Wind speed case study: fitting the Weibull distribution

This function takes input shape parameter k and the data to compute $(1/k)+(1/n)*\sum(\log(x_i))+(1/\alpha)\sum x_i^k \log(x_i) = 0$ where $\alpha = \sum x_i^k$.

```
weibull.shape <- function(k, data)
{
  numer <- colSums(outer(data, k, "^") * log(data))
  denom <- colSums(outer(data, k, "^"))
  numer/denom - 1/k - mean(log(data))
}
```

This function takes input shape parameter k and data to compute k^{th} root of $(1/n)\sum x_i^k$ where n =number of data values.

```
weibull.scale <- function(k, data)
{
  mean(data^k)^(1/k)
}
```

`uniroot` is a built-in function which estimates the roots of a function. Provide function, any arguments needed for function, and a guess of values two values around root. Function values must be opposite signs at lower and upper guess.

Now, we do the data specific commands:

```
Turbine <- read.csv("http://sites.google.com/site/chiharahesterberg/data2/Turbine.csv")
wind <- Turbine$AveSpeed

#alternatively, wind <- subset(Turbine, select = AveSpeed, drop = TRUE)
```

Estimate the shape parameter k:

```
uniroot(weibull.shape, data = wind, lower = 1,upper = 5)
```

```
## $root
## [1] 3.169321
##
## $f.root
## [1] 1.480526e-07
##
## $iter
## [1] 6
##
## $init.it
## [1] NA
##
## $estim.prec
## [1] 6.103516e-05
```

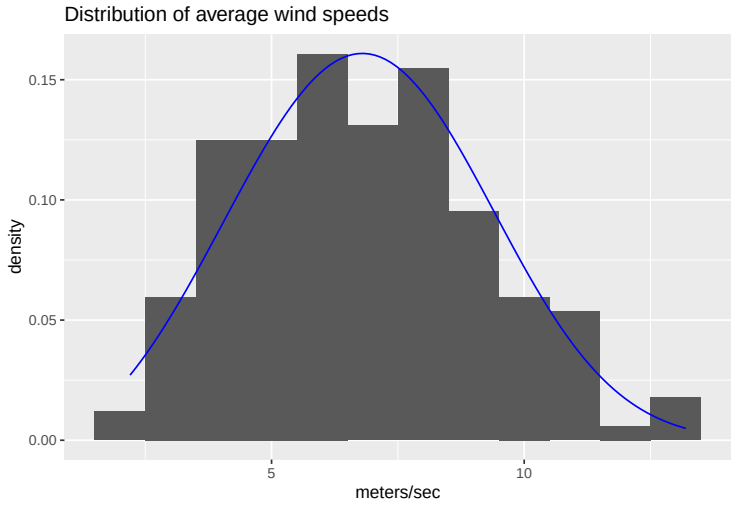
With estimate of shape parameter, now find estimate of scale parameters lambda:

```
weibull.scale(3.169, wind)
```

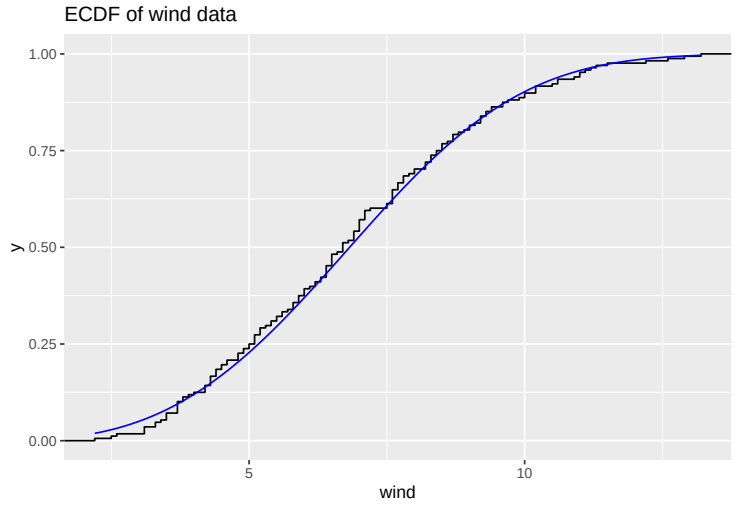
```
## [1] 7.661169
```

Next, plot histogram with density curve overlap. The `prob = TRUE` argument scales histogram to area 1.

```
myfun <- function(x){
  dweibull(x, 3.169, 7.661)
}

df <- data.frame(wind)
ggplot(df, aes(wind)) + geom_histogram(aes(y = stat(density)), bins=12) + labs(title="Distribu
```




```
ggplot(df, aes(wind)) + stat_ecdf() + stat_function(fun=pweibull, args=list(shape=3.169, scale
```



###Goodness-of-fit

Now for the chi-square goodness of fit test Get the deciles of the Weibull distribution with the given parameters and also, the range of the wind data:

```
q <- qweibull(seq(.1, .9, by = .1), 3.169, 7.661)

range(wind)
```

```
## [1]  2.2 13.2
```

Now, encompass the range of wind and get the counts in each sub-interval. The `plot = FALSE` argument suppresses the histogram and just gives statistics:

```
q <- c(0, q, 14)

hist(wind, breaks = q, plot = F)$counts
```

```
##  [1] 17 18 19 13 20 14 17 17 14 19
```

```
#Now store the output
count <- hist(wind,breaks = q, plot = F)$counts

expected <- length(wind)*.1
```

Compute chi-square test statistic:

```
(count-expected)^2/expected
```

```
##  [1] 0.002380952 0.085714286 0.288095238 0.859523810 0.609523810 0.466666667
##  [7] 0.002380952 0.002380952 0.466666667 0.288095238
```

Example 6.13

Simulation comparing two estimators for uniform:

```
Xbar <- numeric(1000)
maxY <- numeric(1000)

#set.seed(100)
for (i in 1:1000)
 {
   x <- runif(25,0,12) #sample n=25 from Unif[0,12]
   Xbar[i] <- 2*mean(x)
   maxY[i] <- 26/25*max(x)
 }

 #mean and standard deviation of the method of moments estimate
 mean(Xbar)
```

```
## [1] 11.99822
```

```
sd(Xbar)
```

```
## [1] 1.351398
```

```
range(Xbar)
```

```
## [1]  7.943627 16.354045
```

```
#mean and sd of estimate from MLE
mean(maxY)
```

```
## [1] 12.01934
```

```
sd(maxY)
```

```
## [1] 0.4402277
```

```
range(maxY)
```

```
## [1]  9.631359 12.479195
```

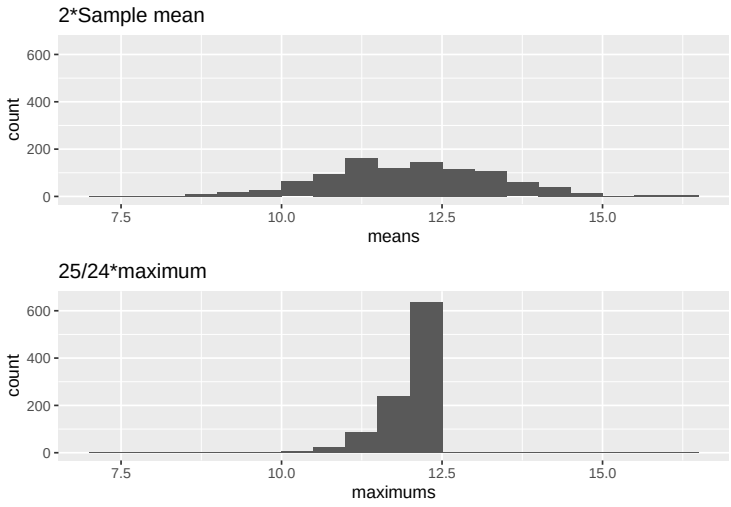
To plot the results:

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
p1 <- ggplot() + geom_histogram(aes(Xbar), breaks=seq(7, 16.5, by =.5))  + ylim(c(0, 650)) + l
p2 <- ggplot() + geom_histogram(aes(maxY), breaks=seq(7, 16.5, by = .5)) + ylim(c(0, 650)) +la
grid.arrange(p1, p2)
```



###Example 6.14 Relative efficiency for the Wind Speed Case Study

```
# Theta = P(wind > 5), nonparametric and parametric estimates
n <- length(wind)
theta1 <- mean(wind > 5)
theta2 <- pweibull(5, 3.169, 7.661, lower.tail = FALSE)
theta1 # 0.75
```

```
## [1] 0.75
```

```
theta2 # 0.7720827
```

```
## [1] 0.7720827
```

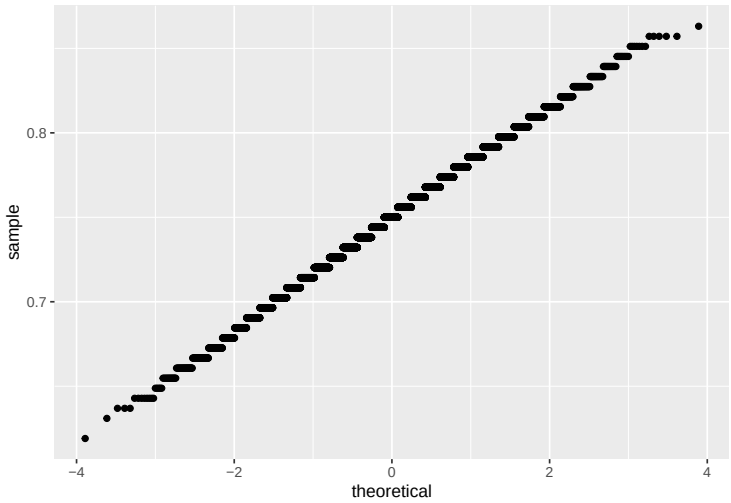
```
eta1 <- quantile(wind, .1)
eta2 <- qweibull(.1, 3.169, 7.661)
eta1 # 3.77
```

```
##  10%
## 3.77
```

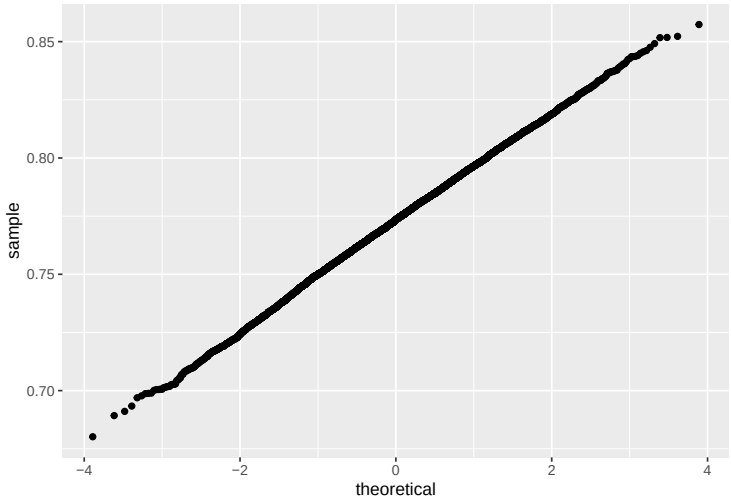
```
eta2 # 3.766038
```

```
## [1] 3.766038
```

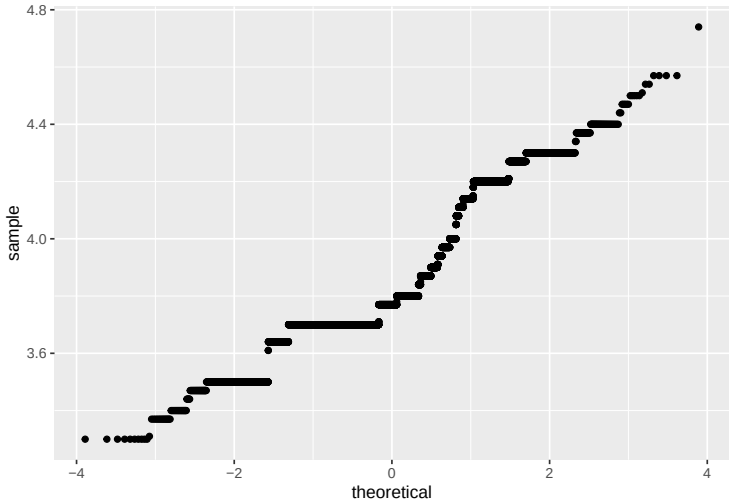
```
# nonparametric bootstrap to find standard errors
set.seed(23)
N <- 10^4
boot.theta1 <- numeric(N)
boot.shape <-  numeric(N)
boot.scale <-  numeric(N)
boot.theta2 <- numeric(N)
boot.eta1 <- numeric(N)
boot.eta2 <- numeric(N)
for (i in 1:N)
{
  boot.wind <- wind[sample(1:n, replace=TRUE)]
  boot.theta1[i] <- mean(boot.wind > 5)
  boot.shape[i] <- uniroot(weibull.shape, data=boot.wind, lower=1,upper=5)$root
  boot.scale[i] <- weibull.scale(boot.shape[i], boot.wind)
  boot.theta2[i] <- pweibull(5, boot.shape[i], boot.scale[i], lower.t = FALSE)
  boot.eta1[i] <- quantile(boot.wind, .1)
  boot.eta2[i] <- qweibull(.1, boot.shape[i], boot.scale[i])
}

ggplot() + stat_qq(aes(sample = boot.theta1)) # discrete, normal
```



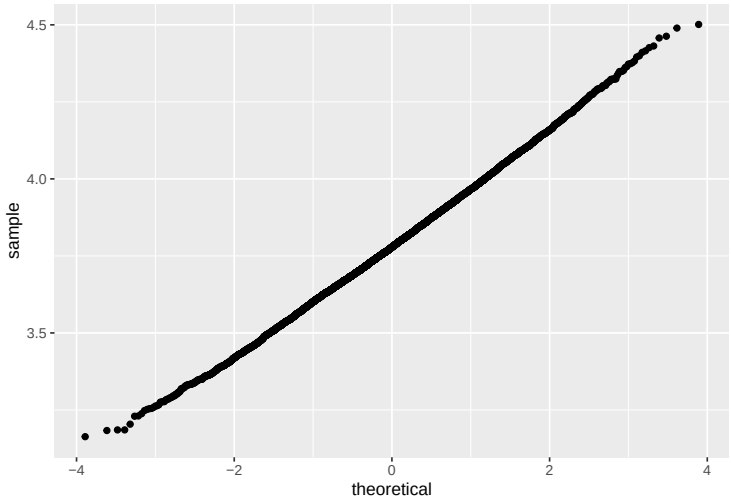

```
ggplot() + stat_qq(aes(sample = boot.theta2)) # continuous, normal
```




```
ggplot() + stat_qq(aes(sample = boot.eta1)) # discrete, irregular, roughly normal
```



```
ggplot() + stat_qq(aes(sample = boot.eta2)) # continuous, very normal
```




```
sd(boot.theta1) # 0.03331767
```

```
## [1] 0.03343642
```

```
sd(boot.theta2) # 0.02342771
```

```
## [1] 0.02352034
```

```
var(boot.theta1) / var(boot.theta2) # 2.022504
```

```
## [1] 2.020935
```

```
# Formula standard error, for comparison
sqrt(theta1 * (1-theta1) / n) # standard error = 0.03340766
```

```
## [1] 0.03340766
```

```
sd(boot.eta1) # 0.2161054
```

```
## [1] 0.2187176
```

```
sd(boot.eta2) # 0.1839509
```

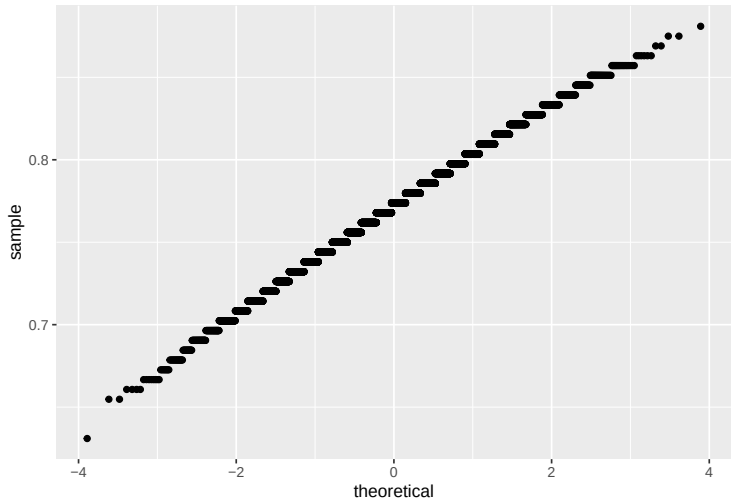
```
## [1] 0.184752
```

```
var(boot.eta1) / var(boot.eta2) # 1.380154
```

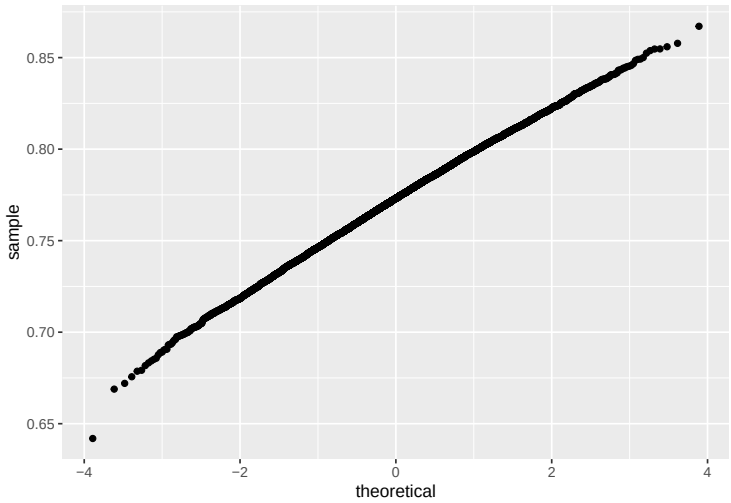
```
## [1] 1.401486
```

```
# parametric bootstrap to find standard errors
set.seed(23)
pboot.theta1 <- numeric(N)
pboot.shape <-  numeric(N)
pboot.scale <-  numeric(N)
```

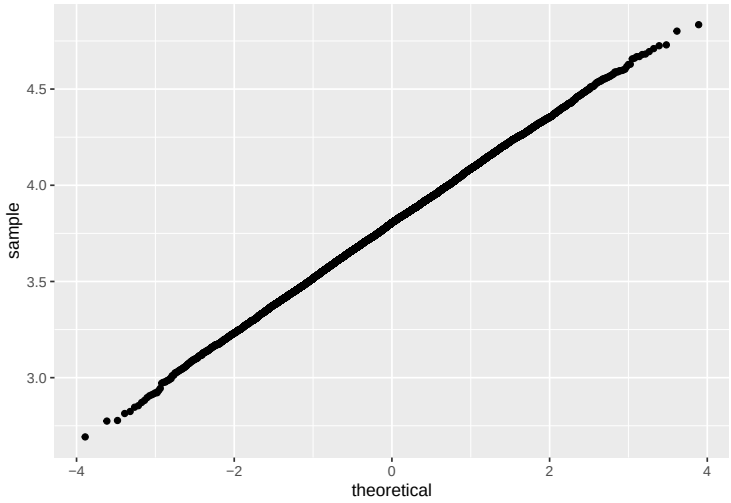
```
pboot.theta2 <- numeric(N)
pboot.eta1 <- numeric(N)
pboot.eta2 <- numeric(N)
for (i in 1:N)
{
  boot.wind <- rweibull(n, 3.169, 7.661)
  pboot.theta1[i] <- mean(boot.wind > 5)
  pboot.shape[i] <- uniroot(weibull.shape, data=boot.wind, lower=1,upper=5)$root
  pboot.scale[i] <- weibull.scale(pboot.shape[i], boot.wind)
  pboot.theta2[i] <- pweibull(5, pboot.shape[i], pboot.scale[i], lower.t=FALSE)
  pboot.eta1[i] <- quantile(boot.wind, .1)
  pboot.eta2[i] <- qweibull(.1, pboot.shape[i], pboot.scale[i])
}
etaRange <- range(boot.eta1, boot.eta2, pboot.eta1, pboot.eta2)

ggplot() + stat_qq(aes(sample = pboot.theta1)) # discrete, slight neg skewness
```



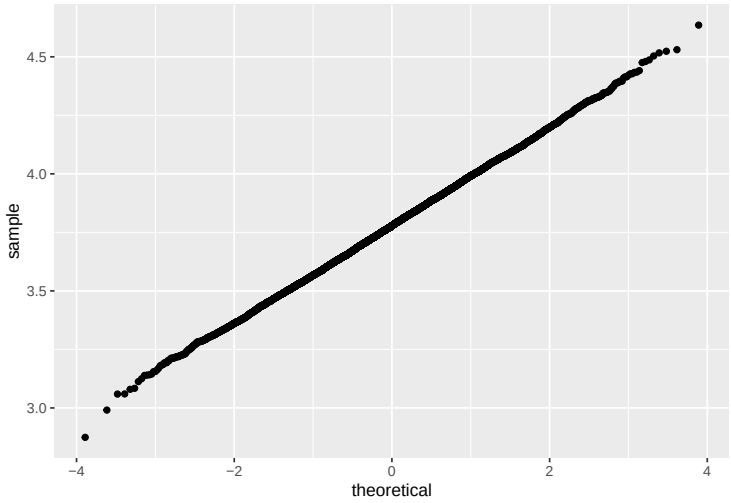

```
ggplot() + stat_qq(aes(sample = pboot.theta2)) # continuous, normal
```




```
ggplot() + stat_qq(aes(sample = pboot.eta1)) # very normal, continuous
```




```
ggplot() + stat_qq(aes(sample = pboot.eta2)) # very normal, continuous
```



```r
sd(pboot.theta1) # 0.03198990
```

```
## [1] 0.0319899
```

```r
sd(pboot.theta2) # 0.02604409
```

```
## [1] 0.02604409
```

```r
var(pboot.theta1) / var(pboot.theta2) # 1.508716
```

```
## [1] 1.508716
```

```r
# Similar to the nonparametric bootstrap, though relative efficiency differs

sd(pboot.eta1) # 0.2823103
```

```
## [1] 0.2823103
```

```r
sd(pboot.eta2) # 0.2100147
```

```
## [1] 0.2100147
```

```r
var(pboot.eta1) / var(pboot.eta2) # 1.806982
```

```
## [1] 1.806982
```