Assignment 1

Johnsen

2022-10-01

Projekt 1: Wind Power Forecast

Descriptive Statistics

```
D <- read.table("tuno.txt", header=TRUE, sep=" ",
                as.is=TRUE)

D$date <- as.Date("2003-01-01")-1+D$r.day
D$pow.obs.norm <- D$pow.obs/5000
```

Read the data tuno.txt into R

Make a graphical presentation of data or parts of the data, and present some summary statistics. Summary statistics:

```
## Dimensions of D (number of rows and columns)
dim(D)
```

```
## [1] 288   8
```

```
##  Column/variable names
names(D)
```

```
## [1] "r.day"        "month"        "day"          "pow.obs"      "ws30"
## [6] "wd30"         "date"         "pow.obs.norm"
```

```
## The first rows/observations
head(D)
```

```
##   r.day month day   pow.obs     ws30      wd30       date pow.obs.norm
## 1     1     1   1  243.0278 6.723611 4.0343405 2003-01-01   0.04860556
## 2     2     1   2 2780.0137 4.272603 2.1365208 2003-01-02   0.55600274
## 3     3     1   3 2118.6164 4.272603 1.6240318 2003-01-03   0.42372329
## 4     4     1   4 1660.8767 6.541096 0.2269022 2003-01-04   0.33217534
## 5     5     1   5 1872.7945 9.713699 5.3161852 2003-01-05   0.37455890
## 6     6     1   6 3212.2603 8.161644 0.9522963 2003-01-06   0.64245205
```

```
## The last rows/observations
tail(D)
```

```
##     r.day month day   pow.obs      ws30      wd30       date pow.obs.norm
## 283   299    10  26  787.0000  9.323288 0.3152175 2003-10-26   0.15740000
## 284   300    10  27 1869.6438 11.280137 5.2411088 2003-10-27   0.37392877
## 285   301    10  28 2551.5205 12.623973 4.7614043 2003-10-28   0.51030411
## 286   302    10  29 2564.5616 11.154795 3.6750237 2003-10-29   0.51291233
## 287   303    10  30  449.5205  5.714384 3.0080934 2003-10-30   0.08990411
## 288   304    10  31  781.8082  6.102740 3.0877370 2003-10-31   0.15636164
```

```
## Selected summary statistics
summary(D)
```

```
##      r.day            month             day           pow.obs
##  Min.   :  1.00   Min.   : 1.000   Min.   : 1.00   Min.   :   0.123
##  1st Qu.: 78.75   1st Qu.: 3.000   1st Qu.: 8.00   1st Qu.: 254.158
##  Median :156.50   Median : 6.000   Median :15.00   Median : 964.123
##  Mean   :154.30   Mean   : 5.594   Mean   :15.47   Mean   :1381.196
##  3rd Qu.:229.25   3rd Qu.: 8.000   3rd Qu.:23.00   3rd Qu.:2196.579
##  Max.   :304.00   Max.   :10.000   Max.   :31.00   Max.   :4681.062
##       ws30             wd30               date             pow.obs.norm
##  Min.   : 1.139   Min.   :0.000095   Min.   :2003-01-01   Min.   :0.0000247
##  1st Qu.: 5.779   1st Qu.:2.474999   1st Qu.:2003-03-19   1st Qu.:0.0508315
##  Median : 8.498   Median :4.079297   Median :2003-06-05   Median :0.1928247
##  Mean   : 9.112   Mean   :3.602390   Mean   :2003-06-03   Mean   :0.2762392
##  3rd Qu.:11.202   3rd Qu.:4.945443   3rd Qu.:2003-08-17   3rd Qu.:0.4393158
##  Max.   :24.950   Max.   :6.274642   Max.   :2003-10-31   Max.   :0.9362123
```

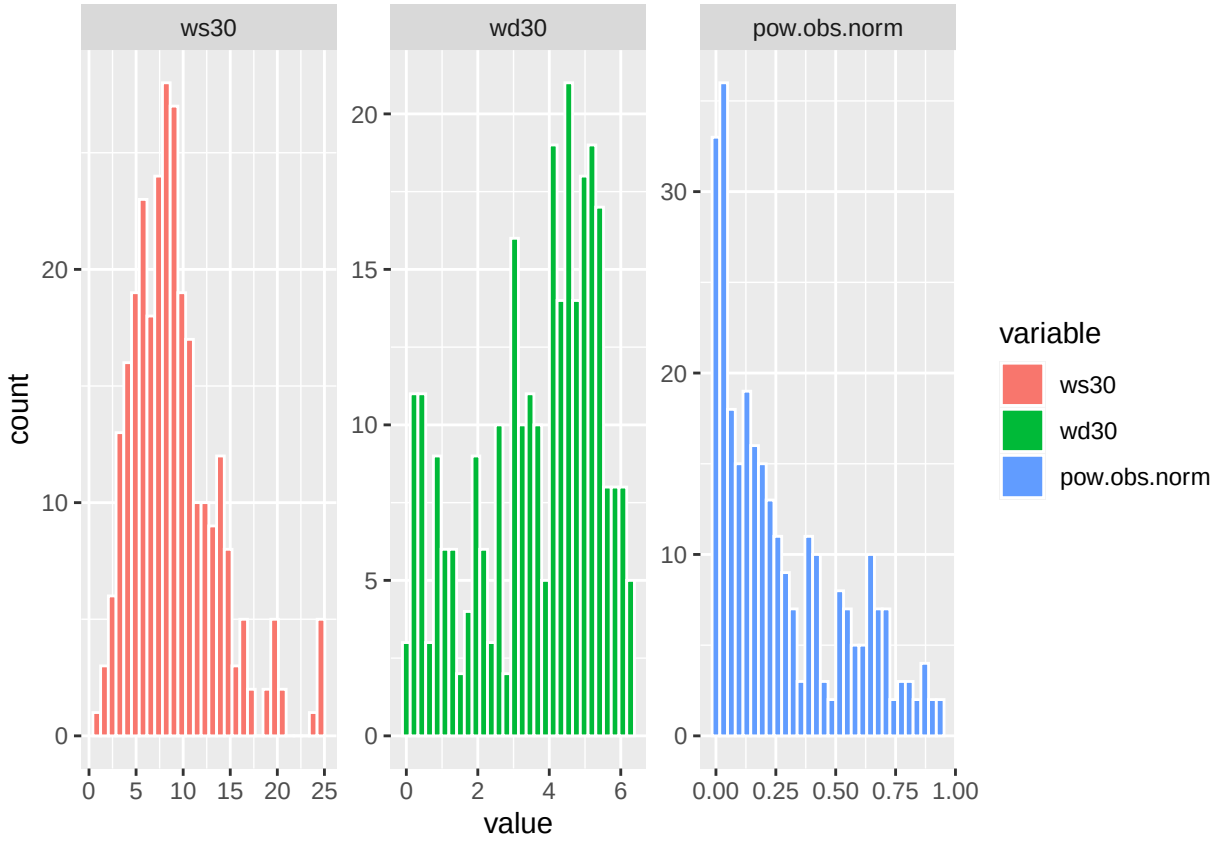
```
## Another type of summary of the dataset
str(D)
```

```
## 'data.frame':    288 obs. of  8 variables:
##  $ r.day       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ day         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ pow.obs     : num  243 2780 2119 1661 1873 ...
##  $ ws30        : num  6.72 4.27 4.27 6.54 9.71 ...
##  $ wd30        : num  4.034 2.137 1.624 0.227 5.316 ...
##  $ date        : Date, format: "2003-01-01" "2003-01-02" ...
##  $ pow.obs.norm: num  0.0486 0.556 0.4237 0.3322 0.3746 ...
```

Visualization of the three relevant variables:

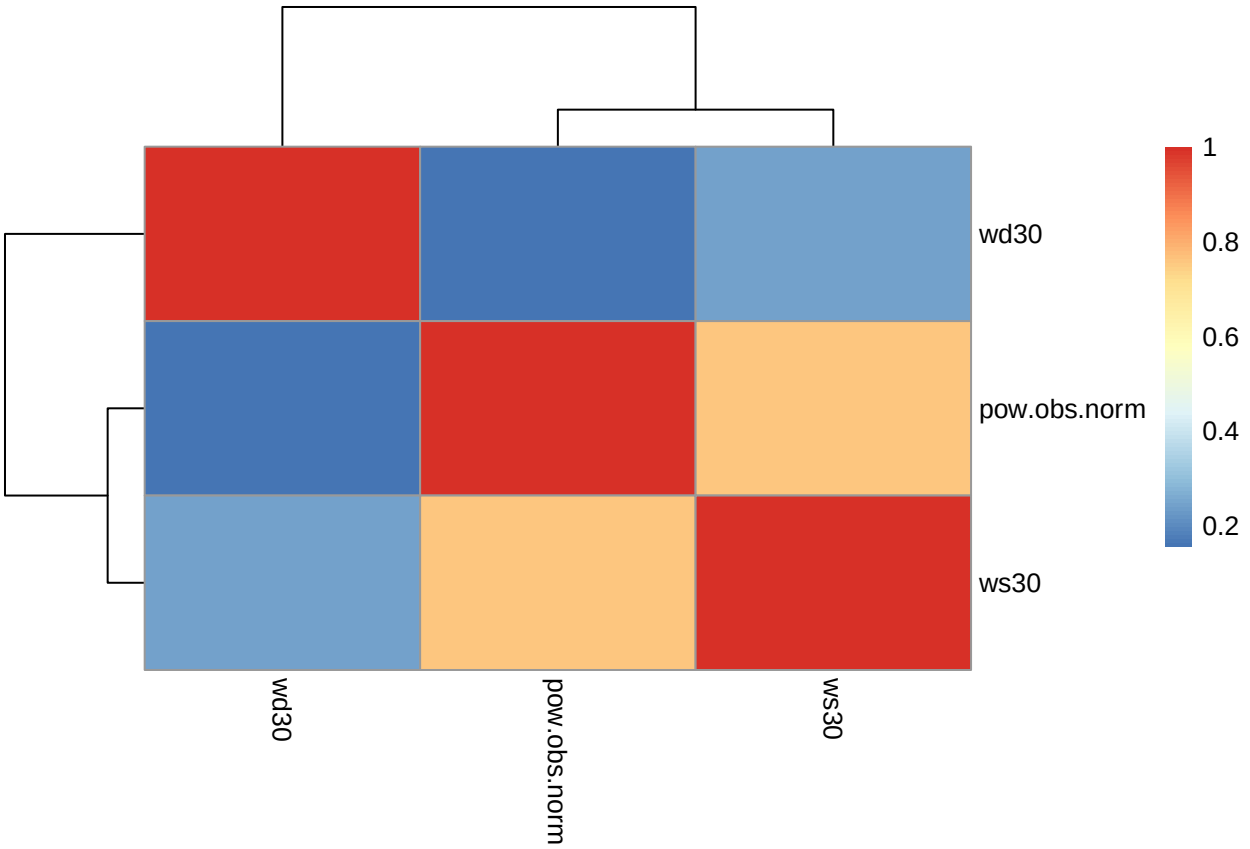
```
meltD <- D %>%
  select(-r.day, -month, -day, -pow.obs) %>%
  melt(id.vars = "date")

ggplot(meltD)+
  geom_histogram(aes(x = value, fill = variable), colour = "white")+
  facet_wrap(~ variable, scales = "free")
```

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

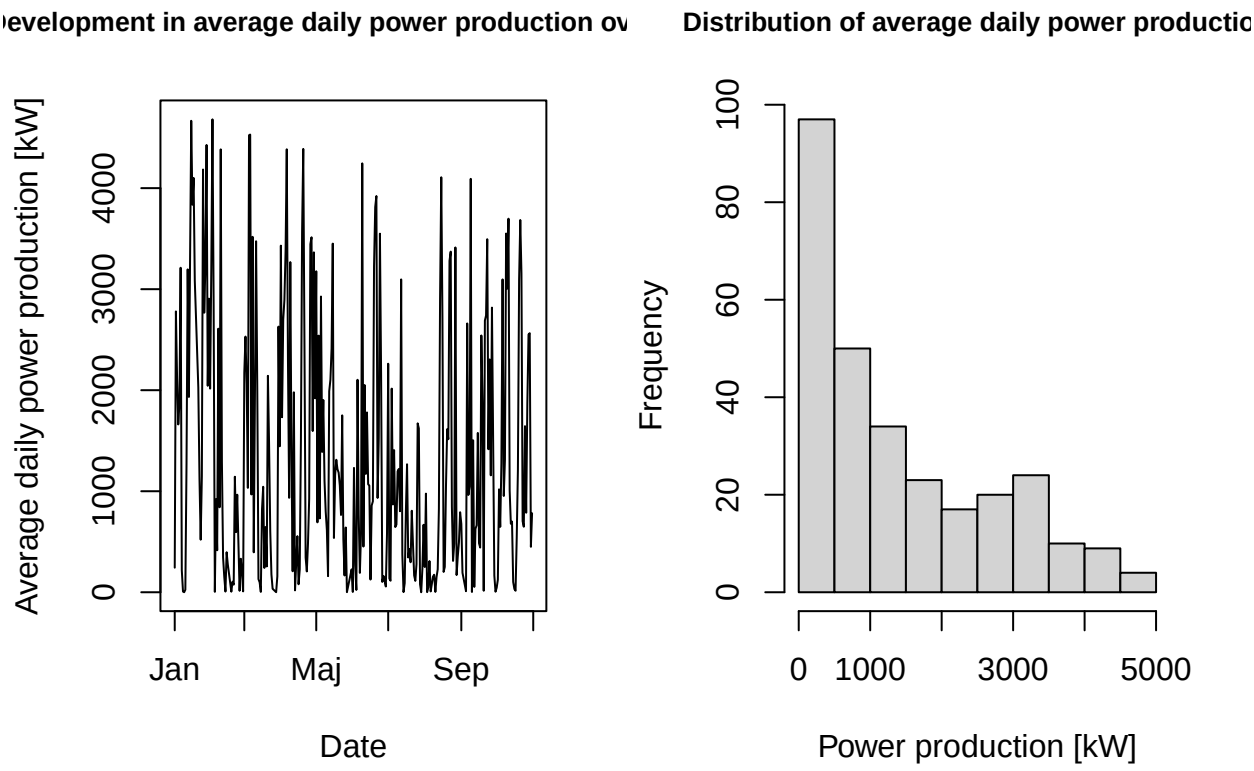


Correlation analysis

```
D %>%
  select(pow.obs.norm, wd30, ws30) %>%
  cor() %>%
  pheatmap()
```

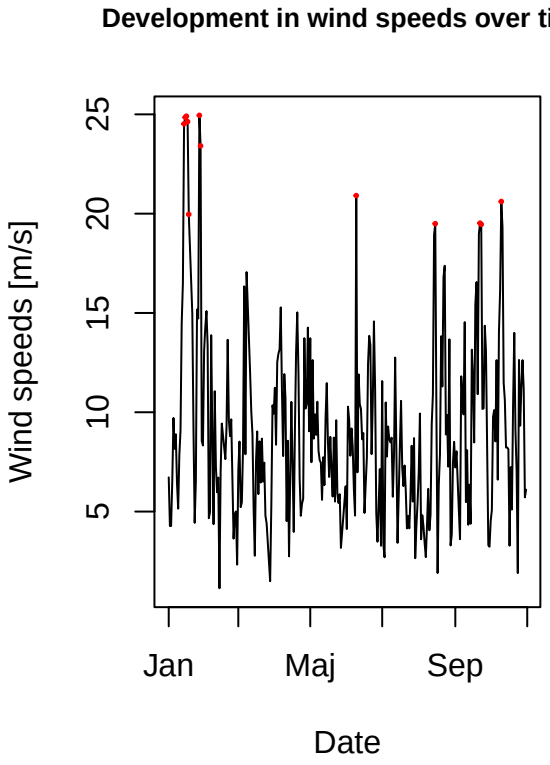


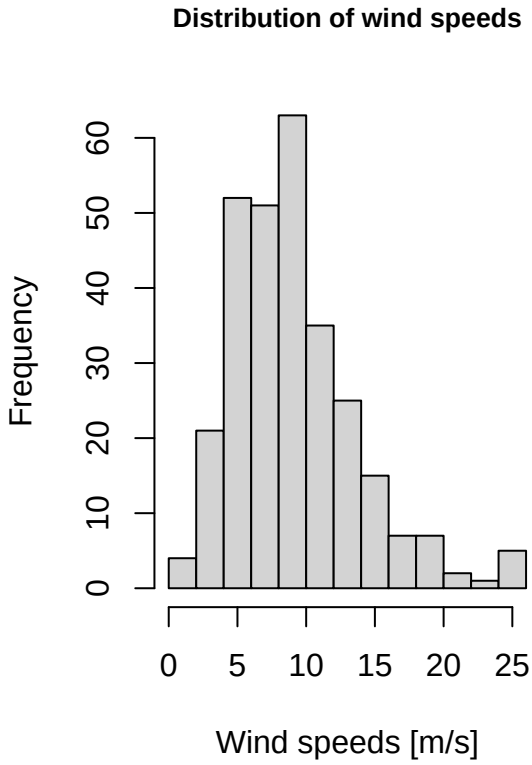

```
par(mfrow=c(1,2))
plot(D$date, D$pow.obs, type = 'l', xlab="Date", ylab="Average daily power production [kW]",
     main = 'Development in average daily power production over time', cex.main = 0.8, col=1)
hist(D$pow.obs, xlab="Power production [kW]", main='Distribution of average daily power production', ce
```



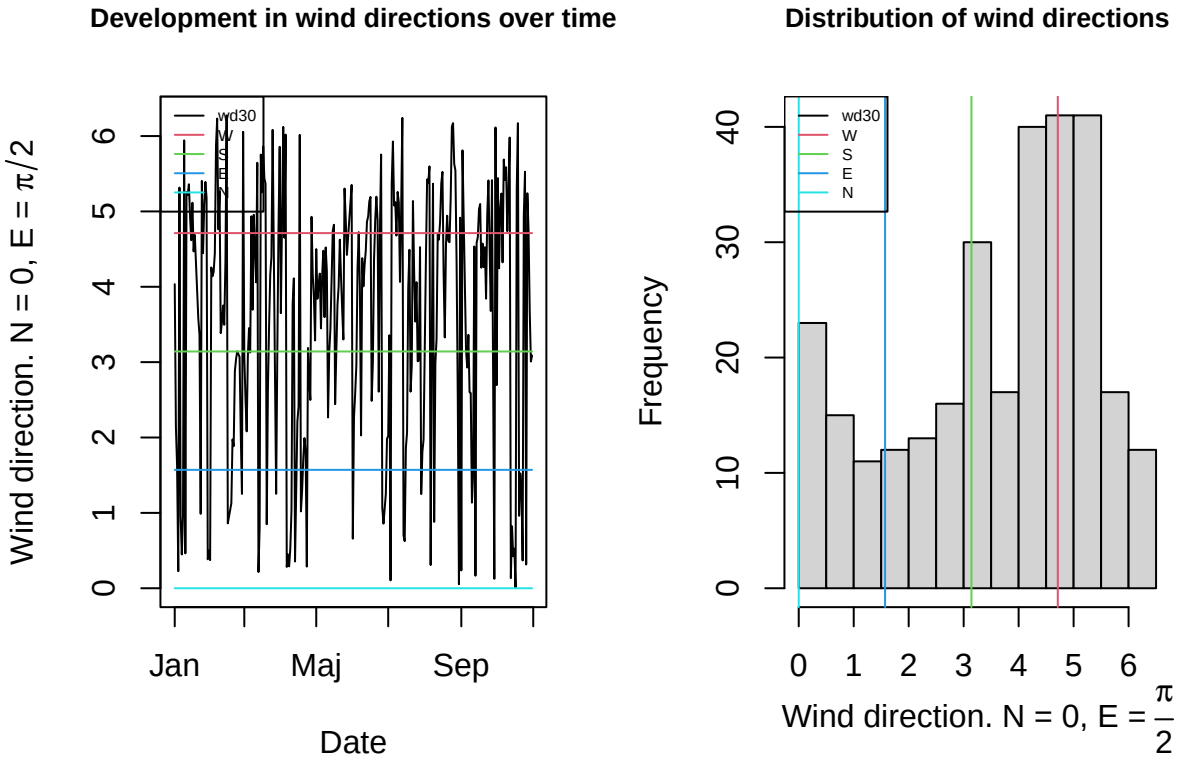
Outlier analysis

```
outlierFUN <- function(data, quantiles){
  v <- quantile(x = data, probs = quantiles)
  IQR <- v[2] - v[1]
  outliers <- ( ( data < (v[1] - 1.5 * IQR) ) | ( data > (v[2] + 1.5 * IQR) ) ) * data
  return (outliers)
}
D$outlierws30 <- outlierFUN(data = D$ws30, quantiles = c(0.25,0.75))
###              ###
par(mfrow=c(1,2))
plot(D$date, D$ws30, type = 'l', xlab="Date", ylab="Wind speeds [m/s]", cex.main = 0.8, col=1,
     main='Development in wind speeds over time')
points(x = D$date ,y = D$outlierws30, type = 'p', pch = 19, col = "red", cex = 0.25)
hist(D$ws30, xlab="Wind speeds [m/s]", main='Distribution of wind speeds', cex.main = 0.8)
```



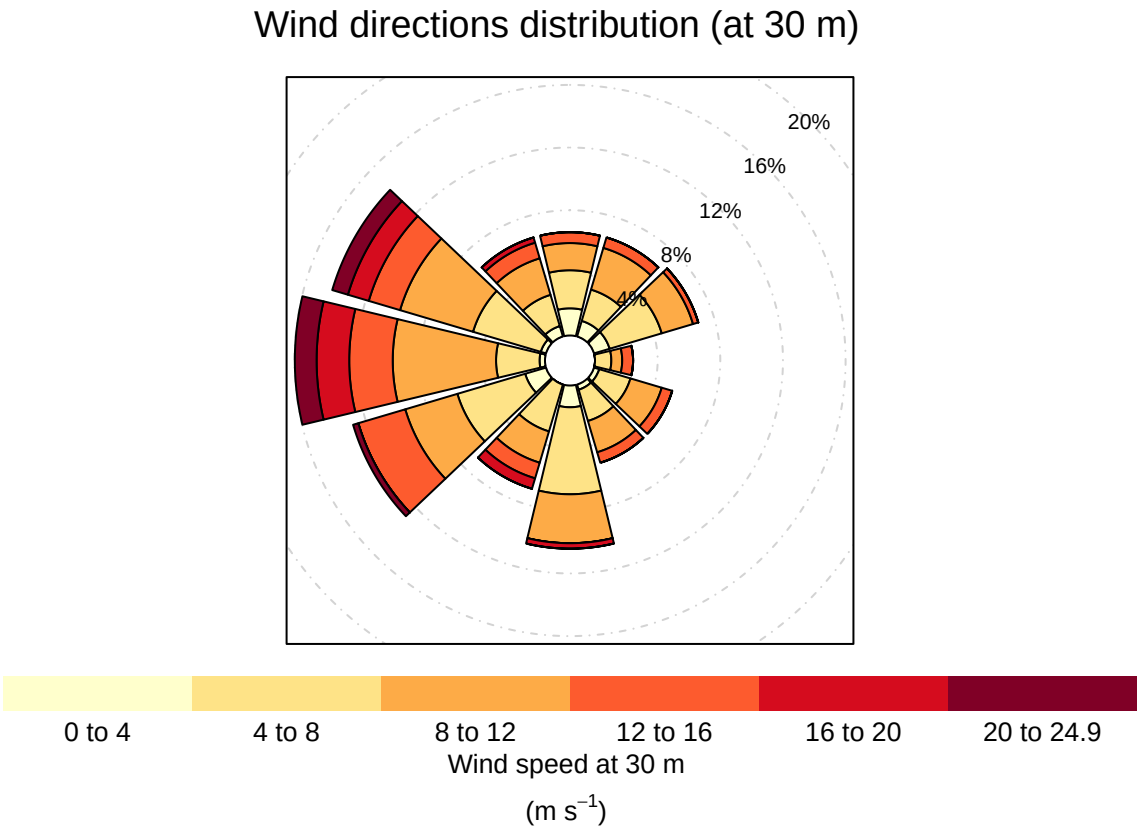



```
par(mfrow=c(1,2))
plot(D$date, D$wd30, type = 'l', xlab="Date", ylab=expression(paste("Wind direction. N = 0, E = ", pi/2
     main='Development in wind directions over time', cex.main = 0.8)
lines(D$date, replicate(length(D$date), 3*pi/2), type='l', col=2) #W
lines(D$date, replicate(length(D$date), pi), type='l', col=3) #S
lines(D$date, replicate(length(D$date), pi/2), type='l', col=4) #E
lines(D$date, replicate(length(D$date), 0), type='l', col=5) #N
legend('topleft', legend = c('wd30', 'W', 'S', 'E', 'N'), col = 1:5, lty = 1, cex = 0.5)
#
hist(D$wd30, xlab=expression(paste('Wind direction. N = 0, E = ', frac(pi,2))),
     main='Distribution of wind directions', cex.main = 0.8, freq = TRUE) #####hist to show that wind r
abline(v = 3*pi/2, col=2)
abline(v = pi, col=3)
abline(v = pi/2, col=4)
abline(v = 0, col=5)
legend('topleft', legend = c('wd30', 'W', 'S', 'E', 'N'), col = 1:5, lty = 1, cex = 0.5)
```



Wind rose

```
D$wd30dg <- D$wd30 * 180/pi
###wind d
intv <- 4
#Dwinter <- D[1:54,] #for testing the season plots above :) D$date[55] = 2003-03-01
#Dsummer <- D[140:230,] #for testing the season plots above :) D$date[140] = 2003-06-01
windRose(D, ws = "ws30", wd = "wd30dg", ws2 = NA, wd2 = NA,
         ws.int = intv, angle = 30, type = "default", bias.corr = TRUE, cols
         = "heat", grid.line = list(value=4, lty=4, col="lightgrey"), width = 1, seg = NULL, auto.text
         = TRUE, breaks = round(max(D$ws30)/intv), offset = 10, normalise = FALSE, max.freq =
           NULL, paddle = FALSE, key.header = "Wind speed at 30 m", key.footer = "(m/s)",
         key.position = "bottom", key = list(height=2), dig.lab = 3, statistic =
           "prop.count", pollutant = NULL, annotate = FALSE, angle.scale =
           45, border = "black", main="Wind directions distribution (at 30 m)",
         cex.main=0.75)
```



Simple Models

```
load("dataWindPower.Rdata")
source("testDistribution.R")
```

Fit different probability density models to wind power, wind speed and wind direction data. You might consider different models e.g. beta, gamma, log normal, and different transformations e.g. (for wind power). It is important that you consider if the distributions/transformations are reasonable for the data that you try to model. Fit an exponential, gamma and beta distribution to the observed wind power data.

```
par.exp <- nlminb(start = 0.2, objective = testDistribution,
                  distribution = "exponential",
                  x = D$pow.obs.norm)

par.exp$objective
```

```
## [1] -82.50862
```

```
par.beta <- nlminb(start = c(2,5)
                   , objective = testDistribution
                   , distribution = "beta"
```

```
                    , x = D$pow.obs.norm
                    , lower = c(0,0.8))
par.beta$objective
```

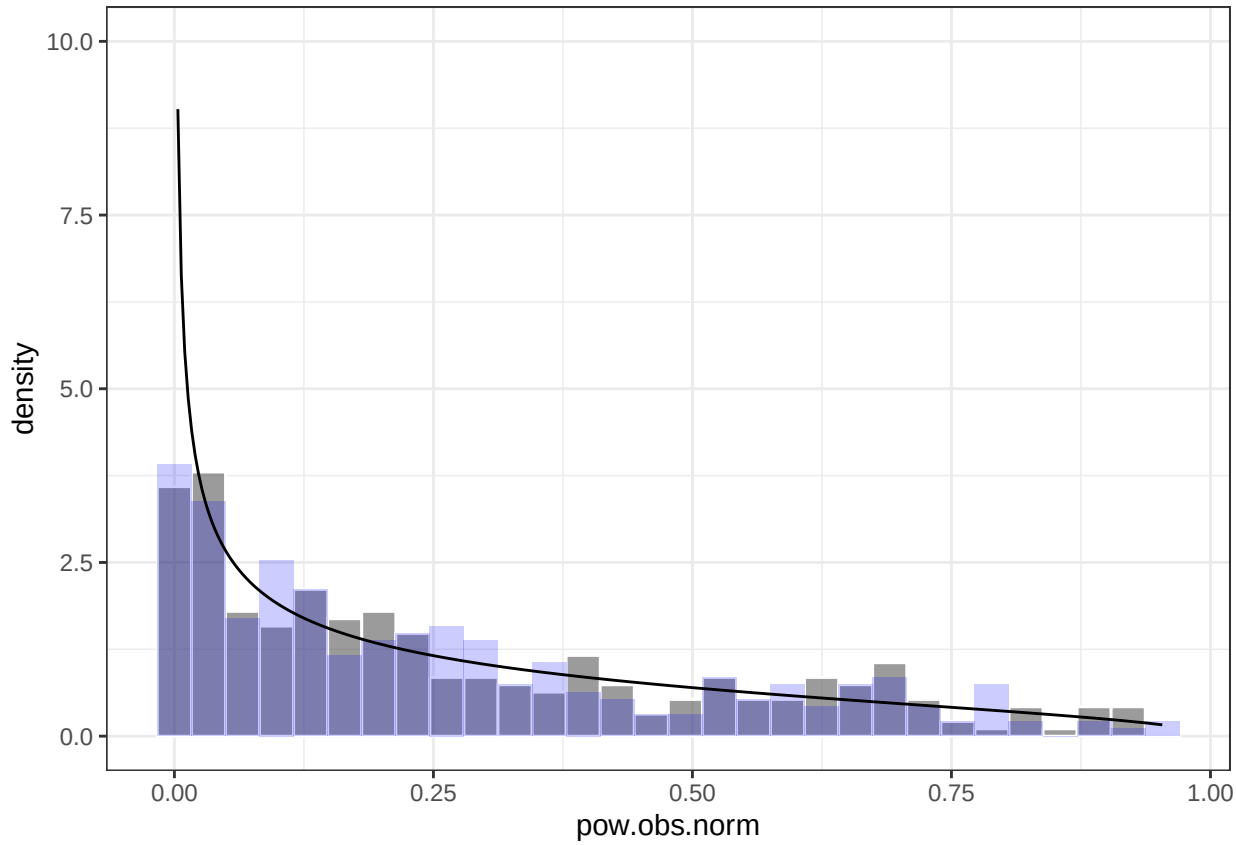
```
## [1] -121.6618
```

```
par.gamma <- nlminb(start = c(2,5)
                    ,objective = testDistribution
                    ,distribution = "gamma"
                    ,x = D$pow.obs.norm)
par.gamma$objective
```

```
## [1] -97.38174
```

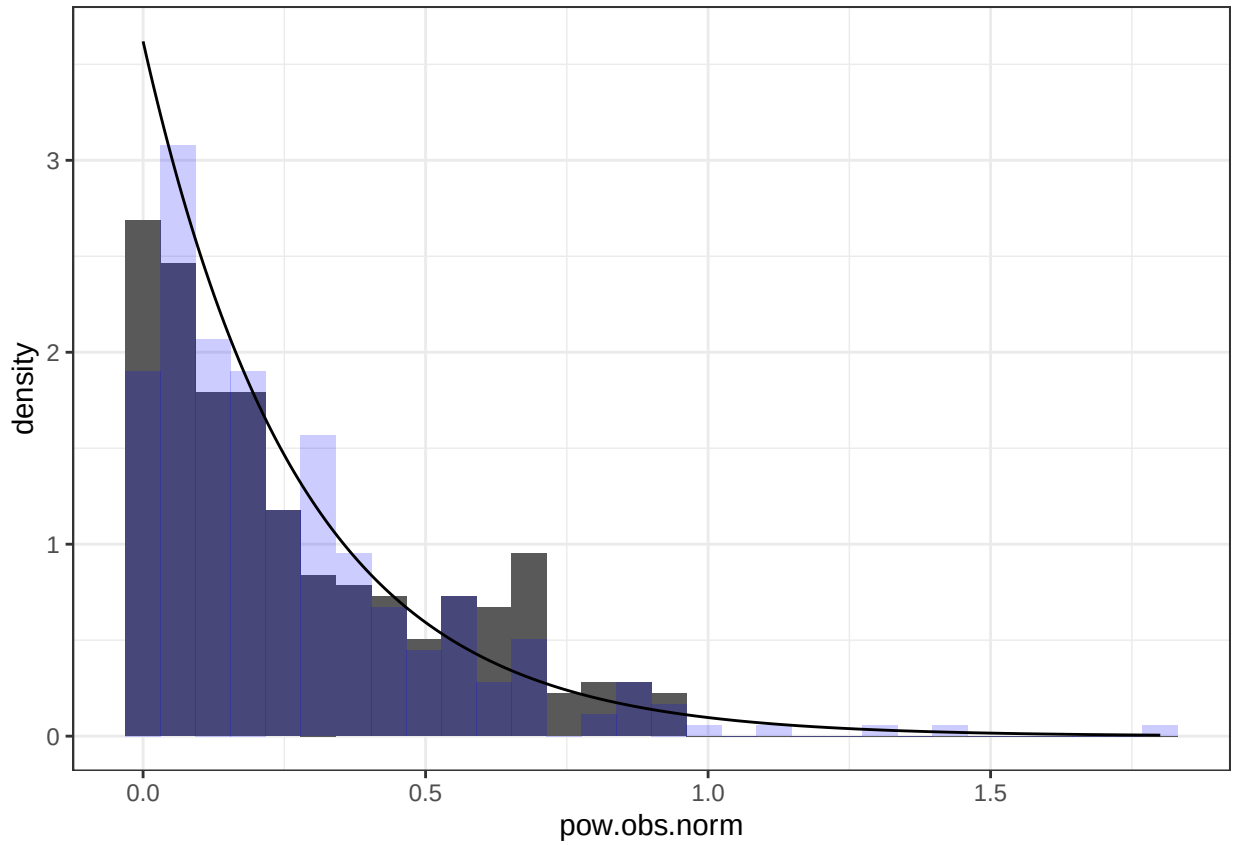
```
#Sampling from the found beta distribution
D$simdata <- rbeta(length(D$pow.obs.norm), shape1 = par.beta$par[1]
                   ,shape2 = par.beta$par[2])
b <- ggplot(D)+
  geom_histogram(aes(x = pow.obs.norm, y =..density..), colour = "white", alpha = 0.6)+
  geom_histogram(aes(x = simdata, y =..density..), alpha = 0.2, fill = "blue")+
  theme_bw()+
  ylim(c(0,10))+
  stat_function(fun = dbeta, n = length(D$pow.obs.norm), args = list(shape1 = par.beta$par[1],shape2 = p
show(b)
```

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



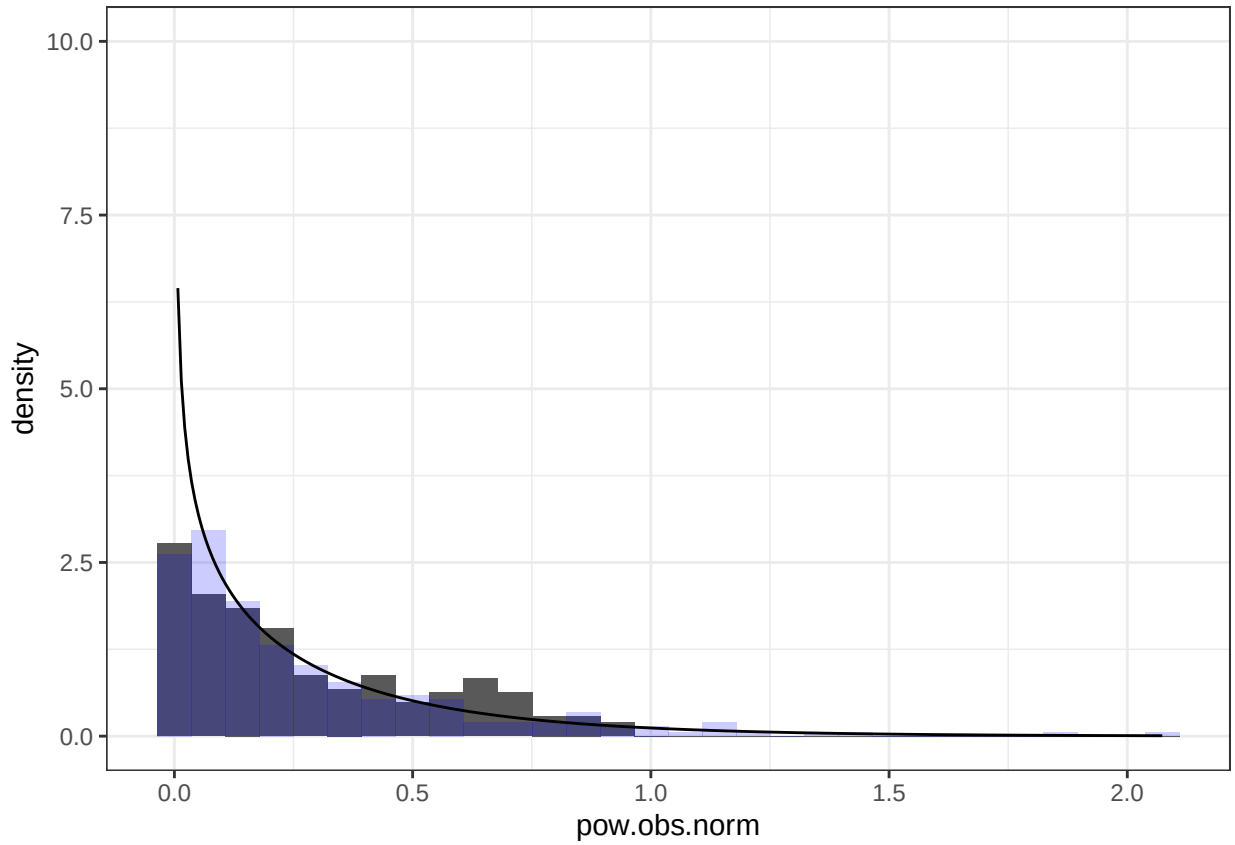

```
#Sampling from the found exp distribution
D$simdata <- rexp(length(D$pow.obs.norm), rate = par.exp$par)
g <- ggplot(D)+
  geom_histogram(aes(x = pow.obs.norm, y = ..density..))+
  geom_histogram(aes(x = simdata, y = ..density..)
                 , alpha = 0.2, fill = "blue")+
  theme_bw()+
  stat_function(fun = dexp, n = length(D$pow.obs.norm), args = list(rate = par.exp$par))
show(g)
```

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




```
#Sampling from the found gamma distribution
D$simdata <- rgamma(length(D$pow.obs.norm), shape = par.gamma$par[1], rate = par.gamma$par[2])
g <- ggplot(D)+
  geom_histogram(aes(x = pow.obs.norm, y = ..density..))+
  geom_histogram(aes(x = simdata, y = ..density..)
                 , alpha = 0.2, fill = "blue")+
  theme_bw()+
  ylim(c(0,10))+
  stat_function(fun = dgamma, n = length(D$pow.obs.norm), args = list(shape = par.gamma$par[1], rate = p
show(g)
```

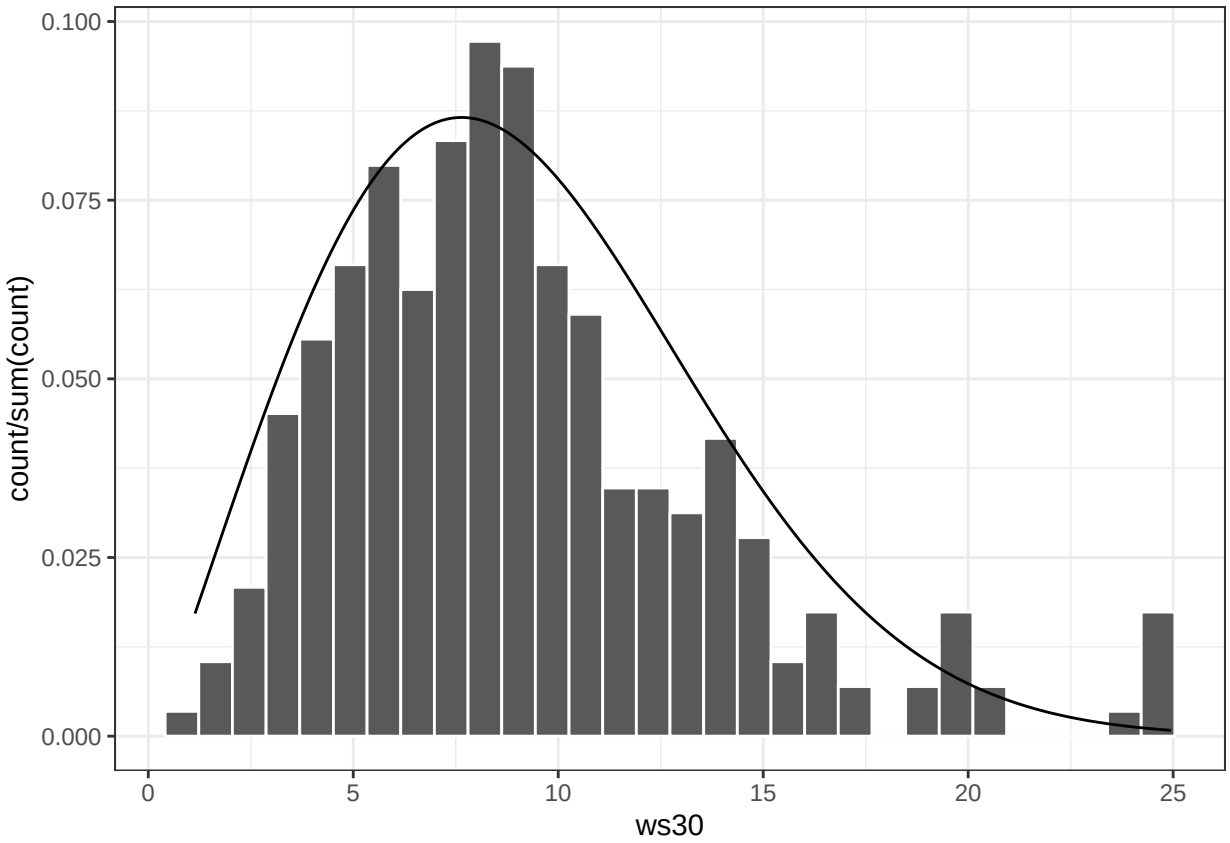
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#Remove simulated data from the data frame
D <- D %>%
  select(-simdata)
```

For wind speed distributions it is common practice to use the weibull distribution.

```
par.ws30 <- nlminb(start = c(1,1), objective = testDistribution
                   , x = D$ws30
                   , distribution = "weibull"
                   , lower = c(0,0))


ggplot(D)+
  geom_histogram(aes(x = ws30, y = ..count../sum(..count..))
                 , colour = "white"
                 , bins = 30)+
  theme_bw()+
  stat_function(fun = dweibull, n = dim(D)[1], args = list(shape = par.ws30$par[1], scale = par.ws30$pa
```



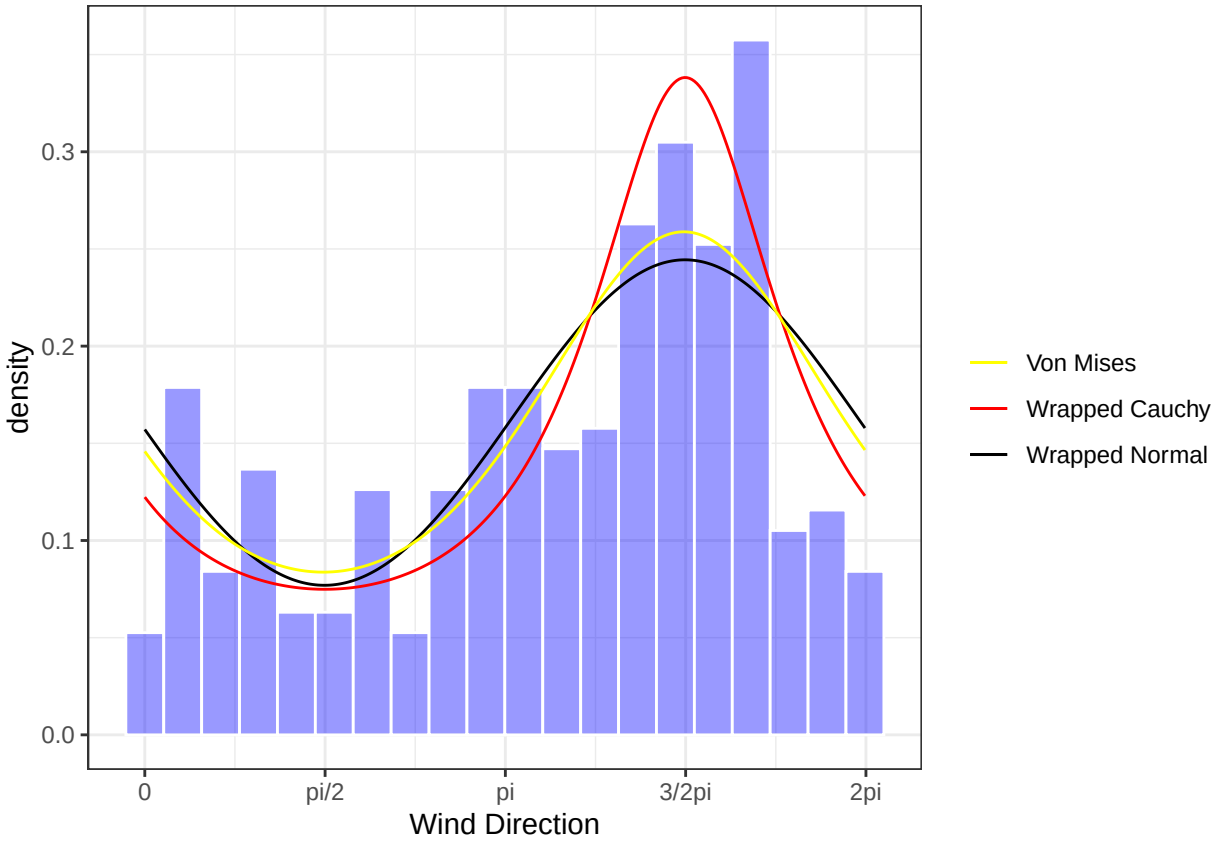
Wind direction are supplied as radians in the dataset, and thus it is appropriate to fit circular distributions to this variable. Here we examine a circular normal distribution, wrapped cauchy and a von Mises distribution.

```
library(circular)
```

```
##
## Vedhæfter pakke: 'circular'

## De følgende objekter er maskerede fra 'package:sn':
##
##     sd, sd.default

## De følgende objekter er maskerede fra 'package:stats':
##
##     sd, var
```

```
nll.wrappedNormal <- function(p,x){
  nll <- -sum(log(dwrappednormal(x, mu = circular(p[1]), rho = NULL, sd = p[2])))
  return(nll)
}

nll.wrappedCauchy <- function(p,x){
  nll <- -sum(log(dwrappedcauchy(x, mu = circular(p[1]), rho = p[2])))
  return(nll)
}
```

```r
nll.vonMises <- function(p,x){
  nll <- -sum(dvonmises(x, mu = circular(p[1]), kappa = p[2], log = T))
  return(nll)
}

wrapped.par <- nlminb(start = c(2,1), objective = nll.wrappedNormal, x = D$wd30)
wrapped.cauc.par <- nlminb(start = c(1,1/10000), lower = c(-Inf, 1/10000), upper = c(Inf, 1),
                           objective = nll.wrappedCauchy, x = D$wd30)
wrapped.vonMises <- nlminb(start = c(0,1), objective = nll.vonMises, x = D$wd30, lower = c(-1000, 0))

ggplot(D)+
  theme_bw()+
  #geom_density(aes(x = wd30.centered, y = ..density..), alpha = .8, colour = "white", fill = "red", co
  geom_histogram(aes(x = wd30, y = ..density..), colour = "white", alpha = .4, fill = "blue", bins = 20)
  scale_x_continuous(breaks = c(0,pi/2,pi,3/2*pi,2*pi)
                     , labels =c("0", "pi/2", "pi", "3/2pi", "2pi"))+
  #stat_function(fun = dnorm, n = dim(D)[1], args = list(mean = par.wd30$par[1], sd = par.wd30$par[2]))
  stat_function(fun = dwrappednormal, n = dim(D)[1], args = list(mu = wrapped.par$par[1], sd = wrapped.p
  stat_function(fun = dwrappedcauchy, n = dim(D)[1], args = list(mu = wrapped.cauc.par$par[1], rho = 0.3
  stat_function(fun = dvonmises, n = dim(D)[1], args = list(mu = wrapped.vonMises$par[1], kappa = wrappe
  labs(x = "Wind Direction", colour = "")+
  scale_colour_manual(values = c("yellow", "red", "black"))
```

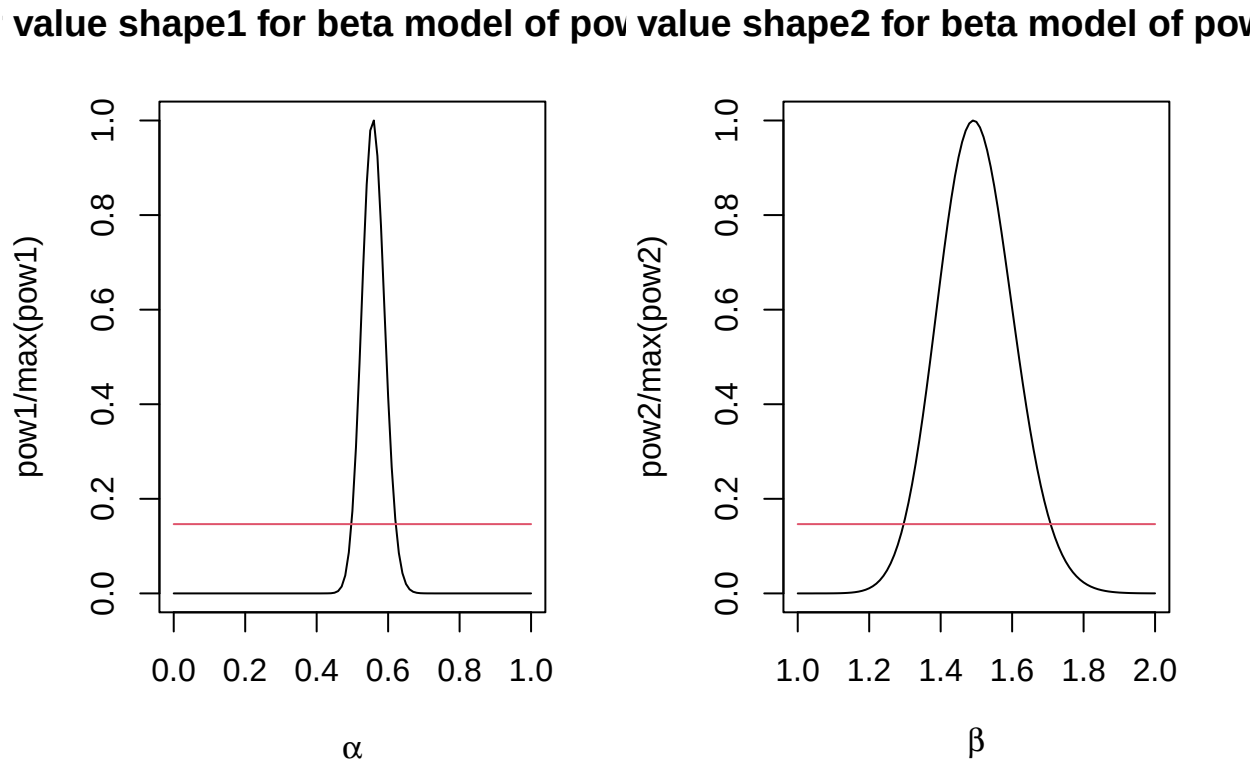


```r
#Calculate AICs
print(paste0("AIC wrapped normal: ", round(-2*log(wrapped.par$objective)+2*2,4), "|"
             ,"AIC wrapped cauchy: ", round(-2*log(wrapped.cauc.par$objective)+2,4), "|"
             ,"AIC von Mises: "     , round(-2*log(wrapped.vonMises$objective)+2*2,4)))
```

```
## [1] "AIC wrapped normal: -8.462|AIC wrapped cauchy: -10.4574|AIC von Mises: -8.4597"
```

```r
## CI ## WIND POWER
par(mfrow=c(1,1))
alpha <- 0.05
c <- exp(-0.5 * qchisq(1-alpha, df = 1))
#likelihood-based
mle.pow <- par.beta$par

pow.fun <- function(shape1, shape2, data){
  return( prod( dbeta(x = data, shape1 = shape1, shape2 = shape2, log = F) ) )
}

l.pow.fun <- function(shape1, shape2, data){
  return( sum( dbeta(x = data, shape1 = shape1, shape2 = shape2, log = T) ) )
}

CIfun.pow <- function(y, first = T){##### T for shape, F for scale
  if(first){
    return( sum( dbeta(x = D$pow.obs.norm, shape1 = mle.pow[1], shape = mle.pow[2], log = T) ) -
      sum( dbeta(x = D$pow.obs.norm, shape1 = y, shape2 = mle.pow[2], log = T) ) -
      0.5 * qchisq(1-alpha, df = 1) )
  } else {
    return( sum( dbeta(x = D$pow.obs.norm, shape1 = mle.pow[1], shape = mle.pow[2], log = T) ) -
      sum( dbeta(x = D$pow.obs.norm, shape1 = mle.pow[1], shape2 = y, log = T) ) -
      0.5 * qchisq(1-alpha, df = 1) )
  }
}

par(mfrow=c(1,2))
shape1s <- seq(0, 1, by = 0.01)
pow1 <- sapply(X = shape1s, FUN = pow.fun, data = D$pow.obs.norm, shape2 = mle.pow[2])
plot(shape1s, pow1/max(pow1), col = 1, type = "l", xlab = expression(paste(alpha)),
     main = "Parameter value shape1 for beta model of power production")
CI.pow1 <- c(uniroot(f = CIfun.pow, interval = c(0, mle.pow[1]), first = T)$root,
             uniroot(f = CIfun.pow, interval = c(mle.pow[1], 1), first = T)$root)
lines(range(shape1s), c*c(1,1), col = 2)

shape2s <- seq(1, 2, by = 0.01)
pow2 <- sapply(X = shape2s, FUN = pow.fun, data = D$pow.obs.norm, shape1 = mle.pow[1])
plot(shape2s, pow2/max(pow2), col = 1, type = "l", xlab = expression(paste(beta)),
     main = "Parameter value shape2 for beta model of power production")
CI.pow2 <- c(uniroot(f = CIfun.pow, interval = c(1, mle.pow[2]), first = F)$root,
             uniroot(f = CIfun.pow, interval = c(mle.pow[2], 2), first = F)$root)
lines(range(shape2s), c*c(1,1), col = 2)
```

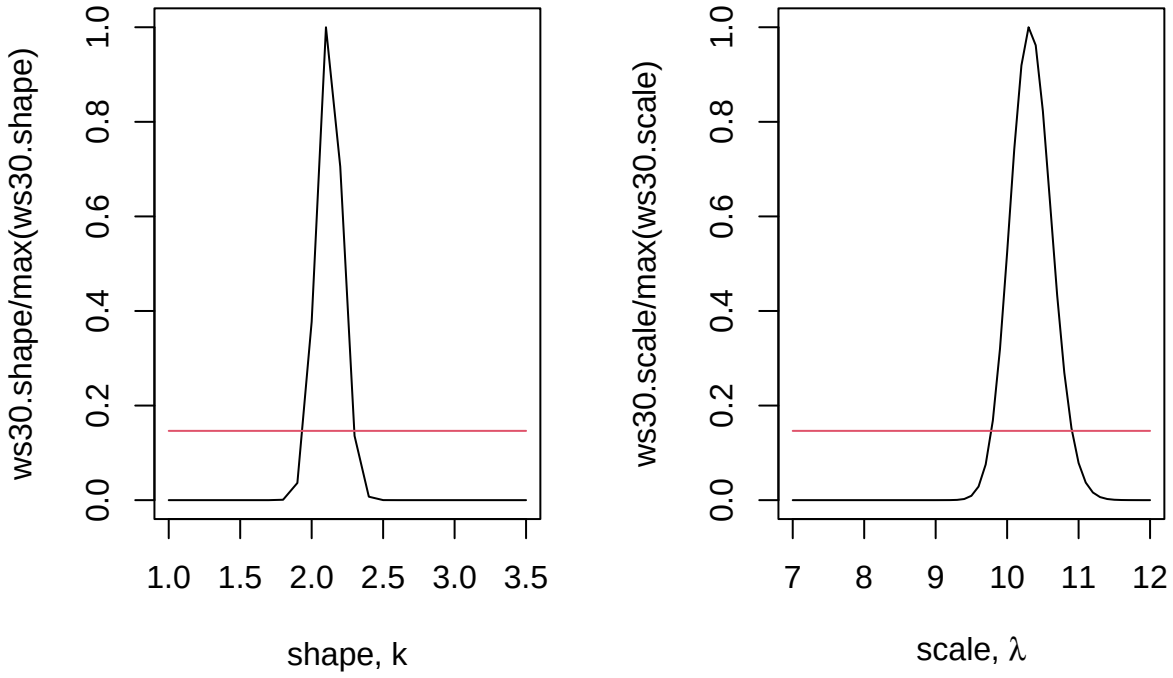
Conclude on the most appropriate model for each variable, also report parameters including assessment of their uncertainty. For models that does not include a transformation you should also give an assessment of the uncertainty of the expected value in the model.




```r
#wald
n <- dim(D)[1]
H.pow.shape1 <- hessian(l.pow.fun, mle.pow[1], shape2 = mle.pow[2], data = D$pow.obs.norm)
V.pow.shape1 <- as.numeric(-1/H.pow.shape1)
H.pow.shape2 <- hessian(l.pow.fun, mle.pow[2], shape1 = mle.pow[1], data = D$pow.obs.norm)
V.pow.shape2 <- as.numeric(-1/H.pow.shape2)
wald.pow.shape1 <- mle.pow[1] + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.pow.shape1)
wald.pow.shape2 <- mle.pow[2] + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.pow.shape2)

## CI ## WIND SPEED
par(mfrow=c(1,2))
#likelihood-based
mle.ws30.weib <- par.ws30$par

ws30.fun <- function(shape, scale, data){#####
  prod(dweibull(x = data, shape = shape, scale = scale, log = F)*2)#to not get full zeros
}

l.ws30.fun <- function(shape, scale, data){#####
  sum(dweibull(x = data, shape = shape, scale = scale, log = T))
}

CIfun.ws30 <- function(y, shape = T){##### T for shape, F for scale
```

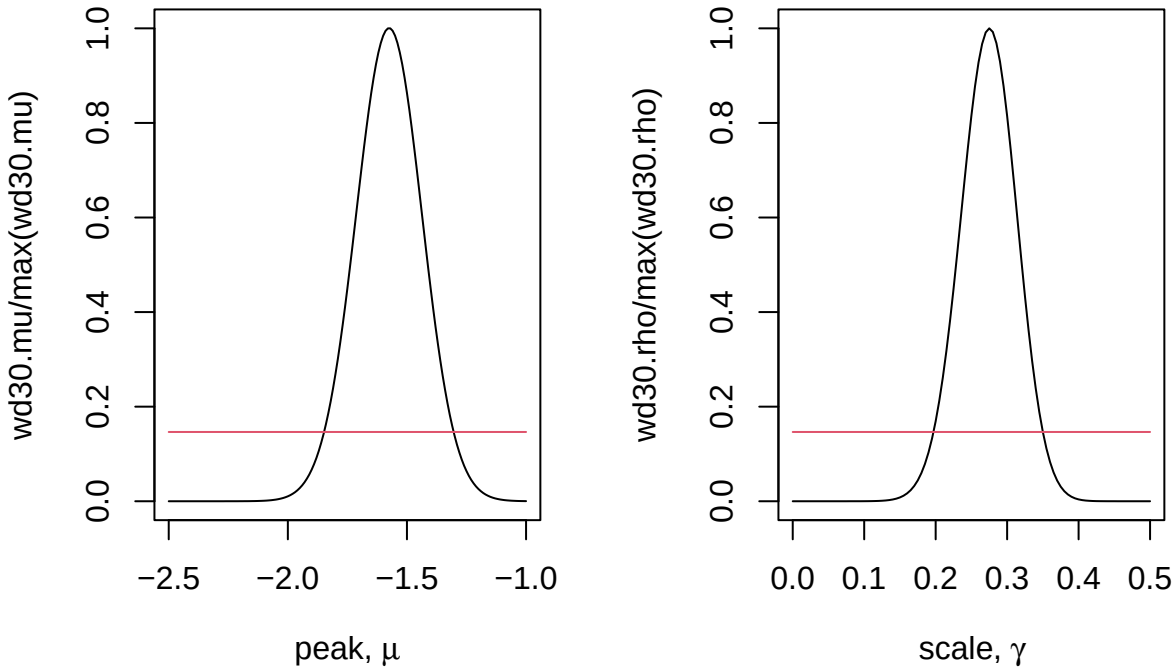
```
  if(shape){
    sum(dweibull(x = D$ws30, shape = mle.ws30.weib[1], scale = mle.ws30.weib[2], log = T)) -
      sum(dweibull(x = D$ws30, shape = y, scale = mle.ws30.weib[2], log = T)) -
      0.5 * qchisq(1-alpha, df = 1)
  } else {
    sum(dweibull(x = D$ws30, shape = mle.ws30.weib[1], scale = mle.ws30.weib[2], log = T)) -
      sum(dweibull(x = D$ws30, shape = mle.ws30.weib[1], scale = y, log = T)) -
      0.5 * qchisq(1-alpha, df = 1)
  }
}
shapes <- seq(1, 3.5, by = 0.1)
ws30.shape <- sapply(X = shapes, FUN = ws30.fun, scale = mle.ws30.weib[2], data = D$ws30)
plot(shapes, ws30.shape/max(ws30.shape), col = 1, type = "l", xlab = "shape, k",
     main = "Parameter value for shape for weibull model of wind speed")
CI.ws30.shape <- c(uniroot(f = CIfun.ws30, interval = c(1, mle.ws30.weib[1]), shape = T)$root,
                   uniroot(f = CIfun.ws30, interval = c(mle.ws30.weib[1], 3.5), shape = T)$root)
lines(range(shapes), c*c(1,1), col = 2)

scales <- seq(7, 12, by = 0.1)
ws30.scale <- sapply(X = scales, FUN = ws30.fun, shape = mle.ws30.weib[1], data = D$ws30)
plot(scales, ws30.scale/max(ws30.scale), col = 1, type = "l", xlab = expression(paste("scale, ", lambda
     main = "Parameter value for scale for weibull model of wind speed")
CI.ws30.scale <- c(uniroot(f = CIfun.ws30, interval = c(7, mle.ws30.weib[2]), shape = F)$root,
                   uniroot(f = CIfun.ws30, interval = c(mle.ws30.weib[2], 12), shape = F)$root)
lines(range(scales), c*c(1,1), col = 2)
```

r value for shape for weibull model r value for scale for weibull model c



```r
#wald
n <- dim(D)[1]
H.ws30.shape <- hessian(l.ws30.fun, mle.ws30.weib[1], scale = mle.ws30.weib[2], data = D$ws30)
V.ws30.shape <- as.numeric(-1/H.ws30.shape)
H.ws30.scale <- hessian(l.ws30.fun, mle.ws30.weib[2], shape = mle.ws30.weib[1], data = D$ws30)
V.ws30.scale <- as.numeric(-1/H.ws30.scale)
wald.ws30.shape <- mle.ws30.weib[1] + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.ws30.shape)
wald.ws30.scale <- mle.ws30.weib[2] + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.ws30.scale)


## CI ## WIND DIRECTION
par(mfrow=c(1,2))
#likelihood-based
mle.wd30 <- wrapped.cauc.par$par

wd30.fun <- function(mu, rho, data){#####
  prod(dwrappedcauchy(x = data, mu = mu, rho = rho))
}

l.wd30.fun <- function(mu, rho, data){#####
  sum( log( dwrappedcauchy(x = data, mu = mu, rho = rho) ) )
}

CIfun.wd30 <- function(y, mu = T){##### T from mean, F for sigma
  if(mu){
    return( sum( log( dwrappedcauchy(x = D$wd30, mu = mle.wd30[1], rho = mle.wd30[2]) ) ) -
      sum( log( dwrappedcauchy(x = D$wd30, mu = y, rho = mle.wd30[2]) ) ) -
      0.5 * qchisq(1-alpha, df = 1) )
  } else {
    return( sum( log( dwrappedcauchy(x = D$wd30, mu = mle.wd30[1], rho = mle.wd30[2]) ) ) -
      sum( log( dwrappedcauchy(x = D$wd30, mu = mle.wd30[1], rho = y) ) ) -
      0.5 * qchisq(1-alpha, df = 1) )
  }
}

mus <- seq(-2.5, -1, by = 0.01)
wd30.mu <- sapply(X = mus, FUN = wd30.fun, rho = mle.wd30[2], data = D$wd30)
plot(mus, wd30.mu/max(wd30.mu), col = 1, type = "l", xlab = expression(paste("peak, ", mu)),
     main = "Parameter value for peak for wrapped cauchy model of wind direction")
CI.wd30.mu <- c(uniroot(f = CIfun.wd30, interval = c(-2.5, mle.wd30[1]), mu = T)$root,
                uniroot(f = CIfun.wd30, interval = c(mle.wd30[1], -1), mu = T)$root)
lines(range(mus), c*c(1,1), col = 2)

rhos <- seq(0, 0.5, by = 0.005)
wd30.rho <- sapply(X = rhos, FUN = wd30.fun, mu = mle.wd30[1], data = D$wd30)
plot(rhos, wd30.rho/max(wd30.rho), col = 1, type = "l", xlab = expression(paste("scale, ", gamma)),
     main = "Parameter value for scale factor for wrapped cauchy model of wind direction")
CI.wd30.rho <- c(uniroot(f = CIfun.wd30, interval = c(0, mle.wd30[2]), mu = F)$root,
                 uniroot(f = CIfun.wd30, interval = c(mle.wd30[2], 0.5), mu = F)$root)
lines(range(rhos), c*c(1,1), col = 2)
```

e for peak for wrapped cauchy mod r scale factor for wrapped cauchy n




```
#wald
n <- dim(D)[1]
H.wd30.mu <- hessian(l.wd30.fun, mle.wd30[1], rho = mle.wd30[2], data = D$wd30)
V.wd30.mu <- as.numeric(-1/H.wd30.mu)
H.wd30.rho <- hessian(l.wd30.fun, mle.wd30[2], mu = mle.wd30[1], data = D$wd30)
V.wd30.rho <- as.numeric(-1/H.wd30.rho)
wald.wd30.mu <- mle.wd30[1] + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.wd30.mu)
wald.wd30.rho <- mle.wd30[2] + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.wd30.rho)

#All CIs of parameters
round( rbind( CI.pow1, wald.pow.shape1, CI.pow2, wald.pow.shape2, mle.pow,
              CI.ws30.shape, wald.ws30.shape, CI.ws30.scale, wald.ws30.scale, mle.ws30.weib,
              CI.wd30.mu, wald.wd30.mu, CI.wd30.rho, wald.wd30.rho, mle.wd30 ), digits = 3 )
```

```
##                   [,1]   [,2]
## CI.pow1          0.497  0.621
## wald.pow.shape1  0.495  0.619
## CI.pow2          1.296  1.708
## wald.pow.shape2  1.286  1.697
## mle.pow          0.557  1.492
## CI.ws30.shape    1.954  2.295
## wald.ws30.shape  1.952  2.294
## CI.ws30.scale    9.781 10.906
## wald.ws30.scale  9.756 10.879
## mle.ws30.weib    2.123 10.318
## CI.wd30.mu      -1.848 -1.304
```

```
## wald.wd30.mu    -1.845 -1.305
## CI.wd30.rho      0.197  0.350
## wald.wd30.rho    0.199  0.352
## mle.wd30        -1.575  0.275
```

```
#CI.E.pow.obs <- 1/par$par + c(-1,1) * qnorm(1-alpha/2) * sqrt(1/par$par^2) / dim(D)[1]
CI.E.pow.obs <- mean(D$pow.obs.norm) + c(-1,1) * qnorm(1-alpha/2) * sd(D$pow.obs.norm) / dim(D)[1]
#par.ws30$par[2]*gamma(1+1/par.ws30$par[1]) #mean = lambda * Gamma(1 + 1/k); lambda = scale, k = shape
E.ws30 <- par.ws30$par[2]*gamma(1+1/par.ws30$par[1])
V.ws30 <- par.ws30$par[2]^2*( gamma(1+2/par.ws30$par[1]) - (gamma(1+1/par.ws30$par[1]))^2)
#CI.E.ws30 <- E.ws30 + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.ws30) / dim(D)[1] #according to Central Limi
CI.E.ws30 <- mean(D$ws30) + c(-1,1) * qnorm(1-alpha/2) * sd(D$ws30) / dim(D)[1]
#CI.E.wd30 <- par.wd30$par[1] + c(-1,1) * qnorm(1-alpha/2) * par.wd30$par[2] / dim(D)[1] #according to
CI.E.wd30 <- mle.wd30[1] + c(-1,1) * qnorm(1-alpha/2) * sd(D$wd30) / dim(D)[1] #mean(D$wd30) instead gi

round(rbind(c(CI.E.pow.obs[1], 1/par.exp$par, CI.E.pow.obs[2]) , c(CI.E.ws30[1], E.ws30, CI.E.ws30[2])
```

```
##          [,1]     [,2]     [,3]
## [1,]  0.27450  0.27624  0.27798
## [2,]  9.08066  9.13771  9.14271
## [3,] -1.58663 -1.57486 -1.56309
```

Projekt 2: Survival Data

Analysis of the Binary Data

```
log.data <- read.table("Logistic.txt", header=TRUE, sep="",
                       as.is=TRUE)

str(log.data)
```

Read the data Logistic.txt into R.

```
## 'data.frame':    2 obs. of  3 variables:
##  $ AZT     : chr  "Yes" "No"
##  $ AIDS_yes: int  25 44
##  $ n       : int  170 168
```

Fit the Binomial distribution to the data (i.e. consider all data as coming from the same population)

```
#all data from one population:
bin.par <- nlminb(start = 0.1, objective = testDistribution
                  , x = c(sum(log.data$AIDS_yes), sum(log.data$n))
                  , distribution = "binomial")
```

```
#separately for the groups
x.AZT <- log.data %>%
  filter(AZT == "Yes") %>%
  select(AIDS_yes, n) %>%
  as.numeric()

AZT.par <- nlminb(start = 0.1, objective = testDistribution
                  , x = c(x.AZT[1], x.AZT[2])
                  , distribution = "binomial")

x.no.AZT <- log.data %>%
  filter(AZT == "No") %>%
  select(AIDS_yes, n) %>%
  as.numeric()

no.AZT.par <- nlminb(start = 0.1, objective = testDistribution
                     , x = c(x.no.AZT[1], x.no.AZT[2])
                     , distribution = "binomial")
```

Fit the Binomial separately to the two distributions and test if there is a difference between the groups. Testing if there's a difference between the two groups:

```
p.hat <- sum(log.data$AIDS_yes)/sum(log.data$n)#bin.par$par


#Calculate expected values for this group based on each group size:
e.A.AZT <- log.data$n[log.data$AZT == "Yes"]*p.hat
e.A.no_AZT <- log.data$n[log.data$AZT == "No"]*p.hat

e.nA.AZT <- log.data$n[log.data$AZT == "Yes"]*(1-p.hat)
e.nA.no_AZT <- log.data$n[log.data$AZT == "No"]*(1-p.hat)

e <- c(e.A.AZT, e.A.no_AZT, e.nA.AZT, e.nA.no_AZT)
#by hand
chi_squared <- sum((c(log.data$AIDS_yes,log.data$n-log.data$AIDS_yes)-e)^2/e)
(chi_squared)
```

```
## [1] 6.859695
```

```
#probability of observing this chi-squared test statistic given that the null-hypothesis is true
rows <- dim(log.data)[1]
columns <- dim(log.data)[2]-1 #-1 because of the AZT column
pchisq(chi_squared,df=(rows-1)*(columns-1),lower.tail=FALSE)
```

```
## [1] 0.008816159
```

```
#WITH CONTINUITY CORRECTION:
#https://en.wikipedia.org/wiki/Yates%27s_correction_for_continuity
chi_squared_yates <- sum((abs(c(log.data$AIDS_yes,log.data$n-log.data$AIDS_yes)-e)-0.5)^2/e)
(chi_squared_yates)
```

```
## [1] 6.171023
```

```
#probability of observing this chi-squared test statistic given that the null-hypothesis is true
rows <- dim(log.data)[1]
columns <- dim(log.data)[2]-1 #-1 because of the AZT column
pchisq(chi_squared_yates,df=(rows-1)*(columns-1),lower.tail=FALSE)
```

```
## [1] 0.01298595
```

```
#direct using R:
log.data.for.chi <- log.data; log.data.for.chi$f <- log.data.for.chi$n - log.data.for.chi$AIDS_yes
prop.test(log.data$AIDS_yes, log.data$n)
```

```
##
##  2-sample test for equality of proportions with continuity correction
##
## data:  log.data$AIDS_yes out of log.data$n
## X-squared = 6.171, df = 1, p-value = 0.01299
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.20593715 -0.02375473
## sample estimates:
##    prop 1    prop 2
## 0.1470588 0.2619048
```

```
#or
chisq.test(as.matrix(log.data.for.chi[,c(2,4)]))
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  as.matrix(log.data.for.chi[, c(2, 4)])
## X-squared = 6.171, df = 1, p-value = 0.01299
```

```
### Result: There's a difference between the two groups.
print(paste0("Mean p for group with AZT treatment: ", round(AZT.par$par,3)))
```

```
## [1] "Mean p for group with AZT treatment: 0.147"
```

```
print(paste0("Mean p for group with no AZT treatment: ", round(no.AZT.par$par,3)))
```

```
## [1] "Mean p for group with no AZT treatment: 0.262"
```

```
#Estimate parameters in the model and report a confidence interval for the parameter
#describing the difference, compare with the result above.
#p_0: Probability of aids in control group
#p_1: Probability of aids in treatment group
```

```r
#calculate likelihood
nll.p_0 <- function(beta, x = log.data$AIDS_yes[2], n = log.data$n[2]){
  p <- exp(beta)/(1+exp(beta))
  nll <- -sum(dbinom(x, size = n, prob = p, log = T))
  return(nll)
}
opt.p_0 <- nlminb(start = 1, objective = nll.p_0, x = log.data$AIDS_yes[2], n = log.data$n[2])
beta_0 <- opt.p_0$par

nll.p_1 <- function(beta_1, beta_0, x = log.data$AIDS_yes[1], n = log.data$n[1]){
  p <- exp(beta_0+beta_1)/(1+exp(beta_0+beta_1))
  nll <- -sum(dbinom(x, size = n, prob = p, log = T))
}
opt.p_1 <- nlminb(start = 1
                  , objective = nll.p_1
                  , beta_0 = beta_0
                  , x = log.data$AIDS_yes[1]
                  , n = log.data$n[1])
beta_1 <- opt.p_1$par

(p_0 <- exp(beta_0)/(1 + exp(beta_0)))
```

Estimate parameters in the model (p0 probability of AIDS in control group, p1 probability of AIDS in treatment group) and report a confidence interval for the parameter describing the difference, compare with the result above.

```
## [1] 0.2619047
```

```r
(p_1 <- exp(beta_0 + beta_1) / (1 + exp(beta_0 + beta_1)))
```

```
## [1] 0.1470588
```

```r
log.data
```

```
##   AZT AIDS_yes   n
## 1 Yes       25 170
## 2  No       44 168
```

```r
logistic <- data.frame("AZT" = c(rep(1,170), rep(0,168))
                       ,"AIDS_yes" = c(rep(c(1,0),c(25,170-25)), rep(c(1,0), c(44, 168-44))))

fit.glm <- glm(AIDS_yes ~ AZT, data = logistic, family = binomial)
print(paste0("with glm model: ", coef(fit.glm)))
```

```
## [1] "with glm model: -1.03609193168383"  "with glm model: -0.721765985868547"
```

```r
print(paste0("By hand (according to slide 19 lect 4): "))
```

```
## [1] "By hand (according to slide 19 lect 4): "
```

```r
print(paste0("beta_0 = ", beta_0, ", beta_1 = ", beta_1))
```

```
## [1] "beta_0 = -1.03609206621491, beta_1 = -0.721765851664904"
```

```r
summary(fit.glm)
```

```
##
## Call:
## glm(formula = AIDS_yes ~ AZT, family = binomial, data = logistic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7793  -0.7793  -0.5640  -0.5640   1.9580
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.0361     0.1755  -5.904 3.54e-09 ***
## AZT          -0.7218     0.2787  -2.590  0.00961 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 342.12  on 337  degrees of freedom
## Residual deviance: 335.19  on 336  degrees of freedom
## AIC: 339.19
##
## Number of Fisher Scoring iterations: 4
```

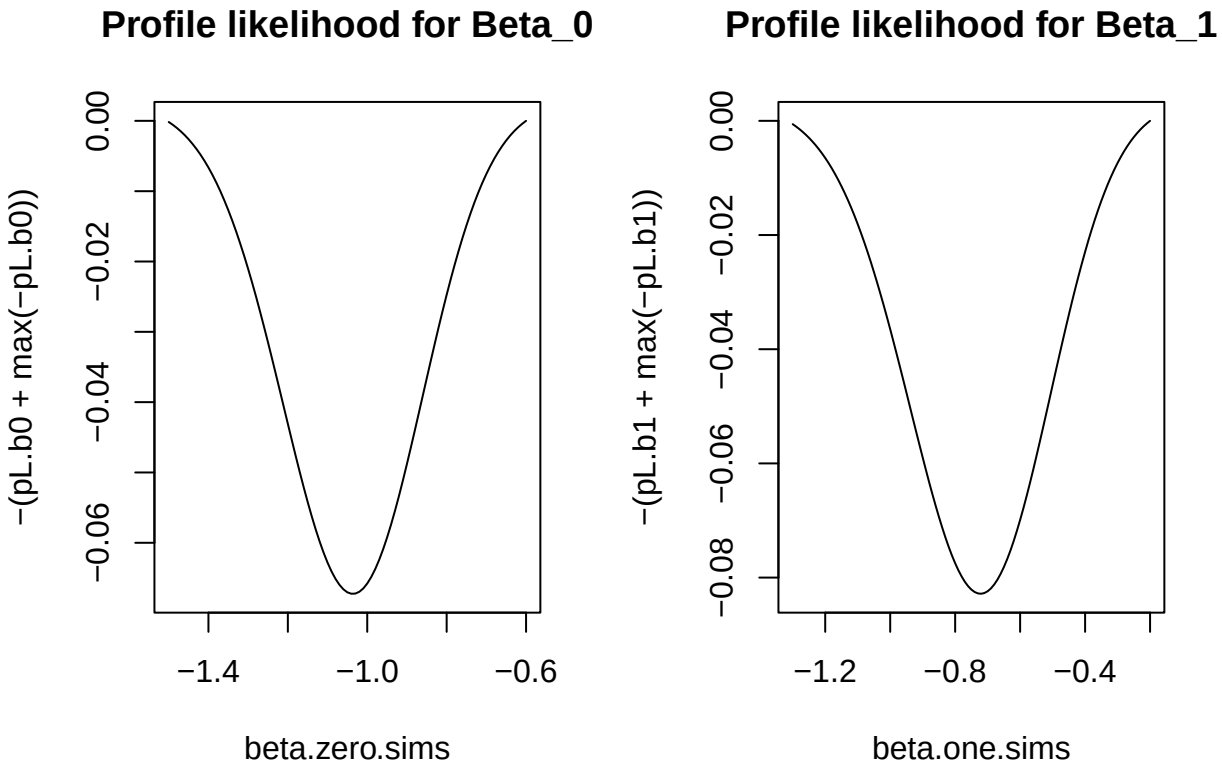
```r
#results show: -0.72 logits(?) for developing AIDS when using the treatment

# Confidence interval for the two beta parameters.
confint(fit.glm)
```

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

```
##                 2.5 %     97.5 %
## (Intercept) -1.390358 -0.7006773
## AZT         -1.279159 -0.1827049
```

```r
#calculate profile likelihoods
prof.b0 <- function(beta0, x = log.data$AIDS_yes[2], n = log.data$n[2]){
  p <- exp(beta0)/(1+exp(beta0))
  return(sum(dbinom(x, size = n, prob = p, log = T)))
}

prof.b1 <- function(beta1, beta0, x = log.data$AIDS_yes[1], n = log.data$n[1]){
  p <- exp(beta0+beta1)/(1+exp(beta0+beta1))
  return(sum(dbinom(x, size = n, prob = p, log = T)))
}
beta.zero.sims <- seq(-1.5,-0.6,0.01)
```

```r
beta.one.sims <- seq(-1.3,-0.2,0.01)
pL.b0 <- exp(sapply(beta.zero.sims, FUN = prof.b0))
pL.b1 <- exp(sapply(beta.one.sims, FUN = prof.b1, beta0 = beta_0))
par(mfrow=c(1,2))
plot(beta.zero.sims
     , -(pL.b0+max(-pL.b0))
     , "l"
     ,main = "Profile likelihood for Beta_0")
abline(h = -qchisq(0.95, df = 1)/2, lty = "dashed")
plot(beta.one.sims
     , -(pL.b1+max(-pL.b1))
     , "l"
     ,main = "Profile likelihood for Beta_1")
abline(h = -qchisq(0.95, df = 1)/2, lty = "dashed")
```




```r
#From these figures it can be concluded that the quadratic approximation
#of the CI through use of fischers information matrix, is a
#good approksimation.
#redefine because x is used

sd_0 <- as.numeric(sqrt(solve(hessian(beta_0, func = nll.p_0))))
sd_1 <- as.numeric(sqrt(solve(hessian(beta_1, func = nll.p_1, beta_0 = beta_0))))

#Wald 95% CIs and profile-likelihoods with approx 95% CI
(W.CI.0 <- beta_0 + c(-1,1)*qnorm(0.975)*sd_0)
```

```
## [1] -1.3800185 -0.6921656
```

```
(W.CI.1 <- beta_1 + c(-1,1)*qnorm(0.975)*sd_1)
```

```
## [1] -1.1462080 -0.2973237
```

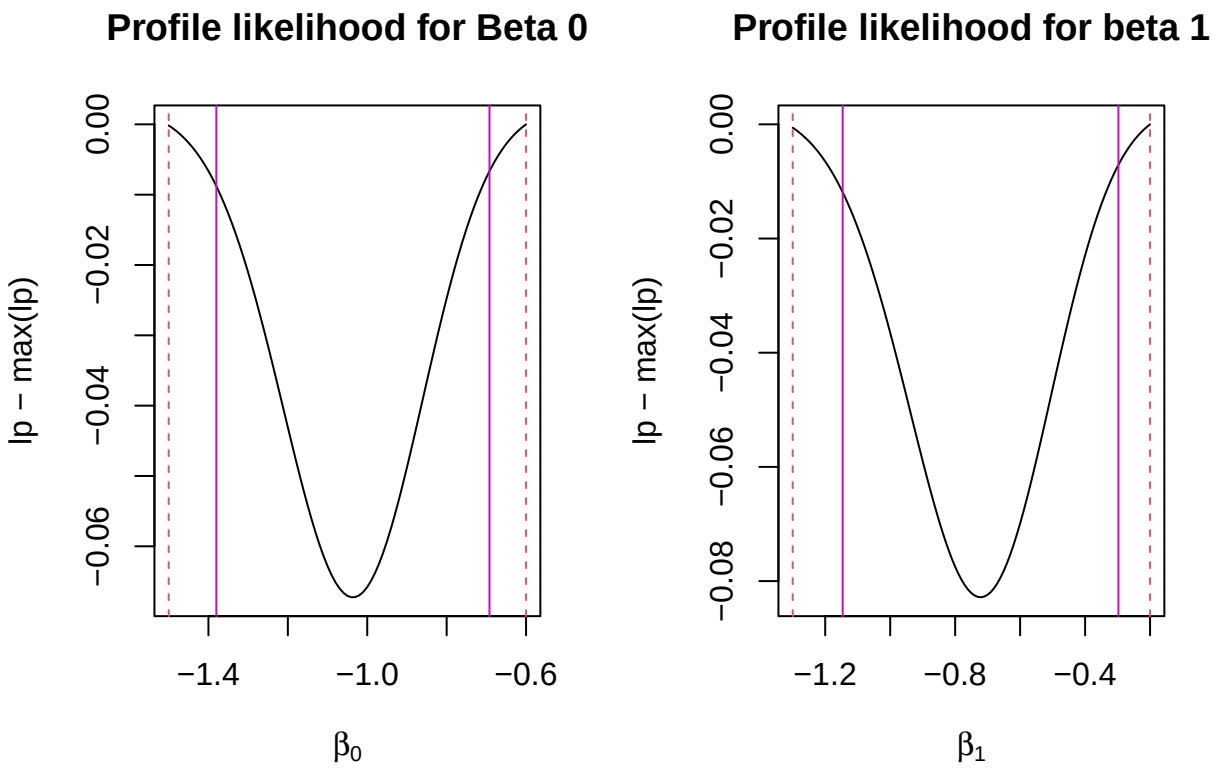
```
#Direkte numerisk approksimation:
(CI.0 <- c(min(beta.zero.sims[-(pL.b0+max(-pL.b0)) > -qchisq(0.95, df = 1)/2])
           ,max(beta.zero.sims[-(pL.b0+max(-pL.b0)) > -qchisq(0.95, df = 1)/2])))
```

```
## [1] -1.5 -0.6
```

```
(CI.1 <- c(min(beta.one.sims[-(pL.b1+max(-pL.b1)) > -qchisq(0.95, df = 1)/2])
           ,max(beta.one.sims[-(pL.b1+max(-pL.b1)) > -qchisq(0.95, df = 1)/2])))
```

```
## [1] -1.3 -0.2
```

```
plot(beta.zero.sims
     , -(pL.b0+max(-pL.b0))
     , "l"
     ,main = "Profile likelihood for Beta 0"
     ,xlab = expression(beta[0])
     ,ylab = "lp - max(lp)")
abline(h = -qchisq(0.95, df = 1)/2, col = 2)
abline(v = c(W.CI.0), col = 6)
text(x = W.CI.0[1]+0.2, y = -3, "Wald CI", col = 6)
text(x = CI.0[1]+0.1, y = -2.5, "CI", col = 2)
abline(v = c(CI.0), lty = "dashed", col = 2)
plot(beta.one.sims
     , -(pL.b1+max(-pL.b1))
     , "l"
     ,main = "Profile likelihood for beta 1"
     ,xlab = expression(beta[1])
     ,ylab = "lp - max(lp)")
abline(h = -qchisq(0.95, df = 1)/2, col = 2)
abline(v = c(W.CI.1), col = 6)
text(x = W.CI.1[1]+0.2, y = -3, "Wald CI", col = 6)
text(x = CI.1[1]+0.1, y = -2.5, "CI", col = 2)
abline(v = c(CI.1),lty = "dashed", col = 2)
```



Analysis of the Survival Time Data

```
#tx: Treatment indicator. 1 = New treatment, 0 = Control treatment
#event: Indicator for AIDS or death. 1 = AIDS diagnosis or death, 0 = Otherwise
#time: Time to AIDS diagnosis or death. Days
#så tiden for event = 0 må angive at personen har været med i studiet time[X] dage uden at være enten d
actg320 <- read.table("actg320.txt", header=TRUE, sep="",
                      as.is=TRUE)

#select time, event and tx as they are the only relevant variables in this project
actg <- actg320 %>%
  select(time, event, tx)
```

Read the data actg320.txt into R. If you are using RStudio you can use the "Import Dataset" button.

```
actg %>%
  group_by(tx) %>%
  summarise("Got AIDS or DIED" = sum(event),
            "Proportion" = sum(event)/n(),
            "Participants Given the Treatment" = n())
```

How many patients got AIDS or died in the two treatment groups? What is the proportion of patients that got AIDS or died in the two group? Other relevant number that could be calculated?

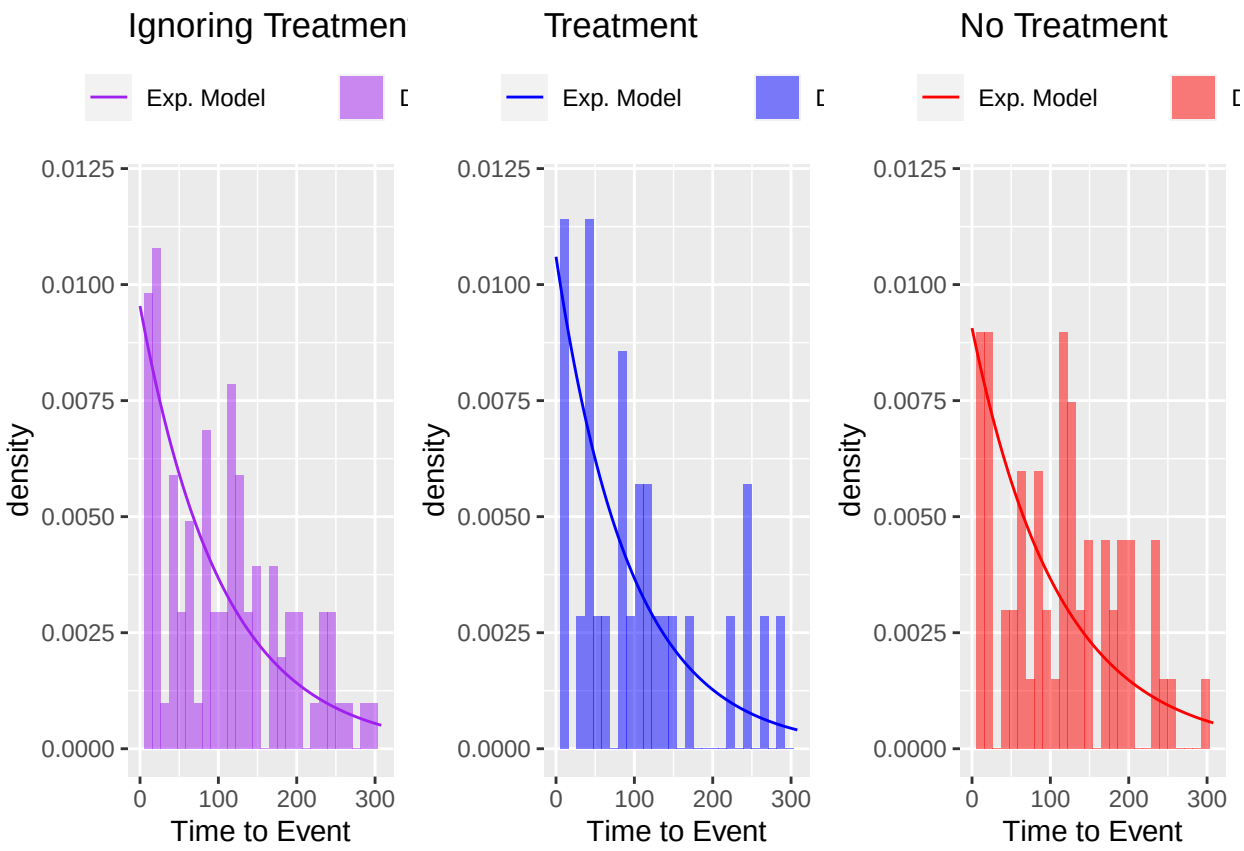
```
## # A tibble: 2 x 4
##      tx `Got AIDS or DIED` Proportion `Participants Given the Treatment`
##   <int>              <int>      <dbl>                              <int>
## 1     0                 63     0.109                                577
## 2     1                 33     0.0575                               574
```

```
#Fitting an exponential model to time for both and for each treatment
#only use times for event = 1, to filter out all the time of event indices with are longer than the rep
#given the fact that the participants in the event = 0 group, has not 'experienced' the event yet.
#Ved sgu ikke om ovenstående er en passende antagelse....

actg_event <- actg %>%
  filter(event == 1)

both <- nlminb(start = 2
               , objective = testDistribution
               , x = actg_event$time
               , distribution = "exponential")

#separate exponential models
t1 <- nlminb(start = 2
             , objective = testDistribution
             , x = filter(actg_event, tx == 1)$time
             , distribution = "exponential")

t0 <- nlminb(start = 2
             , objective = testDistribution
             , x = filter(actg_event, tx == 0)$time
             , distribution = "exponential")
#Potato plots:
p.both <- ggplot(actg_event)+
  geom_histogram(aes(x = time, y = ..density.., fill = "Data"), alpha = 0.5)+
  stat_function(aes(colour = "Exp. Model"), fun = dexp, n = dim(actg_event)[1], args = list(rate = both$
  ggtitle("Ignoring Treatment Effect")+
  theme(legend.position = "top")+
  lims(x = c(0,max(actg_event$time)+10), y = c(0,0.012))+
  labs(fill = "", colour = "", x = "Time to Event")+
  scale_colour_manual(values = "purple")+
  scale_fill_manual(values = "purple")

p.t1 <- ggplot(actg_event[actg_event$tx == 1,])+
  geom_histogram(aes(x = time, y = ..density.., fill = "Data"), alpha = 0.5)+
  stat_function(aes(colour = "Exp. Model"), fun = dexp, n = dim(actg_event)[1], args = list(rate = t1$pa
  ggtitle("Treatment")+
  theme(legend.position = "top")+
  lims(x = c(0,max(actg_event$time)+10), y = c(0,0.012))+
  labs(fill = "", colour = "", x = "Time to Event")+
```

```
  scale_colour_manual(values = "blue")+
  scale_fill_manual(values = "blue")

p.t2 <- ggplot(actg_event[actg_event$tx == 0,])+
  geom_histogram(aes(x = time, y = ..density.., fill = "Data"), alpha = 0.5)+
  stat_function(aes(colour = "Exp. Model"), fun = dexp, n = dim(actg_event)[1], args = list(rate = t0$pa
  ggtitle("No Treatment")+
  theme(legend.position = "top")+
  lims(x = c(0,max(actg_event$time)+10), y = c(0,0.012))+
  scale_colour_manual(values = "red")+
  labs(fill = "", colour = "", x = "Time to Event")+
  scale_fill_manual(values = "red")

grid.arrange(p.both, p.t1, p.t2, nrow = 1)
```

Fit an exponential distribution, using numerical methods, to the time of event (time) in the data set, remember to take into account that some of the data is censored (i.e. we only know that the time to the event is longer that the reported time). 1: Using all data (i.e. ignore the treatment effect) 2: Separately for the two treatments

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

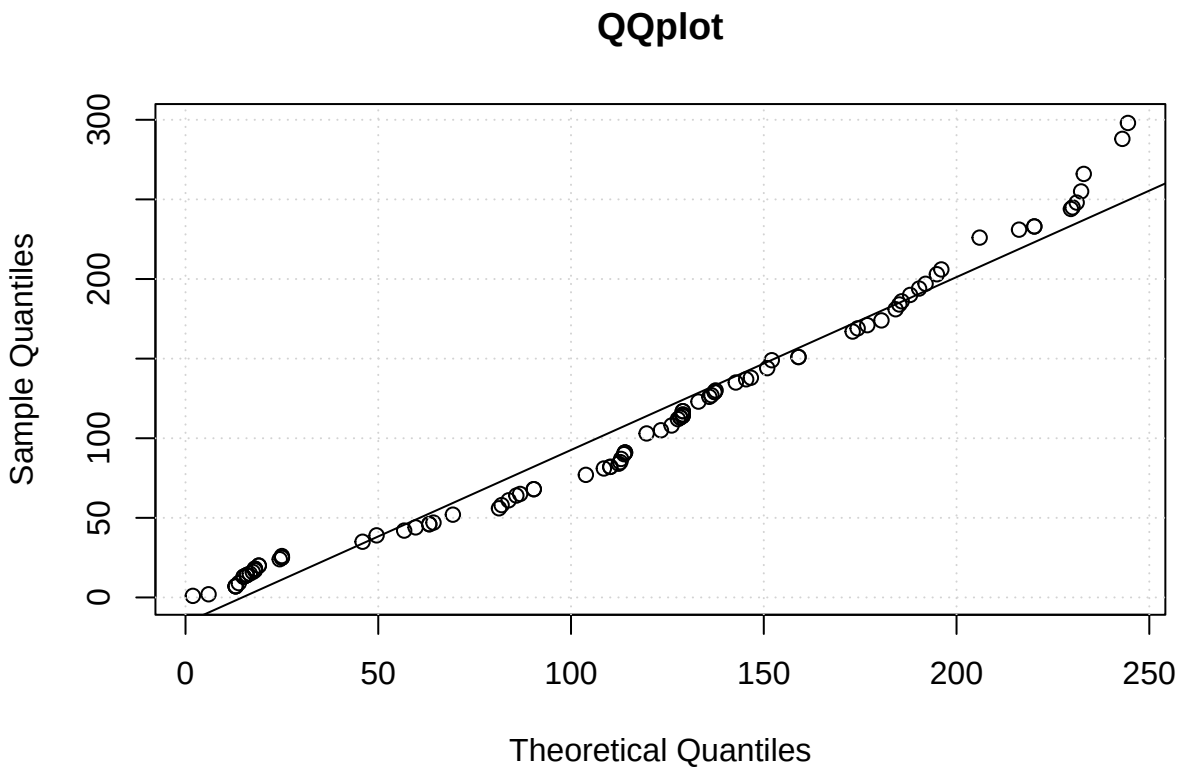


```r
#Likelihood Ratio Test (LRT) comparison
#one model:
chi_squared <- - 2 * ((t1$objective + t0$objective) - both$objective)
(p_value <- 1 - pchisq(chi_squared, df = 1))
```

Compared the likelihood for the above models and conclude

```
## [1] 0.4697636
```

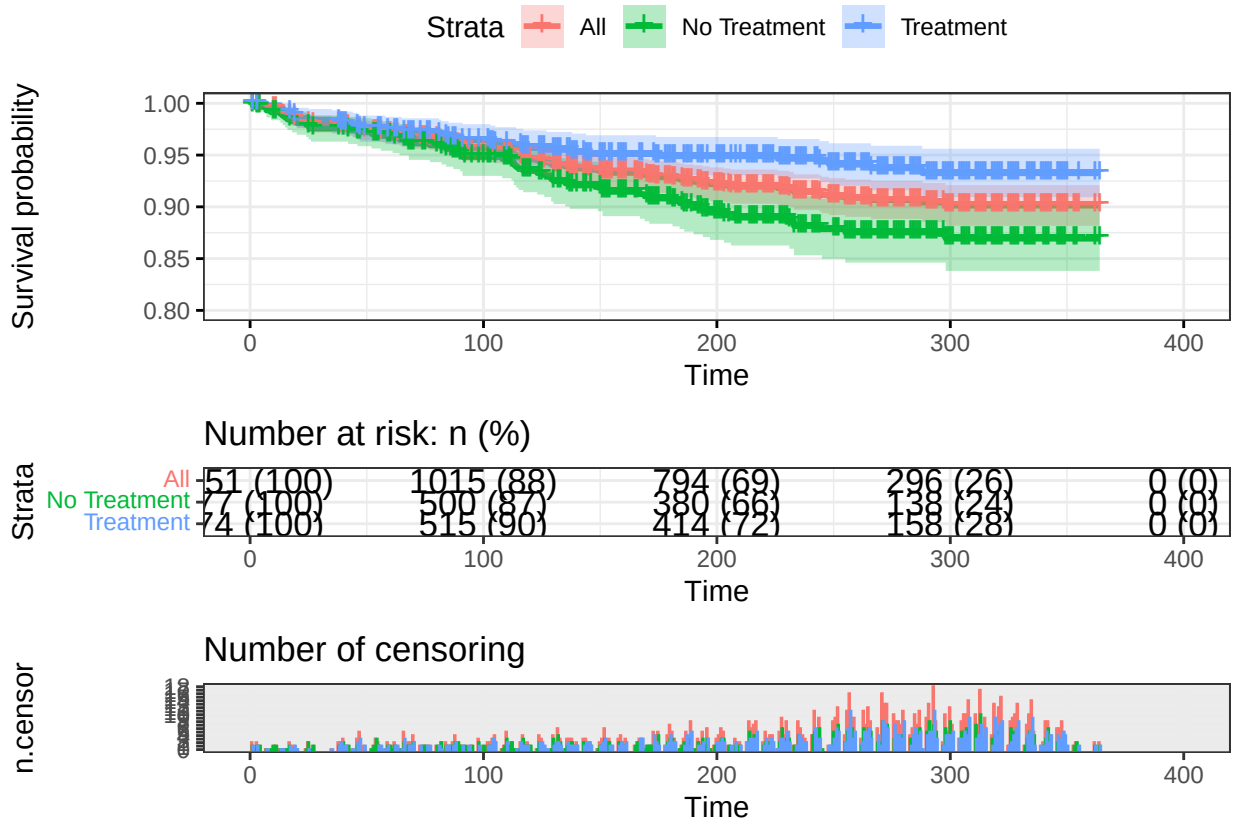
```r
#no difference as p_value: 0.46 > 0.05.

theoretical <- quantile(x = actg_event$time, probs = pexp(q = actg_event$time, rate = both$par))
plot(theoretical, actg_event$time, main = "QQplot", xlab = "Theoretical Quantiles"
     ,ylab = "Sample Quantiles")
grid()
abline(lm(actg_event$time ~ theoretical))
```



Formulate a model where one parameter indicate the treatment effect, find the MLE and compare with the result above. (e.g. $\mathrm{E}[\mathrm{T}] = e_0^{\beta}$ if control group and $\mathrm{E}[\mathrm{T}] = e^{\beta_0+\beta_1}$ if treatment group)

```r
kaplan.meier <- survfit(Surv(time, event) ~ tx, data = actg)
ggsurvplot_add_all(kaplan.meier
```

```
                    , data = actg
                    , conf.int = T
                    , risk.table = "abs_pct"
                    , ylim = c(0.8,1)
                    , pval = T
                    , ncensor.plot = T
                    ,ggtheme = theme_bw()
                    ,legend.labs = c("All", "No Treatment", "Treatment"))
```




```
fit <- survreg(Surv(time, event) ~ tx, data = actg,
               dist = "exponential")
summary(fit)
```

```
##
## Call:
## survreg(formula = Surv(time, event) ~ tx, data = actg, dist = "exponential")
##             Value Std. Error     z       p
## (Intercept) 7.624      0.126 60.52 <2e-16
## tx          0.699      0.215  3.25 0.0011
##
## Scale fixed at 1
##
## Exponential distribution
## Loglik(model)= -851   Loglik(intercept only)= -856.6
##  Chisq= 11.18 on 1 degrees of freedom, p= 0.00083
```

```
## Number of Newton-Raphson Iterations: 6
## n= 1151
```

```
confint(fit)
```

```
##                 2.5 %   97.5 %
## (Intercept) 7.3774309 7.871295
## tx          0.2780026 1.120341
```

```
#Overvej residual plot

#Ifølge ovenstående:
#beta0 = 7.62 95% CI [7.38; 7.87]
#beta1 = 0.699 85% CI [0.28; 1.12]
# => Significant difference.

#ifølge ovenstående er der statistisk signifikant forskel. Herunder regnes i hånden i stedet, så
#vi ved hvad der foregår.

#I hånden (jvf. slides fra uge 7):
#model: T = exp(B0 + B1*tx)*epsilon, epsilon ~ exp(1)
#Der kan opstilles to forskellige modeller afhængigt af tx = 0 eller tx = 1.
#tx = 0: E[T] = exp(b0)*epsilon
#tx = 1: E[T] = exp(b0 + b1)*epsilon

#Likelihood
nll.exp <- function(beta, time = actg$time, event = actg$event, treatment = actg$tx){
  beta0 <- beta[1]
  #dont want to make two functions so let beta1 = 0 if no treatment is not considered/used:
  if (max(treatment) == 0){
    beta1 <- 0
  } else {
    beta1 <- beta[2]
  }

  h <- exp(- beta0 - beta1 * treatment)
  H <- time/exp(beta0 + beta1*treatment)
  nll <- -sum(event*log(h) - H)
  return(nll)
}

beta_hat <- nlminb(start = c(1,1)
                   , objective = nll.exp
                   , time = actg$time
                   , event = actg$event
                   , treatment = actg$tx)
beta_hat$par
```

```
## [1] 7.6243647 0.6991732
```

```
#Comparing likelihoods with the result from bullet-point 4
beta_hat$objective #Ved ikke lige om der skal sammenlignes med de to modeller eller den ene? ahh
```

```
## [1] 851.0115
```

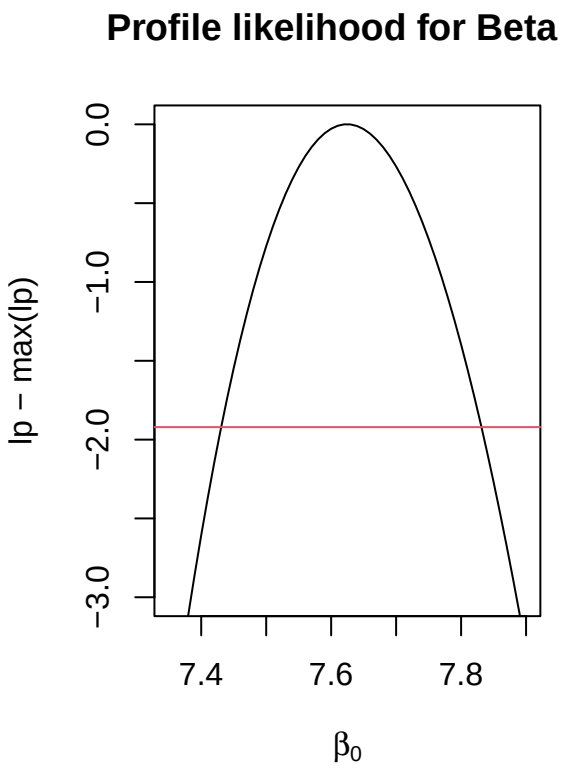
```
#måske skal man undersøge om begge værdier er statistisk signifikante således at vi kan argumentere for
#at der er tale om at behandlingen virker og sammenligne dette resultat med bullet-point 4.

#Calculate LRT
#optimise model without beta1 (no treatment):
beta_no_treatment_effect <- nlminb(start = 1
                                   , objective = nll.exp
                                   , time = actg$time
                                   , event = actg$event
                                   , treatment = rep(0, length(actg$tx)))

beta_no_treatment_effect$par
```

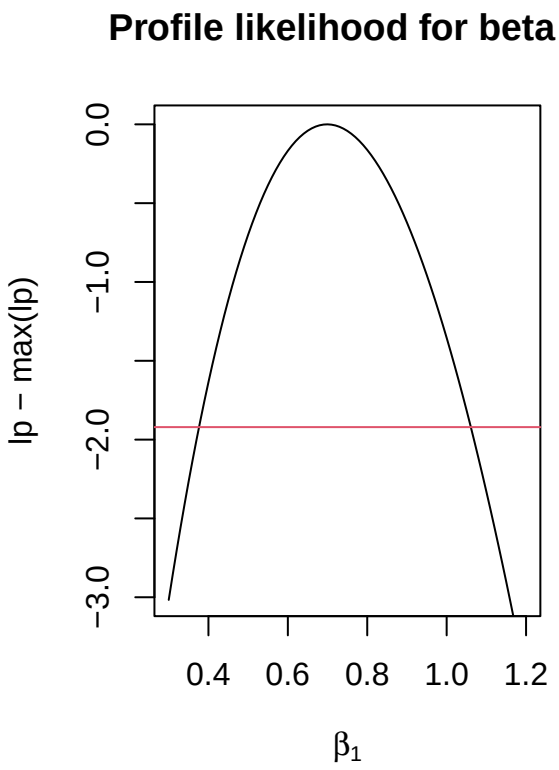
```
## [1] 7.922914
```

```
#LRT:
chi_squared <- - 2 * (beta_hat$objective - beta_no_treatment_effect$objective)
(p_value <- 1 - pchisq(chi_squared, df = 1))
```

```
## [1] 0.0008284118
```

```
#Here, we see that the treatment effect is statistically significant.

#### Bullet point 6 - Wald CI for the treatment parameters beta0 and beta1 ####
#Calculate profile likelihoods to ensure that the quadratic approximation by using Fischers Information
#is acceptable.

beta.zero.sims <- seq(7.35,7.9,0.01)
beta.one.sims <- seq(0.3,1.2,0.01)
pL.beta0 <- apply(X = data.frame(beta.zero.sims,beta_hat$par[2]), MARGIN = 1 , FUN = nll.exp, time = act
pL.beta1 <- apply(X = data.frame(beta_hat$par[1],beta.one.sims), MARGIN = 1 , FUN = nll.exp, time = actg

par(mfrow=c(1,2))
plot(beta.zero.sims
     , -(pL.beta0+max(-pL.beta0))
     , "l"
     ,main = "Profile likelihood for Beta 0"
     ,xlab = expression(beta[0])
     ,ylab = "lp - max(lp)"
     ,ylim = c(-3,0))
abline(h = -qchisq(0.95, df = 1)/2, col = 2)
plot(beta.one.sims
     , -(pL.beta1+max(-pL.beta1))
     , "l"
     ,main = "Profile likelihood for beta 1"
     ,xlab = expression(beta[1])
     ,ylab = "lp - max(lp)"
     ,ylim = c(-3,0))
abline(h = -qchisq(0.95, df = 1)/2, col = 2)
```






```
#CI:
sd <- as.numeric(sqrt(diag(solve(hessian(beta_hat$par, func = nll.exp)))))

#Wald 95 procent CIs and profile-likelihoods with approx 95 procent CI
#Måske er der et eller andet i vejen med de her WALD CIs
(Wald.CI <- beta_hat$par + matrix(c(-1,1), 2,2, byrow = T) * matrix(qnorm(0.975)*sd, 2,2, byrow = F))
```

Find the Wald confidence interval for the treatment parameter in the model above.

```
##           [,1]     [,2]
## [1,] 7.3774323 7.871297
## [2,] 0.2780036 1.120343
```

```
#Direkte numerisk approksimation:
(CI.0 <- c(min(beta.zero.sims[-(pL.beta0+max(-pL.beta0)) > -qchisq(0.95, df = 1)/2])
           ,max(beta.zero.sims[-(pL.beta0+max(-pL.beta0)) > -qchisq(0.95, df = 1)/2])))
```

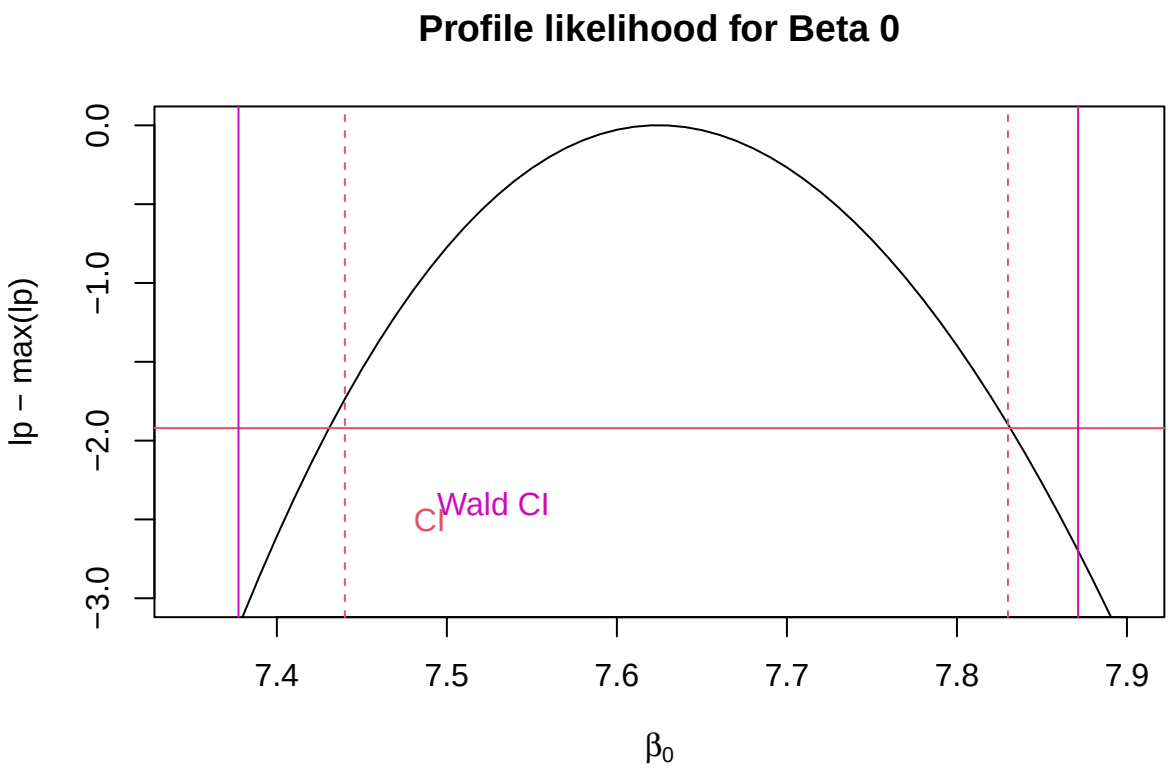
```
## [1] 7.44 7.83
```

```
(CI.1 <- c(min(beta.one.sims[-(pL.beta1+max(-pL.beta1)) > -qchisq(0.95, df = 1)/2])
           ,max(beta.one.sims[-(pL.beta1+max(-pL.beta1)) > -qchisq(0.95, df = 1)/2])))
```

```
## [1] 0.38 1.06
```

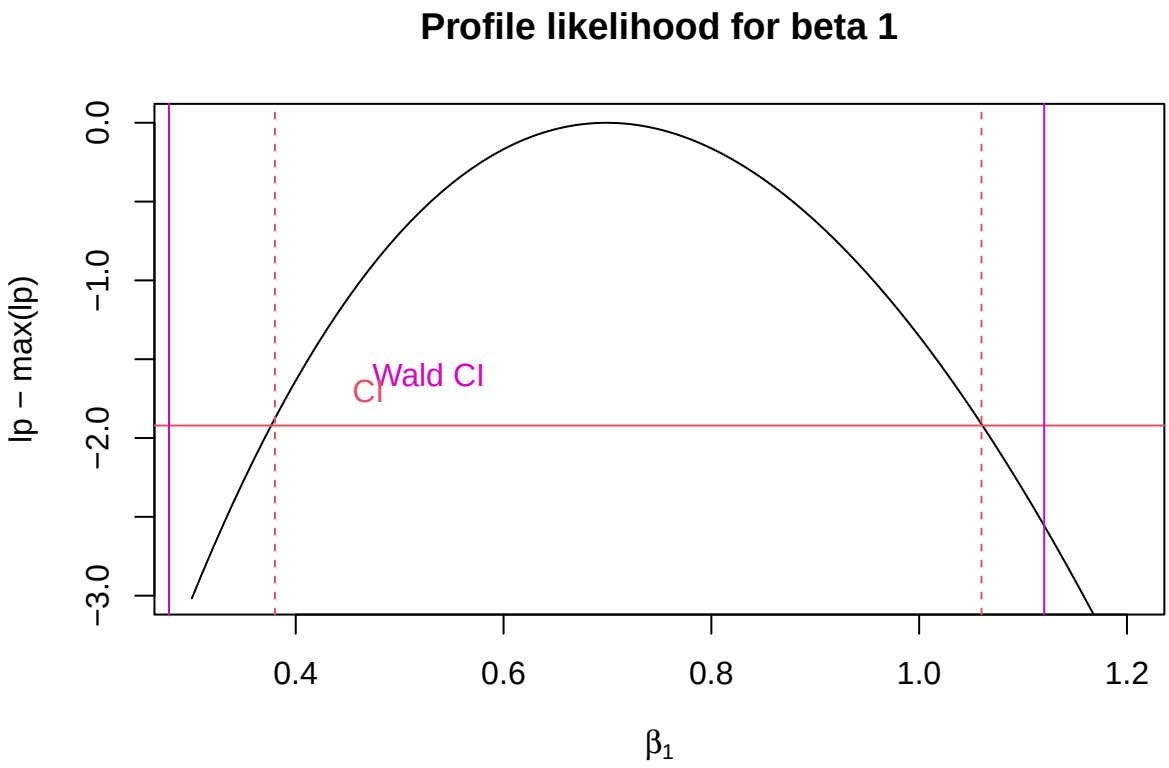
```
plot(beta.zero.sims
     , -(pL.beta0+max(-pL.beta0))
     , "l"
     ,main = "Profile likelihood for Beta 0"
     ,xlab = expression(beta[0])
     ,ylab = "lp - max(lp)"
     ,ylim = c(-3,0))
abline(h = -qchisq(0.95, df = 1)/2, col = 2)
abline(v = Wald.CI[1,], col = 6)
text(x = Wald.CI[1,1]+.15, y = -2.4, "Wald CI", col = 6)
text(x = CI.0[1]+.05, y = -2.5, "CI", col = 2)
abline(v = c(CI.0), lty = "dashed", col = 2)
```

Derive the theoretical results for the models above, including the standard error estimates, use this to formulate and implement the profile likelihood function for the treatment parameter




```
plot(beta.one.sims
     , -(pL.beta1+max(-pL.beta1))
     , "l"
     ,main = "Profile likelihood for beta 1"
     ,xlab = expression(beta[1])
     ,ylab = "lp - max(lp)"
     ,ylim = c(-3,0))
abline(h = -qchisq(0.95, df = 1)/2, col = 2)
```

```
abline(v = Wald.CI[2,], col = 6)
text(x = Wald.CI[2,1]+0.25, y = -1.6, "Wald CI", col = 6)
text(x = CI.1[1]+0.09, y = -1.7, "CI", col = 2)
abline(v = c(CI.1),lty = "dashed", col = 2)
```



(Have not included our analysis based on the weibull distribution)

Projekt 3: Financial Data

Descriptive Statistics and Simple Models

```
D <- read.table("finance_data.csv", header=TRUE, sep=";",
                as.is=TRUE)
## Dimensions of D (number of rows and columns)
dim(D)
```

Present the data, estimate the parameters in a normal model, and asses if the normal model is appropriate.

```
## [1] 454   2
```

```
##  Column/variable names
names(D)
```

```
## [1] "time" "SLV"
```

```
## The first rows/observations
head(D)
```

```
##        time          SLV
## 1  2006-5-5  0.013758146
## 2 2006-5-12  0.032857143
## 3 2006-5-19 -0.128630705
## 4 2006-5-26  0.005555556
## 5  2006-6-5 -0.045777427
## 6 2006-6-12 -0.095119934
```

```
## The last rows/observations
tail(D)
```

```
##          time         SLV
## 449  2015-4-2 -0.01717791
## 450 2015-4-10 -0.01560549
## 451 2015-4-17 -0.01331642
## 452 2015-4-24 -0.03213368
## 453  2015-5-1  0.02722444
## 454  2015-5-8  0.01874596
```

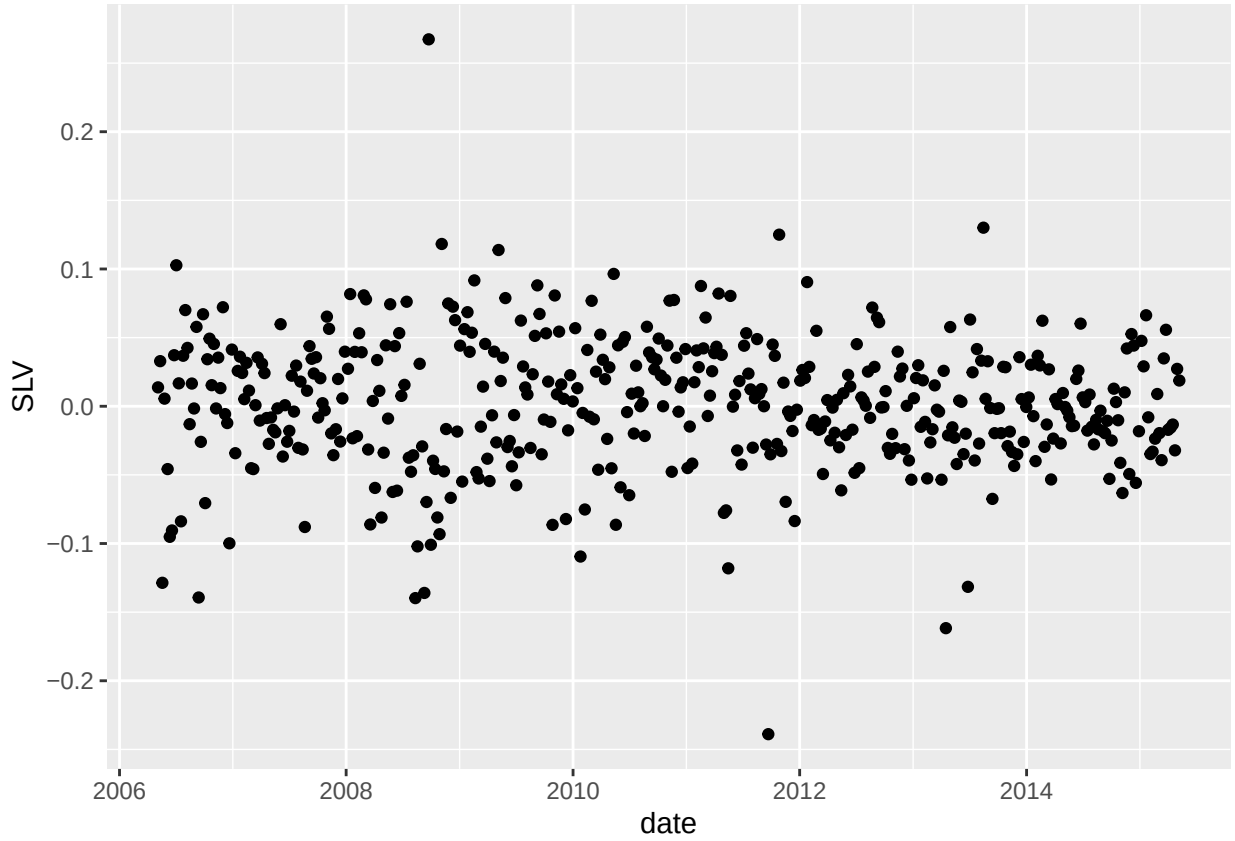
```
## Selected summary statistics
summary(D)
```

```
##      time                SLV
##  Length:454         Min.   :-0.238893
##  Class :character   1st Qu.:-0.026350
##  Mode  :character   Median : 0.002226
##                     Mean   : 0.001468
##                     3rd Qu.: 0.033122
##                     Max.   : 0.267308
```

```
## Another type of summary of the dataset
str(D)
```

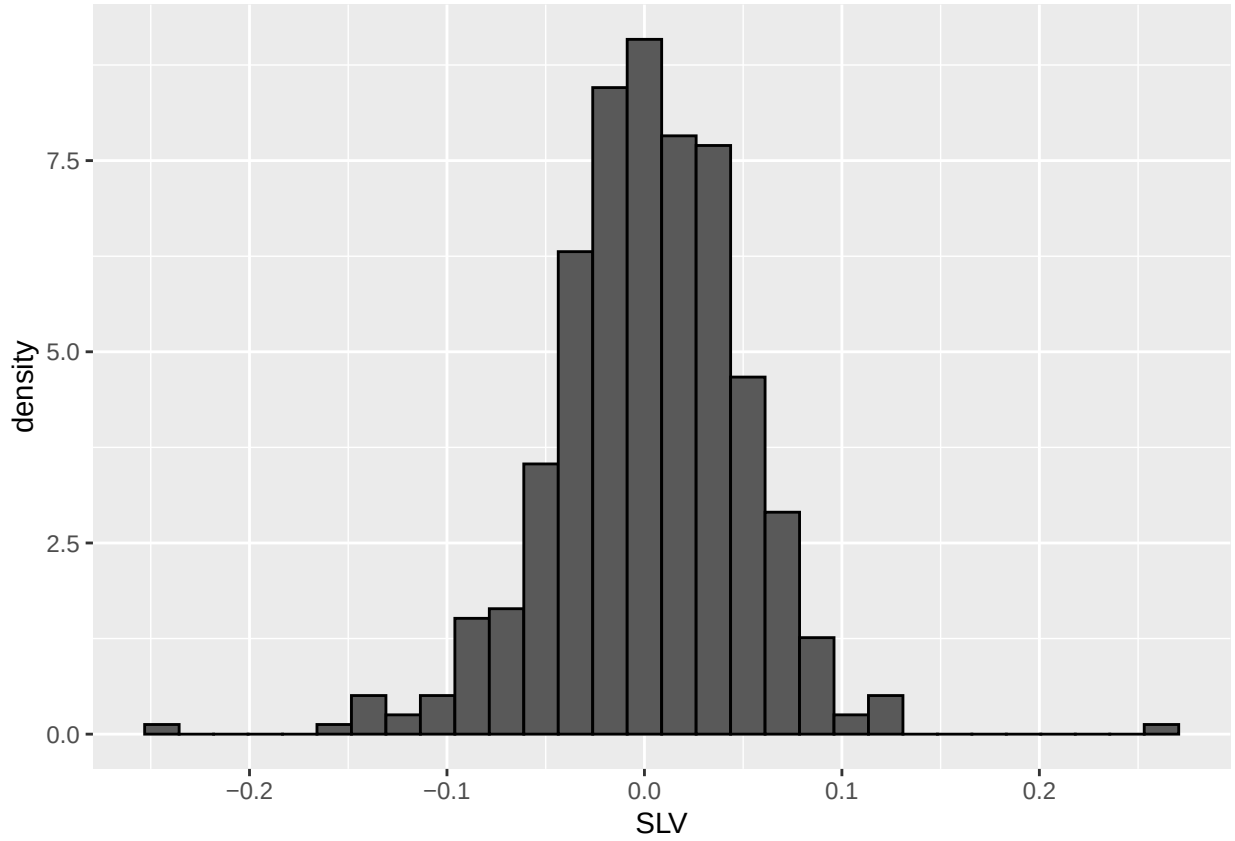
```
## 'data.frame':    454 obs. of  2 variables:
##  $ time: chr  "2006-5-5" "2006-5-12" "2006-5-19" "2006-5-26" ...
##  $ SLV : num  0.01376 0.03286 -0.12863 0.00556 -0.04578 ...
```

```
D$date <- as.Date(D$time)
D$year <- year(D$date)

ggplot(D, aes(x = date, y = SLV)) + geom_point()
```



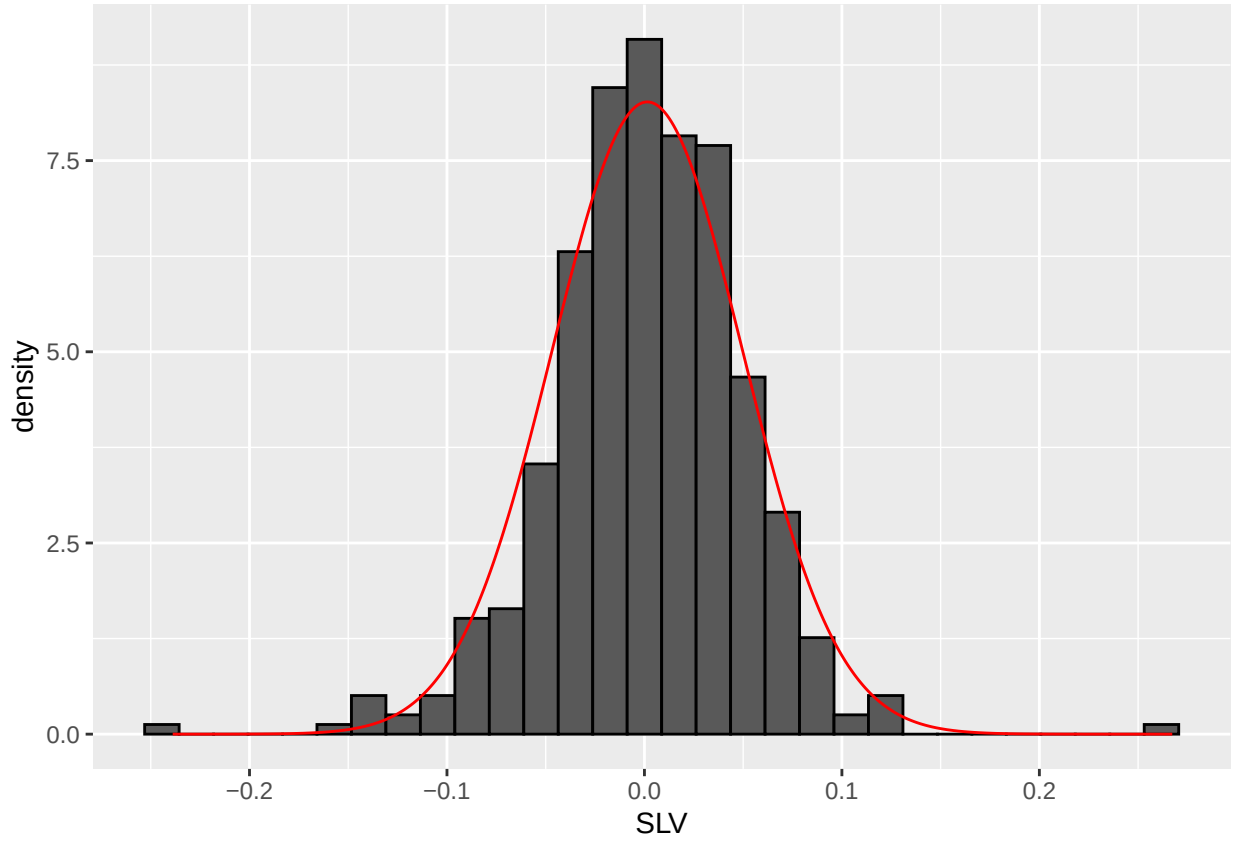

```
ggplot(D, aes(x = SLV)) +
  geom_histogram(aes(y = ..density..), color = 'black')
```

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

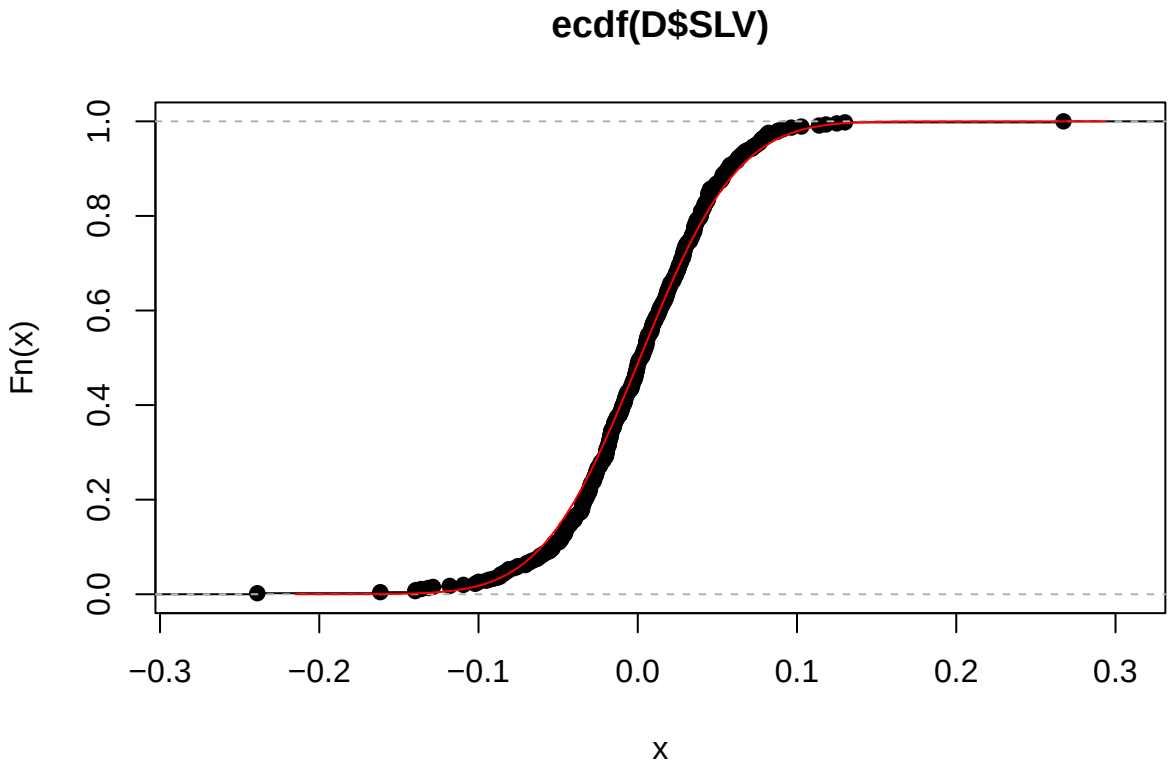



```
par <- nlminb(start = c(1,1), objective = testDistribution,
              distribution = "normal",
              x = D$SLV)
ggplot(D)+
  geom_histogram(aes(x = SLV, y= ..density..,), color='black') +#color, fill
  stat_function(fun = dnorm, n = dim(D)[1], args = list(mean = par$par[1], sd = par$par[2]), color='red
```

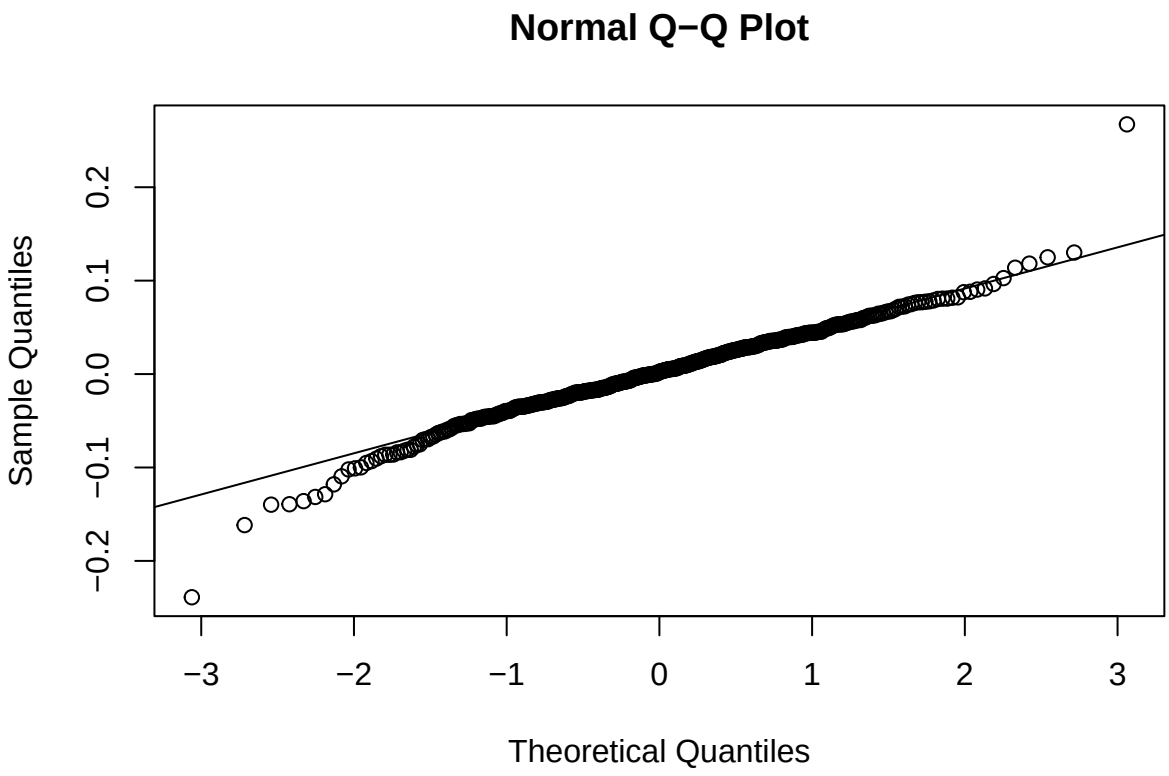
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
plot(ecdf(D$SLV), verticals = T)
xseq <- seq(0.9*min(D$SLV), 1.1*max(D$SLV), length.out=100)
lines(xseq, pnorm(xseq, mean(D$SLV), sd(D$SLV)), col='red')
```




```
#plot(xseq, pnorm(xseq, mean(D$SLV), sd(D$SLV)), col='red')
qqnorm(D$SLV)
qqline(D$SLV)
```




```
lcauchyFUNC <- function(p, data){
  x0 <- p[1] #location R
  gam <- p[2] #scale R > 0
  return(-sum(dcauchy(x = data, location = x0, scale = gam, log = T)))
}
lpownormFUNC <- function(p, data){
  alpha <- p
  return(-sum(log(dpn(x = data, p))))
}
ltFUNC <- function(p, data){
  return(-sum(dt(x = data, df = p, log = T)))
}
lsnFUNC <- function(p, data){ #skewed normal dist
  return(-sum(dsn(x = data, xi = p[1], omega = p[2], alpha = p[3], log = T)))
}
lgnFUNC <- function(p, data){ #symmetric generalized normal dist
  return(-sum(dgnorm(x = data, mu = p[1], alpha = p[2], beta = p[3], log = T)))
}

lasgnFUNC <- function(p, data){ #asymmetric generalized normal dist, when K = 0 has already been checke
  epsilon <- p[1]
  alpha <- p[2]
  kappa <- p[3]
```

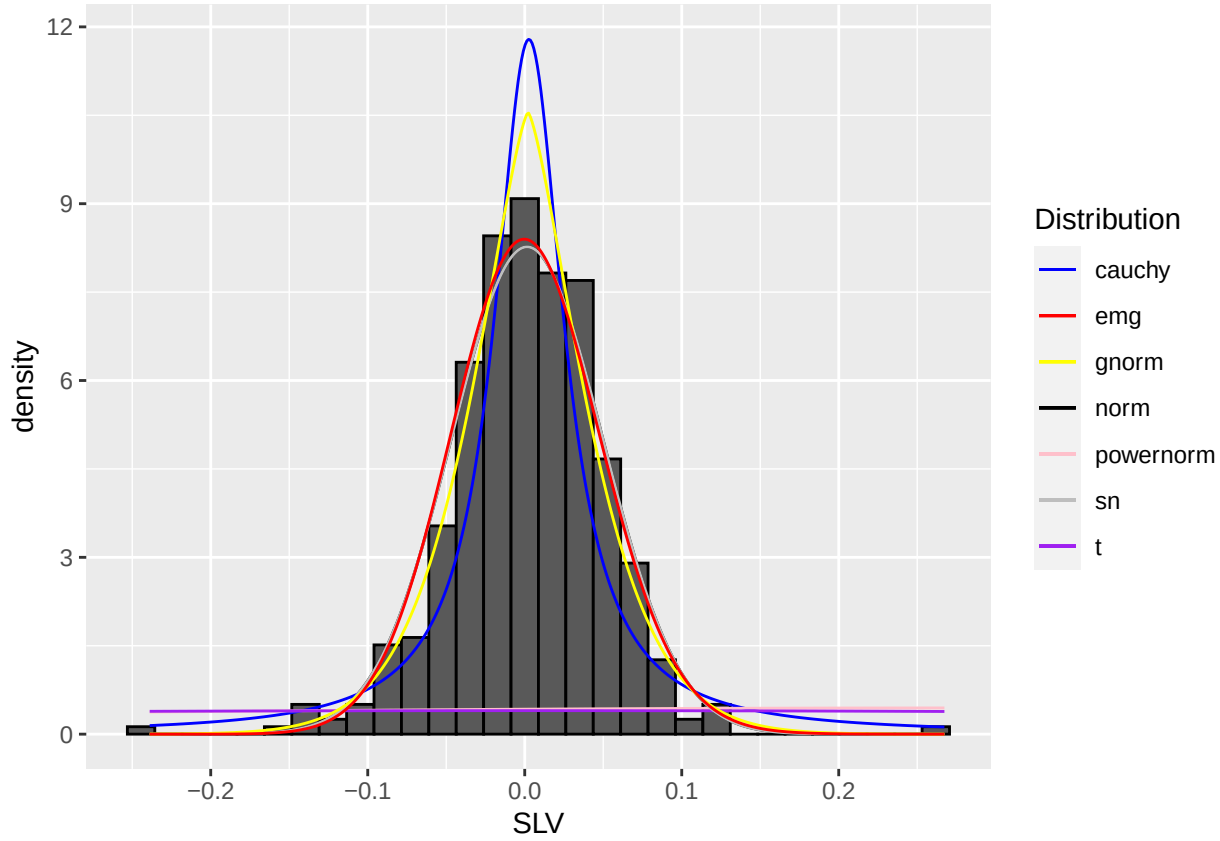
```
  return(-sum( log(dnorm(x = -1/kappa * log(1 - kappa * (data - epsilon) / alpha) ) /
              (alpha - kappa * (data - epsilon)) ) ) )
}

lemgFUNC <- function(p, data){ #exponential modified gaussian dist
  return(-sum(demg(x = data, mu = p[1], sigma = p[2], lambda = p[3], log = T)))
}

par.cauchy <- nlminb(start = c(0,1), objective = lcauchyFUNC, data = D$SLV)
par.pownorm <- nlminb(start = 1, objective = lpownormFUNC, data = D$SLV)
par.t <- nlminb(start = 1, objective = ltFUNC, data = D$SLV)
par.sn <- nlminb(start = c(1,1,1), objective = lsnFUNC, data = D$SLV)
par.gn <- nlminb(start = c(1,1,1), objective = lgnFUNC, data = D$SLV)
```

Hypothesize a model that could fit the data better (Hint: consider tail probabilities), and compare with the normal model estimated above

```
## Not defined for negative values of alpha and/or beta.
## Not defined for negative values of alpha and/or beta.
## Not defined for negative values of alpha and/or beta.
## Not defined for negative values of alpha and/or beta.
```

```
par.asgn <- nlminb(start = c(1,1,1), lower = c(-Inf, -Inf, 0), objective = lasgnFUNC, data = D$SLV)
par.emg <- nlminb(start = c(1,1,1), lower = c(-Inf, 1/1000, 1/1000), objective = lemgFUNC, data = D$SLV

ggplot(D)+
  geom_histogram(aes(x = SLV, y= ..density..,), color='black') + #color, fill
  stat_function(fun = dnorm, n = dim(D)[1], args = list(mean = par$par[1], sd = par$par[2]), aes(colour
  stat_function(fun = dcauchy, n = dim(D)[1], args = list(location = par.cauchy$par[1],
                                                          scale = par.cauchy$par[2]), aes(colour = "cau
  stat_function(fun = dpn, n = dim(D)[1], args = list(alpha = par.pownorm$par), aes(colour = "powernorm
  stat_function(fun = dt, n = dim(D)[1], args = list(df = par.t$par), aes(colour = "t")) +
  stat_function(fun = dsn, n = dim(D)[1], args = list(xi = par.sn$par[1], omega = par.sn$par[2],
                                                      alpha = par.sn$par[3]), aes(colour = "sn")) +
  stat_function(fun = dgnorm, n = dim(D)[1], args = list(mu = par.gn$par[1], alpha = par.gn$par[2],
                                                         beta = par.gn$par[3]), aes(colour = "gnorm")) +
  stat_function(fun = demg, n = dim(D)[1], args = list(mu = par.emg$par[1], sigma = par.emg$par[2],
                                                       lambda = par.emg$par[3]), aes(colour = "emg"))+
  scale_colour_manual(values = c("blue", "red", "yellow", "black", "pink", "grey", "purple"))+
  labs(colour = "Distribution")
```

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




```
#legend('topright', legend=c('normal', 'cauchy', 'power normal', 't'), col=c('red', 'blue', 'green', 'y

AIC.norm <- -2 * sum(dnorm(x = D$SLV, mean = par$par[1], sd = par$par[2], log = T))
+ 2 * length(par$par)
```

```
## [1] 4
```

```
AIC.cauchy <- -2 * sum(dcauchy(x = D$SLV, location = par.cauchy$par[1],
                               scale = par.cauchy$par[2], log = T)) + 2 * length(par.cauchy$par)
AIC.pownorm <- -2 * sum(log(dpn(x = D$SLV, alpha = par.pownorm$par)))
+ 2 * length(par.pownorm$par)
```

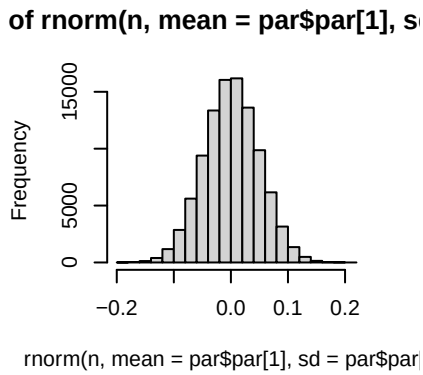
```
## [1] 2
```

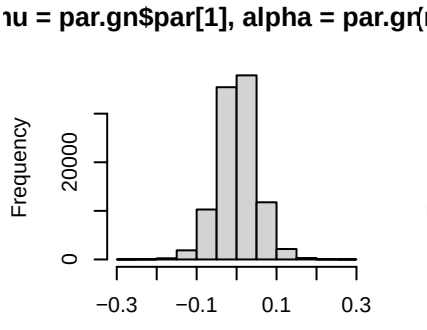
```
AIC.t <- -2 * sum(dt(x=D$SLV, df = par.t$par, log = T)) + 2 * length(par.t$par)
AIC.sn <- -2 * sum(dsn(x=D$SLV, xi = par.sn$par[1], omega = par.sn$par[2], alpha = par.sn$par[3],
                       log = T)) + 2 * length(par.sn$par)
AIC.gn <- -2 * sum(dgnorm(x=D$SLV, mu = par.gn$par[1], alpha = par.gn$par[2], beta = par.gn$par[3],
                          log = T)) + 2 * length(par.gn$par)
AIC.asgn <- -2 * sum( log(dnorm(x = -1/par.asgn$par[3] * log(1 - par.asgn$par[3] * (D$SLV - par.asgn$par
                            (par.asgn$par[2] - par.asgn$par[3] * (D$SLV - par.asgn$par[1])) ) ) + 2 * le
AIC.emg <- -2 * sum(demg(x=D$SLV, mu = par.emg$par[1], sigma = par.emg$par[2], lambda = par.emg$par[3],
                         log = T)) + 2 * length(par.emg$par)
```

```
round(rbind(AIC.norm, AIC.cauchy, AIC.pownorm, AIC.t, AIC.sn, AIC.gn, AIC.asgn, AIC.emg), digits=5)
```

```
##                   [,1]
## AIC.norm    -1463.9996
## AIC.cauchy  -1363.4142
## AIC.pownorm   781.1103
## AIC.t         837.4564
## AIC.sn      -1457.9996
## AIC.gn      -1480.3914
## AIC.asgn    -1458.7443
## AIC.emg     -1461.9880
```

```
n <- 100000
par(mfrow=c(2,3))
hist(rnorm(n, mean = par$par[1], sd = par$par[2]))
hist(D$SLV)
hist(rsn(n, xi = par.sn$par[1], omega = par.sn$par[2], alpha = par.sn$par[3]))
hist(rgnorm(n, mu = par.gn$par[1], alpha = par.gn$par[2], beta = par.gn$par[3]))
hist(rcauchy(n, location = par.cauchy$par[1], scale = par.cauchy$par[2]))
hist(remg(n, mu = par.emg$par[1], sigma = par.emg$par[2], lambda = par.emg$par[3]))
```





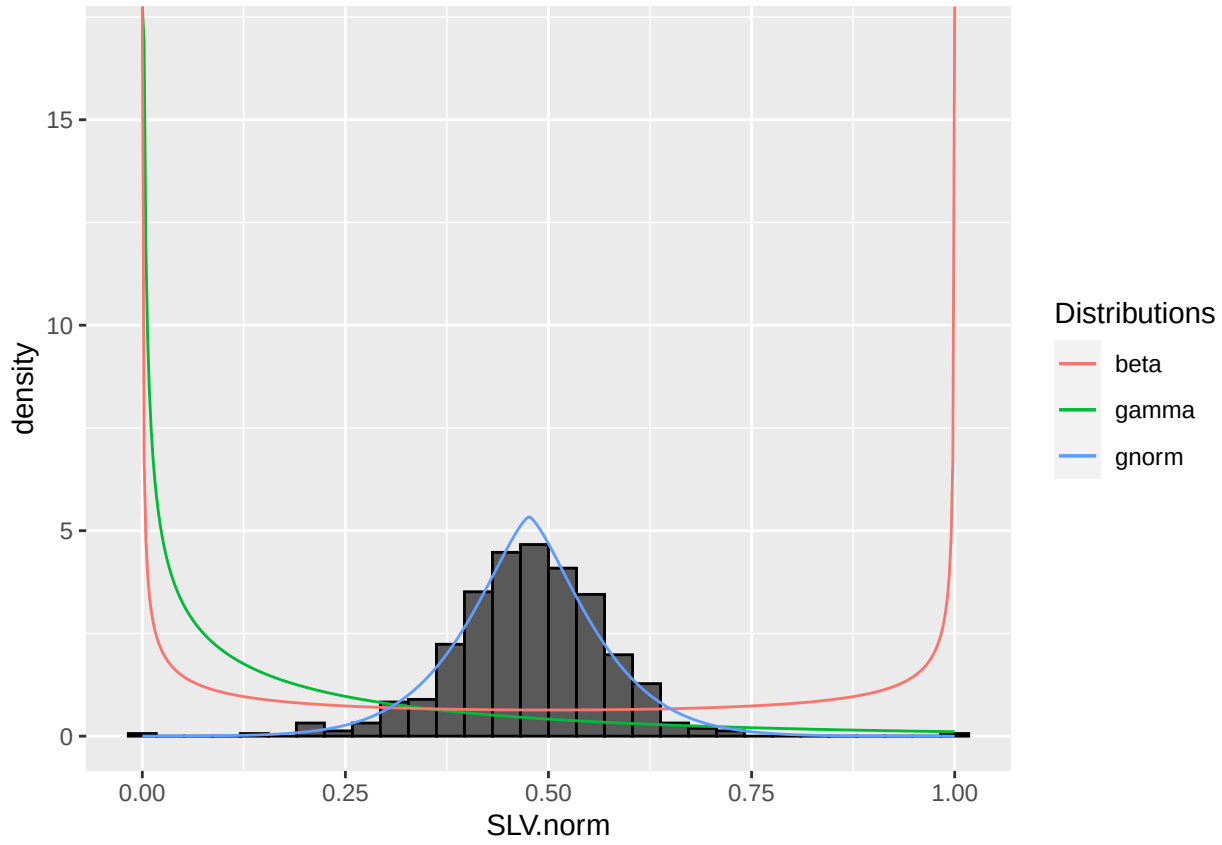
Present the final model (i.e. relevant keynumbers for the estimates)

```r
# ggplot(D)+
#   #geom_histogram(aes(x = rgnorm(dim(D)[1], mu = par.gn$par[1], alpha = par.gn$par[2], beta = par.gn$
#   geom_histogram(aes(x = SLV, y= ..density..,), color='red') + #color, fill
#   geom_histogram(aes(x = rgnorm(dim(D)[1], mu = par.gn$par[1], alpha = par.gn$par[2], beta = par.gn$p
lgamFUNC <- function(p, norm_data){
  k <- p[1] #shape
  beta <- p[2] # rate
  return(-sum(dgamma(x = norm_data, shape = k, rate = beta, log = T)))
}
lbetaFUNC <- function(p, norm_data){
  alpha <- p[1] #shape
  beta <- p[2] #shape
  -sum(dbeta(x = norm_data, shape1 = alpha, shape2 = beta, log = T))
}

D$SLV.norm <- ( D$SLV - min(D$SLV) ) / (max(D$SLV) - min(D$SLV))
par.gam <- nlminb(start = c(0.5, 0.5), objective = lgamFUNC, norm_data = D$SLV.norm)
par.beta <- nlminb(start = c(0.5, 0.5), objective = lbetaFUNC, norm_data = D$SLV.norm)
par.gn.norm <- nlminb(start = c(1,1,1), objective = lgnFUNC, data = D$SLV.norm)
```

```
## Not defined for negative values of alpha and/or beta.
## Not defined for negative values of alpha and/or beta.
## Not defined for negative values of alpha and/or beta.
```

```r
ggplot(D)+
  geom_histogram(aes(x = SLV.norm, y= ..density..,), color='black') +
  stat_function(fun = dgamma, n = dim(D)[1], args = list(shape = par.gam$par[1], scale = par.gam$par[2]
  stat_function(fun = dbeta, n = dim(D)[1], args = list(shape1 = par.beta$par[1], shape2 = par.beta$par
  stat_function(fun = dgnorm, n = dim(D)[1], args = list(mu = par.gn.norm$par[1],
      alpha = par.gn.norm$par[2], beta = par.gn.norm$par[3]), aes(colour='gnorm')) +
  labs(colour = 'Distributions')
```

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




```
-2 * sum(dgamma(x=D$SLV.norm, shape = par.gam$par[1], rate = par.gam$par[2], log = T)) + 2 * length(par
```

```
## [1] -Inf
```

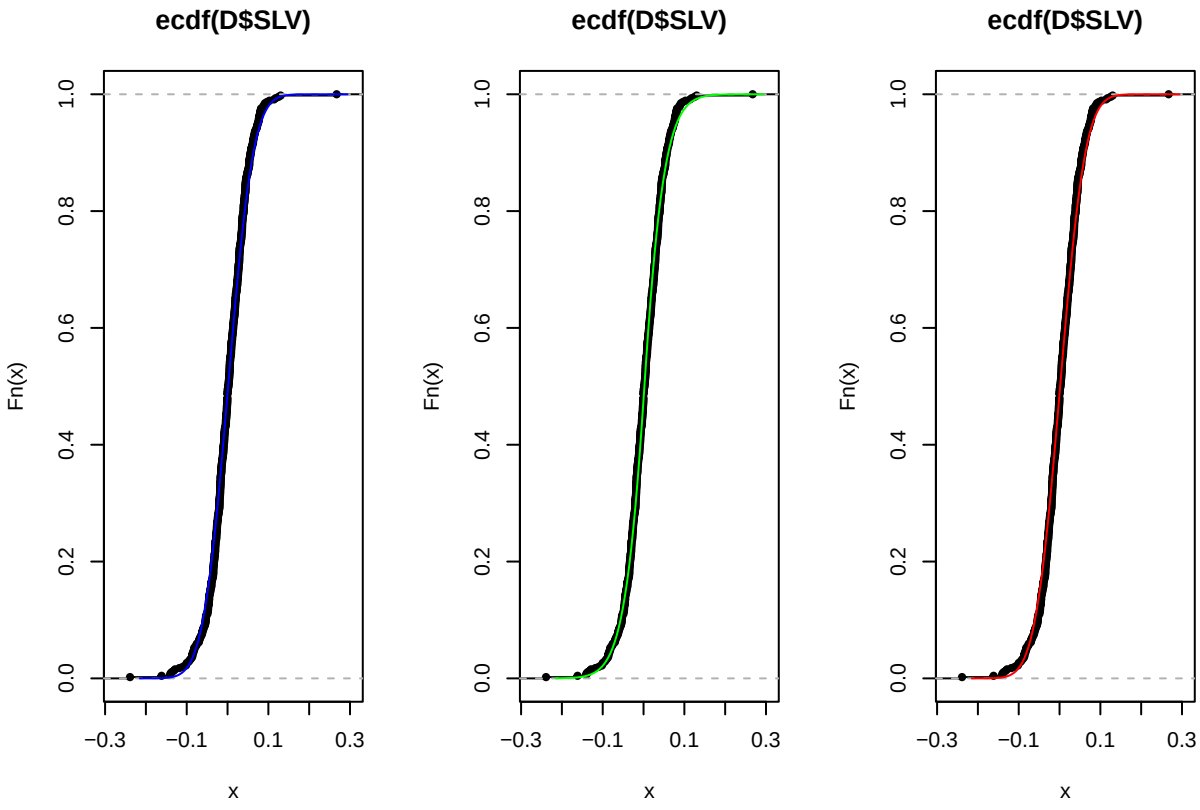
```
-2 * sum(dbeta(x=D$SLV.norm, shape1 = par.beta$par[1], shape2 = par.beta$par[2], log = T)) + 2 * length
```

```
## [1] -Inf
```

```
-2 * sum(dgnorm(x = D$SLV.norm, mu = par.gn.norm$par[1], alpha = par.gn.norm$par[2],
                beta = par.gn.norm$par[3], log = T)) + 2 * length(par.gn.norm$par)
```

```
## [1] -862.2052
```

```
par(mfrow=c(1,3))
plot(ecdf(D$SLV), verticals = T)
xseq <- seq(0.9*min(D$SLV), 1.1*max(D$SLV), length.out=100)
#lines(xseq, pnorm(xseq, mean(D$SLV), sd(D$SLV)), col='red')
lines(xseq, pnorm(xseq, mean = par$par[1], sd = par$par[2]), col='blue')
plot(ecdf(D$SLV), verticals = T)
lines(xseq, pgnorm(xseq, mu = par.gn$par[1], alpha = par.gn$par[2], beta = par.gn$par[3]), col='green')
plot(ecdf(D$SLV), verticals = T)
lines(xseq, psn(xseq, xi = par.sn$par[1], omega = par.sn$par[2], alpha = par.sn$par[3]), col='red')
```


ecdf(D$SLV)
Fn(x)
x
0.0
0.2
0.4
0.6
0.8
1.0
−0.3
−0.1
0.1
0.3
ecdf(D$SLV)
Fn(x)
x
0.0
0.2
0.4
0.6
0.8
1.0
−0.3
−0.1
0.1
0.3
ecdf(D$SLV)
Fn(x)
x
0.0
0.2
0.4
0.6
0.8
1.0
−0.3
−0.1
0.1
0.3