Homework 3

Ginevra Carbone

LEC

Exercise 1

Compute the bootstrap-based confidence interval for the `score` dataset using the studentized method.

```
score <- read.table("student_score.txt", header = TRUE)
psi_fun <- function(data) {
  eig <- eigen(cor(data))$values
  return(max(eig) / sum(eig))
}

psi_obs <- psi_fun(score)

# bootstrap
n <- nrow(score); B <- 10^4
s_vect <- rep(0, B)
for(i in 1:B) {
  ind <- sample(1:n, n, replace = TRUE)
  s_vect[i] <- psi_fun(score[ind,])
}
SE_boot <- sd(s_vect)

# bootstrap for z_vect
n <- nrow(score)
new_s_vect <- rep(0, n)
for(i in 1:B) {
  ind <- sample(1:n, n, replace = TRUE)
  new_s_vect[i] <- psi_fun(score[ind,])
}
SE_z_vect <- sd(new_s_vect)
z_vect <- (s_vect - psi_obs)/SE_z_vect

student_ci <- psi_obs - SE_boot * quantile(z_vect, prob=c(0.975, 0.025))
student_ci
```

```
##     97.5%      2.5%
## 0.5694572 0.8730492
```

Exercise 2

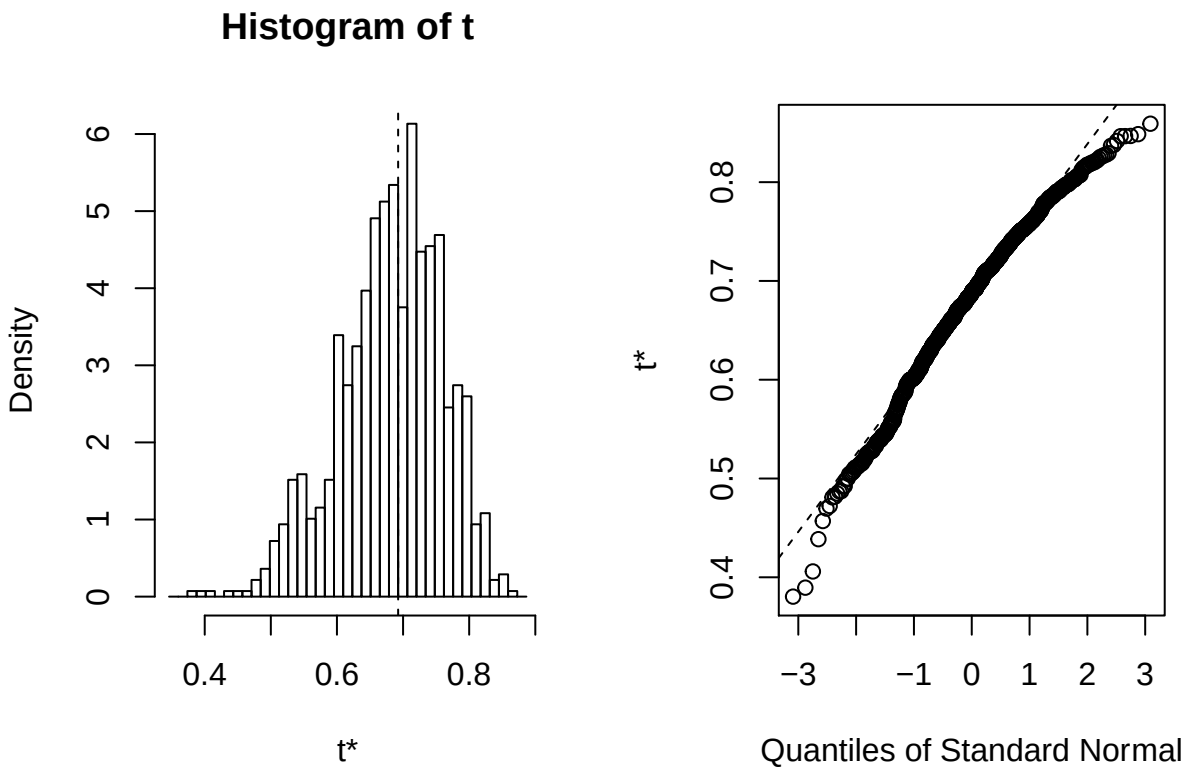
Compute bootstrap-based confidence intervals for the `score` dataset using the `boot` package.

```
library(boot)

score <- read.table("student_score.txt", header = TRUE)

psi_fun <- function(data, ind) {
```

```
  d <- data[ind,] # sample selection
  eig <- eigen(cor(d))$values
  return(max(eig) / sum(eig))
}

# bootstrapping with 1000 replications
results <- boot(data = score, statistic = psi_fun, R = 1000)

# view results
results
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = score, statistic = psi_fun, R = 1000)
##
##
## Bootstrap Statistics :
##      original     bias    std. error
## t1* 0.6925353 -0.0109575   0.0785027
```

```
plot(results)
```




```
# get 95% confidence interval
boot.ci(results, type = c("perc","basic"))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
```

```
##
## CALL :
## boot.ci(boot.out = results, type = c("perc", "basic"))
##
## Intervals :
## Level      Basic              Percentile
## 95%   ( 0.5694,  0.8724 )   ( 0.5127,  0.8156 )
## Calculations and Intervals on Original Scale
```

```r
psi_fun <- function(data, ind) {
  d <- data[ind,] # sample selection
  eig <- eigen(cor(d))$values
  out <- max(eig)/sum(eig)
  # return(c(out, var(out)))
  return(out)
}

psi_fun_var <- function(data, ind){
  d <- data[ind,]
  boot <- boot(data = score, statistic = psi_fun, R = 50)
  eig <- eigen(cor(d))$values
  out <- max(eig)/sum(eig)
  return(c(out, var(boot$t)))
}

resultsVar <- boot(data = score, statistic = psi_fun_var, R = 100)

boot.ci(resultsVar, type = "stud")
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 100 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = resultsVar, type = "stud")
##
## Intervals :
## Level    Studentized
## 95%   ( 0.5567,  0.9054 )
## Calculations and Intervals on Original Scale
## Some studentized intervals may be unstable
```

LAB

Exercise 1

Compute analitically $J(\gamma,\gamma;\gamma), J(\gamma,\beta;y), J(\beta,\beta;y)$.

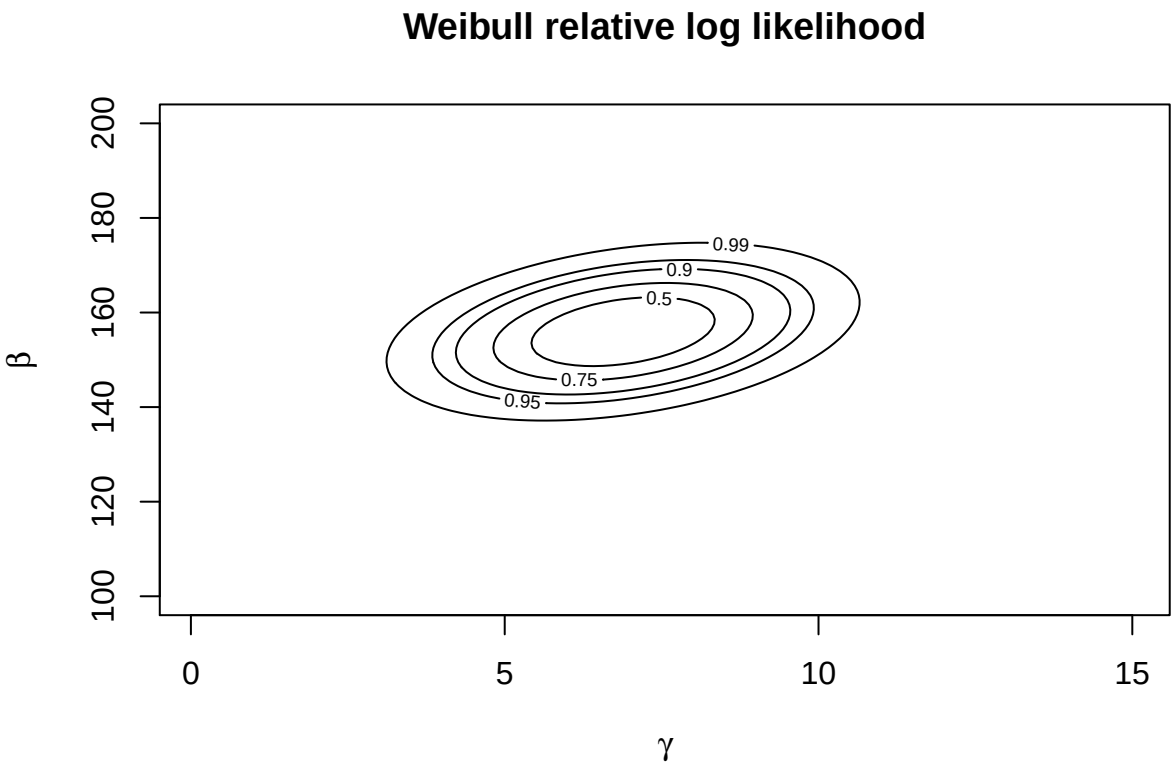
$$J(\theta;y) = \begin{pmatrix} \frac{n}{\gamma^2} + \sum_{i=1}^{n} \left(\frac{y_i}{\beta}\right)^{\gamma} \left(log\frac{y_i}{\beta}\right)^2 & \frac{n}{\beta} - \sum_{i=1}^{n} \frac{y_i^{\gamma}}{\beta^{\gamma+1}}\left(1+\gamma\, log\frac{y_i}{\beta}\right) \\ \frac{n}{\beta} - \sum_{i=1}^{n} \frac{y_i^{\gamma}}{\beta^{\gamma+1}}\left(1+\gamma\, log\frac{y_i}{\beta}\right) & -\frac{n\gamma}{\beta^2} + \gamma\frac{\gamma+1}{\beta^{\gamma+2}} \sum_{i=1}^{n} y_i^{\gamma} \end{pmatrix}$$

Exercise 2

Produce the contour plot for the quadratic approximation of the log-likelihood, based on the Taylor series:

$$l(\theta) - l(\hat{\theta}) \approx -\frac{1}{2}(\theta - \hat{\theta})^T J(\hat{\theta})(\theta - \hat{\theta})$$

```
# observations
y <- c(155.9, 200.2, 143.8, 150.1,152.1, 142.2, 147, 146, 146,
170.3, 148, 140, 118, 144, 97)
n <- length(y)

# thetahat
gammahat<-uniroot(function(x) n/x+sum(log(y))-n* sum(y^x*log(y))/sum(y^x), c(1e-5,15))$root
betahat<- mean(y^gammahat)^(1/gammahat)
weib.y.mle<-c(gammahat,betahat) # thetahat

# observed information matrix
jhat<-matrix(NA,nrow=2,ncol=2)
jhat[1,1]<-n/gammahat^2+sum((y/betahat)^gammahat* (log(y/betahat))^2)
jhat[1,2]<-jhat[2,1]<- n/betahat-sum(y^gammahat/betahat^(gammahat+1)*
(gammahat*log(y/betahat)+1))
jhat[2,2]<- -n*gammahat/betahat^2+gammahat*(gammahat+1)/
betahat^(gammahat+2)*sum(y^gammahat)
jhat <- as.matrix(jhat)

# parameters grid
gamma <- seq(0.1, 15, length=100)
beta <- seq(100, 200, length=100)
parvalues <- expand.grid(gamma,beta)

# quadratic approximation function
approximation <- function(theta){
diff <- theta - weib.y.mle
return(-0.5*t(diff)%*%jhat%*%diff)
}

quadratic_approx <- apply(parvalues, 1, approximation)
quadratic_approx <- matrix(quadratic_approx,nrow=length(gamma), ncol=length(beta), byrow=F )

#contour plot
conf.levels <- c(0,0.5,0.75,0.9,0.95,0.99)
contour(gamma, beta, quadratic_approx,
        levels=-qchisq(conf.levels, 2)/2,
        xlab=expression(gamma),
        labels=as.character(conf.levels),
        ylab=expression(beta))
title('Weibull relative log likelihood')
```



Exercise 3

Use `nlm` for computing the variance for the estimator $\hat{w} = (log(\hat{\gamma}), log(\hat{\beta}))$ and `optimHess` for the variance of $\hat{\theta} = (\hat{\gamma}, \hat{\beta})$.

```
y <- c(155.9, 200.2, 143.8, 150.1,152.1, 142.2, 147, 146, 146,
170.3, 148, 140, 118, 144, 97)
n <- length(y)

log_lik_weibull <- function(data, param){
  -sum(dweibull(data, shape = param[1], scale = param[2], log = TRUE))
}

omega <- function(theta) log(theta)
theta <- function(omega) exp(omega)
log_lik_weibull_rep <- function(data, param) log_lik_weibull(data, theta(param))

weib.y.nlm<-nlm(log_lik_weibull_rep, c(0,0), hessian=T,data=y)

# variance of omegahat
hess <- weib.y.nlm$hessian
diag(solve(hess))
```

```
## [1] 0.032473346 0.001582124
```

```
# variance of thetahat
hess <- optimHess(theta(weib.y.nlm$estimate), log_lik_weibull, data=y)
diag(solve(hess))
```

```
## [1]  1.543241 38.406119
```

Exercise 4

The Wald confidence interval with level $1-\alpha$ is defined as:

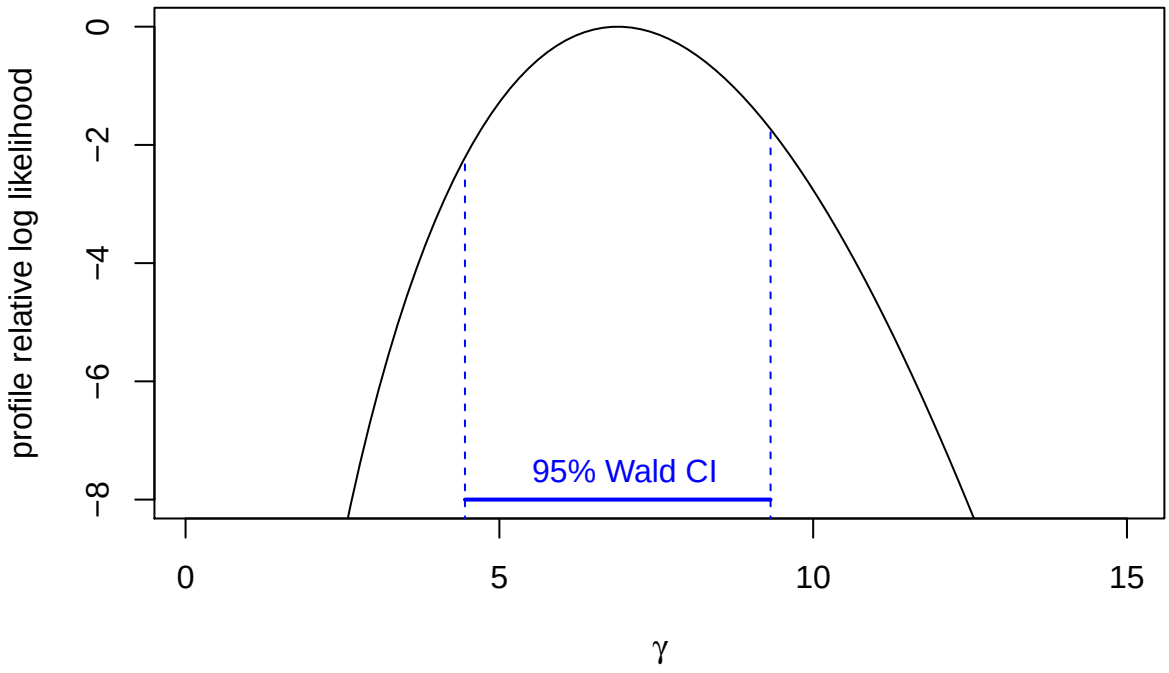
$$\hat{\gamma} \pm z_{1-\alpha/2}\, j_P(\hat{\gamma})^{-1/2}$$

Compute the Wald confidence interval of level 0.95 and plot the results.

```
weib.y.mle<-optim(c(1,1), fn=log_lik_weibull,hessian=T, method='L-BFGS-B',lower=rep(1e-7,2), upper=rep(

conf.level<-0.95
# SE for the estimate from the observed information matrix
weib.y.se <- sqrt(diag(solve(weib.y.mle$hessian)))
# Wald confidence interval for gamma
wald.ci1 <- weib.y.mle$par[1]+c(-1,1)*qnorm(1-(1-conf.level)/2)*weib.y.se[1]
wald.ci1
```

```
## [1] 4.451399 9.321032
```

```
# plot
log_lik_weibull_profile  <- function(data, gamma){
  beta.gamma <- mean(data^gamma)^(1/gamma)
  log_lik_weibull( data, c(gamma, beta.gamma) )
}

log_lik_weibull_profile_v <-Vectorize(log_lik_weibull_profile, 'gamma'  )

plot(function(x)
  -log_lik_weibull_profile_v(data=y, x)
  +weib.y.mle$value,
  from=0.1,to=15,xlab=expression(gamma),
  ylab='profile relative log likelihood',ylim=c(-8,0))

segments( wald.ci1[1],
          -log_lik_weibull_profile_v(y,wald.ci1[1])
          -weib.y.mle$value, wald.ci1[1],
          -log_lik_weibull_profile_v(y,wald.ci1[1])
          +weib.y.mle$value, col="blue", lty=2  )

segments( wald.ci1[2],
          -log_lik_weibull_profile_v(y,wald.ci1[2])
          -weib.y.mle$value, wald.ci1[2],
          -log_lik_weibull_profile_v(y, wald.ci1[2])
          +weib.y.mle$value, col="blue", lty=2  )

segments( wald.ci1[1], -8,  wald.ci1[2], -8, col="blue", lty =1, lwd=2  )
text(7,-7.5,"95% Wald CI",col="blue")
```

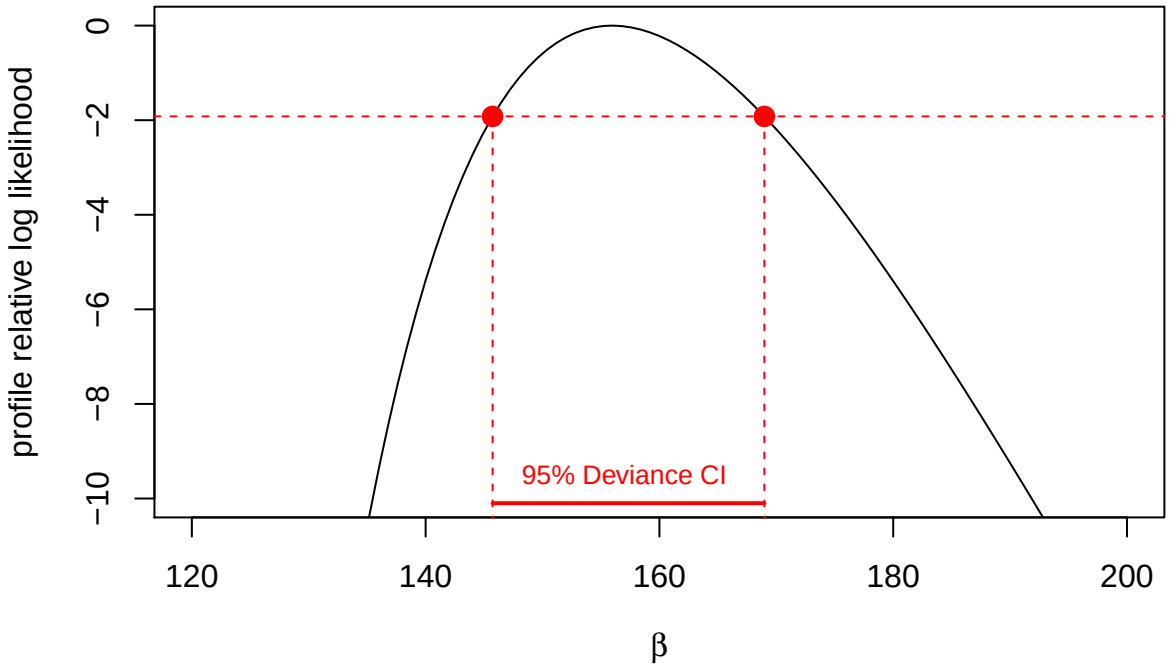


Exercise 5

Repeat the steps above — write the profile log-likelihood, plot it and find the deviance confidence intervals — considering this time γ as a nuisance parameter and β as the parameter of interest.

```
weib.y.mle<-optim(c(1,1), fn=log_lik_weibull,hessian=T, method='L-BFGS-B',lower=rep(1e-7,2), upper=rep(

log_lik_weibull_profile  <- function(data, beta){
  # gamma is fixed
  gamma.beta <- uniroot(function(x) n/x+sum(log(data))-n* sum(data^x*log(data))/sum(data^x),c(1e-5,15))$
  log_lik_weibull( data, c(gamma.beta, beta) )
}

log_lik_weibull_profile_v <- Vectorize(log_lik_weibull_profile, 'beta' )

plot(function(x) -log_lik_weibull_profile_v(data=y, x)
     +weib.y.mle$value,
     from=120,to=200,xlab=expression(beta),
     ylab='profile relative log likelihood',
     ylim=c(-10,0))

conf.level<-0.95
abline(h=-qchisq(conf.level,1)/2,lty='dashed',col=2)

lrt.ci1 <- uniroot(function(x)
  -log_lik_weibull_profile_v(y, x)
  + weib.y.mle$value
  + qchisq(conf.level,1)/2,
  c(1e+7,weib.y.mle$par[2]))$root

lrt.ci1<-c(lrt.ci1,uniroot(function(x)
  -log_lik_weibull_profile_v(y,x)
```

```
  + weib.y.mle$value
  + qchisq(conf.level,1)/2,
  c(weib.y.mle$par[2],15))$root)

segments( lrt.ci1[1],-qchisq(conf.level,1)/2, lrt.ci1[1], -log_lik_weibull_profile_v(y, lrt.ci1[1]), co

segments( lrt.ci1[2],-qchisq(conf.level,1)/2, lrt.ci1[2], -log_lik_weibull_profile_v(y, lrt.ci1[2]), co

points(lrt.ci1[1], -qchisq(0.95,1)/2, pch=16, col=2, cex=1.5)
points(lrt.ci1[2], -qchisq(0.95,1)/2, pch=16, col=2, cex=1.5)
segments( lrt.ci1[1], -10.1, lrt.ci1[2], -10.1, col="red", lty =1, lwd=2  )
text(157,-9.5,"95% Deviance CI",col=2, cex=0.8)
```



Exercise 6

Perform a test as above, but with:

$$\begin{cases} H_0 : \gamma = 1 \\ H_1 : \gamma = 5 \end{cases}$$

```
lambda_lrt <- -2*(log_lik_weibull_profile(y,5)-log_lik_weibull_profile(y,1))
lambda_lrt
```

```
## [1] 3.788422e+16
```

```
p_lrt <- pchisq(lambda_lrt, df =1, lower.tail = FALSE)
p_lrt
```

```
## [1] 0
```

```
# We end up rejecting the null hypotesis, since gamma is closer to 5
```