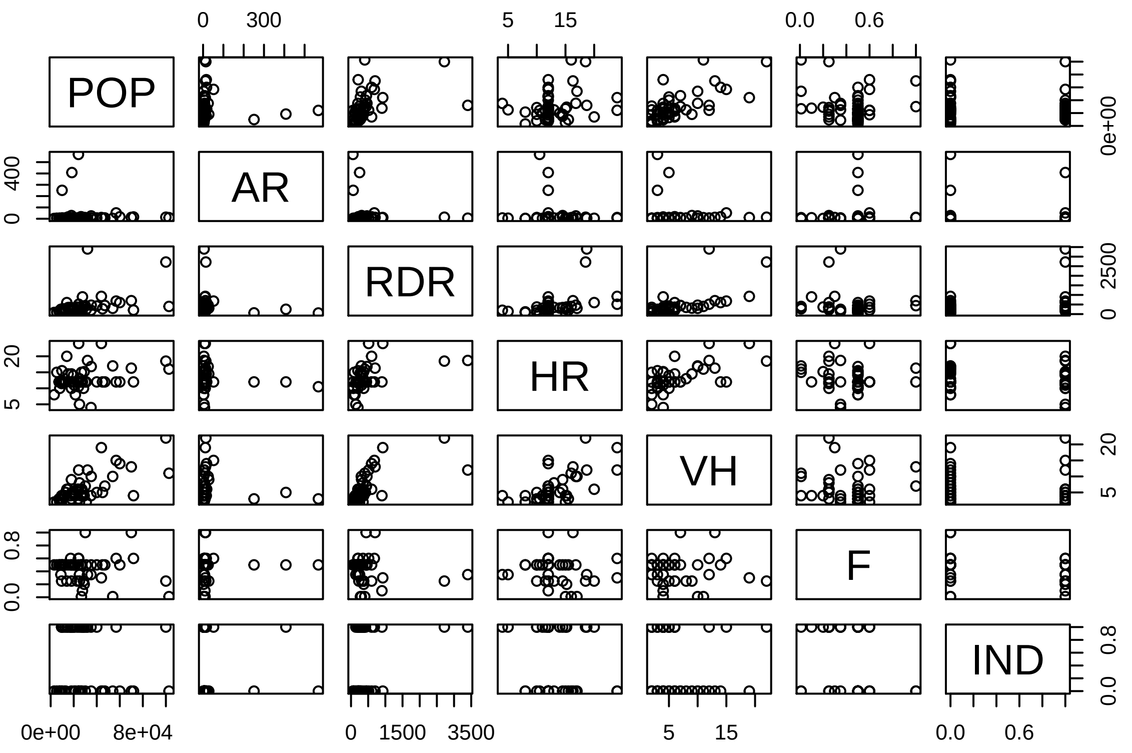
```
head(df)
```

```
##      POP   AR  RDR   HR VH    F IND
## 1 100000 13.6 2718 18.5 22 0.25   1
## 2   8872  2.3  250 12.0  3 0.35   0
## 3  17338  4.3  350 12.0  2 0.60   1
## 4  26170  4.6  186 12.0  4 0.50   0
## 5  60000 17.0  600 12.0 14 0.50   0
## 6  40000  7.0  420 12.0  5 0.50   1
```

```
tail(df)
```

```
##       POP   AR  RDR   HR VH    F IND
## 49  18000 28.0  310 14.5  9 0.25   0
## 50  29103  2.5  369 15.2  4 0.20   1
## 51 102711  9.5  400 16.0 11 0.01   0
## 52  25000  5.0  140  5.0  2 0.35   1
## 53  32000  5.0 3400 18.7 12 0.35   1
## 54  35000  7.0  200  4.0  4 0.35   1
```

```
plot(df)
```



Exercise 2.a)

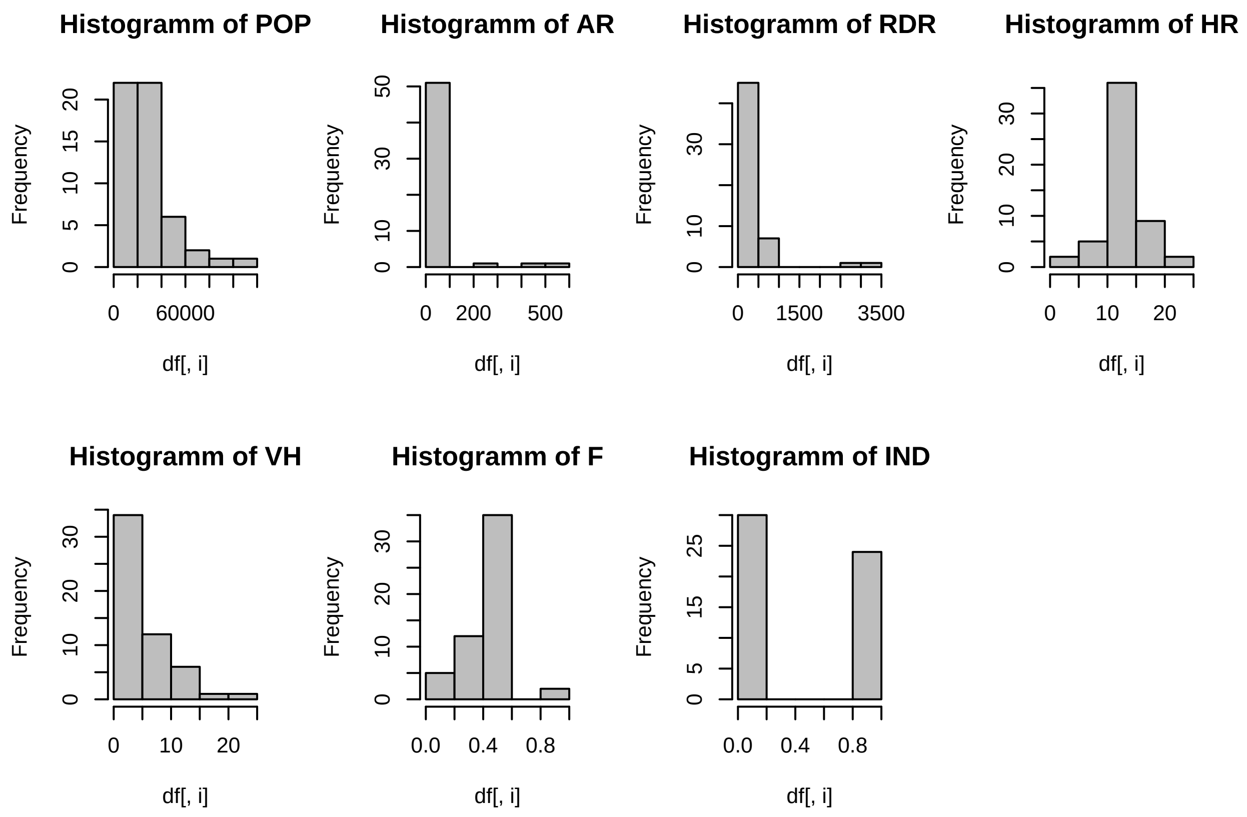
IND is a factor and should be transformed therefore.

```
df$facIND <- as.factor(df$IND)
df$IND
```

```
##  [1] 1 0 1 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 0 1 0 1 0 0 1 0 0 1 0 1 1 0 1 0
## [39] 0 0 1 0 1 1 1 0 1 0 0 1 0 1 1 1
```

Lets look at the data in histogramms:

```
par(mfrow=c(2,4))
for (i in 1:(ncol(df)-1)){
  hist(df[,i], col = 'gray',
  main = paste('Histogramm of', names(df)[i]))
}
```



Some of the variables seem to have values out of the normal range. Let's find out which:

```
which((df$AR > 200) | (df$RDR > 1500))
```

```
## [1]  1 33 35 40 53
```

Exercise 2.b)

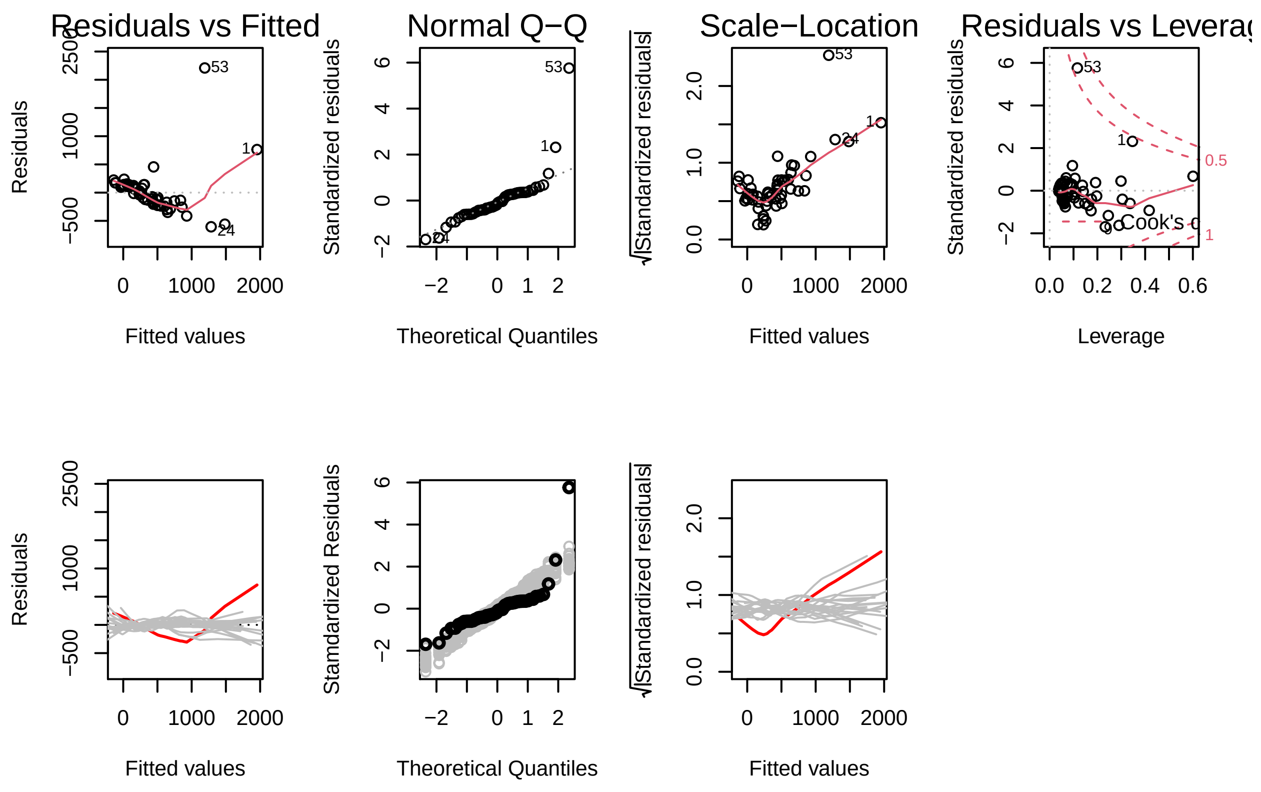
Fitting an ordinary linear regression model to all the data without any transformations:

```
lm2.1 <- lm(RDR ~ POP + AR + HR + VH + F + facIND, data = df)
summary(lm2.1)
```

```
##
## Call:
## lm(formula = RDR ~ POP + AR + HR + VH + F + facIND, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -605.35 -186.66  -55.73  129.47 2208.95
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.242e+02  3.263e+02  -1.300  0.19989
## POP          -4.461e-04  3.910e-03  -0.114  0.90967
## AR           -1.648e-01  5.780e-01  -0.285  0.77681
## HR            1.778e+01  2.161e+01   0.823  0.41486
## VH            7.961e+01  2.399e+01   3.319  0.00175 **
## F            -3.057e+01  3.167e+02  -0.097  0.92353
## facIND1       3.533e+02  1.175e+02   3.006  0.00423 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 407.9 on 47 degrees of freedom
## Multiple R-squared:  0.5288, Adjusted R-squared:  0.4686
## F-statistic:  8.79 on 6 and 47 DF,  p-value: 1.966e-06
```

The model seems the data not to fit very adequately. But lets perform a residual and sensitivity analysis first:

```r
par(mfrow=c(2,4))
plot(lm2.1)
plot.lmSim(lm2.1, SEED = 1)
```



Interpretation:

1. Tukey-Anscombe plot: The smoother has a strong banana form and lies outside the stochastic fluctuation -> outlier in observations i=1, 53.
   => The assumption of constant expectation is violated.

2. Q-Q plot: The residuals lie until the last three observations on the r.h.s. nicely on a straight line but observations 1 and 53 are again outliers. Additionally the residuals are outside of the stochastic fluctuation.
   => The assumption of Gaussian distributed errors is violated.
3. Scale-location plot: The smoother has the strong form of a tick mark with outlier 53. The smoother lies outside of the stochastic fluctuation.
   => There is evidence against the assumption of constant variance of the residuals.
4. Residuals vs. Leverage: No observation hat Cook's Distance > 1 and would therefore be too influential.
   => No too influential (dangerous) observations

CONCLUSION: The model does not fit adequately the data.

Exercise 2.c)

Trying to improve the linear regression model by applying Tukey's First-Aid transformatinos:

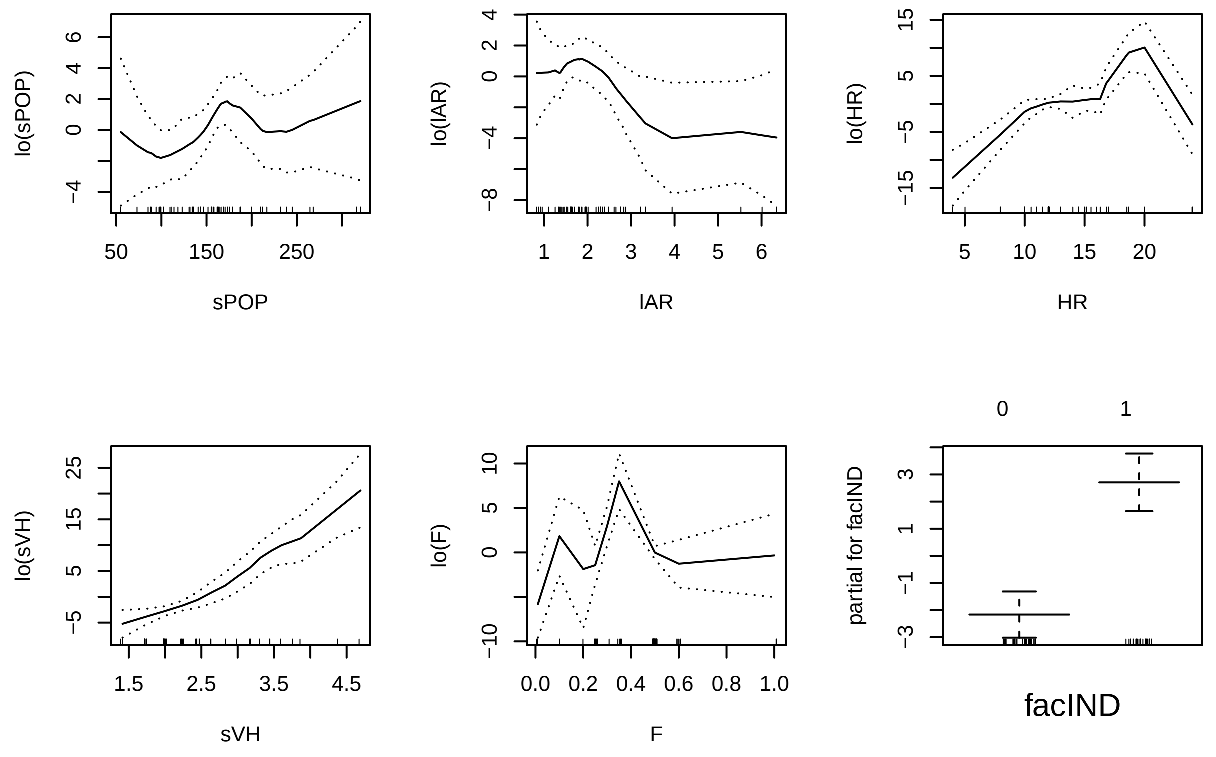
```
# Square root for counts
df$sRDR <- sqrt(df$RDR)
df$sVH <- sqrt(df$VH)
df$sPOP <- sqrt(df$POP)

# And log for continuous values
df$lAR <- log(df$AR)
```

Then using the results of additive model fitting:

```
library(gam)
```

```
## Loading required package: splines

## Loading required package: foreach

## Loaded gam 1.20
```

```
gam2.1 <- gam(sRDR ~ lo(sPOP) + lo(lAR) + lo(HR) + lo(sVH) + lo(F) + facIND,
              data = df, bf.maxit = 100)
par(mfrow=c(2,3))
plot(gam2.1, se = TRUE)
```



The plots do not look very promissing: A straight line could not be drawn in HR and F. Therefore the model fits the data not adequately.

Lets try a robust fitting method:

```
library(robustbase)
lmrob2.1 <- lmrob(sRDR ~ sPOP + lAR + HR + sVH + F + facIND, data = df)
summary(lmrob2.1)
```

```
##
## Call:
## lmrob(formula = sRDR ~ sPOP + lAR + HR + sVH + F + facIND, data = df)
##  \--> method = "MM"
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6945 -0.7731  0.0153  1.0933 30.9145
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.933222   1.505184  -1.949  0.05731 .
## sPOP         0.011563   0.004341   2.664  0.01055 *
## lAR         -0.911592   0.142407  -6.401 6.63e-08 ***
## HR           0.332810   0.120943   2.752  0.00839 **
## sVH          5.237715   0.434643  12.051 5.58e-16 ***
## F            4.532539   1.656366   2.736  0.00874 **
## facIND1      3.772975   0.571014   6.608 3.22e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Robust residual standard error: 1.532
## Multiple R-squared:  0.898,  Adjusted R-squared:  0.885
## Convergence in 15 IRWLS iterations
##
## Robustness weights:
##  3 observations c(1,45,53) are outliers with |weight| = 0 ( < 0.0019);
##  6 weights are ~= 1. The remaining 45 ones are summarized as
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.5261  0.9006  0.9503  0.8991  0.9840  0.9977
## Algorithmic parameters:
##        tuning.chi                bb        tuning.psi        refine.tol
##         1.548e+00         5.000e-01         4.685e+00         1.000e-07
##           rel.tol         scale.tol         solve.tol       eps.outlier
##         1.000e-07         1.000e-10         1.000e-07         1.852e-03
##             eps.x warn.limit.reject warn.limit.meanrw
##         5.830e-10         5.000e-01         5.000e-01
##      nResample         max.it       best.r.s       k.fast.s          k.max
##            500             50              2              1            200
##    maxit.scale      trace.lev            mts     compute.rd fast.s.large.n
##            200              0           1000              0           2000
##                   psi           subsampling                   cov
##            "bisquare"         "nonsingular"         ".vcov.avar1"
## compute.outlier.stats
##                  "SM"
## seed : int(0)
```

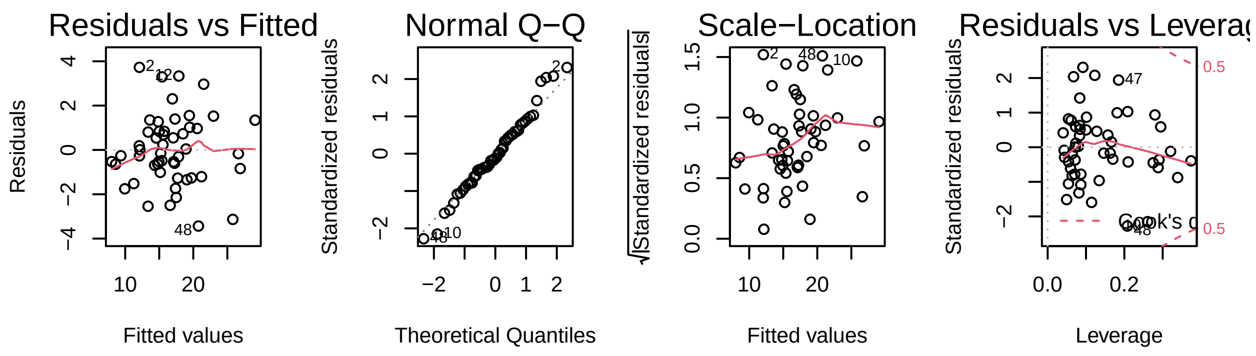
3 observations are outliers (i = 1, 45, 53) with weight = 0 (< 0.0019)

Lets try how the linear model looks when we exclude these found outliers

```
df1 <- df[-c(1, 45, 53),]
lm2.3 <- lm(sRDR ~ sPOP + lAR + HR + sVH + F + facIND, data = df1)
summary(lm2.3)
```

```
##
## Call:
## lm(formula = sRDR ~ sPOP + lAR + HR + sVH + F + facIND, data = df1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4346 -0.9263 -0.1681  0.9040  3.7273
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.674284   1.412854  -1.893 0.064971 .
## sPOP         0.011849   0.006181   1.917 0.061734 .
## lAR         -0.911550   0.217069  -4.199 0.000128 ***
## HR           0.327783   0.091483   3.583 0.000845 ***
## sVH          5.078898   0.610800   8.315 1.43e-10 ***
## F            4.774977   1.362783   3.504 0.001067 **
## facIND1      3.874551   0.509876   7.599 1.53e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.694 on 44 degrees of freedom
## Multiple R-squared:  0.8897, Adjusted R-squared:  0.8747
## F-statistic: 59.16 on 6 and 44 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,4))
plot(lm2.3)
```



This model looks adequate: No model assumptions seem to be violated.

Exercise 2.d)

Question 2.d)

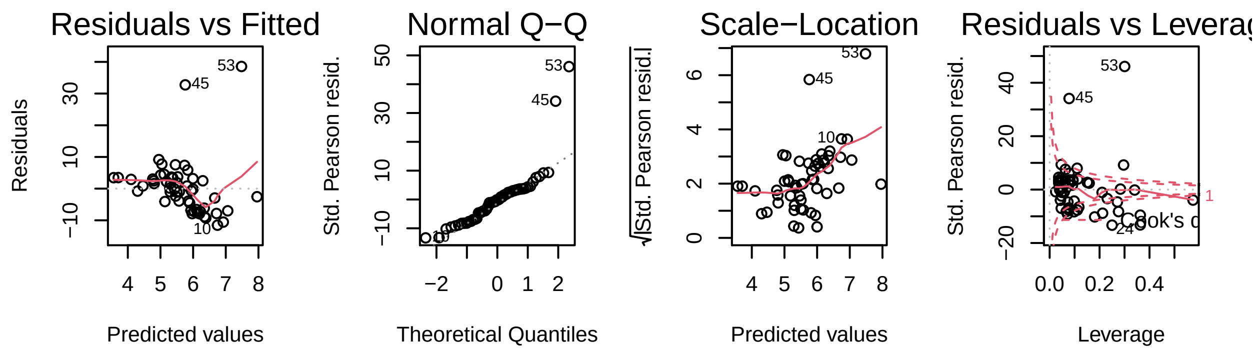
How do we get to this model? Why do we log transform all of the variables? And how do we perform the residual and sensitivity analysis (simulation does not work)?

```
df$lPOP <- log(df$POP)
df$lAR <- log(df$AR)
df$lHR <- log(df$HR)
df$lVH <- log(df$VH)
df$lF <- log(df$F)
```

```
glm2.2 <- glm(RDR ~ lPOP + lAR + lHR + lVH + lF + IND,
                  family=poisson, data=df)
summary(glm2.2)
```

```
##
## Call:
## glm(formula = RDR ~ lPOP + lAR + lHR + lVH + lF + IND, family = poisson,
##     data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -12.462   -6.015   -0.134    3.161   34.191
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.707481   0.197490   3.582  0.00034 ***
## lPOP         0.203218   0.017451  11.645  < 2e-16 ***
## lAR         -0.280124   0.010691 -26.202  < 2e-16 ***
## lHR          0.694960   0.036706  18.933  < 2e-16 ***
## lVH          0.952992   0.024028  39.662  < 2e-16 ***
## lF           0.138048   0.008055  17.137  < 2e-16 ***
## IND          0.858702   0.013794  62.253  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 20655.3  on 53  degrees of freedom
## Residual deviance:  3436.1  on 47  degrees of freedom
## AIC: 3855.3
##
## Number of Fisher Scoring iterations: 4
```

```
par(mfrow=c(2,4))
plot(glm2.2)
```



Exercise 3

```r
path <- file.path('Datasets', 'bacteria.dat')
df <- read.table(path, header=TRUE)

summary(df)
```

Dataset loading and sanity check:

```
##        N              Time     
##  Min.   : 15.0   Min.   : 1.0  
##  1st Qu.: 34.0   1st Qu.: 4.5  
##  Median : 60.0   Median : 8.0  
##  Mean   :103.9   Mean   : 8.0  
##  3rd Qu.:154.0   3rd Qu.:11.5  
##  Max.   :355.0   Max.   :15.0  
```

```r
str(df)
```

```
## 'data.frame':    15 obs. of  2 variables:
##  $ N   : int  355 211 197 166 142 106 104 60 56 38 ...
##  $ Time: int  1 2 3 4 5 6 7 8 9 10 ...
```

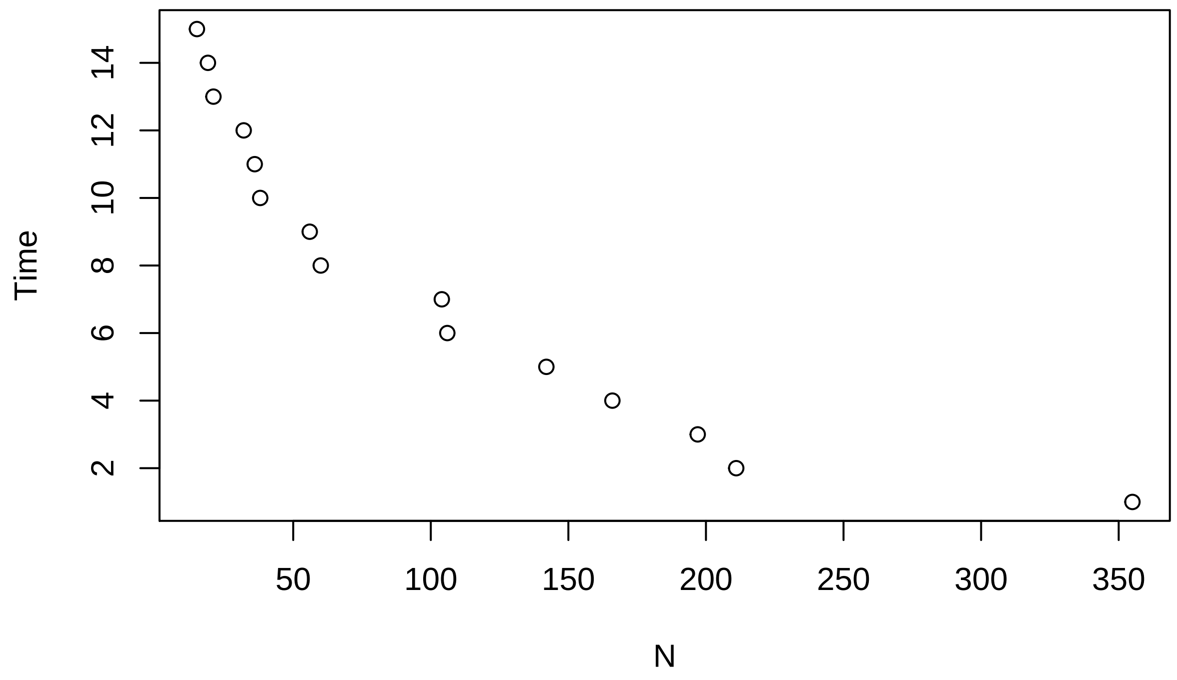
```
head(df)
```

```
##     N Time
## 1 355    1
## 2 211    2
## 3 197    3
## 4 166    4
## 5 142    5
## 6 106    6
```

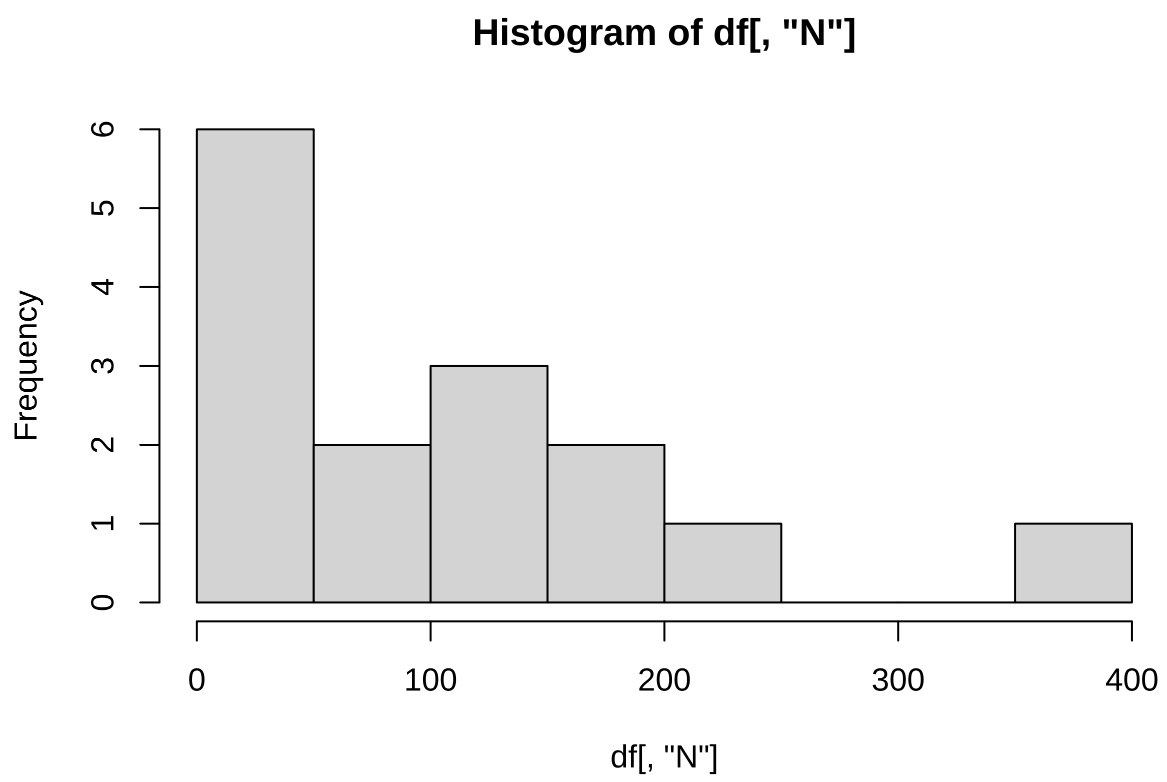
```
tail(df)
```

```
##     N Time
## 10 38   10
## 11 36   11
## 12 32   12
## 13 21   13
## 14 19   14
## 15 15   15
```

```
par(mfrow=c(1,1))
plot(df)
```



```r
hist(df[, 'N'])
```



Datset is sorted in Time and in a string decrease in N in the beginning is apparent.

Exercise 3.a)

- Response: N
- Distribution: Poisson
- Explanatory variables: Time
- Link function: log()

Exercise 3.b)

```r
glm3.1 <- glm(N ~ Time, family = poisson, data = df)
summary(glm3.1)
```

```
##
## Call:
## glm(formula = N ~ Time, family = poisson, data = df)
```

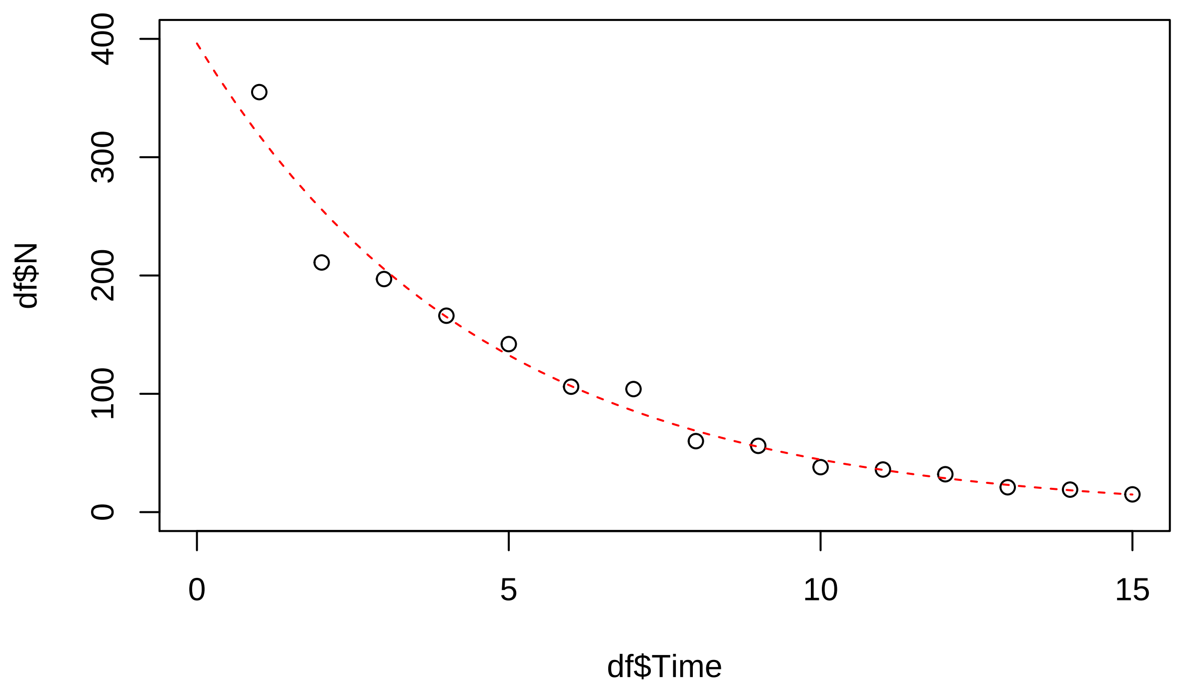
```
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.88228  -0.50786   0.05911   0.36803   2.02191
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.981772   0.041902  142.76   <2e-16 ***
## Time        -0.218920   0.007414  -29.53   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1120.101  on 14  degrees of freedom
## Residual deviance:   19.835  on 13  degrees of freedom
## AIC: 114.84
##
## Number of Fisher Scoring iterations: 4
```

We should be able to interpret the following output part: Time as explanatory variable is significant on the 5% level.

The initial amount of bacteria is $\exp(5.981772) = 396$

Lets plot the model:

```
plot(df$Time, df$N, xlim = c(0, 15), ylim = c(0, 400))
dfPreds3.1 <- data.frame(Time = seq(0, 15, length = 50))
predsGlm3.1 <- predict(glm3.1, type = 'response',
                       newdata = dfPreds3.1)
lines(dfPreds3.1$Time, predsGlm3.1, col = 'red', lty = 2)
```



Exercise 3.c)

```
names(summary(glm3.1))
```

```
##  [1] "call"           "terms"          "family"         "deviance"
##  [5] "aic"            "contrasts"      "df.residual"    "null.deviance"
##  [9] "df.null"        "iter"           "deviance.resid" "coefficients"
## [13] "aliased"        "dispersion"     "df"             "cov.unscaled"
## [17] "cov.scaled"
```

```
summary(glm3.1)$coefficients
```

```
##              Estimate  Std. Error   z value      Pr(>|z|)
## (Intercept)  5.981772 0.041901653 142.75742  0.000000e+00
## Time        -0.218920 0.007413526 -29.52982 1.192951e-191
```

```
(xx <- summary(glm3.1)$coefficients[2,1:2])
```

```
##     Estimate   Std. Error
## -0.218920038  0.007413526
```

```
xx[1] + c(-1,1)*1.96*xx[2]
```

```
## [1] -0.2334505 -0.2043895
```

```
confint(glm3.1, 2)
```

```
## Waiting for profiling to be done...
```

```
##      2.5 %     97.5 %
## -0.2335835 -0.2045186
```

Exercise 4

```
path <- file.path('Datasets', 'transactions.dat')
df <- read.table(path, header=TRUE)

summary(df)
```

Dataset loading and sanity check:

```
##       Time             Type1            Type2
##  Min.   :  48733   Min.   :     0   Min.   : 14833
##  1st Qu.: 361838   1st Qu.:  8487   1st Qu.:151559
##  Median : 558285   Median : 21395   Median :219163
##  Mean   : 660744   Mean   : 28120   Mean   :242172
##  3rd Qu.: 871246   3rd Qu.: 43726   3rd Qu.:317461
##  Max.   :2074134   Max.   :145042   Max.   :579081
```

```
str(df)
```

```
## 'data.frame':    261 obs. of  3 variables:
##  $ Time : int  239627 234827 240326 1351841 1343674 791448 911080 581843 1224988 729993 ...
##  $ Type1: int  0 0 0 51585 62300 39485 40785 24390 53832 1 ...
##  $ Type2: int  116566 165576 89944 331481 396920 308698 292478 148670 409208 279849 ...
```

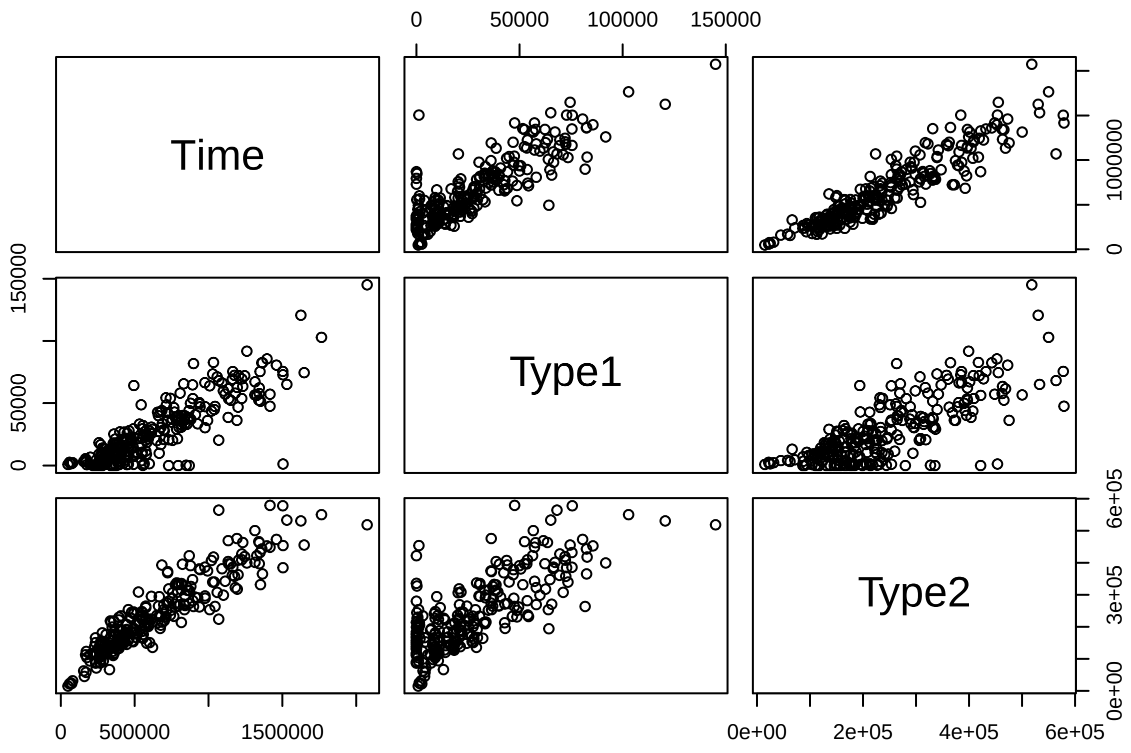
```
head(df)
```

```
##      Time Type1  Type2
## 1  239627     0 116566
## 2  234827     0 165576
## 3  240326     0  89944
## 4 1351841 51585 331481
## 5 1343674 62300 396920
## 6  791448 39485 308698
```

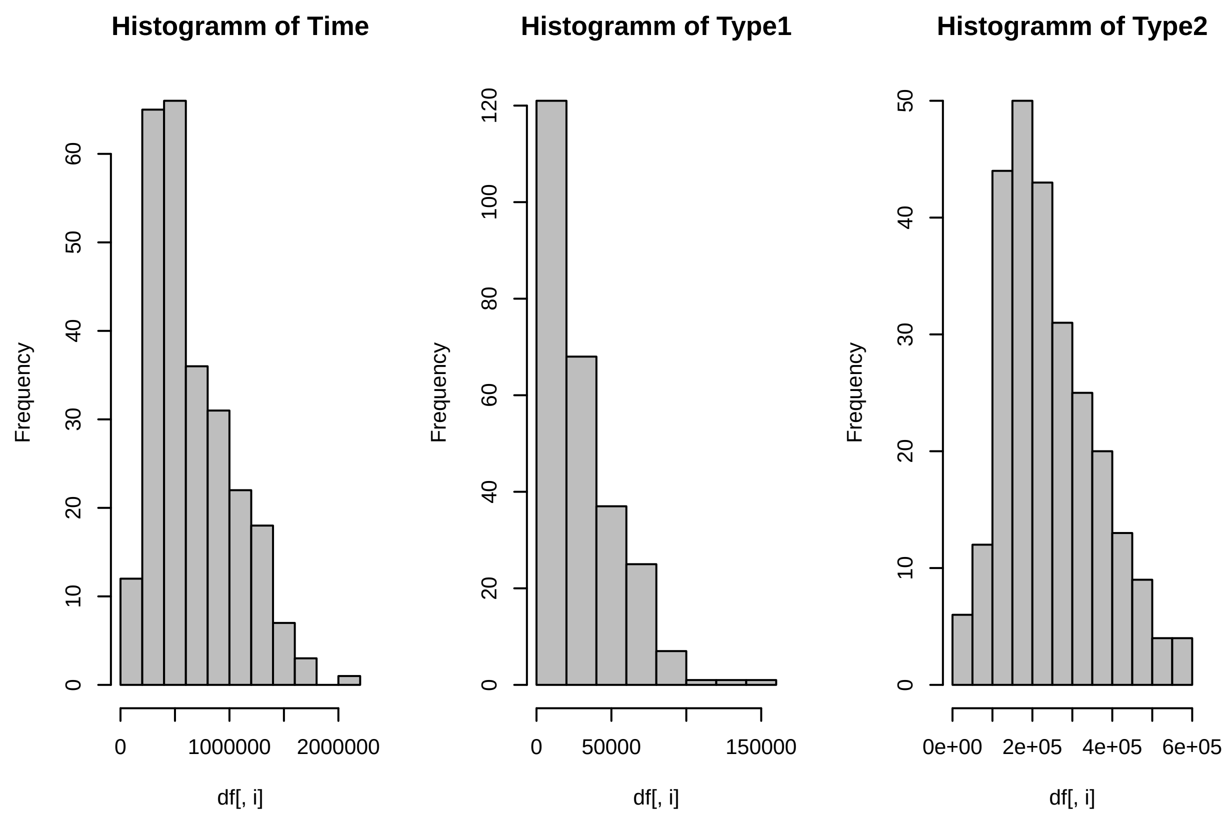
```
tail(df)
```

```
##        Time Type1  Type2
## 256  352612  8487 123703
## 257  444482 19288 191713
## 258  783815 29183 336773
## 259  574792 24493 264411
## 260  792956 36980 264408
## 261 1360991 82453 442893
```

```
par(mfrow=c(1,1))
plot(df)
```



```
par(mfrow=c(1,3))
for (i in 1:ncol(df)){
  hist(df[,i], col = 'gray',
       main = paste('Histogramm of', names(df)[i]))
}
```



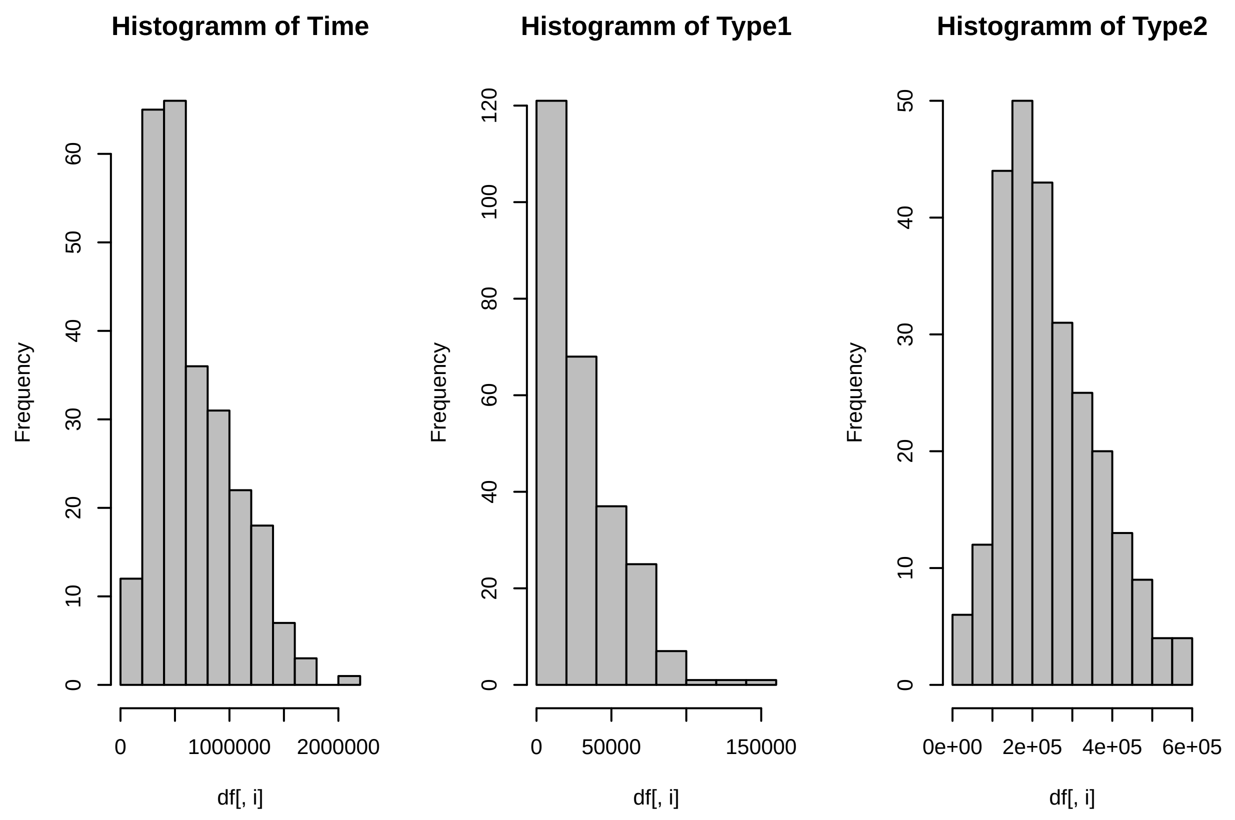
Data looks ok even Time and Type1 look kind of left skewed.

Exercise 4.a)

```
summary(df)
```

```
##       Time              Type1             Type2
##  Min.   :  48733   Min.   :     0   Min.   : 14833
##  1st Qu.: 361838   1st Qu.:  8487   1st Qu.:151559
##  Median : 558285   Median : 21395   Median :219163
##  Mean   : 660744   Mean   : 28120   Mean   :242172
##  3rd Qu.: 871246   3rd Qu.: 43726   3rd Qu.:317461
##  Max.   :2074134   Max.   :145042   Max.   :579081
```

```
par(mfrow=c(1,3))
for (i in 1:ncol(df)){
  hist(df[,i], col = 'gray',
       main = paste('Histogramm of', names(df)[i]))
}
```



Exercise 4.b)

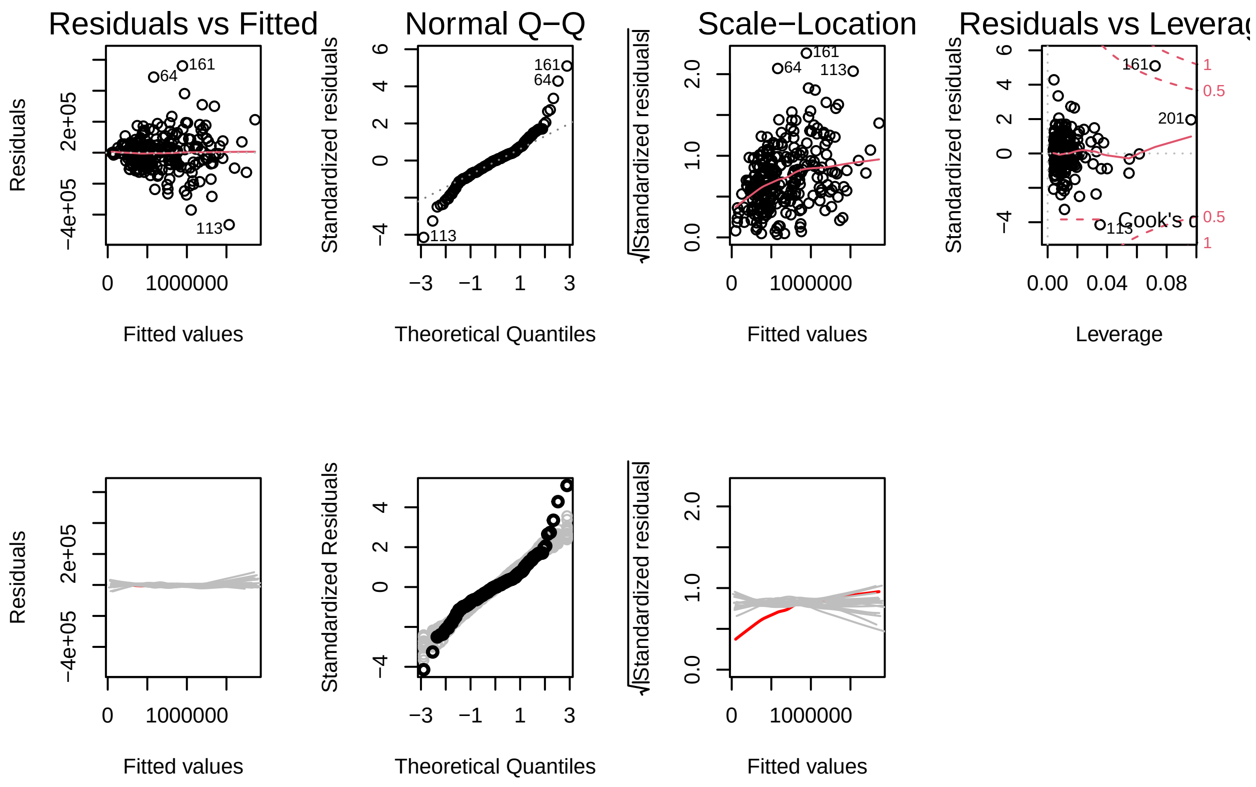
```
lm4.1 <- lm(Time ~ ., data = df)
summary(lm4.1)
```

```
##
## Call:
## lm(formula = Time ~ ., data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -465032  -59840     254   45592  560646
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.446e+04  1.705e+04   0.848    0.397
## Type1       5.463e+00  4.332e-01  12.609   <2e-16 ***
## Type2       2.034e+00  9.433e-02  21.567   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 114200 on 258 degrees of freedom
## Multiple R-squared:  0.9091, Adjusted R-squared:  0.9084
## F-statistic:  1290 on 2 and 258 DF,  p-value: < 2.2e-16
```

On the first sight the model looks not to bad with both explanatory variables as significant and an R^2 of 0.909.

But lets look at it in the residual and sensitivity analysis:

```
par(mfrow=c(2,4))
plot(lm4.1)
plot.lmSim(lm4.1, SEED = 1)
```



Interpretation:

1. Tukey-Anscombe plot: The smoother is a straight line and lies perfectly in the stochastic fluctuation. => The assumption of constant expectation is not violated.
2. Q-Q plot: The residuals deviate in on the r.h.s. and the l.h.s. from the straight line and are not within the stochastic fluctuation.
   => The assumption of Gaussian distributed errors is violated.
3. Scale-location plot: The smoother has a strong increasing trend which is outside the stochastic fluctuation.
   => The assumption of constant variance of the residuals is violated.
4. Residuals vs. Leverage: No observation hat Cook's Distance > 1 and would therefore be too influential. => No too influential (dangerous) observations.

CONCLUSION: The model does not fit adequately the data.

Exercise 4.c)

- Distribution: Gamma
- Link function: $-\frac{1}{\mu}$ ### Question 4.c) Why is the link function identity and not $-\frac{1}{\mu}$?

```
glm4.1 <- glm(Time ~ ., family = Gamma(link = identity), data = df)
summary(glm4.1)
```

```
##
## Call:
## glm(formula = Time ~ ., family = Gamma(link = identity), data = df)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.46888  -0.10719   0.00193   0.08619   0.67961
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.536e+04  5.183e+03   2.964  0.00332 **
## Type1       5.705e+00  4.257e-01  13.401  < 2e-16 ***
## Type2       2.007e+00  5.803e-02  34.582  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.02938966)
##
##     Null deviance: 92.602  on 260  degrees of freedom
## Residual deviance:  7.478  on 258  degrees of freedom
## AIC: 6725.5
##
## Number of Fisher Scoring iterations: 4
```

Exercise 5

```
path <- file.path('Datasets', 'nambeware.txt')
df <- read.table(path, header=TRUE)

summary(df)
```

Dataset loading and sanity check:

```
##      Type                Diam            Time            Price
##  Length:59          Min.   : 5.00   Min.   : 12.02   Min.   : 21.50
##  Class :character   1st Qu.: 8.25   1st Qu.: 22.21   1st Qu.: 47.25
##  Mode  :character   Median :11.00   Median : 31.46   Median : 75.00
##                     Mean   :10.93   Mean   : 35.82   Mean   : 86.38
##                     3rd Qu.:13.00   3rd Qu.: 45.03   3rd Qu.:107.00
##                     Max.   :25.00   Max.   :109.38   Max.   :260.00
```

```
str(df)
```

```
## 'data.frame':    59 obs. of  4 variables:
##  $ Type : chr  "CassDish" "CassDish" "CassDish" "Bowl" ...
##  $ Diam : num  10.7 14 9 8 10 10.5 16 15 6.5 5 ...
##  $ Time : num  47.6 63.1 58.8 34.9 55.5 ...
##  $ Price: num  144 215 105 69 134 129 155 99 38.5 36.5 ...
```

```
head(df)
```

```
##       Type Diam  Time Price
## 1 CassDish 10.7 47.65   144
## 2 CassDish 14.0 63.13   215
## 3 CassDish  9.0 58.76   105
## 4     Bowl  8.0 34.88    69
## 5     Dish 10.0 55.53   134
## 6 CassDish 10.5 43.14   129
```

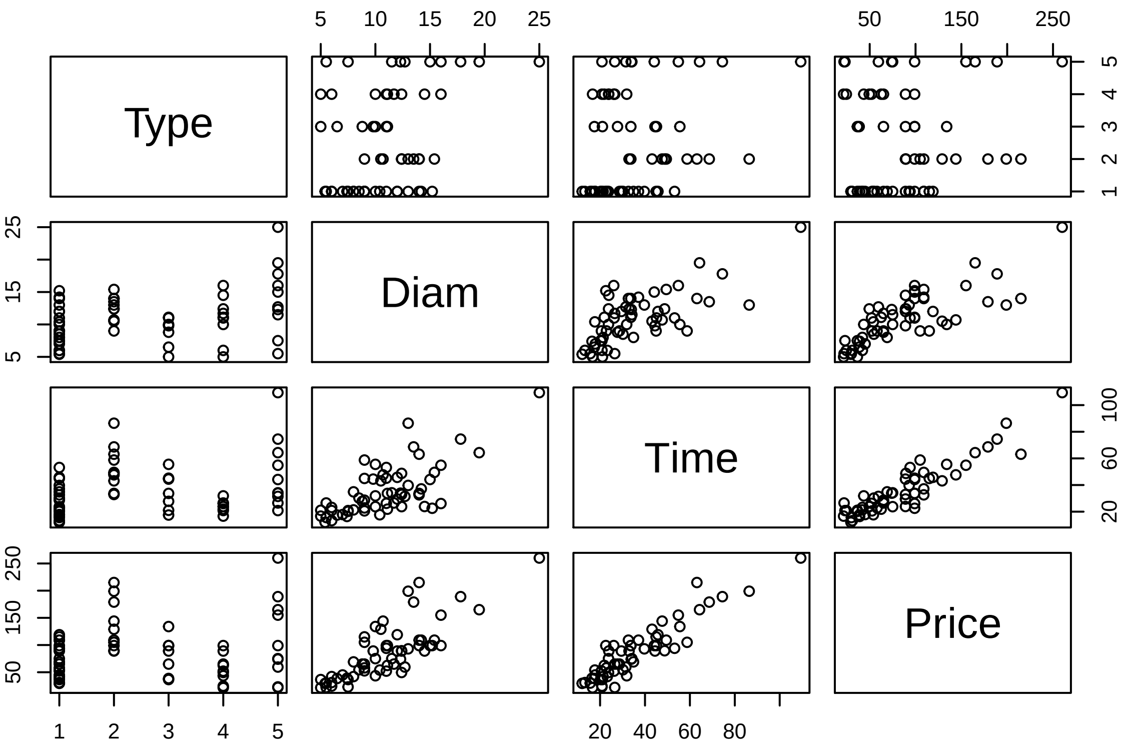
```
tail(df)
```

```
##     Type Diam  Time Price
## 54  Bowl  8.5 30.20  54.5
## 55 Plate  6.0 20.85  24.5
## 56 Plate 11.0 26.25  52.0
## 57 Plate 11.1 21.87  62.5
## 58 Plate 14.5 23.88  89.0
## 59 Plate  5.0 16.66  21.5
```

```
par(mfrow=c(1,1))
plot(df)

library(purrr)
```

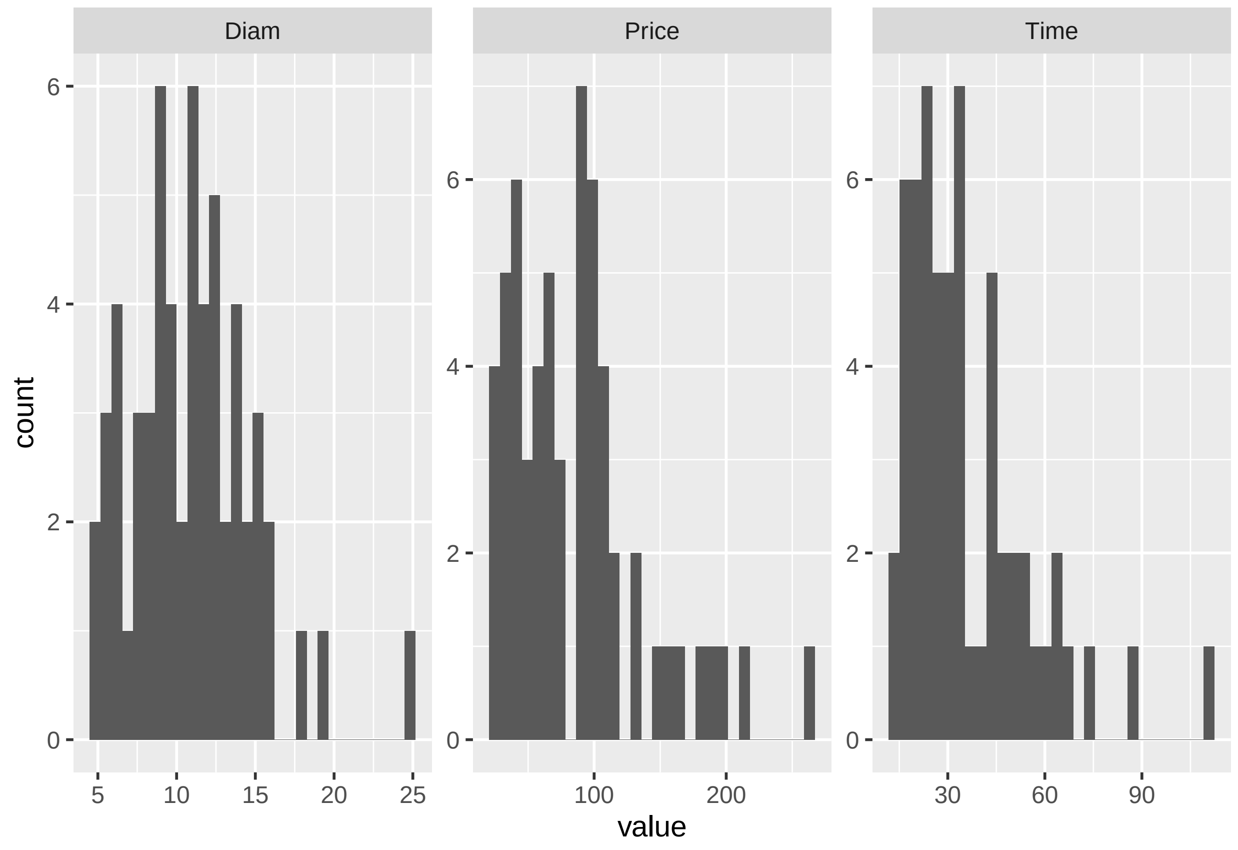
```
##
## Attaching package: 'purrr'

## The following objects are masked from 'package:foreach':
##
##     accumulate, when
```

```
library(tidyr)
library(ggplot2)
```



```
df %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Type is a factor variable, so lets transform it to that:

```r
df$Type <- as.factor(df$Type)
unique(df$Type)
```

```
## [1] CassDish Bowl     Dish     Tray     Plate
## Levels: Bowl CassDish Dish Plate Tray
```

Exercise 5.a)

```r
glm5.1 <- glm(Time ~ Diam + Type, family = Gamma(link=log), data = df)
summary(glm5.1)
```

```
##
## Call:
## glm(formula = Time ~ Diam + Type, family = Gamma(link = log),
##     data = df)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.54489  -0.20244  -0.06442   0.13852   0.64306
##
## Coefficients:
```

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.54897    0.12721  20.038  < 2e-16 ***
## Diam          0.07671    0.01176   6.525 2.62e-08 ***
## TypeCassDish  0.47516    0.11855   4.008 0.000193 ***
## TypeDish      0.28940    0.12894   2.244 0.029000 *
## TypePlate    -0.18791    0.11847  -1.586 0.118639
## TypeTray      0.14472    0.12652   1.144 0.257816
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.08899162)
##
##     Null deviance: 14.0053  on 58  degrees of freedom
## Residual deviance:  4.5039  on 53  degrees of freedom
## AIC: 438.65
##
## Number of Fisher Scoring iterations: 4
```

```r
coef(glm5.1)
```

```
##  (Intercept)         Diam TypeCassDish     TypeDish    TypePlate     TypeTray
##   2.54897318   0.07670807   0.47516081   0.28939601  -0.18791439   0.14472101
```

Exercise 5.b)

The expected response is
Time = exp(2.548 + 0.076 * Diam + β_2) which is
Time = exp(2.548) * exp(0.076 * Diam) * exp(β_2)
where β_2 depends wether Type is CassDish, Bowl, Dish, Tray, Plate

Interpreting a gamma regression model with linear predictor:

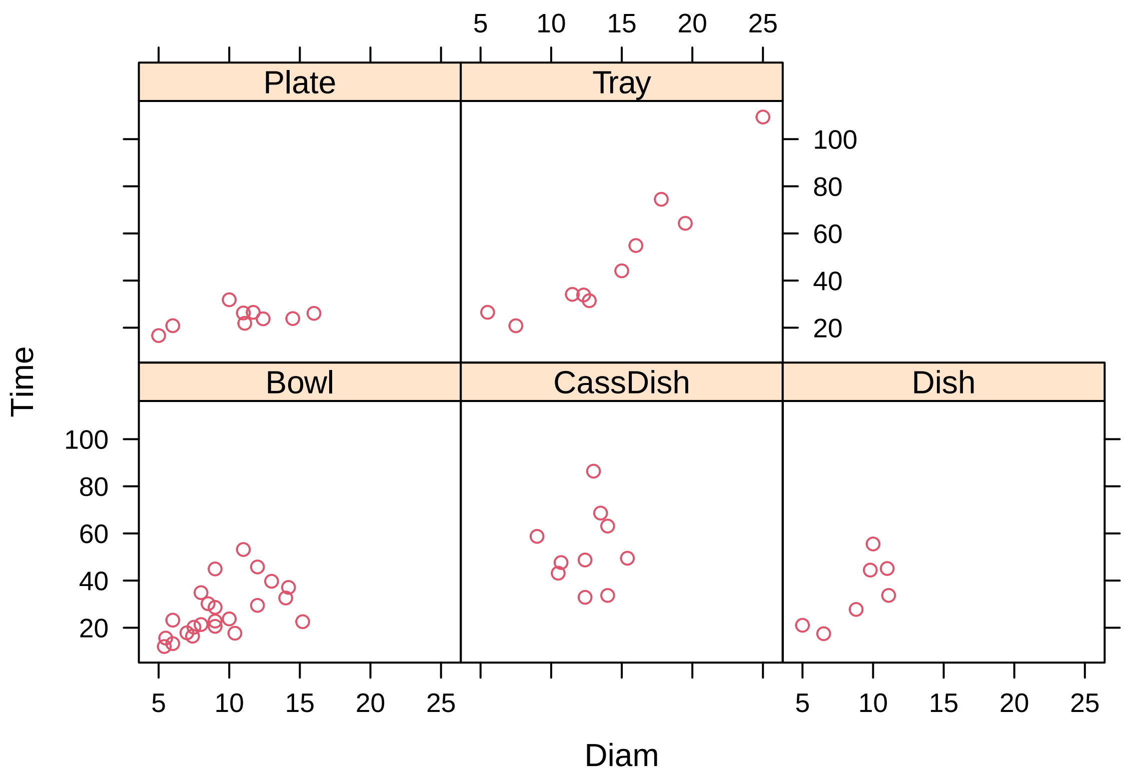
```r
glm5.2 <- glm(Time ~ Diam * Type, family = Gamma(link=log), data = df)
summary(glm5.2)
```

```
##
## Call:
## glm(formula = Time ~ Diam * Type, family = Gamma(link = log),
##     data = df)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.60856  -0.16983  -0.08072   0.13720   0.63314
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         2.419740   0.213420  11.338 2.65e-15 ***
## Diam                0.090333   0.021672   4.168 0.000124 ***
## TypeCassDish        1.414662   0.669735   2.112 0.039784 *
## TypeDish           -0.264869   0.515910  -0.513 0.609978
## TypePlate           0.448015   0.392218   1.142 0.258897
## TypeTray            0.161472   0.335958   0.481 0.632918
## Diam:TypeCassDish -0.079086   0.054758  -1.444 0.155021
## Diam:TypeDish      0.061891   0.055765   1.110 0.272483
## Diam:TypePlate    -0.061396   0.036170  -1.697 0.095962 .
## Diam:TypeTray     -0.005815   0.027533  -0.211 0.833616
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.0851398)
##
##     Null deviance: 14.005  on 58  degrees of freedom
## Residual deviance:  3.921  on 49  degrees of freedom
## AIC: 438.38
##
## Number of Fisher Scoring iterations: 5
```

This model is identical to
Time = 1 + Diam + Type + Diam:Type

Like that the estimated expected response is not just affected by a factor expt(β_2) but also the factor exp(β_1*Diam) depends on the type of product because the slope β_1 depends on the type of the product. So the coefficients 'Type…' get added to the intercept and the coefficients 'Diam:…' get added to the slope of Diam (β_1) depending on the corresponding Type.

```r
library(lattice)
xyplot(Time ~ Diam | Type, data=df, col=2)
```



Works?