***Exam***

***Fu Linjing 831028***

***Assignment 1: Likelihood optimization***

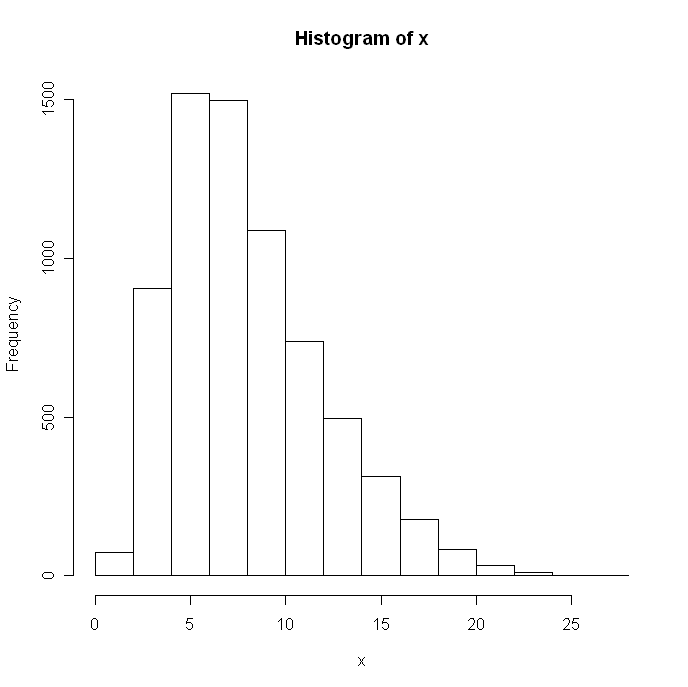
The data set ***wind.xls*** represents wind speed records for twelve Meridian sites distributed around the New Zealand. It has long been known that the wind speed matches well with the Weibull distribution. Your task is to make some inference about the parameter values for the city **Otago**

1. Load necessary information concerning Otago city to R. Plot the histogram of the wind speed and comment whether it looks like Weibull (you may also compare it with the shape of the related Rayleigh distribution)

Code:

wind<-read.table("z:/wind.txt",header=TRUE)

x<-wind$Otago



**\*0.5p**

1. Write your own function as a parameter of the data that finds the maximum log-likelihood estimates of the parameters *shape* and *scale* of the Weibull distribution by using an optimization method such as BFGS or CG. Compute the parameter estimates (if you get warnings, ignore them). What kind of problems may one encounter when these methods are used for the search of the maximum log-likelihood estimates? What other methods can you see as alternatives that do not have these problems?

z<-c()

s<-c()

myll<-function(z)

{k<-z[1]

s<-z[**1**] **\*HERE IS A MISTAKE**

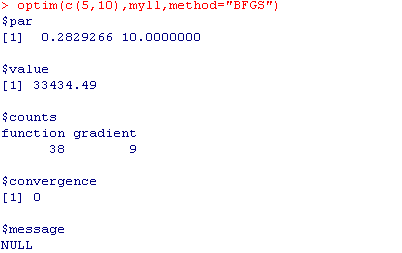
return(-sum(dweibull(x,k,scale=s,log=TRUE)))

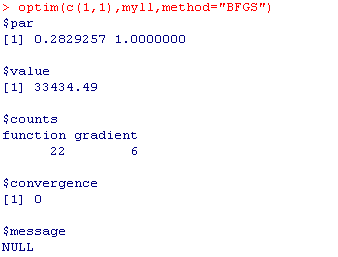
}

optim(c(1,1),myll,method="BFGS")

or:

optim(c(1,1),myll,method="CG")





I found whatever the initial scale value I choose; the function cannot optimize the scale. Because the weibull distribution is not continually and the constraint of shape and scale both greater than 0

So ,I loop the scale value with the different value, find the optimal shape value,and compare the maximum log-likelihood.

Finally , I get the best scale value=9 which provide the optimal object value: **\*1p Well, it is a “trick” that helped you to get a solution, but “9” is not optimal**

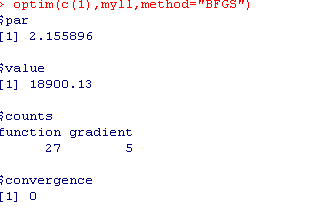
myll<-function(k)

{

-sum(dweibull(x,k,scale=9,log=TRUE))

}

final<-optim(c(1),myll,method=”BFGS”)



1. Modify your function in such way that it returns only *shape* parameter. Consider this function as an estimator of *shape* and generate 100 bootstrap estimates of this parameter. Produce an appropriate plot and make comments. Compute 95% confidence bounds (**using percentile method only!**) for the parameter *shape* and answer whether the data can in principle come from Rayleigh distribution.

wll<-function(data,index) **\*1.5p**

{

myll<-function(k)

{

-sum(dweibull(data[index],k,scale=9,log=TRUE))

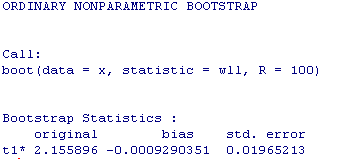
}

final<-optim(c(1),myll,method="BFGS")

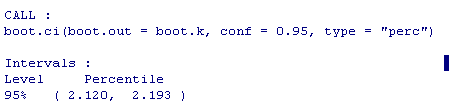
return(final$par)

}

boot.k<-boot(x,wll,R=100)



boot.ci(boot.k,conf=0.95,type="perc")



**\*0.5p**

Because from the previous function, I found the optimal object is with regard to:

shape=2,16 and scale=9, the k is around 2,so I think it is in principle a Rayleigh distribution. **\*0p “2” is outside confidence limits**

***Assignment 2: Computing an integral***

Consider the distribution with density

1. Write your own function that can generate sample of size *n* from the distribution above using the acceptance-rejection method with uniform distribution as the majorizing density. The function should also compute what percent ***R***of the totally generated random numbers was rejected in this method. What was the value of the scaling constant *c* you used in deriving the majorizing function and why you have chosen this value?

Because c must be greater or equal target density/majorizing density

C>=/dunif(x,0,1)

So c should be equal to pi/2 **\*1p**

1. Generate a sample of size 1000 by using your function and plot the histogram. Does the sample look like it should?

z<-c()

y<-c()

k<-c()

u<-c()

p<-c()

count<-c()

myexp1<-function(z)

{

pi/2\*sin(pi\*z)}

myfun<-function(n,c)

{

count<-0

for (i in 1:n)

{

for (k in 1:10000)

{

u<-runif(1)

y<-runif(1)

alpha<-myexp1(y)/c

if(u<=alpha)

{p[i]<-y

break

}

}

count<-count+k

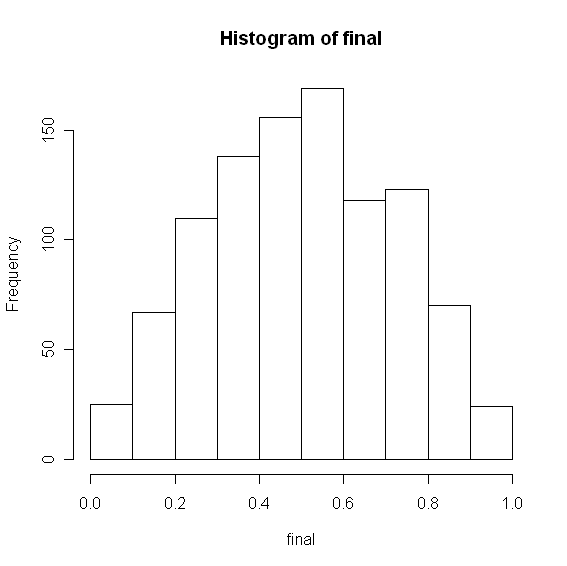
}

print(count)

p

}

**\*1.5p**

**\*0.5p**

The random number is between (0,1),and the most frequency number is in the number 0,5, and the wave is quite similar to the sine wave. So I think the histogram looks like it should. **\*0.5p**

1. Print out the computed rejection rate ***R***. Find out without using computer what would be the rejection rate ***R*** if the sample size would be very large (i.e., the expected value of ***R***).



The average rejection rate is (1-1000/1547)\*100%=35%**\*0.5p**

If the sample size is very large, the rejection rate should be the same around 0,35 **\*0p Why?**

1. Use the obtained sample to compute the value of the integral to derive the value of the integral by applying the importance sampling. Why using the introduced above as importance function is better than using the density of the uniform distribution?

=g(x)\*p(x)

use the above f(x) as p(x),so g(x)=2/(pi\*(5+x^2))

I can estimate the integral=E(2/(pi\*(5+x^2))) x is the random generate number from above f(x)

**\*1p**

=0,1202549

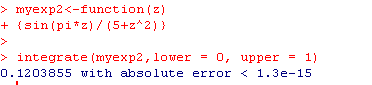
Using the f(x) as importance function is better than using uniform distribution function(mean(sin(pi\*final)/(5+final^2))

[1] 0.1469452)

That is because both of the two methods should use the same generate random x, but if use the uniform distribution the bias is bigger than use f(x):

mean(sin(pi\*final)/(5+final^2)) has large bias than mean(2/(pi\*(5+x^2))) ,because the complex. **\*0.5p Not totally correct**

1. Compute the same integral by using the function *integrate* in R and comment on the result.



The result is quite similar, that means the generate number I made is quite good to meet the given density function.

**\*1p**

## Submission procedure

Submit your solutions via It’s learning at latest by 24 May, 13:00