A1 Project 1

Johnsen & Johnsen

2022-10-05

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
```

The dataset contains 288 observations of the 8 variables: r.day, month, day, pow.obs, ws30, wd30, date, pow.obs.norm.

Summary statistics of the 8 variables:

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

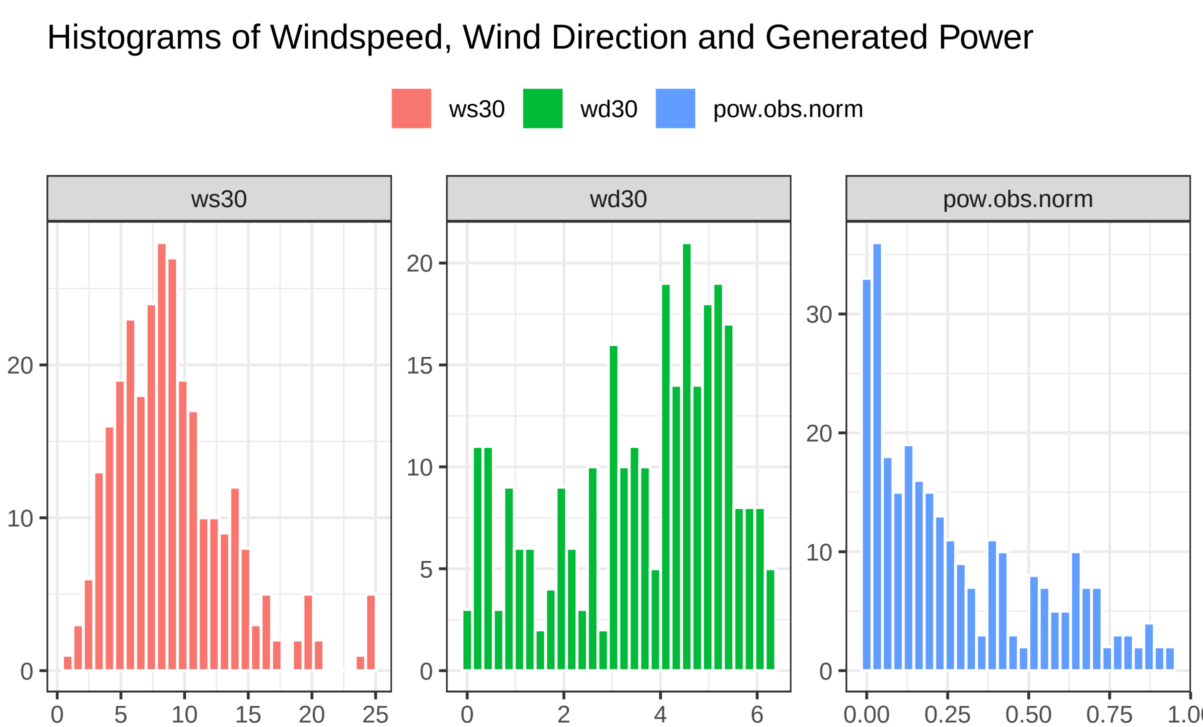
```
##       ws30             wd30                date               pow.obs.norm
##  Min.   : 1.139   Min.   :0.000095   Min.   :2003-01-01   Min.   :0.0000247
##  1st Qu.: 5.779   1st Qu.:2.474999   1st Qu.:2003-03-19   1st Qu.:0.0508315
##  Median : 8.498   Median :4.079297   Median :2003-06-05   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)
```

As can be seen from the table above, the highest observed generated power was 4681.062, and thus our normalization based on a max of 5000 will yield an observed normalized power to be between [0;1[.

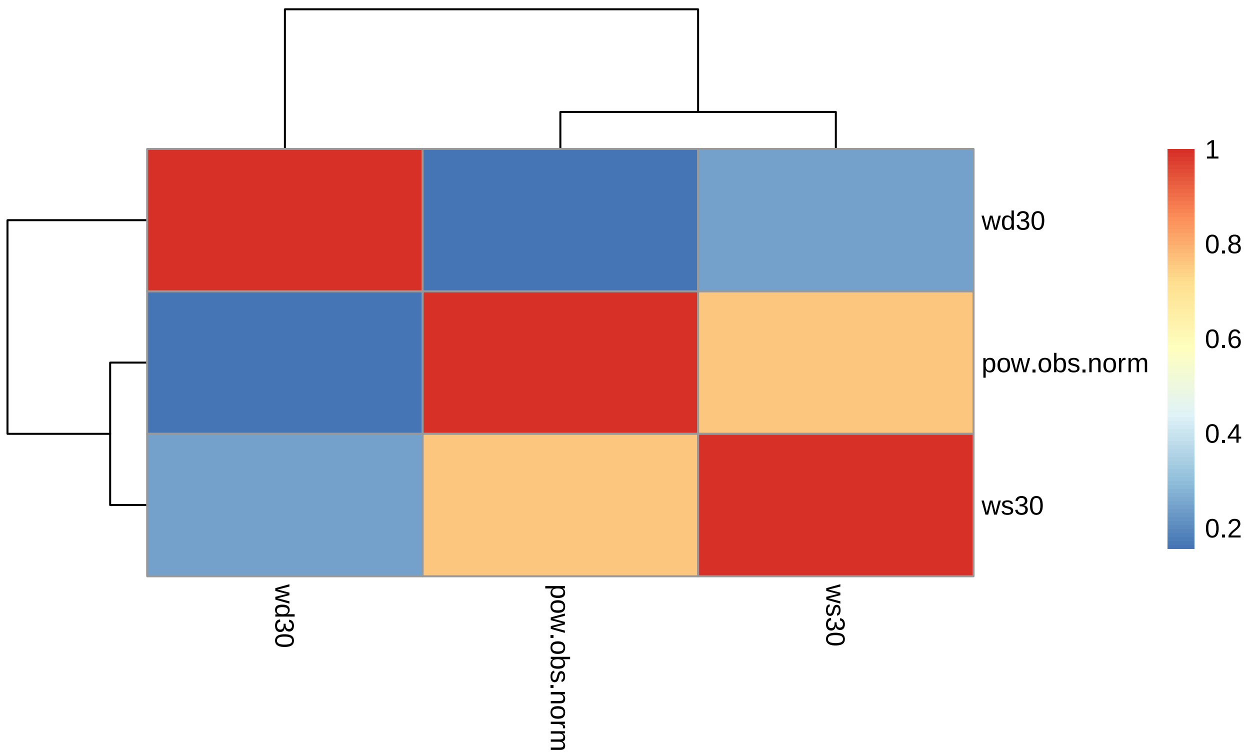
Visualization of the three relevant variables:

```
meltD <- D %>%
  dplyr::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")+
  theme_bw()+
  labs(fill = "")+
  theme(legend.position = "top")+
  labs(y = "", x = "")+
  ggtitle("Histograms of Windspeed, Wind Direction and Generated Power")
```



The heatmap below, shows that the observed power and the wind speed show the strongest correlation in the dataset.

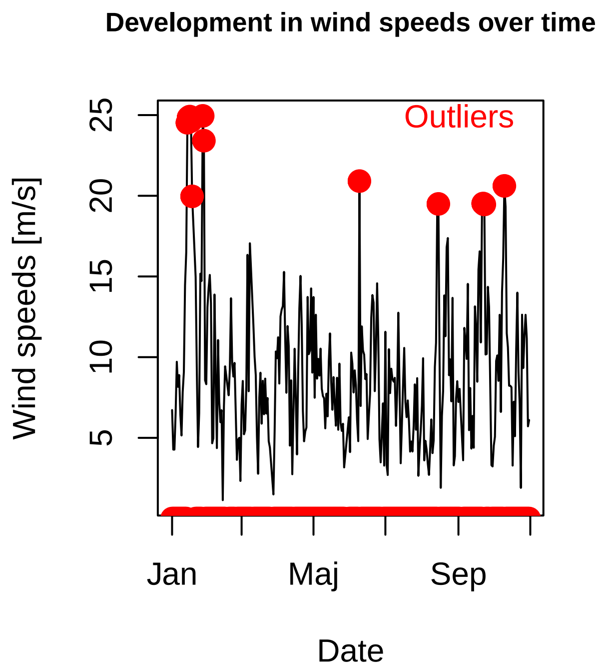
```
D %>%
  dplyr::select(pow.obs.norm, wd30, ws30) %>%
  cor() %>%
  pheatmap()
```

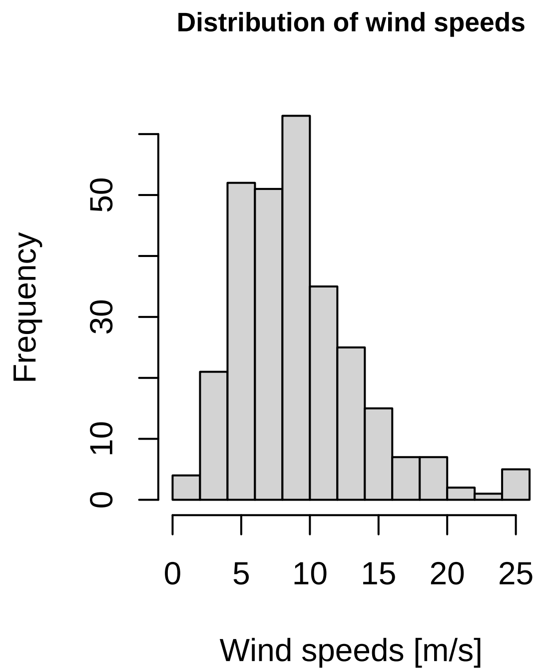



```
#par(mfrow=c(1,2))
#plot(D$date, D$pow.obs, type = 'l', xlab="Date", ylab="Average daily power production [kW]",
#     main = 'Development in average daily power production over time', cex.main = 0.8, col=1)
#hist(D$pow.obs, xlab="Power production [kW]", main='Distribution of average daily power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

Outlier analysis: No outliers were found for the wind direction.

```
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, lwd = 10)
text(as.Date(quantile(D$r.day,probs = 0.8), origin = min(D$date)), quantile(D$ws30, probs = 1), "Outlie
hist(D$ws30, xlab="Wind speeds [m/s]", main='Distribution of wind speeds', cex.main = 0.8)
```



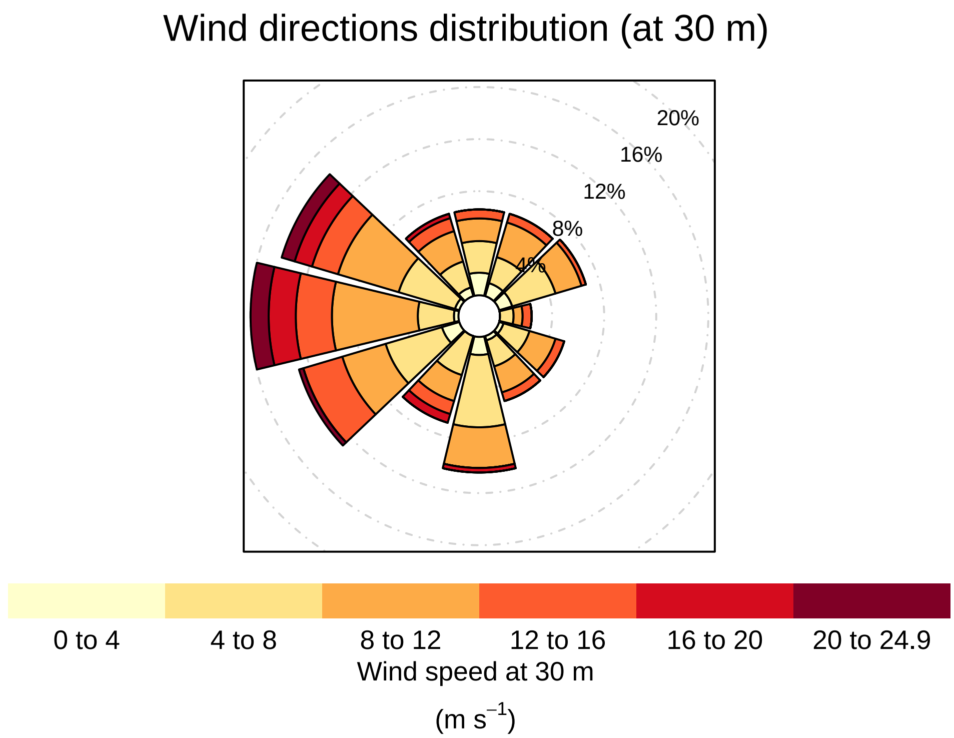



```
# par(mfrow=c(1,2))
# plot(D$date, D$wd30, type = 'l', xlab="Date", ylab=expression(paste("Wind direction. N = 0, E = ", pi/
#      main='Development in wind directions over time', cex.main = 0.8)
# lines(D$date, replicate(length(D$date), 3*pi/2), type='l', col=2) #W
# lines(D$date, replicate(length(D$date), 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
# 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)
```

The distribution of the wind direction can also be examined through a wind rose visualization as to capture the fact that the wind direction have been supplied in radians, and should as such be treated as a circular distribution.

```
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)
```



Here we see that the main wind direction is West. It is also from this direction we see the highest wind speeds.

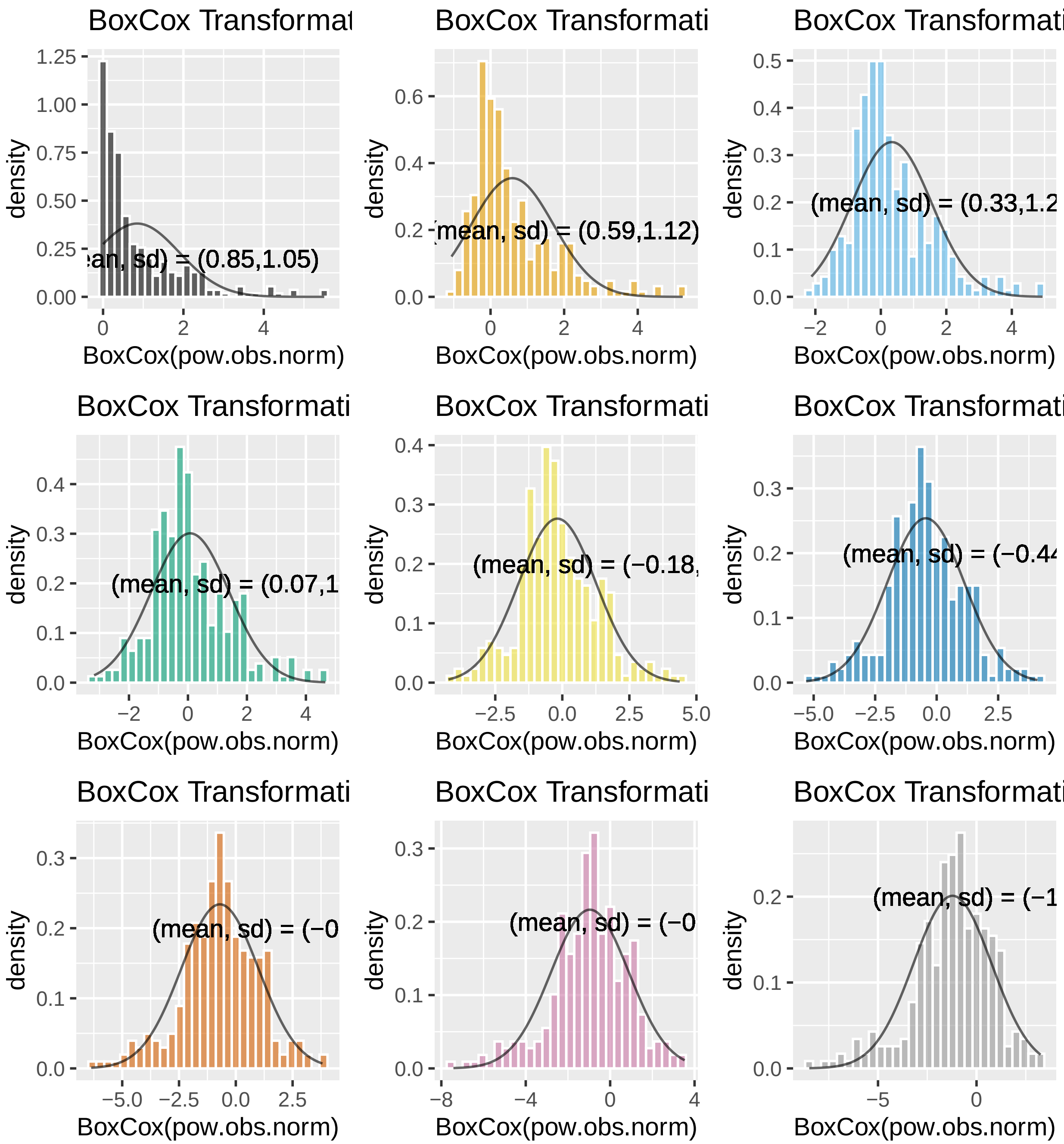
Simple Models

```
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. We try-out three different approaches: 1) Box-Cox transformation; 2) Transformation based on Eq. 1 in the assignment description; 3) fit distributions directly to the standardized pow.obs.

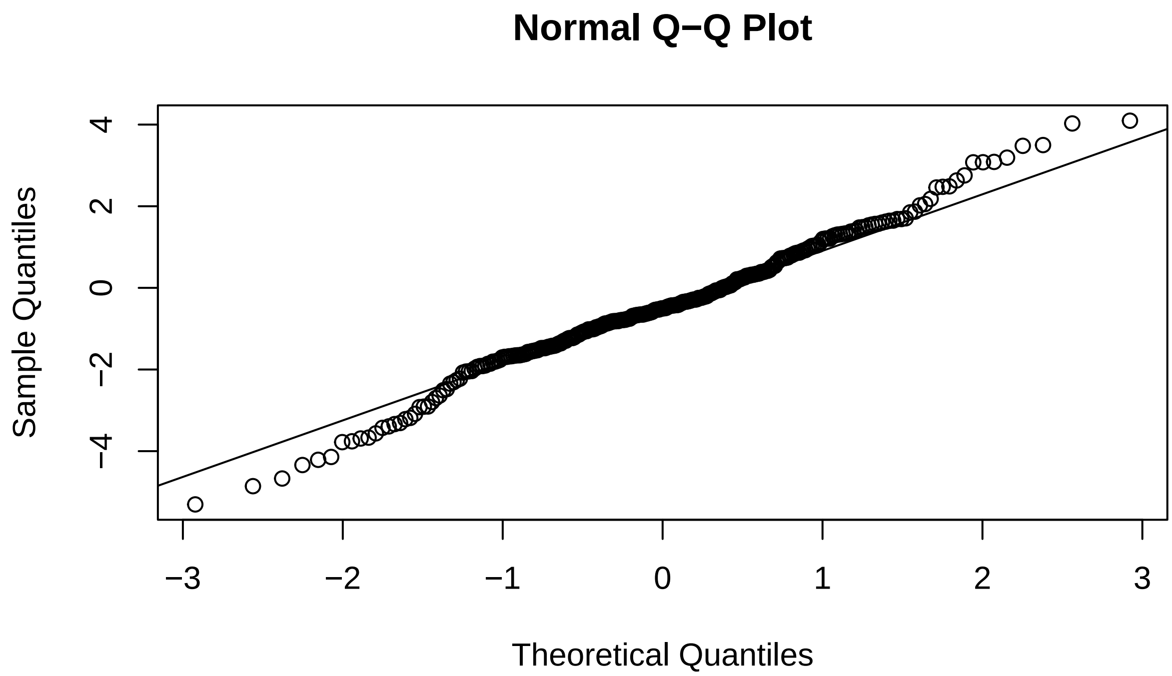
```
#Box-Cox transformation of pow.obs.norm
#Examine different transformations and the achieved fit when fitting a normal
```

```r
#distribution

lambda <- c(0.0, 0.05, 0.10, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4)
pal <- palette.colors(length(lambda))
BoxCoxPlot <- list()
for (i in 1:length(lambda)){
  xData <- 2*log(D$pow.obs.norm^lambda[i]/(1-D$pow.obs.norm)^(1-lambda[i]))
  n <- nlminb(start = c(-1,1)
              , objective = testDistribution
              , x = xData
              , distribution = "normal")

  #simData <- rnorm(n = length(D$pow.obs.norm), mean = n$par[1], sd = n$par[2])
  #D$sim <- simData
  D$BoxCox <- xData

  BoxCoxPlot[[paste0(lambda[i])]] <- ggplot(D)+
    #geom_histogram(aes(x = sim, y = ..density..)
    #               , colour = "white"
    #               , alpha = 0.5
    #               , fill = "black")+
    geom_histogram(aes(x = BoxCox, y = ..density..)
                   , colour = "white"
                   , alpha = 0.6
                   , fill = pal[[i]])+
    labs(x = "BoxCox(pow.obs.norm)")+
    ggtitle(paste0("BoxCox Transformation of Windpower, ", expression(lambda), " = ", lambda[i]))+
    #geom_text(x = 2, y = 0.23, label = paste0("NLL = ", round(n$objective,2)))+
    geom_text(x = 2, y = 0.2, label = paste0("(mean, sd) = (", round(n$par[1],2), ",", round(n$par[2],2
    stat_function(fun = dnorm, n = length(D$pow.obs.norm), args = list(mean = n$par[1], sd = n$par[2]),
}

grid.arrange(grobs = BoxCoxPlot)
```

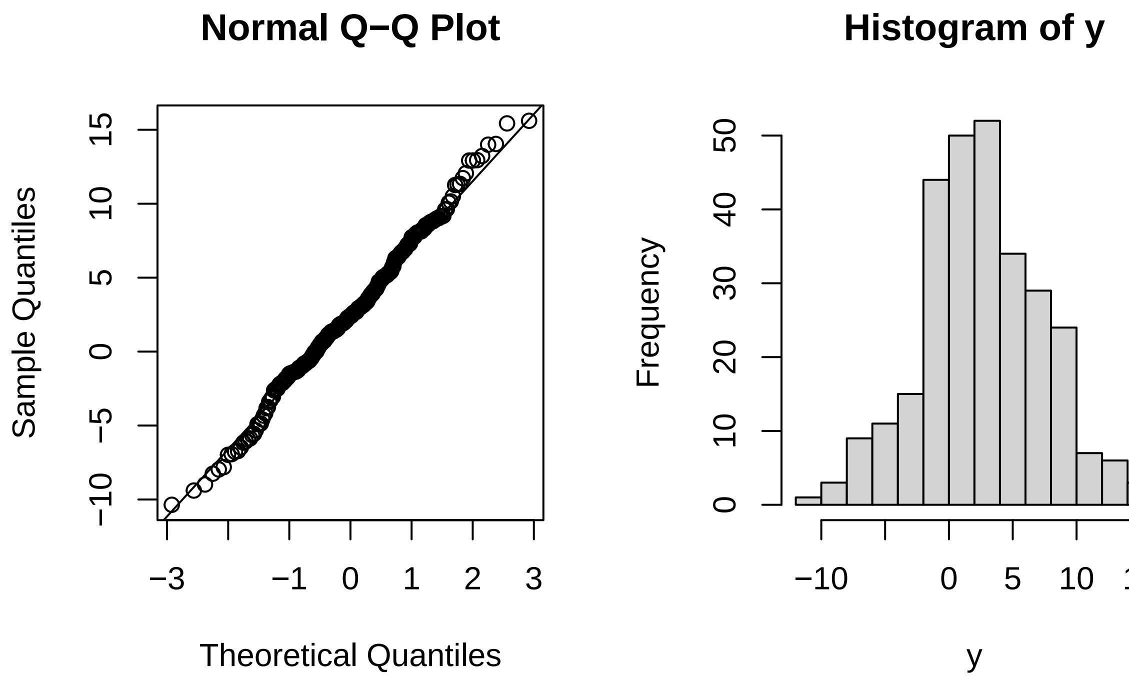


Box-Cox:
From here it can be seen that a box-cox transformation with lambda = 0.25 might be appropriate. It is however not a good approximation as can be seen from the qqplot below.

```
xData <- 2*log(D$pow.obs.norm^0.25/(1-D$pow.obs.norm)^(1-0.25))
qqnorm(xData)
qqline(xData)
```




```
#Define transformation function
Trans.eq1 <- function(lambda, y){
  y_lambda <- 1/lambda * log(y^lambda/(1-y^lambda))#, lambda > 0
  return(y_lambda)
}

#Optimization function
lambda_NLL <- function(lambda, x = D$pow.obs.norm){
  y <- Trans.eq1(lambda, x)
  NLL <- -as.numeric(shapiro.test(y)$statistic)
  return(NLL)
}

theta.hat <- nlminb(start = 0.2, objective = lambda_NLL)
#round to two decimal points.
lambda <- round(theta.hat$par, 2)
D$transformed.pow.obs.norm <- Trans.eq1(lambda, D$pow.obs.norm)

#Check qqplot:
par(mfrow = c(1,2))
y <- Trans.eq1(lambda, D$pow.obs.norm)
qqnorm(y)
qqline(y)
hist(y)
```



Eq. 1 Transformation
Change of variable

In order to calculate the new likelihood function for the transformed pow.obs, we use change of variable:

$$y = g(x) = \frac{1}{0.26} log\left(\frac{x^{0.26}}{1 - x^{0.26}}\right)$$

Now we can calculate the pdf of y as:

$$f_y(y) = \frac{f_x(x)}{\left|\frac{dg}{dx}\right|}$$

This finally yields:

$$f_x(x) = f_y(y)\left|\frac{dg}{dx}\right| = f_y(y)\frac{1}{x(-1 + x^{\lambda})}$$

Here, we can either solve our transformation expression $y = g(x)$ for $x = g^{-1}(y)$ and insert this expression, or we can simply supply the normalized pow.obs to the derivative function and our transformed data to the pdf $f_y(y)$. The likelihood of function for the distribution can now be expressed as:

$$L(y) = \prod_{i=1}^{n} f_y(y)\frac{1}{g^{-1}(y)(-1 + g^{-1}(y)^{\lambda})}$$

Calculating the negative log-likelihood we get:

$$\mathcal{L}(y) = \sum_{i=1}^{n} log\left(f_y(y)\frac{1}{g^{-1}(y)(-1 + g^{-1}(y)^{\lambda})}\right)$$

Where $f_y(y)$ is the normal distribution pdf with mean=μ and variance=σ^2). We define this log-likelihood in R and get:

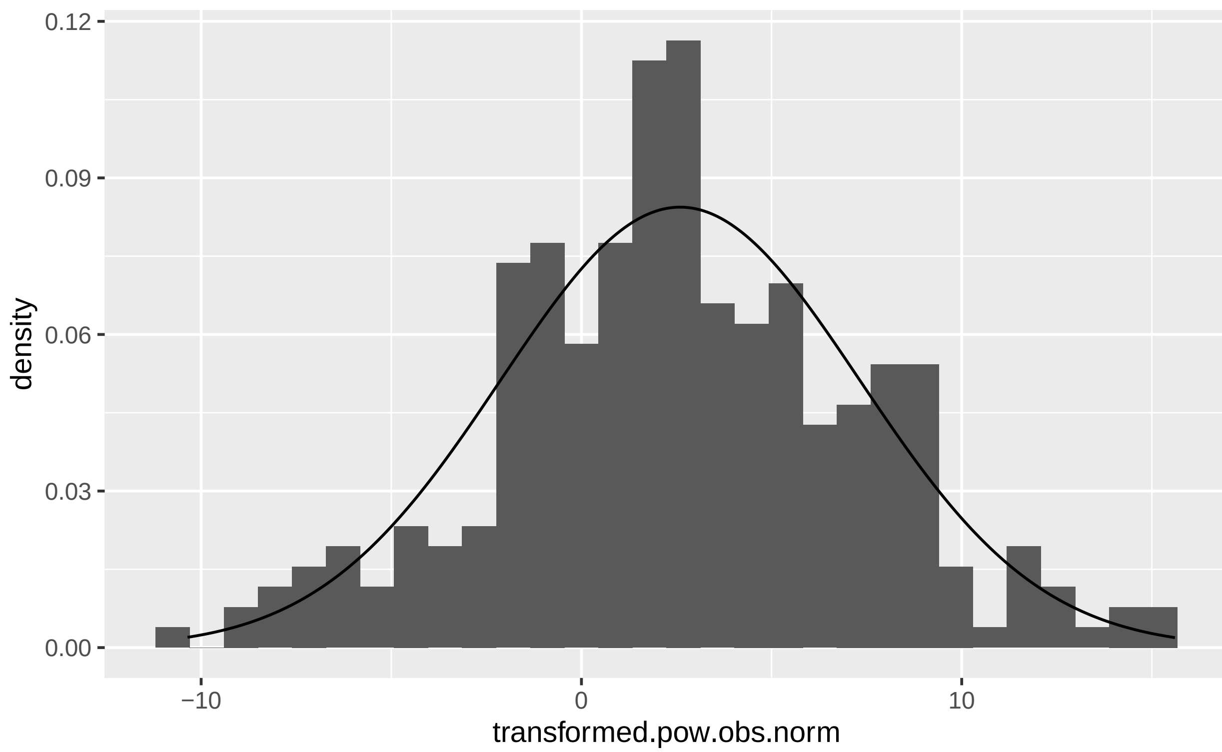
```r
dgdx <- function(x,lambda){
  return(-1/(x*(x^lambda - 1)))
}

#Density for y is simply the normal distribution:
nll.y <- function(theta, x){
  y <- Trans.eq1(lambda, x)
  return(-sum(log(dnorm(y, mean = theta[1], sd = theta[2])*dgdx(x, lambda))))
}
theta.hat.y <- nlminb(start = c(0,1), objective = nll.y
       , x = D$pow.obs.norm)

#### alternative using g^-1(y) ####
g_inverse <- function(y,lambda){
  return((1/(exp(y*lambda)+1))^(1/lambda) * exp(y))
}

nll.y.alt <- function(theta, y){
  return(-sum(log(dnorm(y, mean = theta[1], sd = theta[2])*dgdx(g_inverse(y,lambda), lambda))))
}

theta.hat.y.alt <- nlminb(start = c(0,1), objective = nll.y.alt
       , y = Trans.eq1(lambda, D$pow.obs.norm))

theta.hat.y$par
```

```
## [1] 2.596304 4.726883
```

```r
theta.hat.y.alt$par
```

```
## [1] 2.596304 4.726883
```

```r
#This yields the same estimates#
#### Alternative end ####

#plot the transformed data alongside the found distribution
ggplot(D)+
  geom_histogram(aes(x = transformed.pow.obs.norm, y = ..density..))+
  stat_function(fun = dnorm, n = dim(D)[1], args = list(mean = theta.hat.y$par[1]
                                                     , sd = theta.hat.y$par[2]))
```



No Transformation 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
```

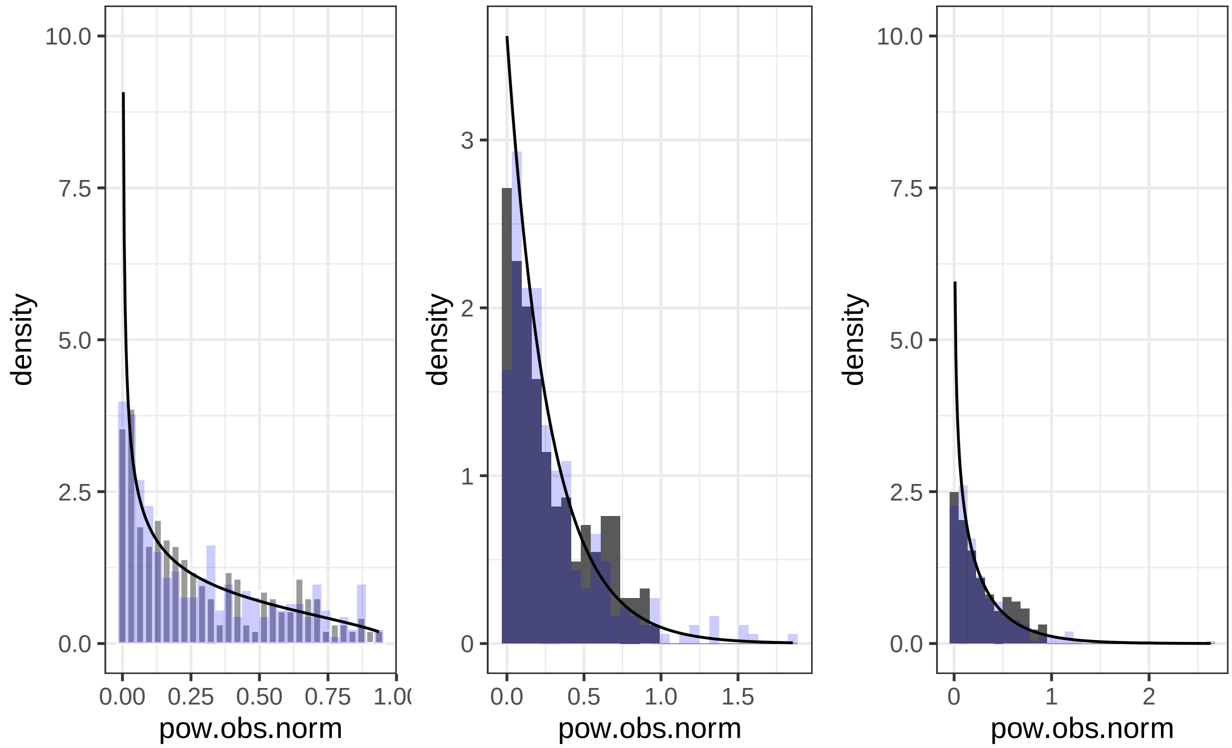
```
par.beta <- nlminb(start = c(2,5)
                   , objective = testDistribution
                   , distribution = "beta"
                   , x = D$pow.obs.norm
                   , lower = c(0,0.8))
par.beta$objective
```

```
## [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])
sam.plot.pow.beta <- 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

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

#Sampling from the found gamma distribution
D$simdata <- rgamma(length(D$pow.obs.norm), shape = par.gamma$par[1], rate = par.gamma$par[2])
sam.plot.pow.gamma <- 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

grid.arrange(sam.plot.pow.beta, sam.plot.pow.exp, sam.plot.pow.gamma, ncol = 3)
```




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

Comparing likelihoods We compare the likelihoods achieved through the different models by calculation of AIC:

```r
print(paste("AIC normal transformation: ", 2*theta.hat.y$objective+2*2))
```

```
## [1] "AIC normal transformation:  -240.692759888533"
```

```r
print(paste("AIC beta distribution: ", 2*par.beta$objective+2*2))
```

```
## [1] "AIC beta distribution:  -239.323586939173"
```

```r
print(paste("AIC exponential distribution: ", 2*par.exp$objective+2*1))
```

```
## [1] "AIC exponential distribution:  -163.017243829812"
```

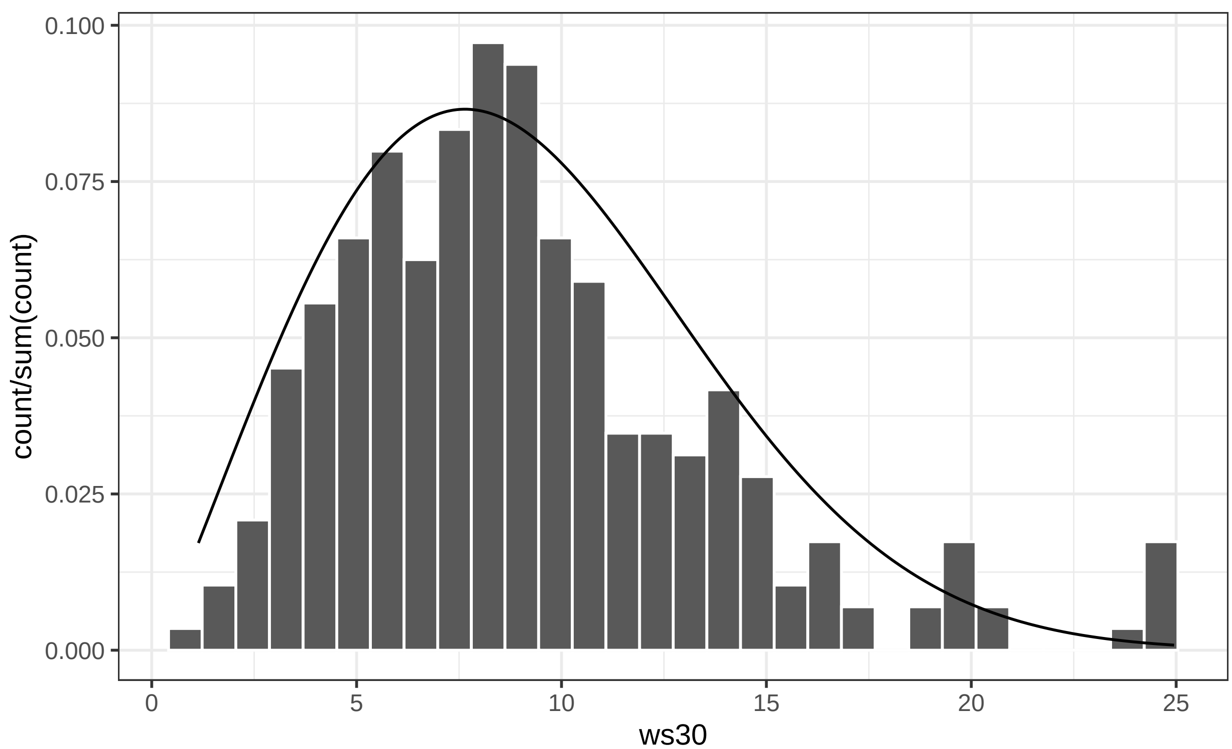
```r
print(paste("AIC gamma distributioun: ", 2*par.gamma$objective+2*2))
```

```
## [1] "AIC gamma distributioun:  -190.76348287363"
```

From this comparison we see that the applied transformation and subsequent fitting of a normal distribution is the most appropriate model. The normal model is 1.005627 times more likely than the beta distribution (p. 30).

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

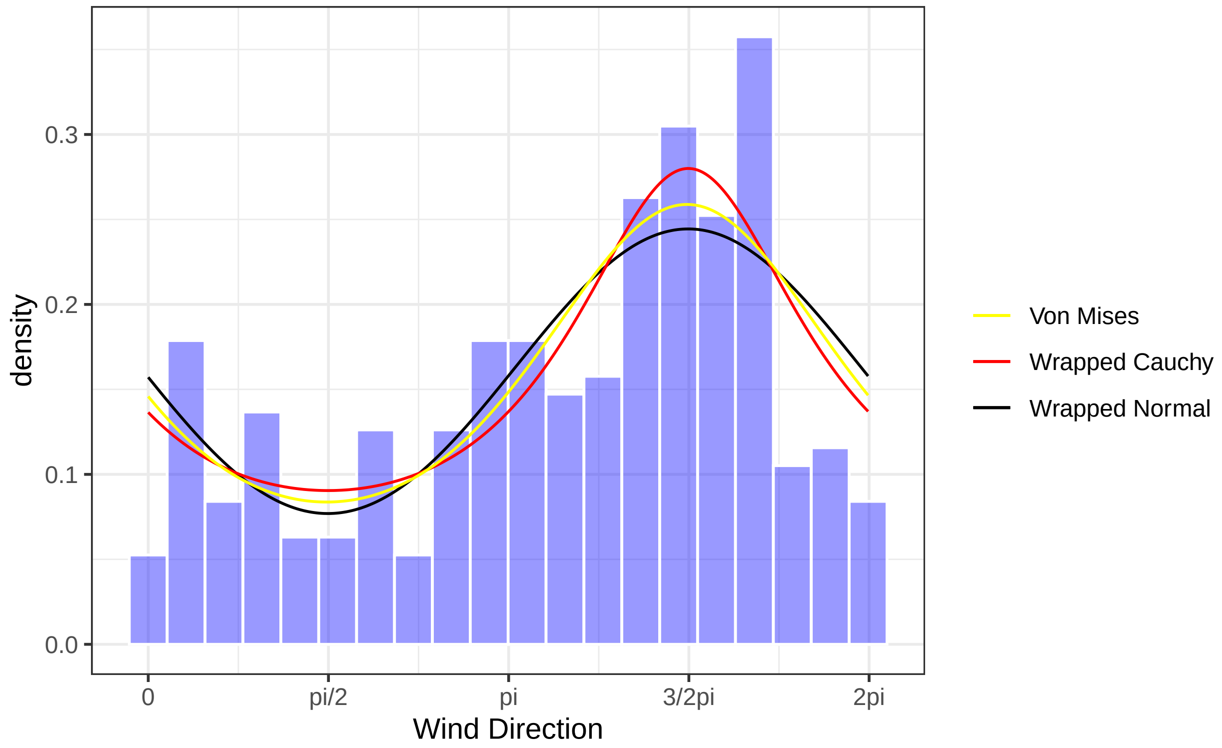
```
par.ws30 <- nlminb(start = c(1,1), objective = testDistribution
                   , x = D$ws30
                   , distribution = "weibull"
                   , lower = c(0,0))


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$par[1], scale = par.ws30$par
```



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.

```
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)
}

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$par[1], sd = wrapped.
  stat_function(fun = dwrappedcauchy, n = dim(D)[1], args = list(mu = wrapped.cauc.par$par[1],rho = wrap
  #stat_function(fun = dwrappedcauchy, n = dim(D)[1], args = list(mu = -1.5748695, rho = 0.2751607), ae
  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", "blue"))
```

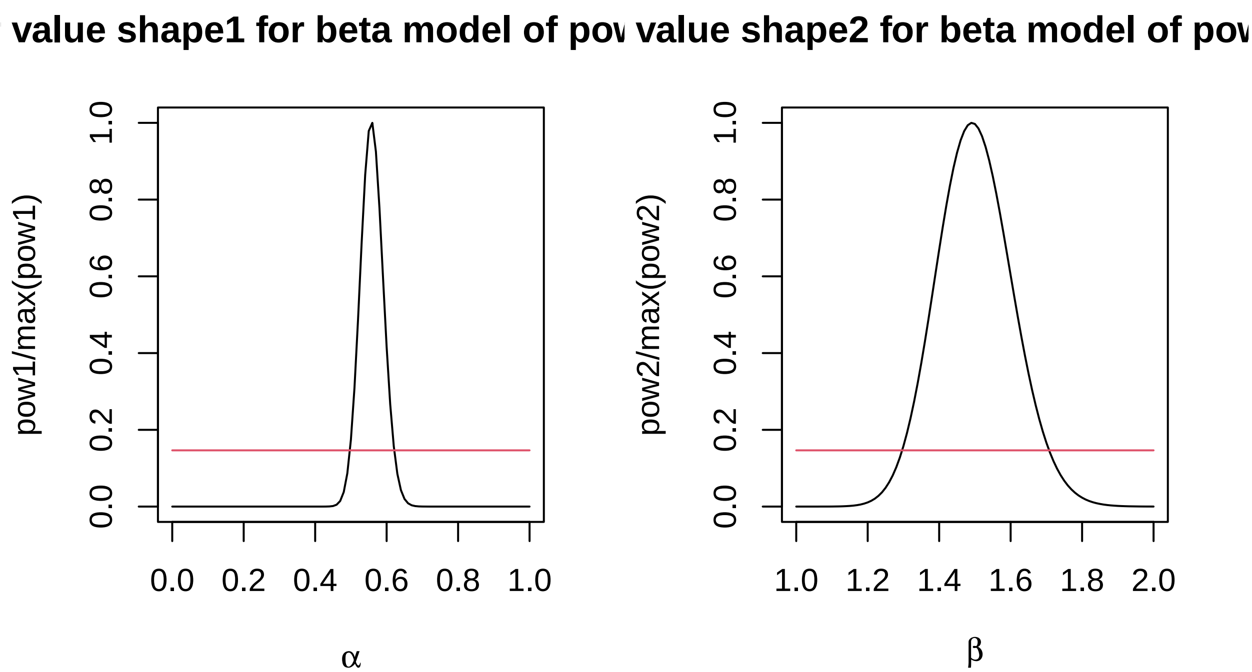


```r
#Calculate AICs
print(paste0("AIC wrapped normal: ", round(-2*log(exp(-wrapped.par$objective     ))+2*2,4), "|"
            ,"AIC wrapped cauchy: ", round(-2*log(exp(-wrapped.cauc.par$objective))+2*2,4), "|"
            ,"AIC von Mises: "     , round(-2*log(exp(-wrapped.vonMises$objective))+2*2,4)))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

```
## [1] "AIC wrapped normal: 1020.5519|AIC wrapped cauchy: 1018.1731|AIC von Mises: 1019.3436"
```

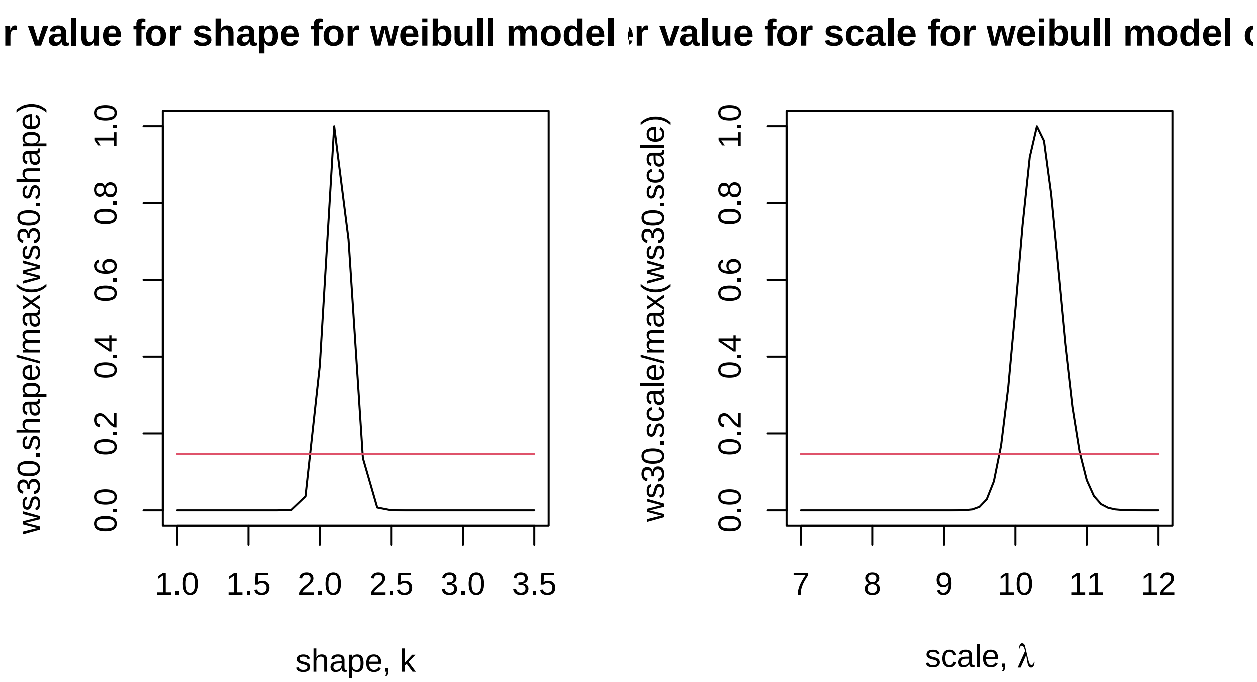
```r
## CI ## WIND POWER
par(mfrow=c(1,1))
alpha <- 0.05
c <- exp(-0.5 * qchisq(1-alpha, df = 1))
#likelihood-based
mle.pow <- par.beta$par

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.



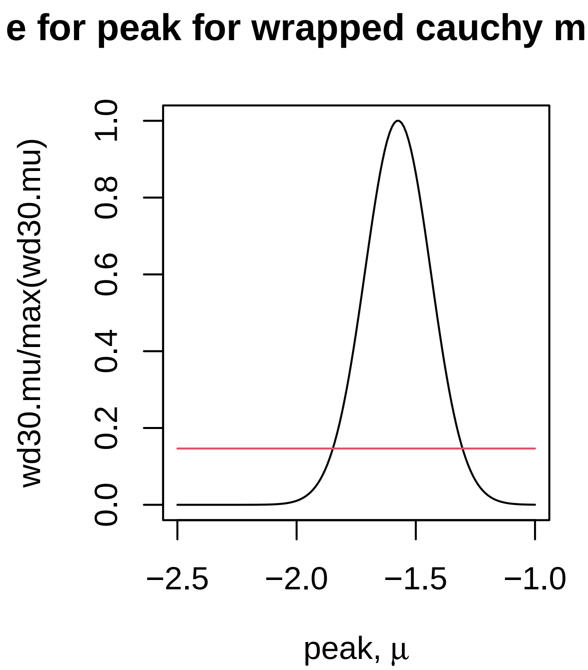

```r
#wald
n <- dim(D)[1]
H.pow.shape1 <- hessian(l.pow.fun, mle.pow[1], shape2 = mle.pow[2], data = D$pow.obs.norm)
V.pow.shape1 <- as.numeric(-1/H.pow.shape1)
H.pow.shape2 <- hessian(l.pow.fun, mle.pow[2], shape1 = mle.pow[1], data = D$pow.obs.norm)
V.pow.shape2 <- as.numeric(-1/H.pow.shape2)
wald.pow.shape1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

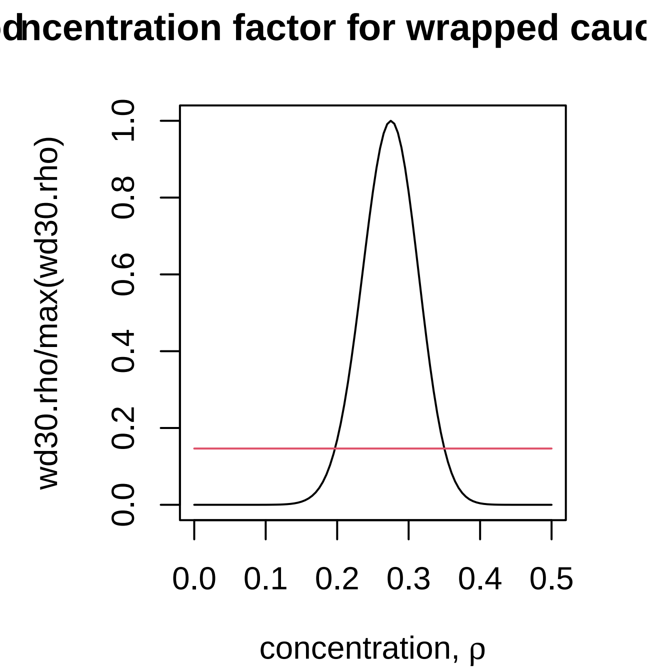
```
      0.5 * qchisq(1-alpha, df = 1)
  } else {
    sum(dweibull(x = D$ws30, shape = mle.ws30.weib[1], scale = mle.ws30.weib[2], log = T)) -
      sum(dweibull(x = D$ws30, shape = mle.ws30.weib[1], scale = y, log = T)) -
      0.5 * qchisq(1-alpha, df = 1)
  }
}
shapes <- seq(1, 3.5, by = 0.1)
ws30.shape <- 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)
```




```
#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)
```

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






```
#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
```

```
alpha <- par.beta$par[1]; beta <- par.beta$par[2]
#Beta: E[X] = alpha/(alpha + beta), Var[X] = alpha*beta/((alpha+beta)^2*(alpha+beta+1))
E.pow.obs <- alpha/(alpha + beta)
CI.E.pow.obs <- alpha/(alpha + beta) + c(-1,1) * qnorm(1-alpha/2) * alpha*beta/((alpha+beta)^2*(alpha+be
#(CI.E.pow.obs <- mean(D$pow.obs.norm) + c(-1,1) * qnorm(1-alpha/2) * sd(D$pow.obs.norm) / dim(D)[1])

#Weibull: E[X] = lambda * gamma(1+1/k); Var[X] = lambda^2*(gamma(1+2/k) - (gamma(1+1/k))^2)
#par.ws30$par[2]*gamma(1+1/par.ws30$par[1]) #mean = lambda * Gamma(1 + 1/k); lambda = scale, k = shape
scale <- par.ws30$par[2]; shape <- par.ws30$par[1]
E.ws30 <- scale*gamma(1+1/shape)
V.ws30 <- scale^2*(gamma(1+2/shape) - (gamma(1+1/shape))^2)
CI.E.ws30 <- E.ws30 + c(-1,1) * qnorm(1-alpha/2) * sqrt(V.ws30) / dim(D)[1] #according to Central Limit
#(CI.E.ws30 <- mean(D$ws30) + c(-1,1) * qnorm(1-alpha/2) * sd(D$ws30) / dim(D)[1])

#Wrapped Cauchy: E[X] = mu, Var[X] = 1 - exp(-gamma)
#relationship between rho and gamma: gamma = -ln(rho)
mu <- wrapped.cauc.par$par[1]; gamma = -log(wrapped.cauc.par$par[2])
(E.wd30 <- mu)
```

```
## [1] -1.574857
```

```
(V.wd30 <- 1 - exp(-gamma)) #or 1 - rho
```

```
## [1] 0.7248408
```

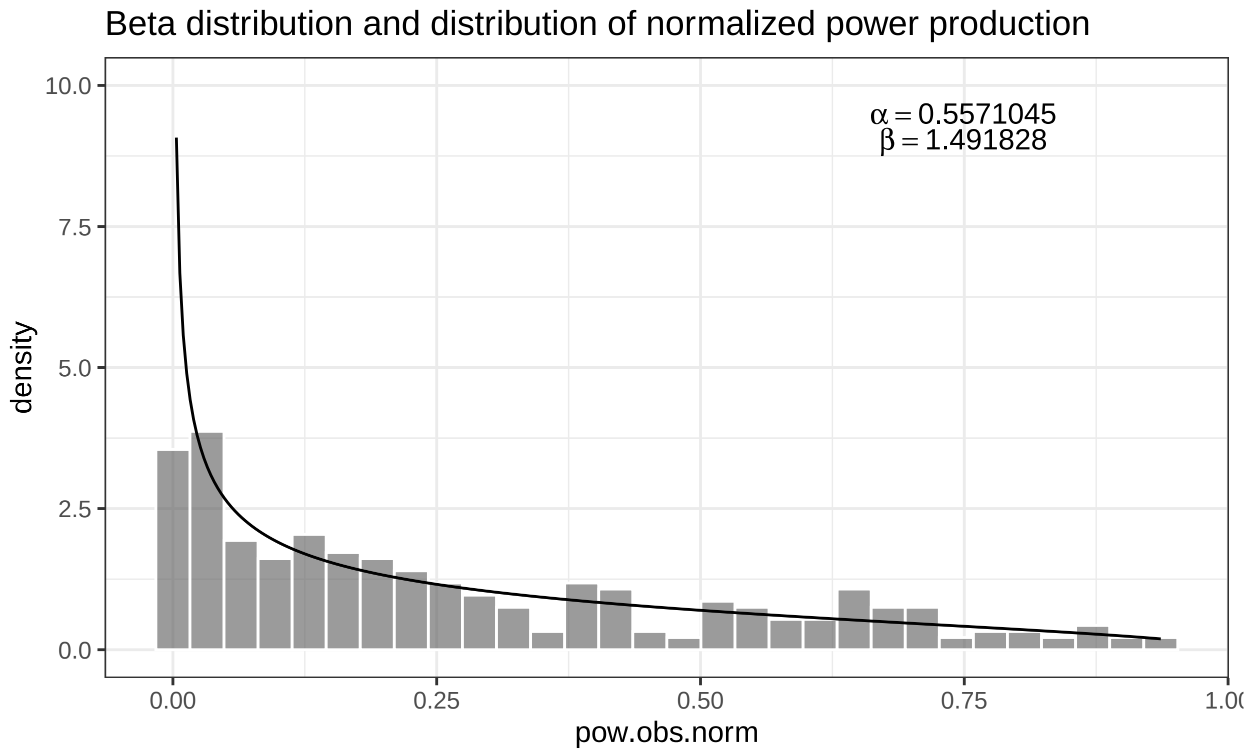
```
(CI.E.wd30 <- E.wd30 + c(-1,1) * qnorm(1-alpha/2) *V.wd30 / dim(D)[1]) #according to Central Limit Theo
```

```
## [1] -1.576335 -1.573380
```

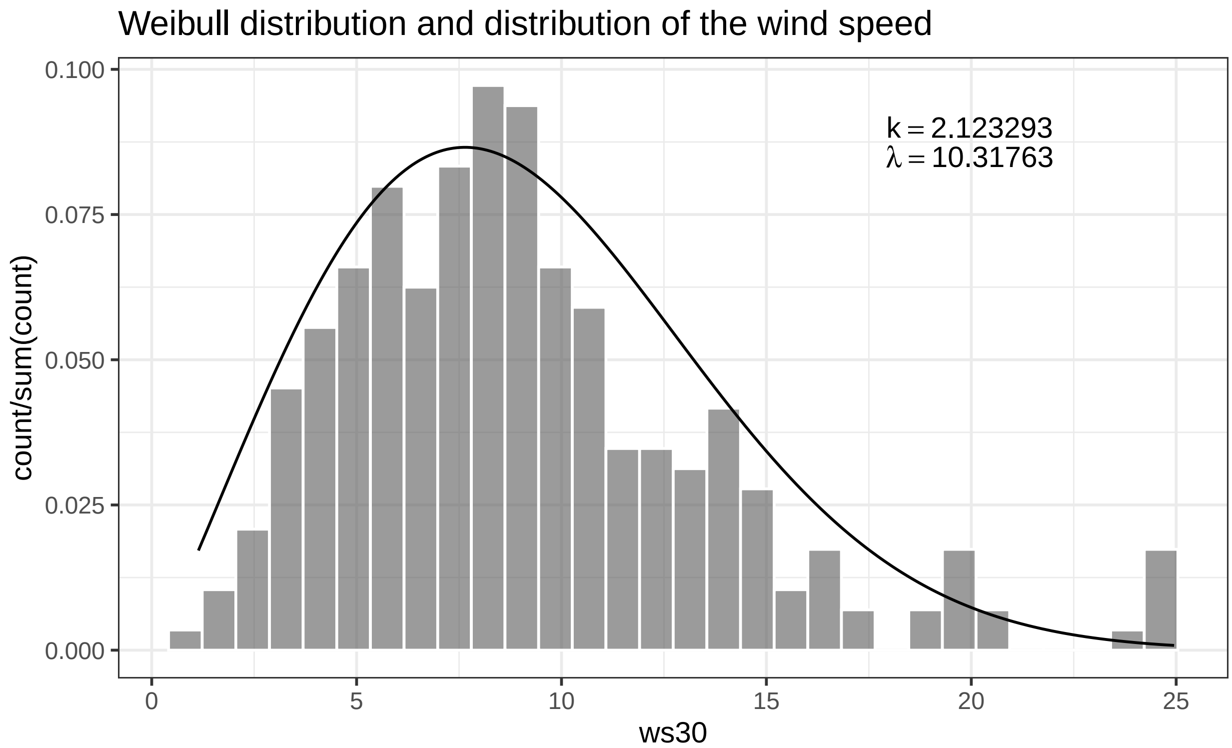
```
#(CI.E.wd30 <- mle.wd30[1] + c(-1,1) * qnorm(1-alpha/2) * sd(D$wd30) / dim(D)[1]) #mean(D$wd30) instead

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,]  0.27177  0.27624  0.27203
## [2,]  9.12849  9.13771  9.14694
## [3,] -1.57634 -1.57486 -1.57338
```

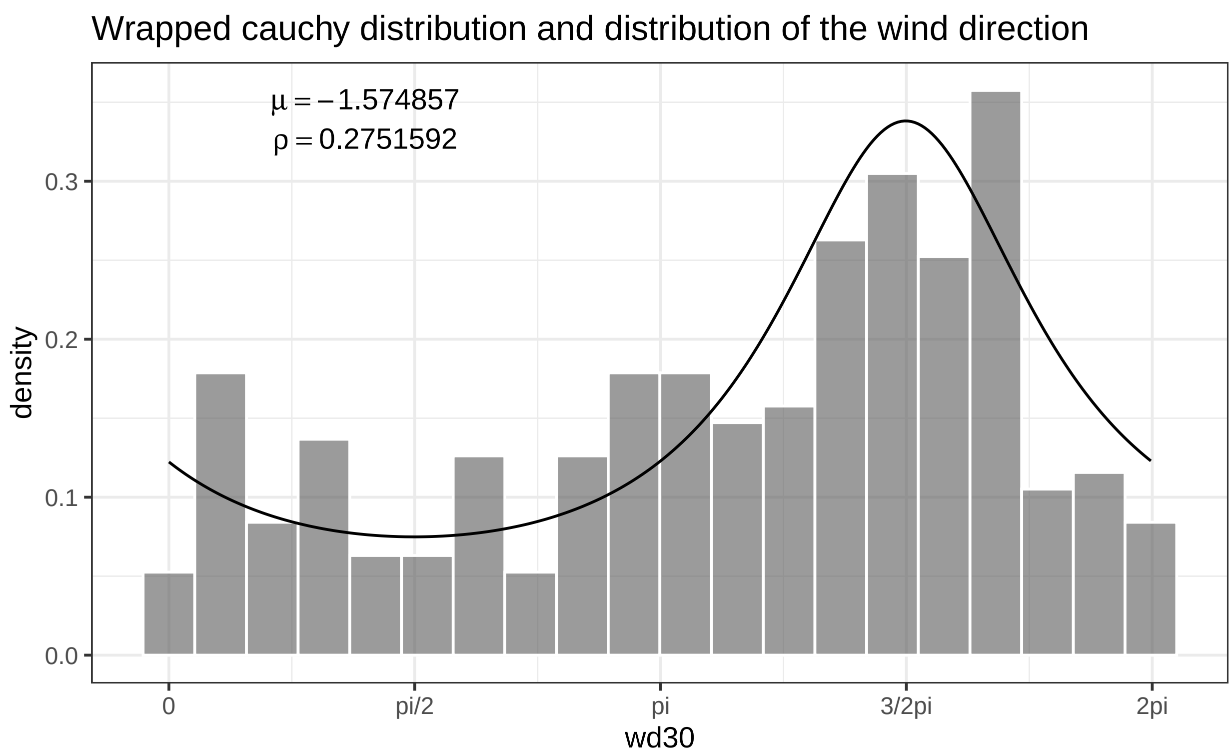
```
par(mfrow=c(1,3))

temp1 <- paste("alpha == ", mle.pow[1]) #par.beta$par[1]
temp2 <- paste("beta == ", mle.pow[2]) #par.beta$par[2]
temp <- c(temp1, temp2)

ggplot(D)+
  geom_histogram(aes(x = pow.obs.norm, y = ..density..), colour='white', alpha=0.6, bins=30)+
  theme_bw()+
  stat_function(fun = dbeta, n = dim(D)[1], args = list(shape1 = mle.pow[1], shape2 = mle.pow[2]))+
  ylim(c(0,10))+
  annotate( "text", x = 4/5*max(D$pow.obs.norm), y = c(9.5, 9.0), label = temp, parse = T  ) +
  ggtitle("Beta distribution and distribution of normalized power production")
```




```
temp1 <- paste("k == ", mle.ws30.weib[1]) #par.ws30$par[1]
temp2 <- paste("lambda == ", mle.ws30.weib[2]) #par.ws30$par[2]
temp <- c(temp1, temp2)

ggplot(D)+
  geom_histogram(aes(x = ws30, y = ..count../sum(..count..)) , colour = "white", alpha=0.6, bins = 30)+
  theme_bw()+
  stat_function(fun = dweibull, n = dim(D)[1], args = list(shape = par.ws30$par[1], scale = par.ws30$pa
  annotate( "text", x = 4/5*max(D$ws30), y = c(0.09,0.085), label = temp, parse = T  ) +
  ggtitle("Weibull distribution and distribution of the wind speed")
```




```
temp1 <- paste("mu ==", mle.wd30[1]) #wrapped.cauc.par$par[1]
temp2 <- paste("rho ==", mle.wd30[2]) #wrapped.cauc.par$par[2]
temp <- c(temp1, temp2)

ggplot(D)+
  theme_bw()+
  geom_histogram(aes(x = wd30, y = ..density..), colour = "white", alpha = 0.6, 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 = dwrappedcauchy, n = dim(D)[1], args = list(mu = wrapped.cauc.par$par[1], rho = 0.3
  annotate( "text", x = 1/5*max(D$wd30), y = c(0.35, 0.325), label = c(temp1,temp2), parse = T  ) +
  ggtitle("Wrapped cauchy distribution and distribution of the wind direction")
```



Wrapped cauchy distribution and distribution of the wind direction
μ = −1.574857
ρ = 0.2751592
0.3
0.2
0.1
0.0
density
0
pi/2
pi
3/2pi
2pi
wd30