Practice

gherardo varando

Load the data of the practice exam:

```
load("exam.RData")
```

Problem 1

```
head(ToothGrowth)
```

```
##    len supp dose
## 1  4.2   VC  0.5
## 2 11.5   VC  0.5
## 3  7.3   VC  0.5
## 4  5.8   VC  0.5
## 5  6.4   VC  0.5
## 6 10.0   VC  0.5
```

We transform the dose variable to a factor,

```
ToothGrowth$dose <- as.factor(ToothGrowth$dose)
```

1.1

We compute the mean tooth length for all the six combinations of supplement types and levels.

```
combinations <- expand.grid(supp = levels(ToothGrowth$supp), dose = levels(ToothGrowth$dose))
temp <- apply(combinations, MARGIN = 1, function(x){
  ix <- ToothGrowth$supp == x[1] &
    ToothGrowth$dose == x[2]
  return( c(mean = mean(ToothGrowth[ix, 1]),
  se = sd(ToothGrowth[ix, 1]) / sqrt(sum(ix) )))
} )

means <- cbind(combinations, t(temp) )
means
```

```
##   supp dose  mean        se
## 1   OJ  0.5 13.23 1.4102837
## 2   VC  0.5  7.98 0.8685620
## 3   OJ    1 22.70 1.2367520
## 4   VC    1 16.77 0.7954104
## 5   OJ    2 26.06 0.8396031
## 6   VC    2 26.14 1.5171757
```

1.2

We will investigate whether different dose levels have the same effect. Perform 0.05-level two sample t-tests with unequal variances to check whether to reject the following null hypotheses, and explain the result for each hypothesis

With the OJ method, the dose levels 0.5 and 1.0 mg/day have the same effect in tooth length:

```
t.test(x = ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "0.5", 1],
      y = ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "1", 1], var.equal = FALSE )
```

```
##
##  Welch Two Sample t-test
##
## data:  ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "0.5",  and ToothGrowth[ToothGrowt
## t = -5.0486, df = 17.698, p-value = 8.785e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -13.415634  -5.524366
## sample estimates:
## mean of x mean of y
##     13.23     22.70
```

We reject at $\alpha = 0.05$ (p-value $= 8.785 \times 10^{-05}$) the null hypothesis that the mean value of the tooth length are the same for subject treated with OJ and dose levels 0.5 and 1.

With the OJ method, the dose levels 1.0 and 2.0 mg/day have the same effect in tooth length.

```
t.test(x = ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "1", 1],
      y = ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "2", 1], var.equal = FALSE )
```

```
##
##  Welch Two Sample t-test
##
## data:  ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "1",  and ToothGrowth[ToothGrowth$
## t = -2.2478, df = 15.842, p-value = 0.0392
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -6.5314425 -0.1885575
## sample estimates:
## mean of x mean of y
##     22.70     26.06
```

We reject at $\alpha = 0.05$ (p-value $= 0.0392$) the null hypothesis that the mean value of the tooth length are the same for subject treated with OJ and dose levels 1 and 2.

With the VC method, the dose levels 0.5 and 1.0 mg/day have the same effect in tooth length:

```
t.test(x = ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "0.5", 1],
      y = ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "1", 1], var.equal = FALSE )
```

```
##
##  Welch Two Sample t-test
##
## data:  ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "0.5",  and ToothGrowth[ToothGrowt
## t = -7.4634, df = 17.862, p-value = 6.811e-07
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -11.265712  -6.314288
## sample estimates:
## mean of x mean of y
##      7.98     16.77
```

We reject at $\alpha = 0.05$ (p-value $= 6.811 \times 10^{-07}$) the null hypothesis that the mean value of the tooth length are the same for subject treated with VC and dose levels 0.5 and 1.

With the VC method, the dose levels 1.0 and 2.0 mg/day have the same effect in tooth length.

```
t.test(x = ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "1", 1],
      y = ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "2", 1], var.equal = FALSE )
```

```
##
##  Welch Two Sample t-test
##
## data:  ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "1",  and ToothGrowth[ToothGrowth$
## t = -5.4698, df = 13.6, p-value = 9.156e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -13.054267  -5.685733
## sample estimates:
## mean of x mean of y
##     16.77     26.14
```

We reject at $\alpha = 0.05$ (p-value $= 9.156 \times 10^{-5}$) the null hypothesis that the mean value of the tooth length are the same for subject treated with VC and dose levels 1 and 2.

1.3

We are interested in whether OJ is more effective than VC. Perform 0.05-level two sample t-tests with unequal variances to check whether to reject the following null hypotheses:

With 0.5 mg/day dose level, OJ is less effective than or as effective as VC in tooth growth.

```
t.test(x = ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "0.5", 1],
      y = ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "0.5", 1], var.equal = FALSE, alte
```

```
##
##  Welch Two Sample t-test
##
## data:  ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "0.5",  and ToothGrowth[ToothGrowt
## t = 3.1697, df = 14.969, p-value = 0.003179
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  2.34604      Inf
## sample estimates:
## mean of x mean of y
##     13.23      7.98
```

We reject the null hypothesis at $\alpha = 0.05$ (p-value $= 0.003179$).

With 1.0 mg/day dose level, OJ is less effective than or as effective as VC in tooth growth.

```
t.test(x = ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "1", 1],
      y = ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "1", 1], var.equal = FALSE, altern
```

```
##
##  Welch Two Sample t-test
```

```
##
## data:  ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "1",  and ToothGrowth[ToothGrowth$
## t = 4.0328, df = 15.358, p-value = 0.0005192
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  3.356158      Inf
## sample estimates:
## mean of x mean of y
##     22.70     16.77
```

We reject the null hypothesis at $\alpha = 0.05$ (p-value = 0.0005192).

With 2.0 mg/day dose level, OJ is less effective than or as effective as VC in tooth growth.

```r
t.test(x = ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "2", 1],
      y = ToothGrowth[ToothGrowth$supp == "VC" & ToothGrowth$dose == "2", 1], var.equal = FALSE, altern
```

```
##
##  Welch Two Sample t-test
##
## data:  ToothGrowth[ToothGrowth$supp == "OJ" & ToothGrowth$dose == "2",  and ToothGrowth[ToothGrowth$
## t = -0.046136, df = 14.04, p-value = 0.5181
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -3.1335     Inf
## sample estimates:
## mean of x mean of y
##     26.06     26.14
```

We can not reject the null hypothesis at $\alpha = 0.05$ (p-value = 0.5181).

Under which dose level(s) can we say OJ is more effective than VC?

We can say that OJ is more effective than VC under dose levels 0.5 and 1.0.

Problem 2

2.1

Show that when $k = 1$, the Weibull distribution with parameters $k = 1, \lambda$, reduces to the exponential distribution.

The Weibull density is

$$f_{WB}(x|k,\lambda) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k}, \quad x \geq 0$$

Thus is $k = 1$ it reduces to

$$f_{WB}(x|k=1,\lambda) = \frac{1}{\lambda}\left(\frac{x}{\lambda}\right)^{0} e^{-(x/\lambda)} = \frac{1}{\lambda}e^{-(x/\lambda)} = f_{Exp}(x|r = \frac{1}{\lambda})$$

Where f_{Exp} is the density function of an exponential random variable.

What is the rate parameter of the obtained exponential distribution?

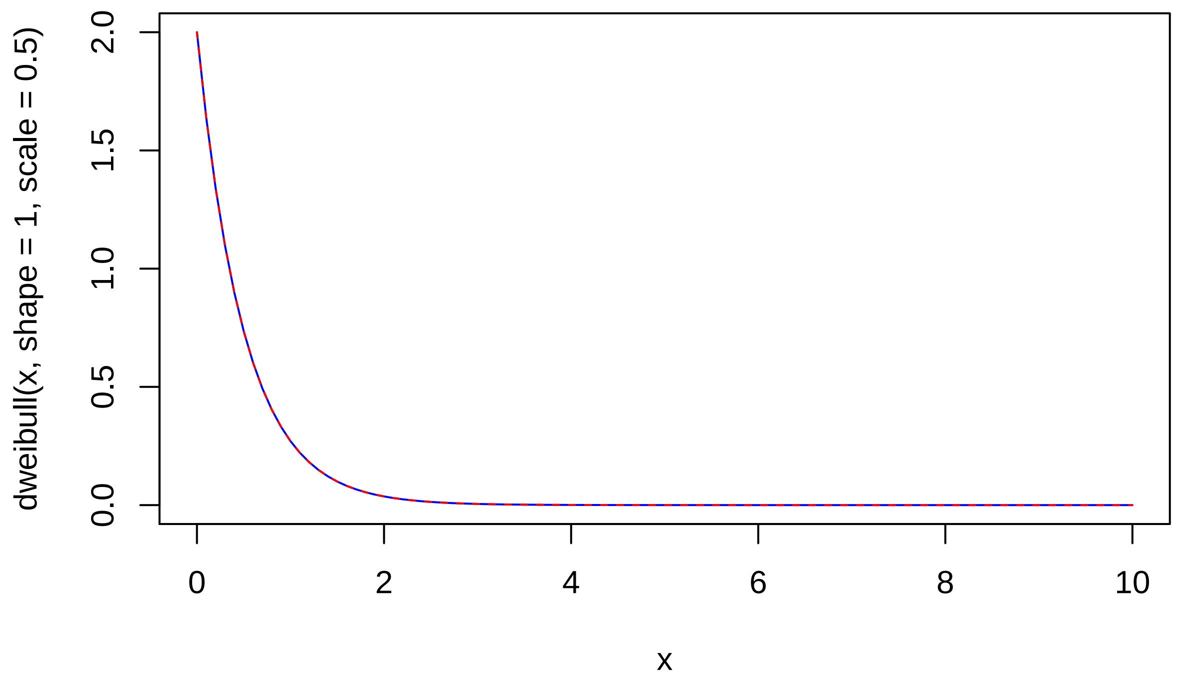
As we can see from the equation above

$$f_{WB}(x|k=1,\lambda) = f_{Exp}(x|r = \frac{1}{\lambda})$$

Thus the rate of the obtained exponential distribution is $r = \frac{1}{\lambda}$ where λ is the parameter of the Weibull distribution.

We check it graphically

```
curve(dweibull(x, shape = 1, scale = 0.5), col = "blue",
      from= 0 , to = 10)
curve(dexp(x, rate = 1/(0.5)), col = "red", add = TRUE, lty = 2)
```



They coincide for $\lambda = 0.5 = \frac{1}{r}$.

2.2

The implementation of the minus log-likelihood is:

```
mll_wb <- function(par, data){
  -sum(dweibull(data, shape = par[1], scale = par[2], log = TRUE))
}
```

Now we can minimize the minus log-likelihood for the ISI data,

```
res <- optim(par = c(1,1), fn = mll_wb, data = neuron$isi)
res$par
```

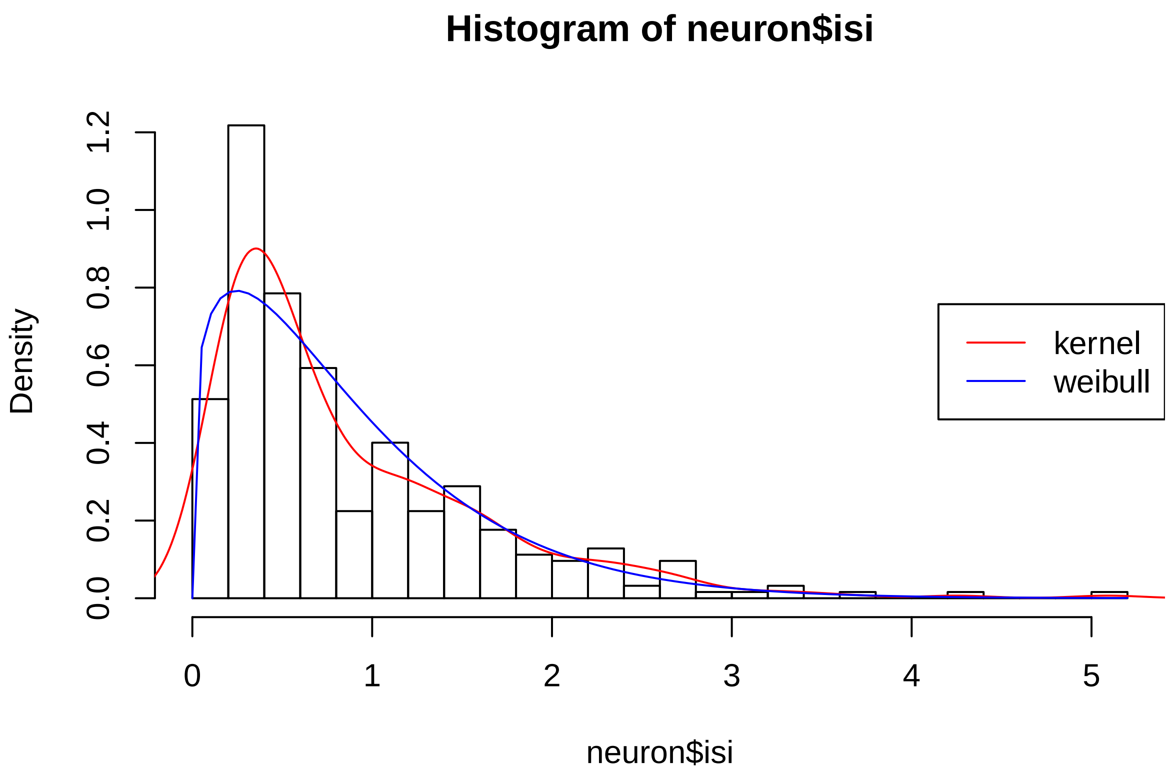
```
## [1] 1.2358668 0.9398702
```

2.3

Investigate how the Weibull model fits the neuron data by a Q-Q plot and comparing with the kernel density estimation.

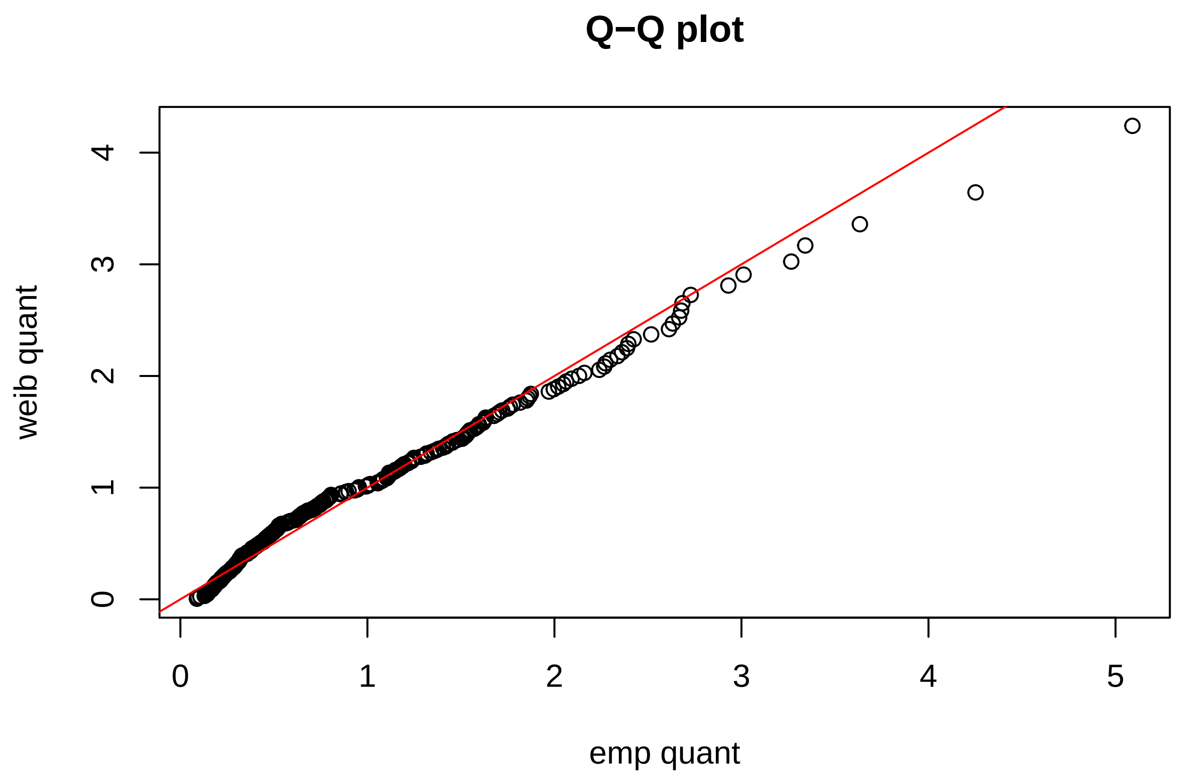
We plot histogram, kernel density estimation and fitted Weibull density.

```r
hist(neuron$isi, probability = TRUE, breaks = "FD")
lines(density(neuron$isi), col = "red")
curve(dweibull(x, shape = res$par[1], scale = res$par[2]),
      col = "blue", add = TRUE)
legend("right", legend = c("kernel", "weibull"), col = c("red", "blue"),
       lty = 1)
```



Q-Q plot

```r
p <- ppoints(neuron$isi)
q_wb <- qweibull(p, shape = res$par[1], scale = res$par[2])
plot(sort(neuron$isi), q_wb, xlab = "emp quant", ylab = "weib quant",
     main = "Q-Q plot")
abline(0, 1, col = "red")
```



From the two plots we can see that the Weibull distribution fits quite well the data.

2.4

Compute confidence intervals for k and λ using parametric and non-parametric bootstrap, use both normal confidence interval and percentile confidence intervals.

Using non-parametric bootstrap,

```r
M <- 1000
par_bt <- replicate(M, {
  temp <- sample(neuron$isi, replace = TRUE)
  pr <- optim(par = c(1,1), fn = mll_wb, data = temp)$par
  return(c(k = pr[1], lambda = pr[2]))
})
se <- apply(par_bt, MARGIN = 1, function(x) sd(x))
```

We show 95% confidence intervals for k and λ, using asymptotic normality,

```r
a <- 0.05
z <- qnorm(1 - a / 2)

matrix(res$par + z * se %*% t(c(-1, +1)), dimnames = list(c("k", "lambda"), c("a", "b")), ncol = 2  )
```

```
##                 a        b
## k      1.1497907 1.321943
## lambda 0.8523159 1.027425
```

The percentile confidence intervals can be obtained directly from the sample of the bootstrap,

```
t(apply(par_bt, MARGIN = 1, function(x) quantile(x, probs =
                                                  c(a/2, 1- a/2))))
```

```
##             2.5%    97.5%
## k      1.1601390 1.336471
## lambda 0.8551832 1.029755
```

Parametric bootstrap is similar but the generation of the sample is done using the Weibull distribution (in this case),

```
par_bt <- replicate(M, {
  temp <- rweibull(length(neuron$isi), shape = res$par[1],
                   scale = res$par[2] )
  pr <- optim(par = c(1,1), fn = mll_wb, data = temp)$par
  return(c(k = pr[1], lambda = pr[2]))
})

## normal CI
matrix(res$par + z * se %*% t(c(-1, +1)), dimnames = list(c("k", "lambda"), c("a", "b")), ncol = 2  )
```

```
##                a        b
## k      1.1497907 1.321943
## lambda 0.8523159 1.027425
```

```
##percentile CI
t(apply(par_bt, MARGIN = 1, function(x) quantile(x, probs =
                                                  c(a/2, 1- a/2))))
```

```
##             2.5%    97.5%
## k      1.1442605 1.356640
## lambda 0.8551029 1.030719
```

2.5

We fit the exponential distribution to the data,

```
r_est <- 1 / mean(neuron$isi)
```

We compute AIC, BIC for both model

```
aic_exp <- -2*sum(dexp(neuron$isi, rate = r_est, log = TRUE)) + 2
bic_exp <- -2*sum(dexp(neuron$isi, rate = r_est, log = TRUE)) + 2 * log(length(neuron$isi))

aic_wb <- 2 * res$value + 2 * 2
bic_wb <- 2 * res$value + 2 * log(length(neuron$isi))

matrix(c(aic_exp, bic_exp, aic_wb, bic_wb), ncol = 2,
       dimnames = list(c("aic", "bic"), c("exp", "weibull")))
```

```
##          exp weibull
## aic 540.4776 520.074
## bic 549.9636 527.560
```

The Weibull model is selected by both AIC and BIC.

We can also perform likelihood-ratio test since, as we observe at the beginning, the exponential model is nested in the Weibull model.

```r
ll_exp <- sum(dexp(neuron$isi, rate = r_est, log = TRUE))
ll_weib <- -res$value
delta <- -2 * ( ll_exp - ll_weib   )
## p-value
pchisq(delta, lower.tail = FALSE, df = 1)
```

```
## [1] 2.209674e-06
```

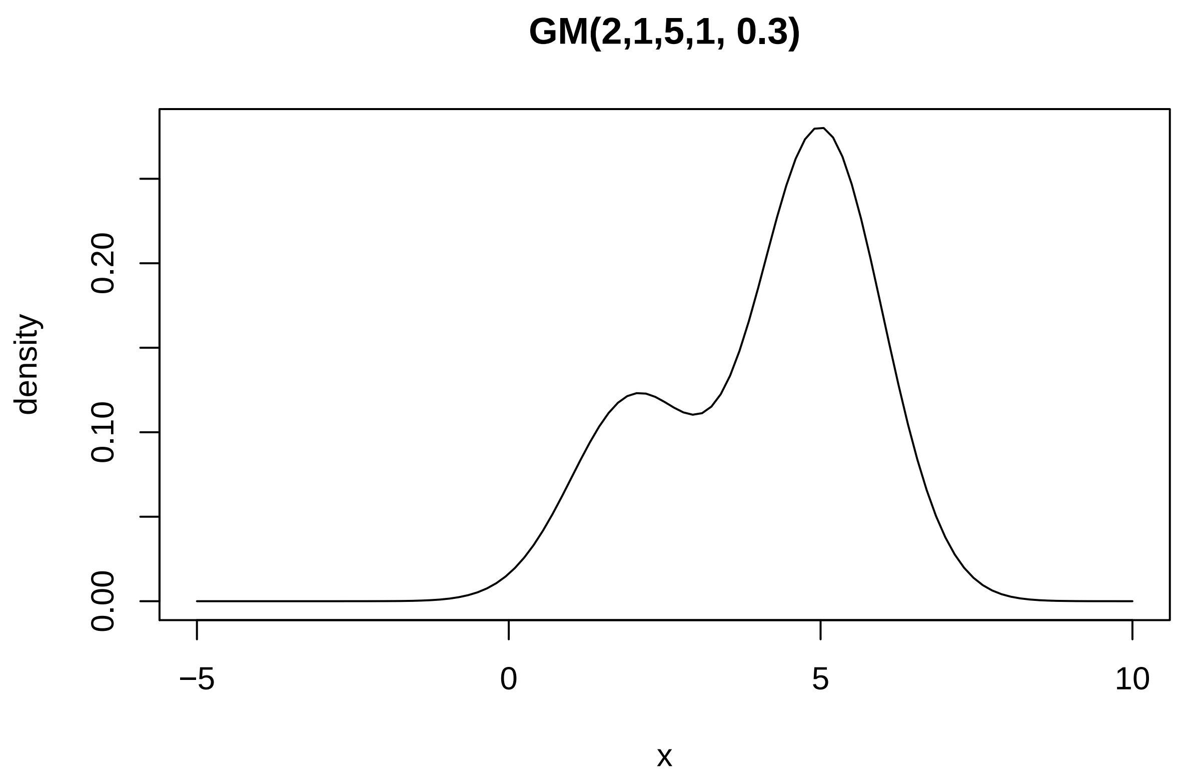
The p-value is equal to 2.21×10^{-6} thus we reject the null hypothesis that $k = 1$ (Exp model) at $\alpha = 0.001$ (for example). Also the likelihood-ratio test indicates that the Weibull model it to prefer.

Problem 3

3.1

We implement the density of the Gaussian mixture, we parametrize it with standard deviations σ_1, σ_2.

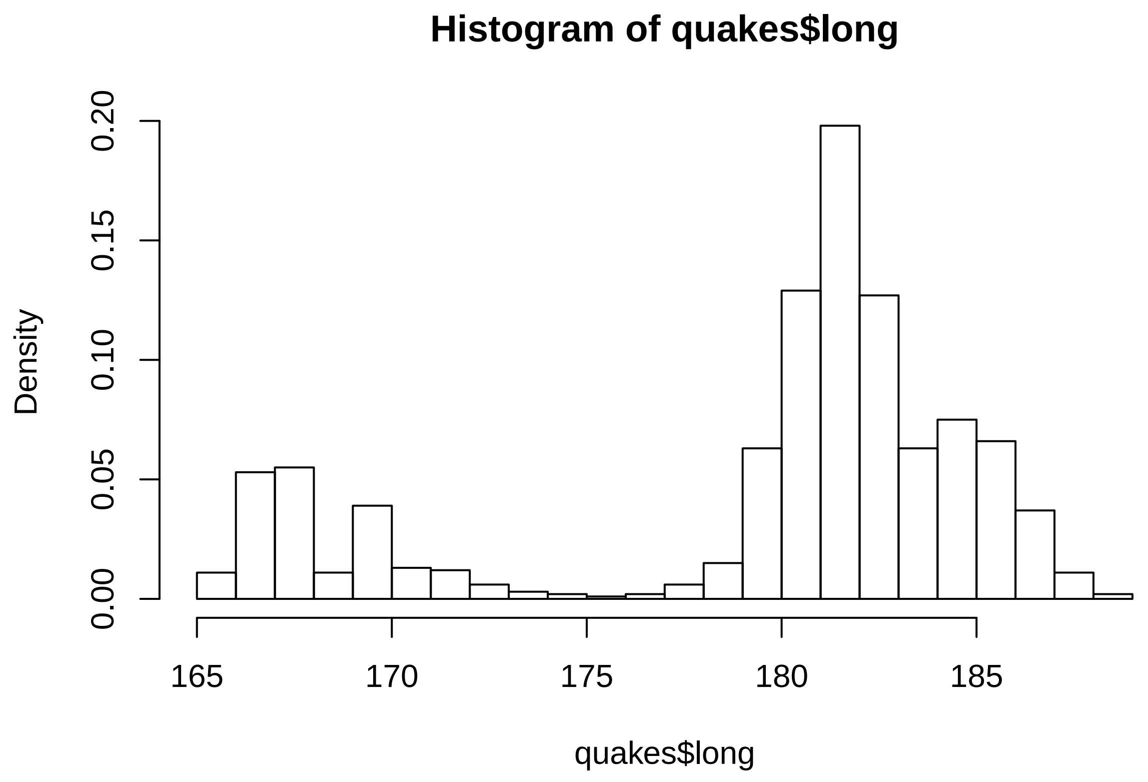
```r
dgaussmix <- function(x, mean1, sd1, mean2, sd2, w){
  stopifnot(w <= 1 && w >= 0)
  stopifnot(sd1 > 0 && sd2 > 0)
  w * dnorm(x, mean1, sd1) + (1 - w) * dnorm(x, mean2, sd2)
}

curve(dgaussmix(x, 2, 1, 5, 1, 0.3), from = -5, to = 10,
      main = "GM(2,1,5,1, 0.3)", ylab = "density")
```



3.2

To obtain initial guess for the parameter of the Gaussian mixture for the longitude locations we plot the histogram

```
hist(quakes$long, probability = TRUE, breaks = "FD")
```



We can divide the longitude location observations in two groups, before and after 175.

```
bef <- quakes$long[quakes$long < 175]
aft <- quakes$long[quakes$long > 175]
```

We can now consider the following initial estimate for the Gaussian mixture:

```
m1.init <- mean(bef)
sd1.init <- sd(bef)
m2.init <- mean(aft)
sd2.init <- sd(aft)
w.init <- length(bef) / length(aft)
par.init <- c(m1.init, sd1.init, m2.init, sd2.init, w.init)
par.init
```

```
## [1] 168.2598049   1.9746871 182.3506415   2.1433270   0.2578616
```

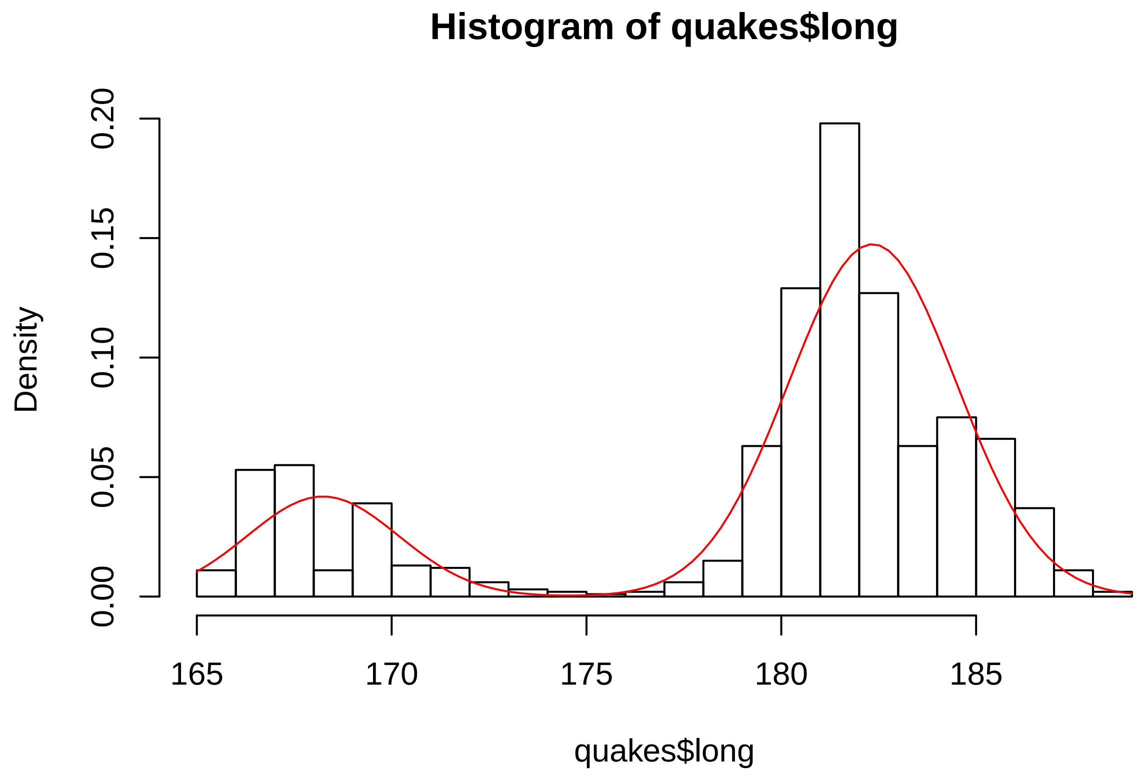
We define now the minus log-likelihood and then start the optimization,

```
mll <- function(par, data){
  if (par[5] > 1 || par[5] < 0 ){
    return(Inf)
  }
  if (par[2] < 0 || par[4] < 0){
    return(Inf)
  }
  -sum(log(dgaussmix(data, par[1], par[2], par[3], par[4], par[5])))
}
par.est <- optim(par = par.init, fn = mll, data = quakes$long)$par
par.est
```

```
## [1] 168.236709   1.944461 182.340729   2.153245   0.204089
```

We now plot the fitted mixture on top of the histogram.

```
hist(quakes$long, probability = TRUE, breaks = "FD")
curve(dgaussmix(x, par.est[1], par.est[2], par.est[3], par.est[4], par.est[5]), add = TRUE, col = "red"
```



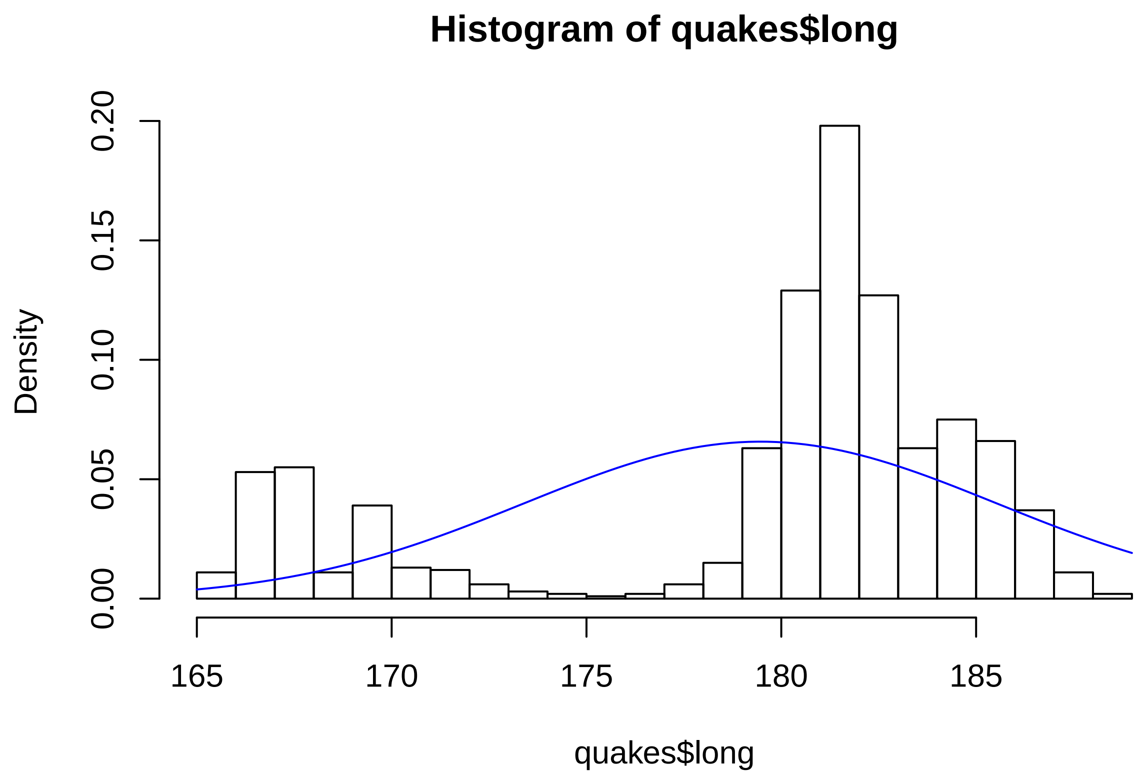
3.3

We here fit the longitudinal data to a simple Gaussian model, we use the known formula for the MLE of a Gaussian model,

$$\hat{\mu} = \overline{X} \quad \hat{\sigma} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \hat{\mu})^2}$$

```
mu_est <- mean(quakes$long)
sigma_est <- sd(quakes$long)
```

```
hist(quakes$long, probability = TRUE, breaks = "FD")
curve(dnorm(x, mu_est, sigma_est), add = TRUE, col = "blue")
```



It seems that the mixture model fit the data much better.

3.4

We compare now the mixture and the simple Gaussian model using AIC

```
aic.mixture <- 2*mll(par.est, data = quakes$long) +  2 * length(par.est)
aic.gauss <- -2*sum(dnorm(quakes$long, mu_est, sigma_est, log = TRUE)) + 2 * 2
c(mixture = aic.mixture, gauss = aic.gauss)
```

```
##  mixture    gauss
## 5349.808 6447.428
```

Thus by the AIC score the mixture model should be preferred.

The BIC:

```
n <- nrow(quakes)
bic.mixture <- 2*mll(par.est, data = quakes$long) + log(n) * length(par.est)
bic.gauss <- -2*sum(dnorm(quakes$long, mu_est, sigma_est, log = TRUE)) + 2 * log(n)
c(mixture = bic.mixture, gauss = bic.gauss)
```

```
##  mixture    gauss
## 5374.347 6457.244
```

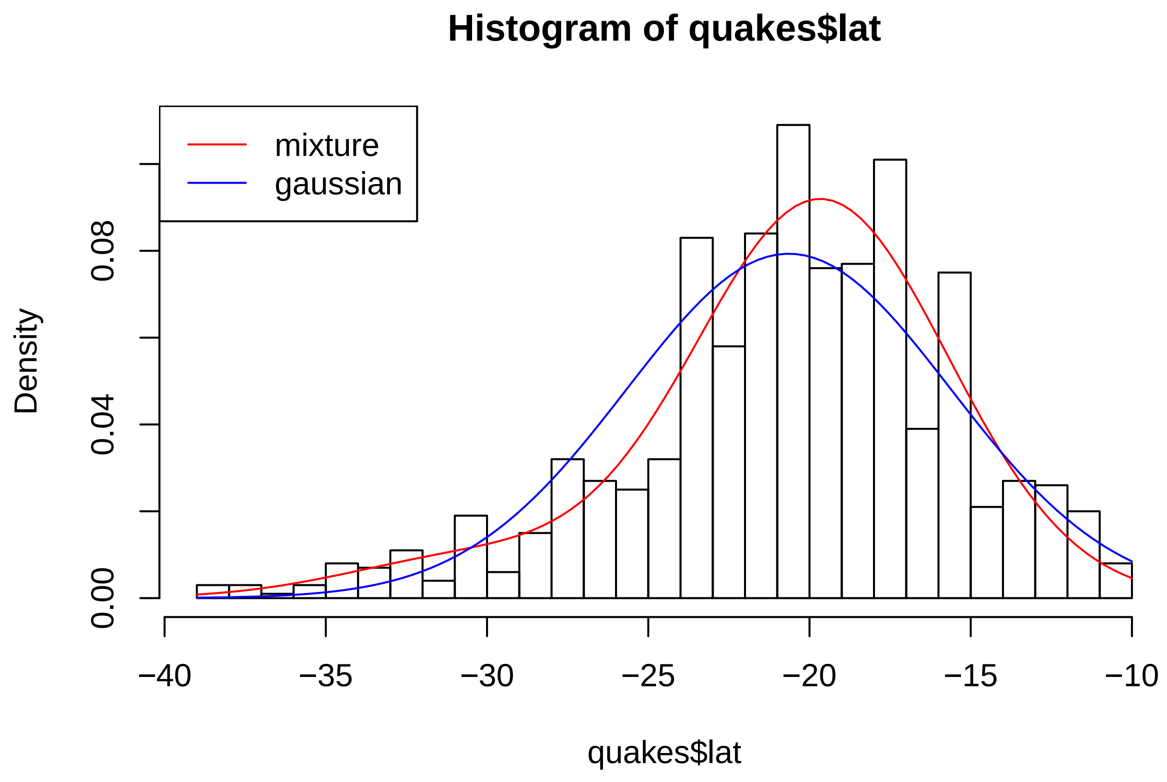
Similarly also the BIC score indicates that the mixture model should be selected

3.5

We repeat now the model selection between mixture and Gaussian model for other variables `lat`, `depth`.

For `lat`:

```
par.lat <- optim(par = c(-25, 4, -15, 4, 0.5), fn = mll,
                 data = quakes$lat)$par
mu.lat <- mean(quakes$lat)
sigma.lat <- sd(quakes$lat)
hist(quakes$lat, probability = TRUE, breaks = "FD")
curve(dgaussmix(x, par.lat[1], par.lat[2],
                par.lat[3], par.lat[4], par.lat[5]), add = TRUE,
      col = "red")
curve(dnorm(x, mean = mu.lat, sd = sigma.lat), col = "blue", add = TRUE)
legend("topleft", legend = c("mixture", "gaussian"),
       col = c("red", "blue"), lty = 1)
```



```
aic.mixture <- 2*mll(par.lat, data = quakes$lat) +  2 * length(par.lat)
aic.gauss <- -2*sum(dnorm(quakes$lat, mu.lat, sigma.lat, log = TRUE)) + 2 * 2
c(mixture = aic.mixture, gauss = aic.gauss)
```

```
##  mixture    gauss
## 5995.203 6071.236
```

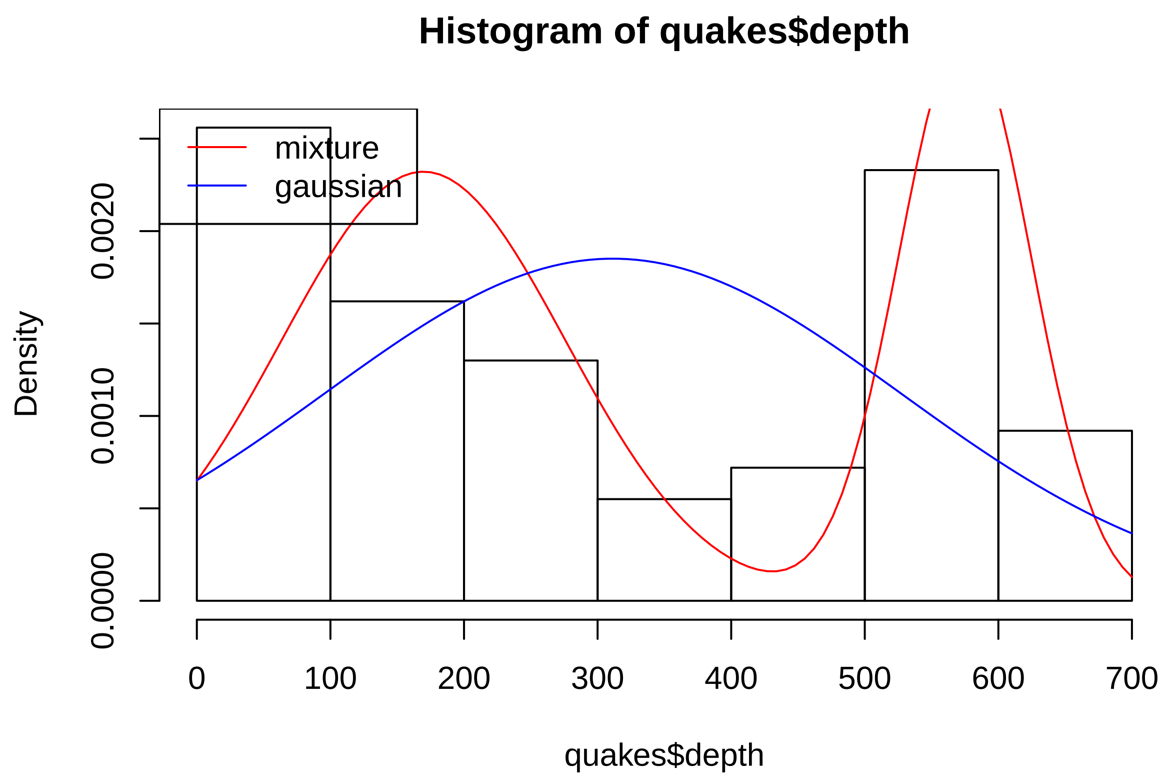
```r
bic.mixture <- 2*mll(par.lat, data = quakes$lat) + log(n) * length(par.lat)
bic.gauss <- -2*sum(dnorm(quakes$lat, mu.lat, sigma.lat, log = TRUE)) + 2 * log(n)
c(mixture = bic.mixture, gauss = bic.gauss)
```

```
##  mixture    gauss
## 6019.741 6081.052
```

The mixture model is to prefer (AIC and BIC).

For the `depth` observations:

```r
par.depth <- optim(par = c(100, 100, 600, 100, 0.5), fn = mll,
                  data = quakes$depth)$par
mu.depth <- mean(quakes$depth)
sigma.depth <- sd(quakes$depth)
hist(quakes$depth, probability = TRUE, breaks = "FD")
curve(dgaussmix(x, par.depth[1], par.depth[2],
                par.depth[3], par.depth[4], par.depth[5]), add = TRUE,
      col = "red")
curve(dnorm(x, mean = mu.depth, sd = sigma.depth), col = "blue", add = TRUE)
legend("topleft", legend = c("mixture", "gaussian"),
       col = c("red", "blue"), lty = 1)
```



```r
aic.mixture <- 2*mll(par.depth, data = quakes$depth) +  2 * length(par.depth)
aic.gauss <- -2*sum(dnorm(quakes$depth, mu.depth, sigma.depth, log = TRUE)) + 2 * 2
c(mixture = aic.mixture, gauss = aic.gauss)
```

```
##  mixture    gauss
## 12942.26 13587.13
```

```
bic.mixture <- 2*mll(par.depth, data = quakes$depth) + log(n) * length(par.depth)
bic.gauss <- -2*sum(dnorm(quakes$depth, mu.depth, sigma.depth, log = TRUE)) + 2 * log(n)
c(mixture = bic.mixture, gauss = bic.gauss)
```

```
##  mixture    gauss
## 12966.80 13596.94
```

The mixture model is again selected by both AIC and BIC

3.6

In this question we consider a generalized linear model with the log link and stations follows a Gaussian distribution.

$$\log(\mathbb{E}(\texttt{stations}|...)) = \beta_0 + \beta_1\texttt{lat} + \beta_2\texttt{long} + \beta_3\texttt{depth} + \beta_4\texttt{mag} \tag{1}$$

```
fit1 <- glm(stations ~ ., data = quakes,
            family = gaussian(link = "log"))
summary(fit1)
```

```
##
## Call:
## glm(formula = stations ~ ., family = gaussian(link = "log"),
##     data = quakes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -67.512   -6.380   -1.474    4.391   46.943
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.2425771  0.2908656 -14.586  < 2e-16 ***
## lat          0.0079305  0.0018178   4.363 1.42e-05 ***
## long         0.0140097  0.0014976   9.355  < 2e-16 ***
## depth        0.0002845  0.0000392   7.257 7.94e-13 ***
## mag          1.1290611  0.0170562  66.197  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 107.1978)
##
##     Null deviance: 479147  on 999  degrees of freedom
## Residual deviance: 106661  on 995  degrees of freedom
## AIC: 7519.5
##
## Number of Fisher Scoring iterations: 5
```

Since it is intuitive that stronger earthquakes are more likely to be detected, we assume that stations is more related to mag. Fit the following model:

$$\log(\mathbb{E}(\texttt{stations}|...)) = \beta_0 + \beta_1\texttt{lat} + \beta_2\texttt{long} + \beta_3\texttt{depth} + \beta_4\texttt{mag} + \beta_5\texttt{mag}^2 \tag{2}$$

```r
fit2 <- glm(stations ~ . + I(mag^2), data = quakes,
            family = gaussian(link = "log"))
summary(fit2)
```

```
##
## Call:
## glm(formula = stations ~ . + I(mag^2), family = gaussian(link = "log"),
##     data = quakes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -43.310   -5.327   -0.270    5.469   42.920
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.159e+01  7.478e-01 -15.501  < 2e-16 ***
## lat          9.271e-03  1.693e-03   5.475 5.53e-08 ***
## long         1.098e-02  1.391e-03   7.893 7.79e-15 ***
## depth        2.904e-04  3.628e-05   8.005 3.33e-15 ***
## mag          4.233e+00  2.891e-01  14.642  < 2e-16 ***
## I(mag^2)    -3.013e-01  2.811e-02 -10.716  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 94.54226)
##
##     Null deviance: 479147  on 999  degrees of freedom
## Residual deviance:  93974  on 994  degrees of freedom
## AIC: 7394.9
##
## Number of Fisher Scoring iterations: 5
```

The recorded magnitude is actually in the Richter scale which is a log scale of the earthquake wave amplitude. We thus transform now the Richter scale back to the original scale. Fit the model:

$$\log(\mathbb{E}(\texttt{stations}|...)) = \beta_0 + \beta_1\texttt{lat} + \beta_2\texttt{long} + \beta_3\texttt{depth} + \beta_4\exp(\texttt{mag}) + \beta_5\exp(\texttt{mag})^2 \tag{3}$$

```r
fit3 <- glm(stations ~ lat + long + depth + exp(mag) +
              I(exp(mag)^2),
            data = quakes, family = gaussian(link = "log"))
summary(fit3)
```

```
##
## Call:
## glm(formula = stations ~ lat + long + depth + exp(mag) + I(exp(mag)^2),
##     family = gaussian(link = "log"), data = quakes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -45.060   -6.508   -1.353    4.451   77.861
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    6.818e-01  2.529e-01   2.696  0.00713 **
## lat            9.407e-03  1.768e-03   5.322 1.27e-07 ***
## long           8.423e-03  1.470e-03   5.731 1.32e-08 ***
## depth          2.534e-04  3.837e-05   6.605 6.48e-11 ***
## exp(mag)       1.465e-02  3.701e-04  39.577  < 2e-16 ***
## I(exp(mag)^2) -1.935e-05  8.130e-07 -23.796  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 104.8708)
##
##     Null deviance: 479147  on 999  degrees of freedom
## Residual deviance: 104240  on 994  degrees of freedom
## AIC: 7498.6
##
## Number of Fisher Scoring iterations: 11
```

3.7

Perform the log likelihood ratio test selection between model 1 and model 2.

```r
anova(fit1, fit2, test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: stations ~ lat + long + depth + mag
## Model 2: stations ~ lat + long + depth + mag + I(mag^2)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       995     106661
## 2       994      93974  1    12688 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value is very small and we thus reject the null hypothesis (e.g. at a level 0.0005) that the simpler model `fit1` is sufficient.

Use instead AIC and BIC to perform model selection between model 1, model 2 and model 3.

```r
models <- list(fit1 = fit1, fit2 = fit2, fit3 = fit3)
sapply(models, function(m){
  c(AIC = AIC(m), BIC = BIC(m))
})
```

```
##         fit1     fit2     fit3
## AIC 7519.536 7394.894 7498.570
## BIC 7548.983 7429.248 7532.924
```

Model 2 is selected by both AIC and BIC.

3.8

We observe now that `stations` *are actually positive counts. It is thus natural to use the Poisson regression model. Fit then the Poisson regression models with the log link function:*

$$\log(\mathbb{E}(\texttt{stations}|...)) = \beta_0 + \beta_1\texttt{lat} + \beta_2\texttt{long} + \beta_3\texttt{depth} + \beta_4\texttt{mag} \tag{4}$$

```r
fit4 <- glm(stations ~ ., data = quakes,
            family = poisson(link = "log"))
summary(fit4)
```

```
##
## Call:
## glm(formula = stations ~ ., family = poisson(link = "log"), data = quakes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -7.3543  -1.1201  -0.1238   0.9457   5.9071
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.9057762  0.1848110 -21.134  < 2e-16 ***
## lat          0.0068245  0.0011614   5.876  4.2e-09 ***
## long         0.0098097  0.0009717  10.095  < 2e-16 ***
## depth        0.0002722  0.0000257  10.591  < 2e-16 ***
## mag          1.2088383  0.0118700 101.840  < 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: 12198.5  on 999  degrees of freedom
## Residual deviance:  2764.3  on 995  degrees of freedom
## AIC: 7950.4
##
## Number of Fisher Scoring iterations: 4
```

$$\log(\mathbb{E}(\texttt{stations}|...)) = \beta_0 + \beta_1\texttt{lat} + \beta_2\texttt{long} + \beta_3\texttt{depth} + \beta_4\texttt{mag} + \beta_5\texttt{mag}^2 \tag{5}$$

```r
fit5 <- glm(stations ~ lat + long + depth + mag +
              I(mag^2),
            data = quakes, family = poisson(link = "log"))
summary(fit5)
```

```
##
## Call:
## glm(formula = stations ~ lat + long + depth + mag + I(mag^2),
##     family = poisson(link = "log"), data = quakes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.6110  -1.0989  -0.0992   0.9355   5.9666
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.774e+00  5.158e-01 -17.011  < 2e-16 ***
## lat          7.597e-03  1.163e-03   6.529 6.61e-11 ***
## long         9.576e-03  9.686e-04   9.887  < 2e-16 ***
## depth        2.868e-04  2.565e-05  11.180  < 2e-16 ***
## mag          3.209e+00  1.979e-01  16.216  < 2e-16 ***
## I(mag^2)    -2.014e-01  1.991e-02 -10.113  < 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: 12198.5  on 999  degrees of freedom
## Residual deviance:  2657.2  on 994  degrees of freedom
## AIC: 7845.3
##
## Number of Fisher Scoring iterations: 4
```

$$\log(\mathbb{E}(\texttt{stations}|...)) = \beta_0 + \beta_1\texttt{lat} + \beta_2\texttt{long} + \beta_3\texttt{depth} + \beta_4\exp(\texttt{mag}) + \beta_5\exp(\texttt{mag})^2 \tag{6}$$

```
fit6 <- glm(stations ~ lat + long + depth + exp(mag) +
              I(exp(mag)^2),
            data = quakes, family = poisson(link = "log"))
summary(fit6)
```

```
##
## Call:
## glm(formula = stations ~ lat + long + depth + exp(mag) + I(exp(mag)^2),
##     family = poisson(link = "log"), data = quakes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -7.0498  -1.2112  -0.1699   0.8832   8.6805
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)    8.342e-01  1.675e-01   4.980 6.36e-07 ***
## lat            6.868e-03  1.157e-03   5.938 2.88e-09 ***
## long           6.846e-03  9.692e-04   7.064 1.62e-12 ***
## depth          2.497e-04  2.567e-05   9.727  < 2e-16 ***
## exp(mag)       1.523e-02  2.403e-04  63.390  < 2e-16 ***
## I(exp(mag)^2) -1.981e-05  5.629e-07 -35.187  < 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: 12198.5  on 999  degrees of freedom
## Residual deviance:  2818.3  on 994  degrees of freedom
## AIC: 8006.4
##
## Number of Fisher Scoring iterations: 4
```

Perform model selection between model 4 and model 5 using the **anova** *function. Perform model selection between the three Poisson regression models using AIC and BIC.*

anova:

```
anova(fit4, fit5, test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: stations ~ lat + long + depth + mag
## Model 2: stations ~ lat + long + depth + mag + I(mag^2)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       995     2764.3
## 2       994     2657.2  1   107.05 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

model 5 (`fit5`) is selected, since we reject the null hypothesis that the simpler model is sufficient.

```
models <- list(fit4 = fit4, fit5 = fit5, fit6 = fit6)
sapply(models, function(m){
  c(AIC = AIC(m), BIC = BIC(m))
})
```

```
##         fit4     fit5     fit6
## AIC 7950.386 7845.340 8006.409
## BIC 7974.925 7874.787 8035.855
```

Model 5 (`fit5`) is selected by both AIC and BIC.

3.9

Consider the generalized linear regression model with the in- verse link function l(y) = 1/y:

$$\frac{1}{\mathbb{E}(\texttt{stations}|\ldots)} = \beta_0 + \beta_1\texttt{lat} + \beta_2\texttt{long} + \beta_3\texttt{depth} + \beta_4\texttt{mag} + \beta_5\texttt{mag}^2$$

Where `stations`|... *follows a gamma distribution. Fit this model to the* `quakes` *data. Take a look to the relevant information about the distribution and the link function* `?family`.

```
fit7 <- glm(stations ~ . + I(mag^2), data = quakes,
            family = Gamma(link = "inverse" ))
summary(fit7)
```

```
##
## Call:
## glm(formula = stations ~ . + I(mag^2), family = Gamma(link = "inverse"),
##     data = quakes)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.09679  -0.22536  -0.02369   0.16825   1.03649
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.310e-01  2.573e-02  24.519  < 2e-16 ***
## lat         -1.483e-04  5.194e-05  -2.856 0.004384 **
## long        -1.517e-04  4.290e-05  -3.536 0.000425 ***
## depth       -5.664e-06  1.101e-06  -5.144 3.24e-07 ***
## mag         -2.038e-01  9.891e-03 -20.603  < 2e-16 ***
## I(mag^2)     1.740e-02  9.739e-04  17.865  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.0913272)
##
```

```
##     Null deviance: 355.601  on 999  degrees of freedom
## Residual deviance:  92.448  on 994  degrees of freedom
## AIC: 7148.8
##
## Number of Fisher Scoring iterations: 5
```

Problem 4

In this problem we analyze the beerfoam data. The data set contains 13 observations of measurements of wet foam height and beer height at various time points for Shiner Bock at 20C.

4.1

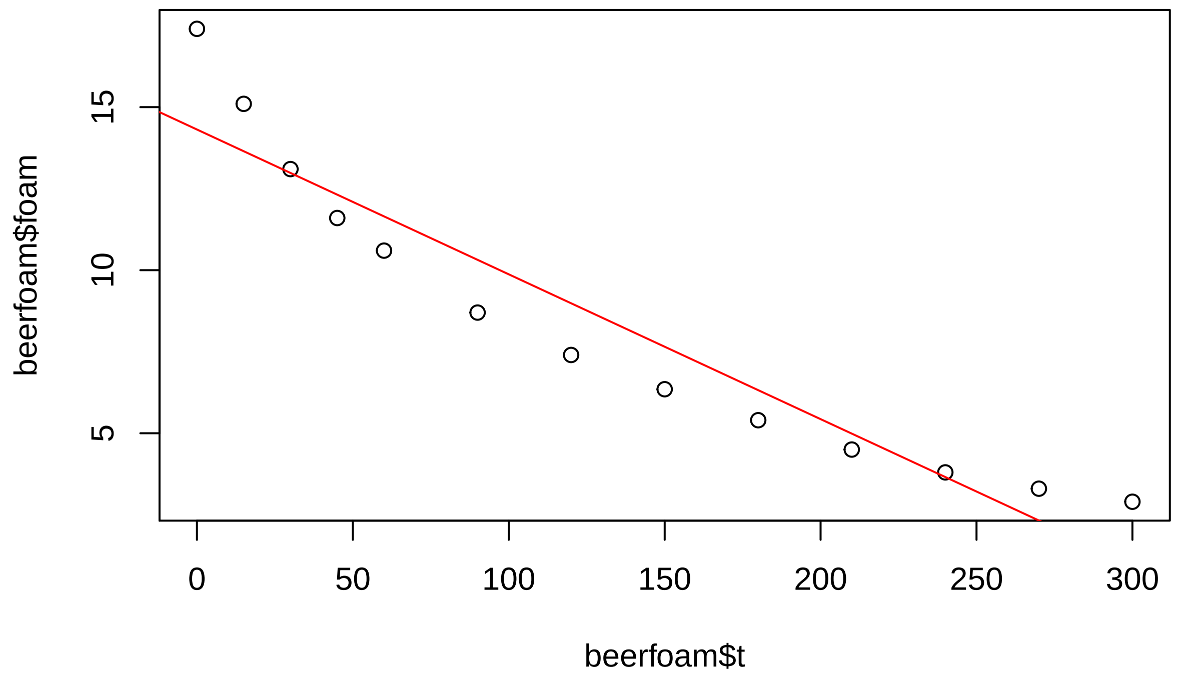
We fit a simple linear regression model for the foam height as a function of the time.

```r
fit1 <- lm(foam ~ t, data = beerfoam)
summary(fit1)
```

```
##
## Call:
## lm(formula = foam ~ t, data = beerfoam)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6175 -1.0496 -0.4891  0.9750  3.0862
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 14.313796   0.712673   20.09 5.11e-10 ***
## t           -0.044403   0.004351  -10.21 6.03e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.531 on 11 degrees of freedom
## Multiple R-squared:  0.9045, Adjusted R-squared:  0.8958
## F-statistic: 104.1 on 1 and 11 DF,  p-value: 6.034e-07
```

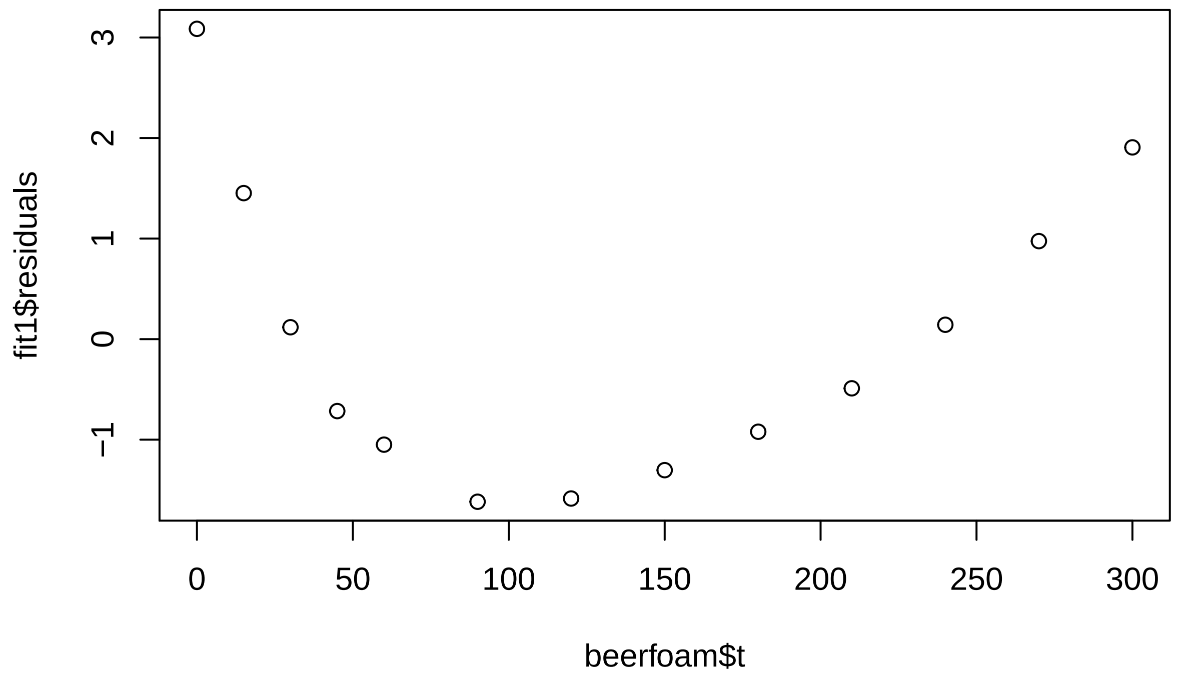
We plot the observations (black) and the fitted regression (red)

```r
##we plot also the observations
plot(beerfoam$t, beerfoam$foam)
abline(fit1, col = "red")
```



The model does not seem very good, we check also the residual vs the predictor and the residuals normal Q-Q plot.

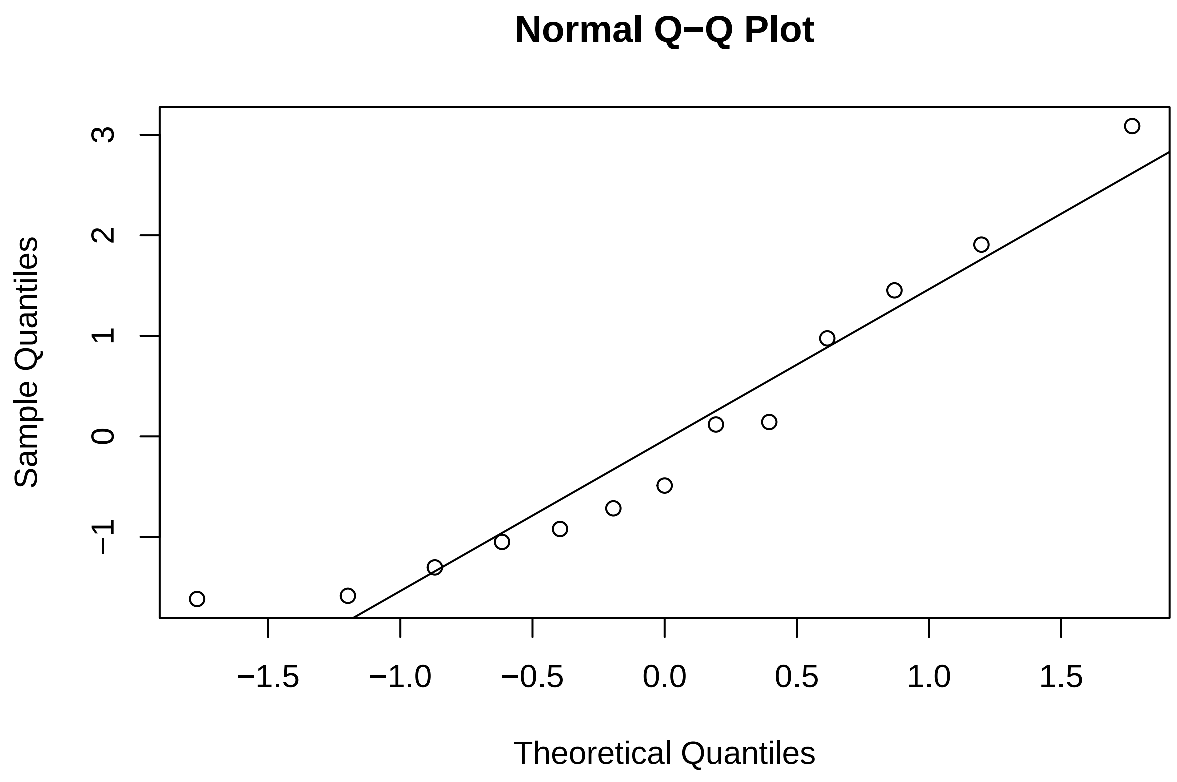
```r
plot(beerfoam$t, fit1$residuals)
```



The residuals vs predictor plot shows a clear dependency between time and residuals, that is a clear hint that the model is incorrect.

Moreover the Q-Q plot against normal quantiles shows a departure from normality:

```r
qqnorm(fit1$residuals)
qqline(fit1$residuals)
```



4.2

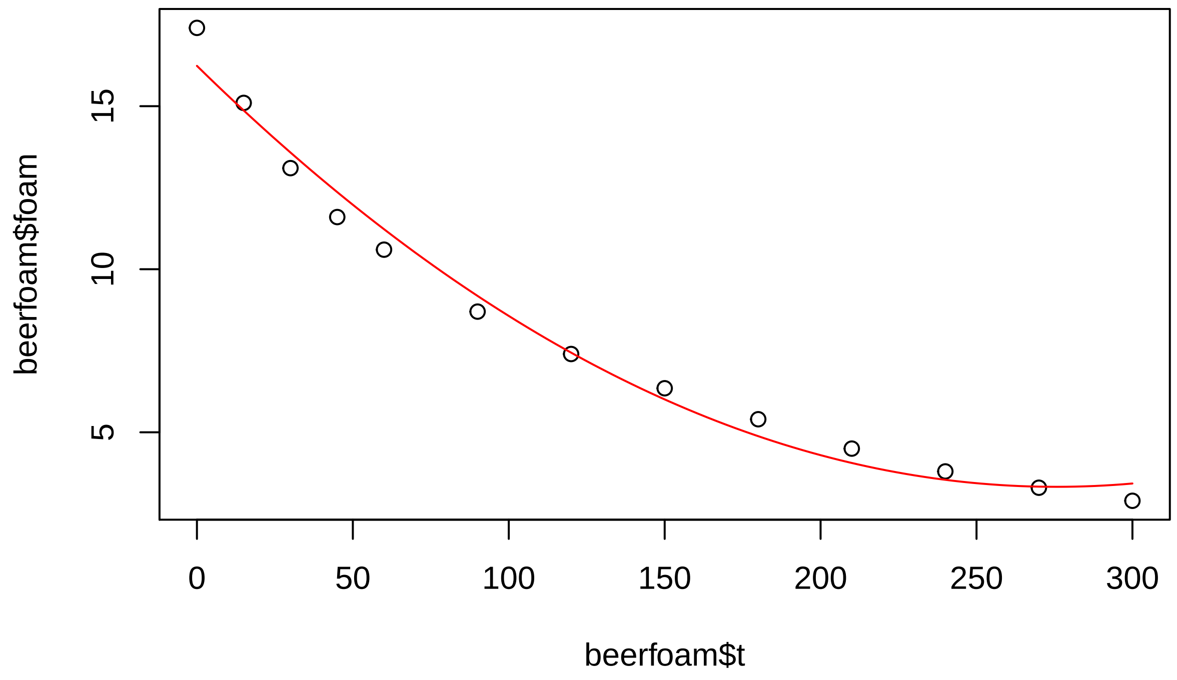
We fit now a quadratic regression model.

```r
fit2 <- lm(foam ~ t + I(t^2), data = beerfoam)
summary(fit2)
```

```
##
## Call:
## lm(formula = foam ~ t + I(t^2), data = beerfoam)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.76423 -0.48128 -0.03335  0.34182  1.16444
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.624e+01  3.811e-01  42.605 1.22e-12 ***
## t           -9.368e-02  6.687e-03 -14.008 6.73e-08 ***
## I(t^2)       1.700e-04  2.227e-05   7.633 1.77e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6146 on 10 degrees of freedom
## Multiple R-squared:  0.986,  Adjusted R-squared:  0.9832
## F-statistic: 352.3 on 2 and 10 DF,  p-value: 5.366e-10
```

We plot the observations and the fitted curve

```
plot(beerfoam$t, beerfoam$foam)
tt <- seq(min(beerfoam$t), max(beerfoam$t), length.out = 100)
yy <- predict(fit2, newdata = data.frame(t = tt))
lines(tt, yy, col = "red")
```



The curve fitted seems good, surely better than the simple linear regression.

We perform model selection with AIC, BIC

```
cands <- list(fit1 = fit1, fit2 = fit2)
sapply(cands, function(m) c(aic = AIC(m), bic = BIC(m)))
```

```
##         fit1     fit2
## aic 51.79606 28.82456
## bic 53.49091 31.08436
```

As expected the quadratic regression model is selected by both scores.

We perform also the F-test

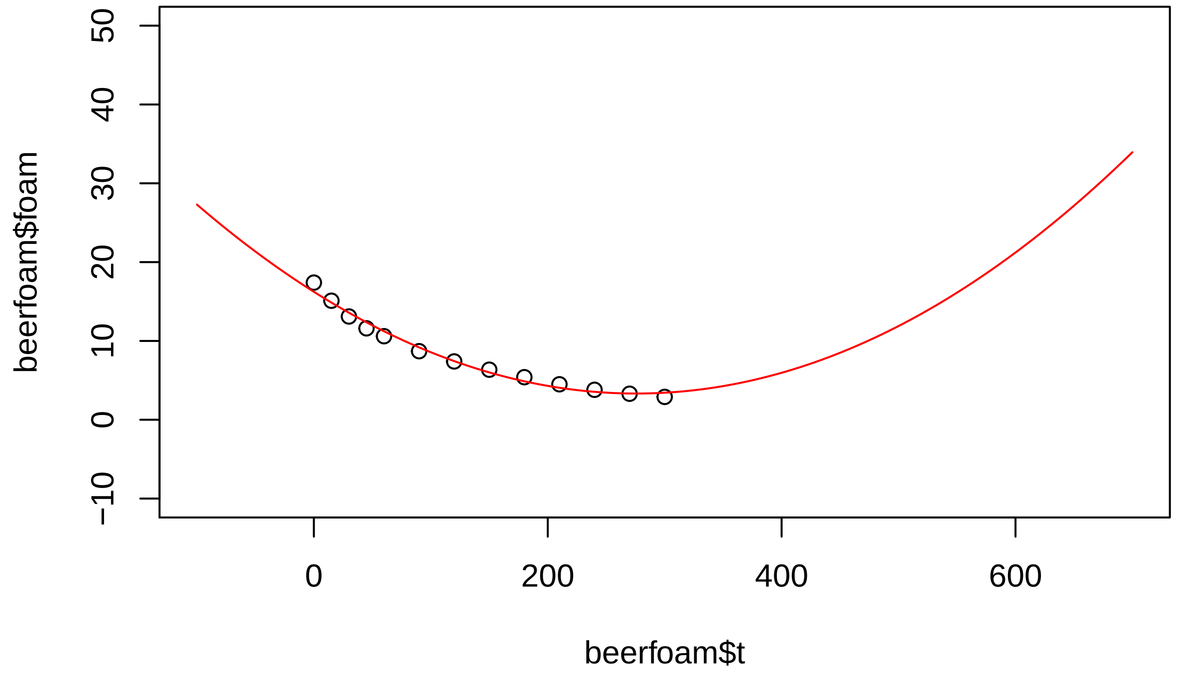
```
anova(fit1, fit2)
```

```
## Analysis of Variance Table
##
## Model 1: foam ~ t
## Model 2: foam ~ t + I(t^2)
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1     11 25.7864
## 2     10  3.7771  1    22.009 58.27 1.771e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And clearly the null hypothesis that the simple regression is sufficient is rejected (the p-value is very small).

We now observe that even if the quadratic model fit well the data in the range of the observed values, the behavior of the model is at least strange when we predict outside of the range of the observations.

```
plot(beerfoam$t, beerfoam$foam, xlim = c(-100, 700), ylim = c(-10,+50))
tt <- seq(-100, 700, length.out = 500)
yy <- predict(fit2, newdata = data.frame(t = tt))
lines(tt, yy, col = "red")
```



In particular the model predict that the height of the beer foam will increase after some time (that is a strange behavior)

4.3

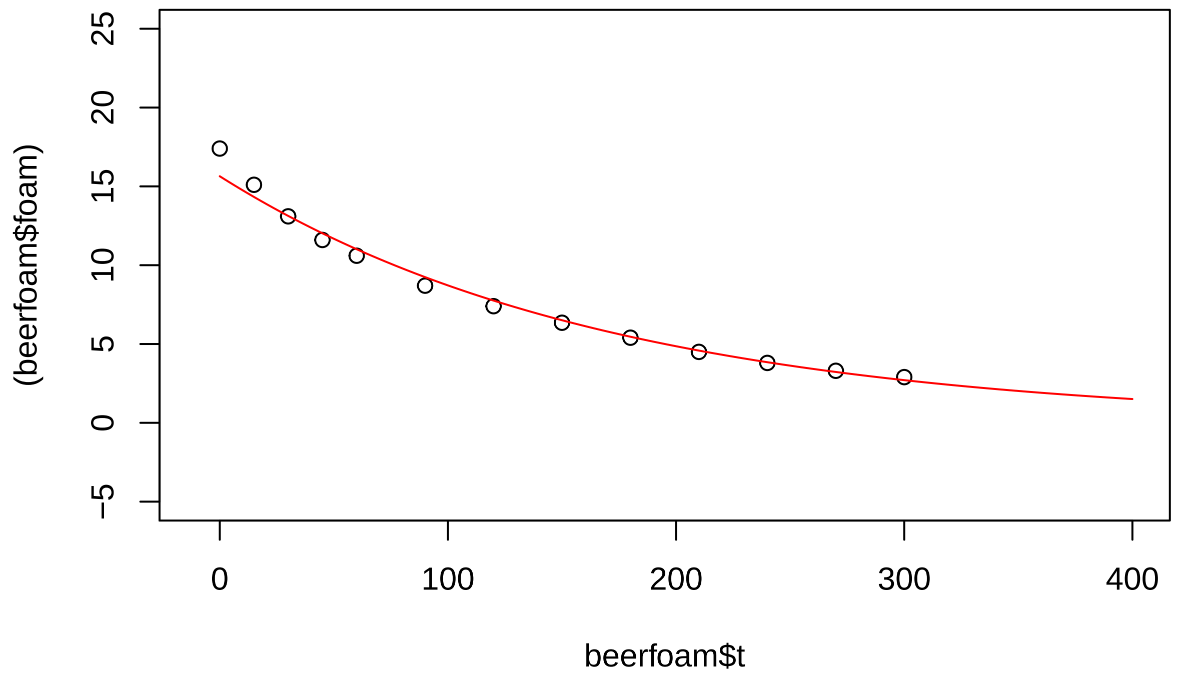
We fit the linear regression for log(`foam`).

```
fit3 <- lm(log(foam) ~ t, data = beerfoam)
```

And we plot the regression function on top of the points:

```
plot(beerfoam$t, (beerfoam$foam), xlim = c(-10, 400), ylim = c(-5,+25))
xx <- seq(0, 400, length.out = 1000)
```

```r
yy <- exp(predict(fit3, newdata = data.frame(t = xx)))
points(xx, yy, type = "l", col = "red")
```



Model selection:

```r
AIC(fit1, fit2, fit3)
```

```
##      df       AIC
## fit1  3  51.79606
## fit2  4  28.82456
## fit3  3 -36.42118
```

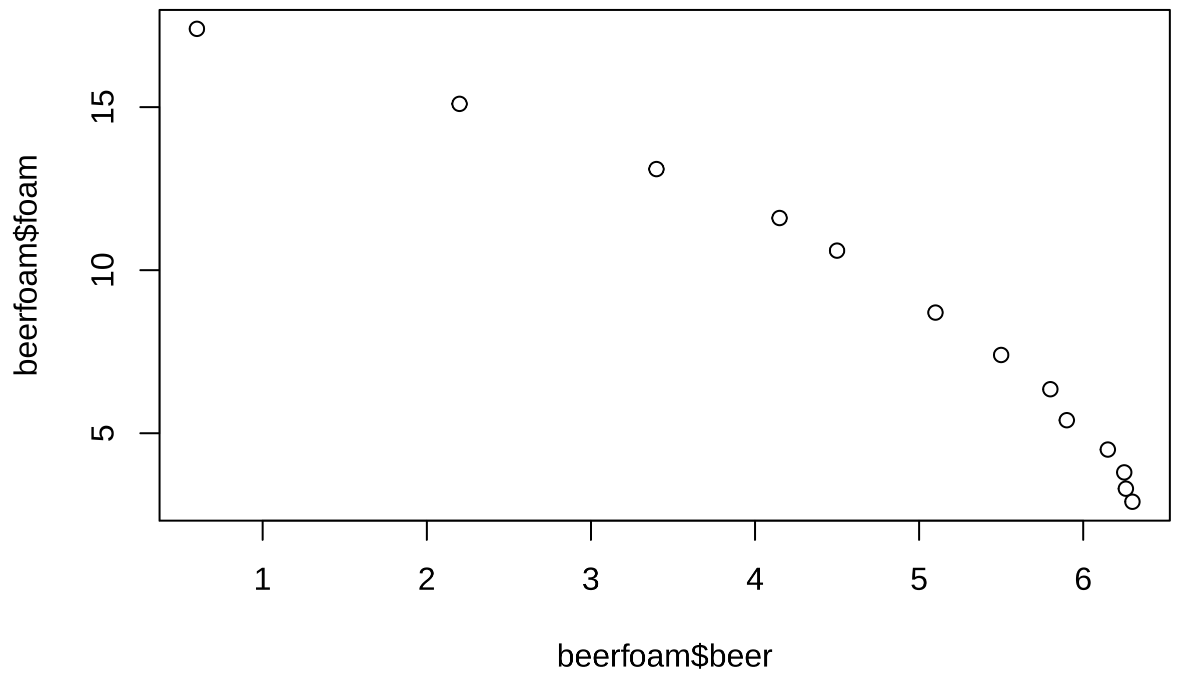
```r
BIC(fit1, fit2, fit3)
```

```
##      df       BIC
## fit1  3  53.49091
## fit2  4  31.08436
## fit3  3 -34.72634
```

4.4

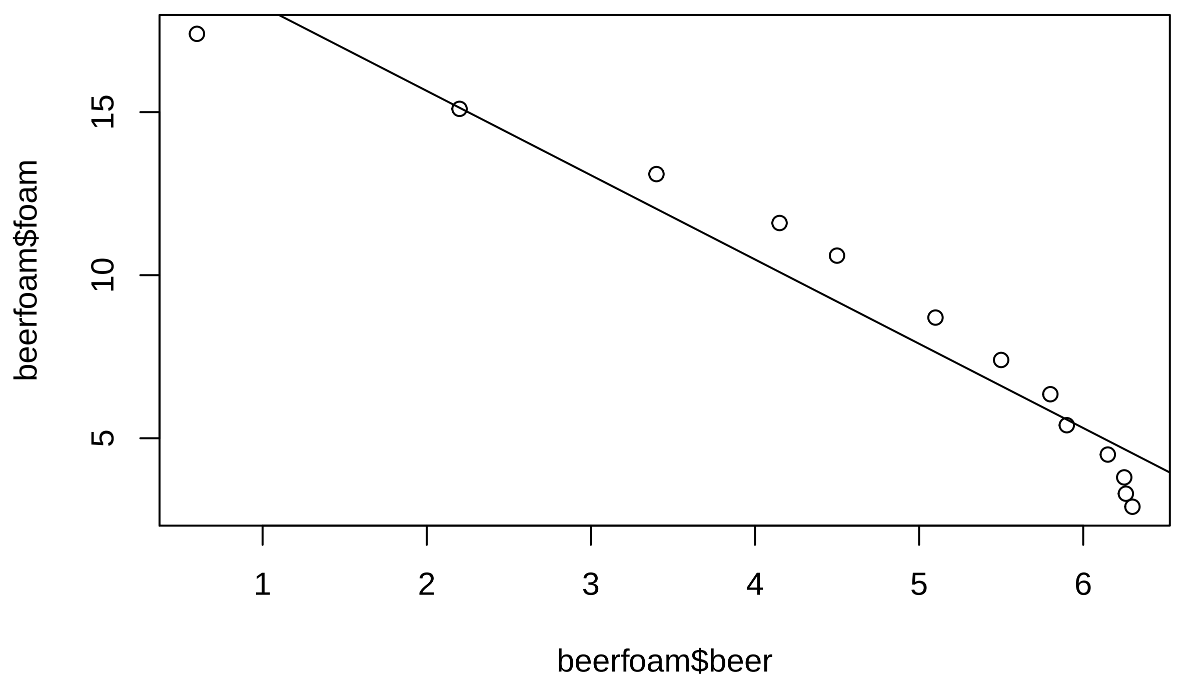
Plot the points:

```r
plot(beerfoam$beer, beerfoam$foam)
```

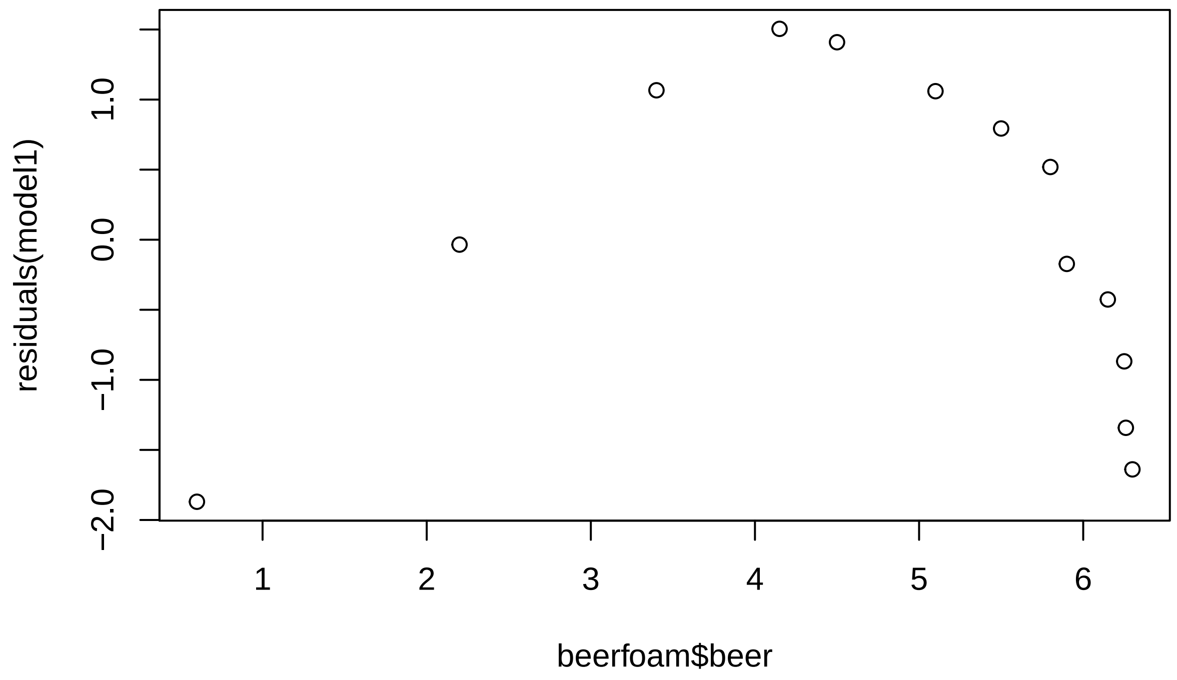


We could try a simple straight line (but we can guess that will not work well):

```r
model1 <- lm(foam ~ beer, data = beerfoam)
plot(beerfoam$beer, beerfoam$foam)
abline(model1)
```



```r
plot(beerfoam$beer, residuals(model1))
```



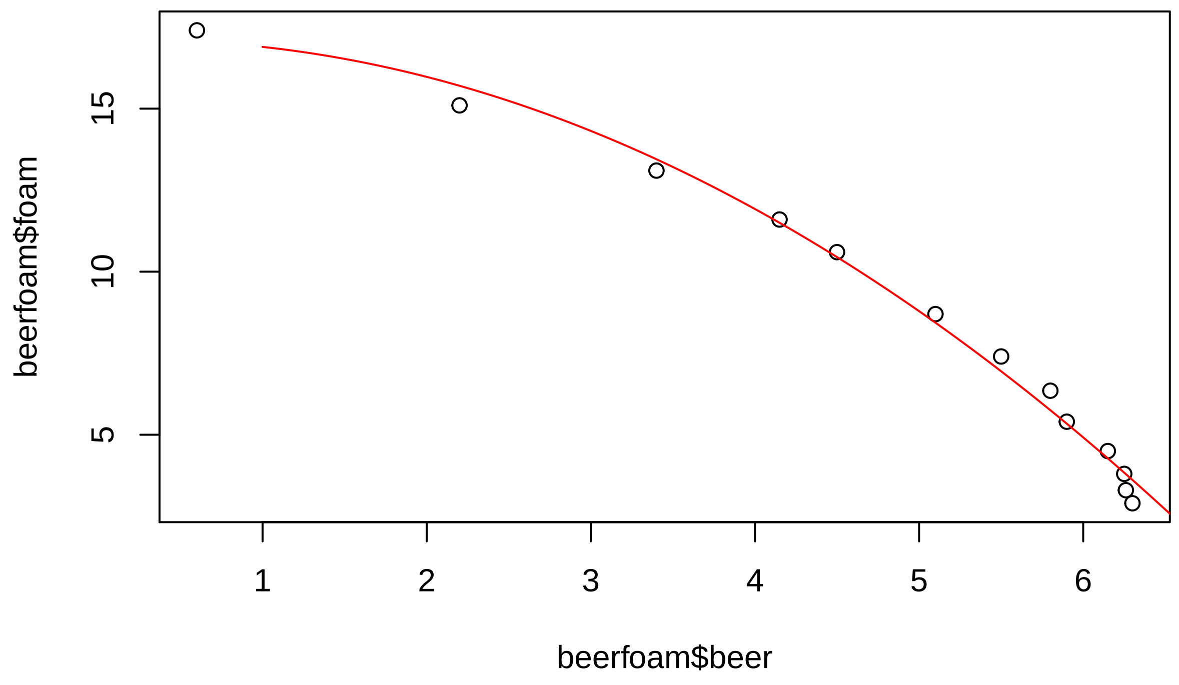
From the plot of the residuals we observe that the simple linear model is probably not appropriate.

We could try a polynomial regression.

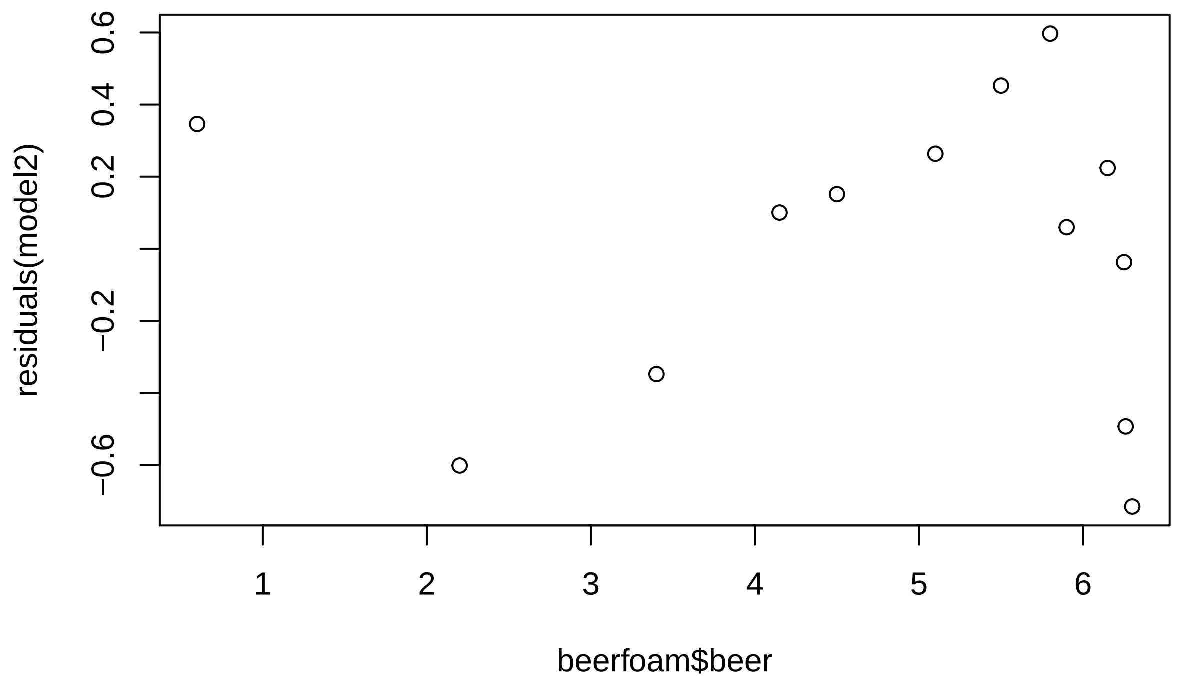
```
model2 <- lm(foam ~ beer + I(beer^2), data = beerfoam)
summary(model2)
```

```
##
## Call:
## lm(formula = foam ~ beer + I(beer^2), data = beerfoam)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7152 -0.3478  0.1003  0.2636  0.5968
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.07398    0.58717  29.078 5.40e-11 ***
## beer         0.18768    0.34226   0.548    0.595
## I(beer^2)   -0.36889    0.04447  -8.295 8.56e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4556 on 10 degrees of freedom
## Multiple R-squared:  0.9923, Adjusted R-squared:  0.9908
## F-statistic: 645.1 on 2 and 10 DF,  p-value: 2.691e-11
```

```r
##
plot(beerfoam$beer, beerfoam$foam)
xx <- seq(1,7,length.out = 100)
yy <- predict(model2, newdata = data.frame(beer = xx))
points(xx, yy, type = "l", col = "red")
```



```r
## residuals
plot(beerfoam$beer, residuals(model2))
```



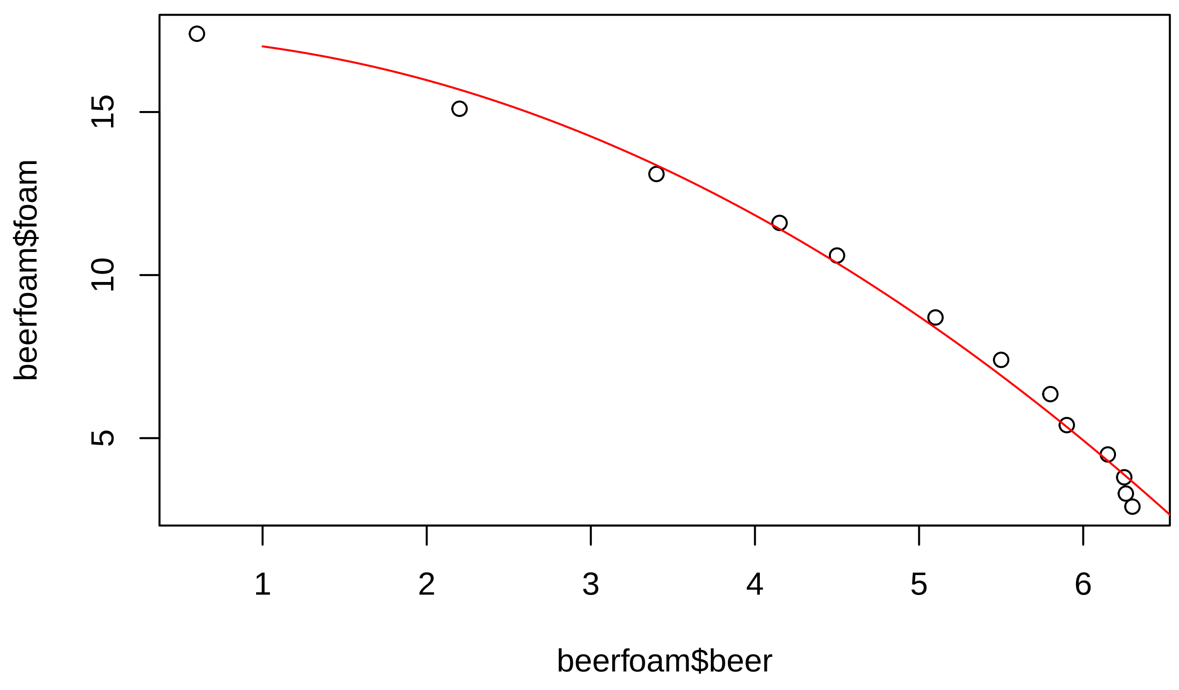
We can also try the polynomial model foam = beer^2, since in model2 the coefficient for beer is not significant.

```
model3 <- lm(foam ~ I(beer^2), data = beerfoam)
summary(model3)
```

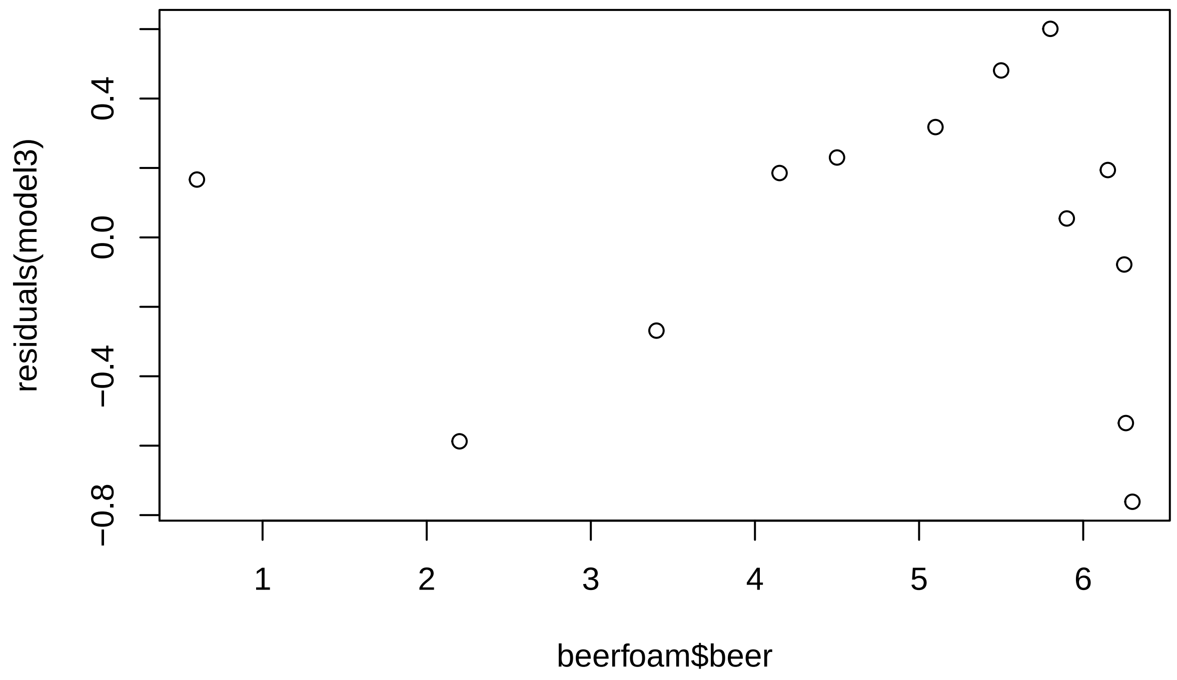
```
##
## Call:
## lm(formula = foam ~ I(beer^2), data = beerfoam)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7615 -0.2685  0.1666  0.2302  0.6008
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.357644   0.268807   64.57 1.52e-15 ***
## I(beer^2)   -0.345078   0.009298  -37.11 6.55e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4409 on 11 degrees of freedom
## Multiple R-squared:  0.9921, Adjusted R-squared:  0.9914
## F-statistic:  1377 on 1 and 11 DF,  p-value: 6.554e-13
```

```
##
plot(beerfoam$beer, beerfoam$foam)
xx <- seq(1,7,length.out = 100)
```

```r
yy <- predict(model3, newdata = data.frame(beer = xx))
points(xx, yy, type = "l", col = "red")
```



```r
## residuals
plot(beerfoam$beer, residuals(model3))
```



And:

```r
AIC(model1, model2, model3)
```

```
##        df      AIC
## model1  3 45.88113
## model2  4 21.04328
## model3  3 19.42840
```

```r
BIC(model1, model2, model3)
```

```
##        df      BIC
## model1  3 47.57598
## model2  4 23.30307
## model3  3 21.12325
```

Moreover the F-test

```r
anova(model3, model2)
```

```
## Analysis of Variance Table
##
## Model 1: foam ~ I(beer^2)
## Model 2: foam ~ beer + I(beer^2)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     11 2.1383
## 2     10 2.0759  1  0.062419 0.3007 0.5955
```

From the above model selection procedure we can argue that among the three models tested `model3` is to prefer.