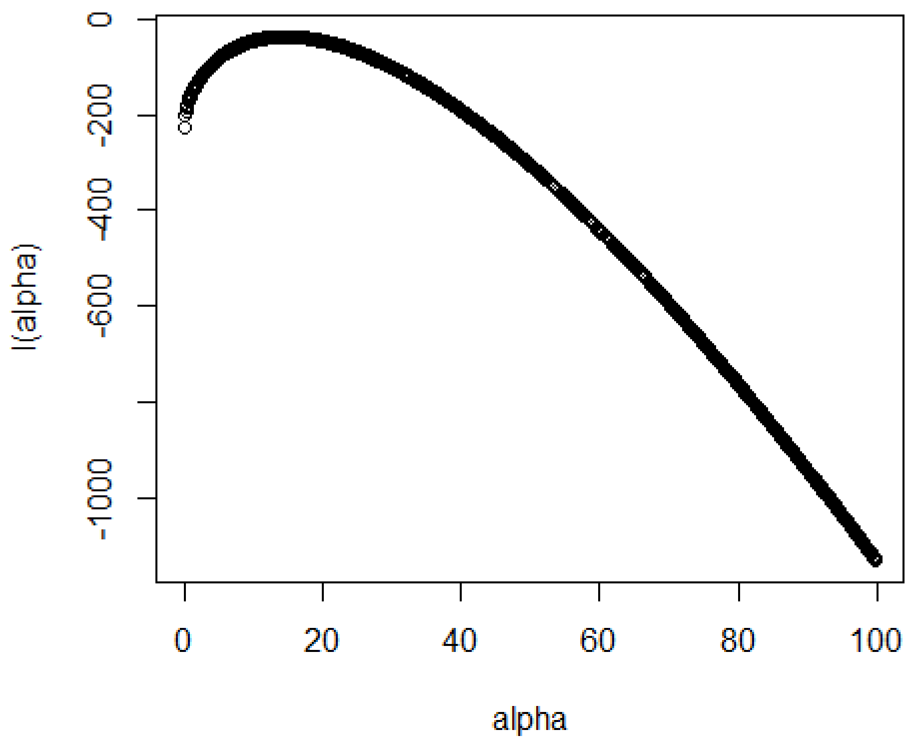
Exercises I3

Exercise 3.1

i) With the R script

```
o = c(5.1, 7.4, 10.9, 21.3, 12.3, 15.4, 25.4, 18.2, 17.4,
22.5)
loglikGamma = c()
alphas = seq(from = 0.01, to = 100, by = 0.1)



for (alpha_i in alphas) {
  loglikGamma =
append(loglikGamma,sum(log(dgamma(o,alpha_i))))
}

plot(alphas,loglikGamma,xlab = "alpha",ylab = "l(alpha)")
```

we plot the log likelihood of the observed sample:



ii) With the R script

```
l_alpha = function(alphas) {
  loglikGamma = c()
  for (alpha_i in alphas) {
    loglikGamma = append(loglikGamma, sum(log(dgamma(o,
alpha_i))))
  }
  return(loglikGamma)
}

derivative_l_alpha = function(alphas) {
  return(numDeriv::grad(l_alpha, alphas))
}
```

```
observed_info = function(alphas) {
  return(-numDeriv::grad(derivative_l_alpha, alphas))
}
newton_raphson = function(starting_value, n_iterations) {
  alfa = starting_value
  for (i in 0:n_iterations) {
    print(alfa)
    alfa = alfa + (derivative_l_alpha(alfa) /
observed_info(alfa))
  }
  return(alfa)
}
interval = c(0.01, 100)
uniroot_result = uniroot(derivative_l_alpha, interval =
interval)$root
newton_raphson_result = newton_raphson(1, 100)
```

and we get 14.54997 using `uniroot` and 14.54997 using my implementation of Newton-Raphson with 100 iterations.

iii) We can use the `observed_info` function we defined previously to get $J(\hat{\alpha}) = 0.711444$

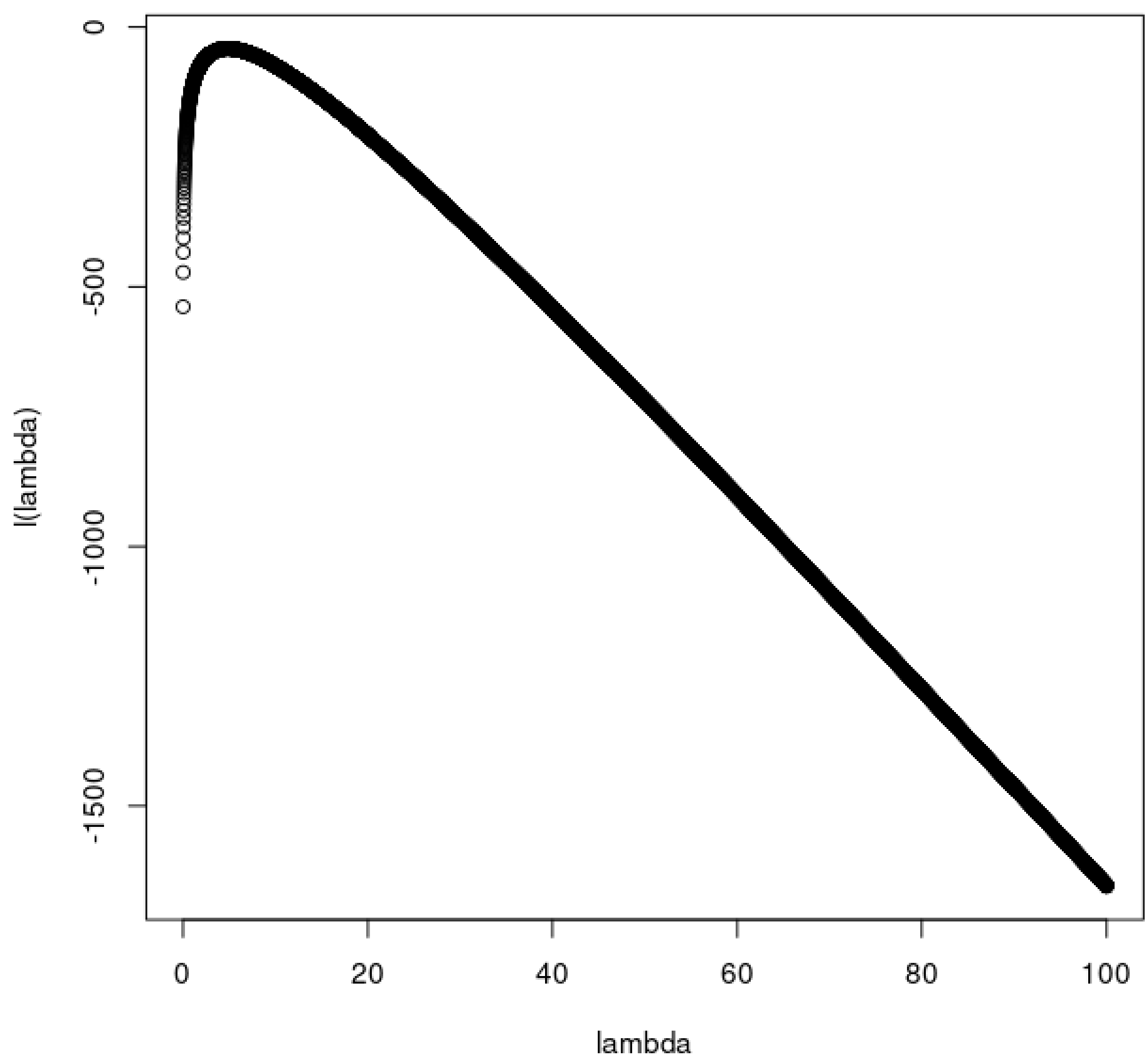
iv)...

Exercise 3.2

i) With:

```
o = c(7, 4, 2, 4, 3, 2, 5, 10, 7, 7, 3, 5, 5, 5, 4, 3, 7, 3,
6, 4)
samples=o
l_lambda = function(lambda) {
  loglikPoi = c()
  for (lambda_i in lambda) {
    loglikPoi = append(loglikPoi, sum(log(dpois(samples,
lambda_i))))
  }
  return(loglikPoi)
}

lambdas = seq(from = 0.01, to = 100, by = 0.01)

plot(lambdas,l_lambda(lambdas),xlab = "lambda",ylab =
"l(lambda)")
```

we plot:



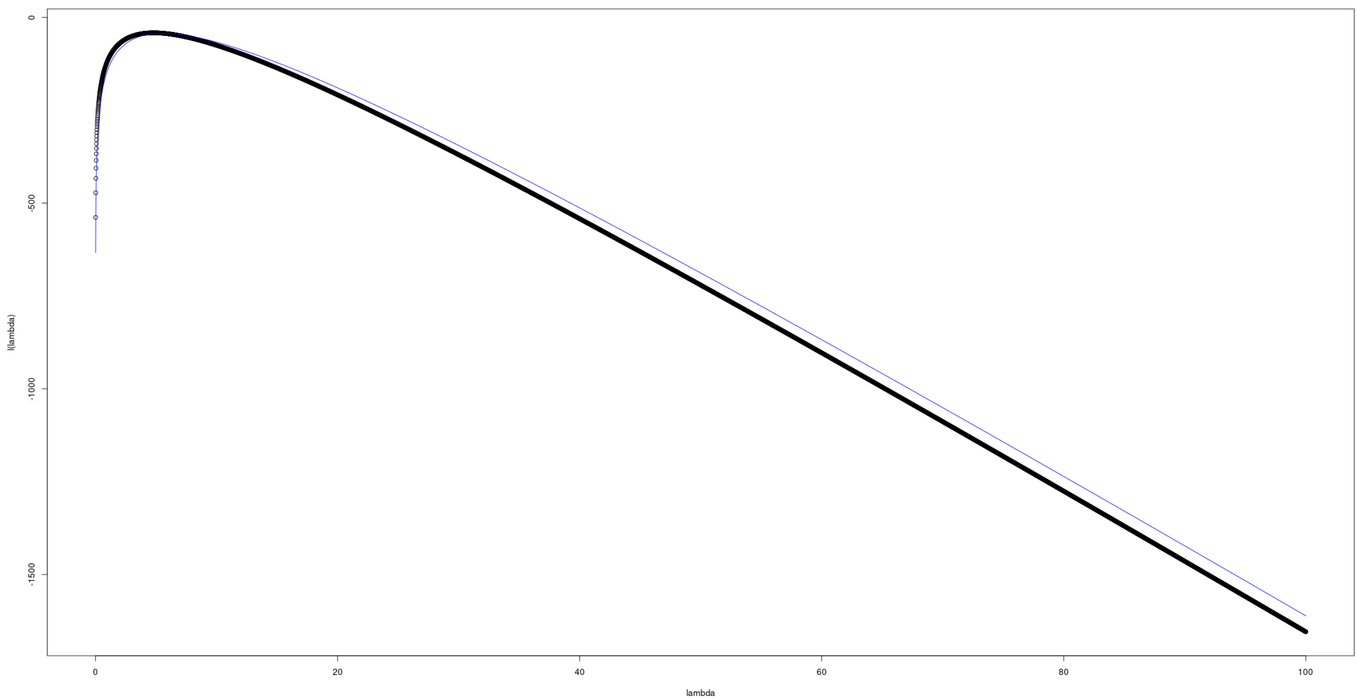
and using again `uniroot` we find that $\hat{\alpha} = 4.80001$

iii) with the script

```
o = c(7, 4, 2, 4, 3, 2, 5, 10, 7, 7, 3, 5, 5, 5, 4, 3, 7, 3,
6, 4)
samples=o

l_lambda = function(lambda) {
  loglikPoi = c()
```

```
  for (lambda_i in lambda) {
    loglikPoi = append(loglikPoi, sum(log(dpois(samples,
lambda_i))))
  }
  return(loglikPoi)
}

lambdas = seq(from = 0.01, to = 100, by = 0.01)

plot(lambdas,l_lambda(lambdas),xlab = "lambda",ylab =
"l(lambda)")

derivative_l_lambda = function(lambda) {
  return(numDeriv::grad(l_lambda, lambda))
}

interval = c(0.01, 100)
uniroot_result = uniroot(derivative_l_lambda, interval =
interval)$root

resampled_o = sample(o,20,replace = TRUE)
samples=resampled_o
lines(lambdas,l_lambda(lambdas),col="blue")
```

we obtain the following plot:

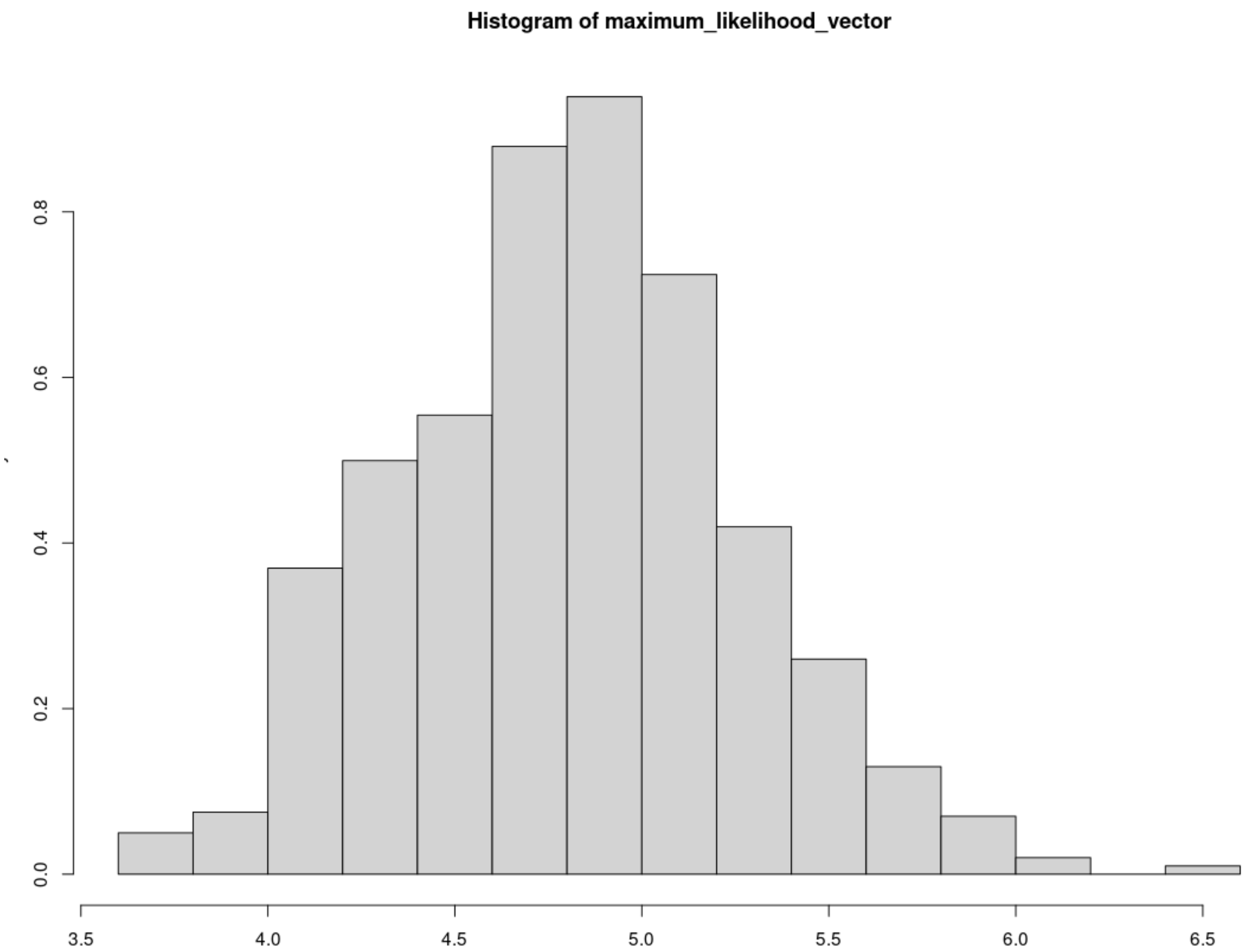


where the blue line is the one obtained drawing 20 numbers with replacement from the sample.

iii) With the script:

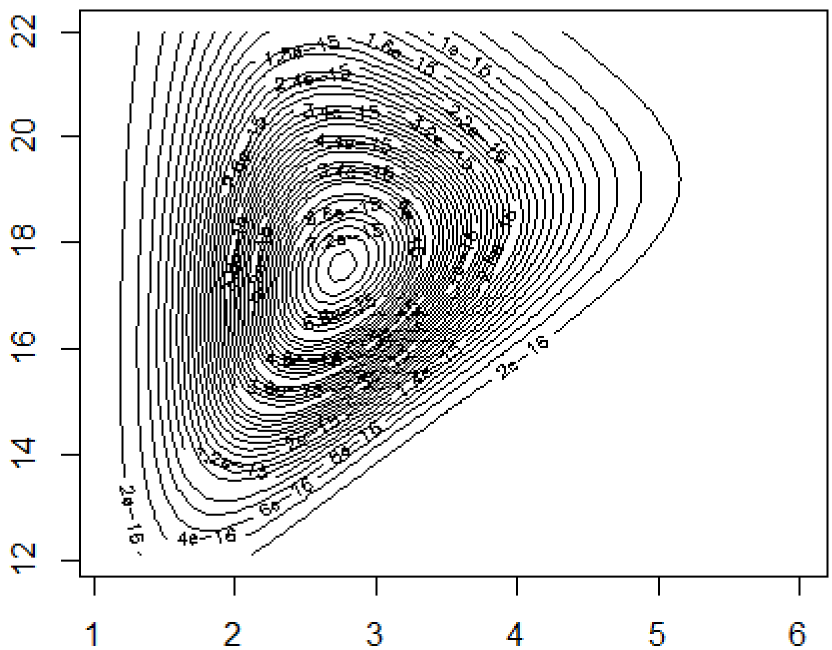
```
maximum_likelihood_vector = c()
for(i in (0:1000)){
  resampled_o = sample(o,20,replace = TRUE)
  ML = uniroot(function(lambda)
derivative_l_lambda(lambda,resampled_o), interval =
interval)$root

maximum_likelihood_vector=append(maximum_likelihood_vector,ML
)
}

hist(maximum_likelihood_vector,freq = FALSE)
```

we get

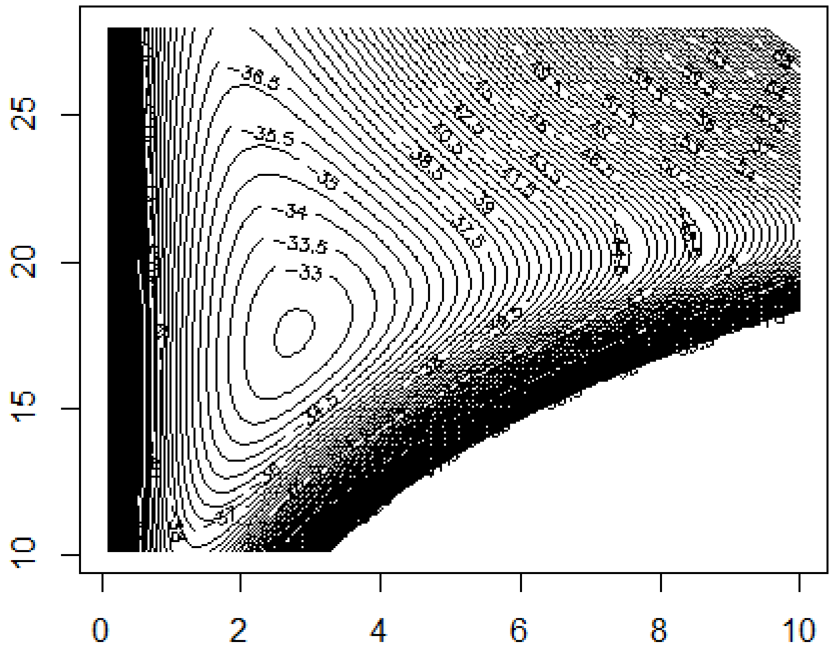


Exercise 3.3

Contours of likelihood



Contours of loglikelihood:



computed with ```

like_wei<- function(sample,alpha,beta){

return(prod((sapply(sample, dweibull,shape=alpha,scale=beta))))

}

log_like_wei <- function(sample,alpha,beta){

return(sum(log(sapply(sample, dweibull,shape=alpha,scale=beta))))

}```

ii) Implementing multivariate Newton-Raphson:

```
llikeF<-function(v) {
  return(sum(log(dweibull(observed_samples,v[1],v[2]))))
}
hessianLlikeF <- function(v) {
  return(numDeriv::hessian(llikeF, v))
}
```

```
gradientLlikeF<- function(v) {
  return(numDeriv::jacobian(llikeF, v))
}
root = c(2,17)
for (i in 0:1000){
  gradient=gradientLlikeF(root)
  inverseHess=solve(hessianLlikeF(root))
  root = root - inverseHess%*%t(gradient)
}

print(root)
```

we get the optimum $\alpha = 2.757154, \beta = 17.556447$
iii) Using `optim`:

```
result = optim(par = c(2.5,10),fn = function(parameters) -
log_like_wei(observed_samples,parameters[1],parameters[2]),me
thod = "BFGS")
```

we get the optimum $\alpha = 2.757227, \beta = 17.557161$
which is similar to the Newton-Raphson result.