Inferential Statistics Assignment

Costalonga Andrea 2019164

Introduction

I went through the first and the third assignment, here reported in this order. This report has been created using r markdown and latex.

1. First Assignment

First of all I've defined some constants and a function to make my code clearer and understandable. I've also included some libraries that I've used in this part of the assignment.

```
library(plotrix)
library(rmarkdown)
set.seed(29)

#tau function
tau <- function(theta){
    out <- theta/(1-theta)
    out2 <- log(out)
    return(out2)
}

theta0 = 2/3 ##Neg bin success probability
r = 10 ##Neg bin size param
n <- c(10,50) ## Sample sizes
alpha = 0.05 ## Wald c.i. .95
```

In this segment I've created 2*N matrix with N = 10^5 to collect some results to simulate the distribution of tau hat. Every $\hat{\tau}$ is obtained evaluating $\hat{\theta}$ with the function tau. $\hat{\theta}$ is defined as

$$\hat{\theta} = \frac{r * n}{r * n + \bar{\mathbf{y}}}$$

where r is the number of odds of the negative binomial, n is the size of the r. ve. and $\bar{\mathbf{y}}$ is the sum of the values of the observed sample vector. This results comes as the maximum of the likelihood function.

```
N = 1e5
sim.N.tau <-matrix(NA, nrow = N, ncol = 2)
for(i in 1:N){
    xnegbin10 <- rnbinom(n=10,size=10,prob=theta0)
    xnegbin50 <- rnbinom(n=50,size=10,prob=theta0)
    sim.N.tau[i,1] <- tau(r*n[1]/(r*n[1]+sum(xnegbin10)))
    sim.N.tau[i,2] <- tau(r*n[2]/(r*n[2]+sum(xnegbin50)))
}
```

In the next cell I've managed to calculate what would have been the true value of mean and variance of the distribution of $\hat{\tau}$. In order to do so I've calculated the variance of $\hat{\theta}$ with Cramér-Rao's theorem, knowing that the Maximum Likelihood Estimator is asymptotically efficient. In formulas:

$$Var(\hat{\theta}) = I_{\mathrm{n}}(\theta)^{-1}$$

where $I_{\mathrm{n}}(\theta)$ is the Fisher information

$$I_{\mathrm{n}}(\theta) = n * I(\theta)$$

knowing that our r. ve. is built from $Y1, ..., Yni.i.d.$ variables and

$$I(\theta) = \frac{\theta^2 * (1-\theta)}{r * n}$$

, which is obtained by the definition of Fisher information. After that I've used the Delta Method to obtain the variance and the mean of the gaussian curve that represent our ideal distribution, and I've got those results: $\hat{\tau}$'s variance:

$$var(\hat{\tau}) = (\frac{dtau(\theta)}{d\theta})^2 * Var(\hat{\theta})$$

$\hat{\tau}$'s mean:

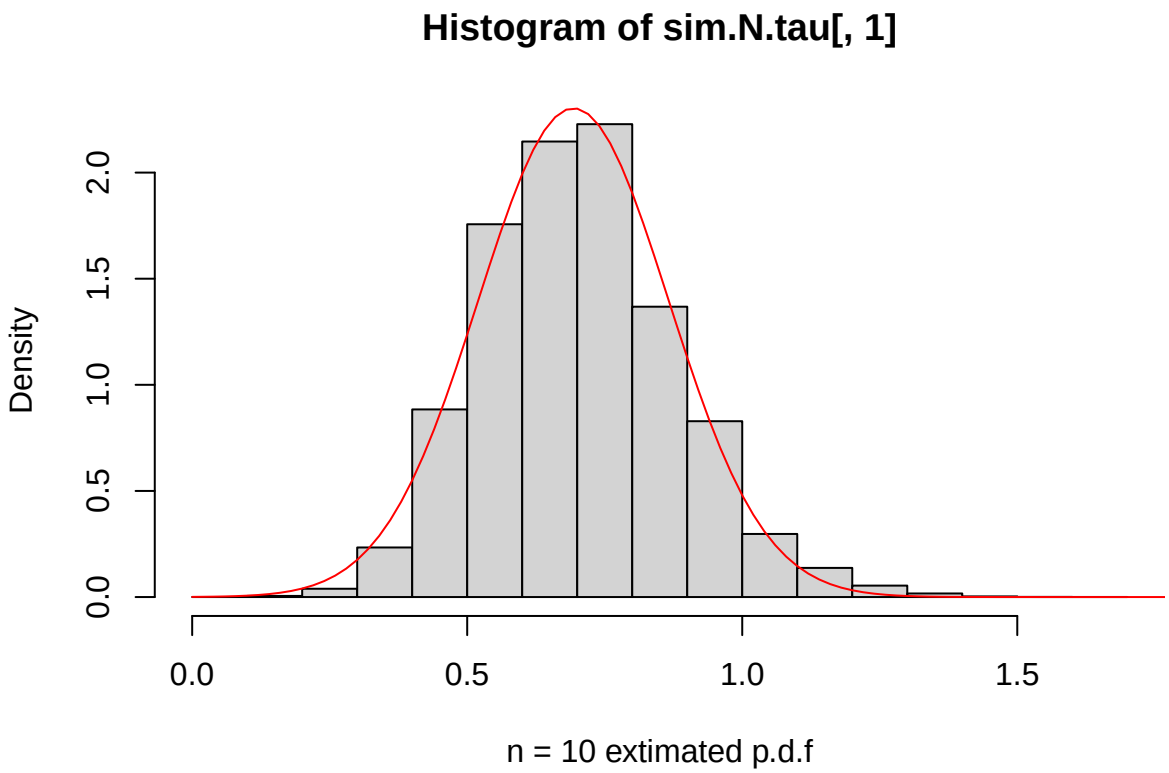
$$E[\hat{\tau}] = tau(\theta)$$

Evaluating variance and mean in θ_0 bring us to the real distribution of $\hat{\tau}$

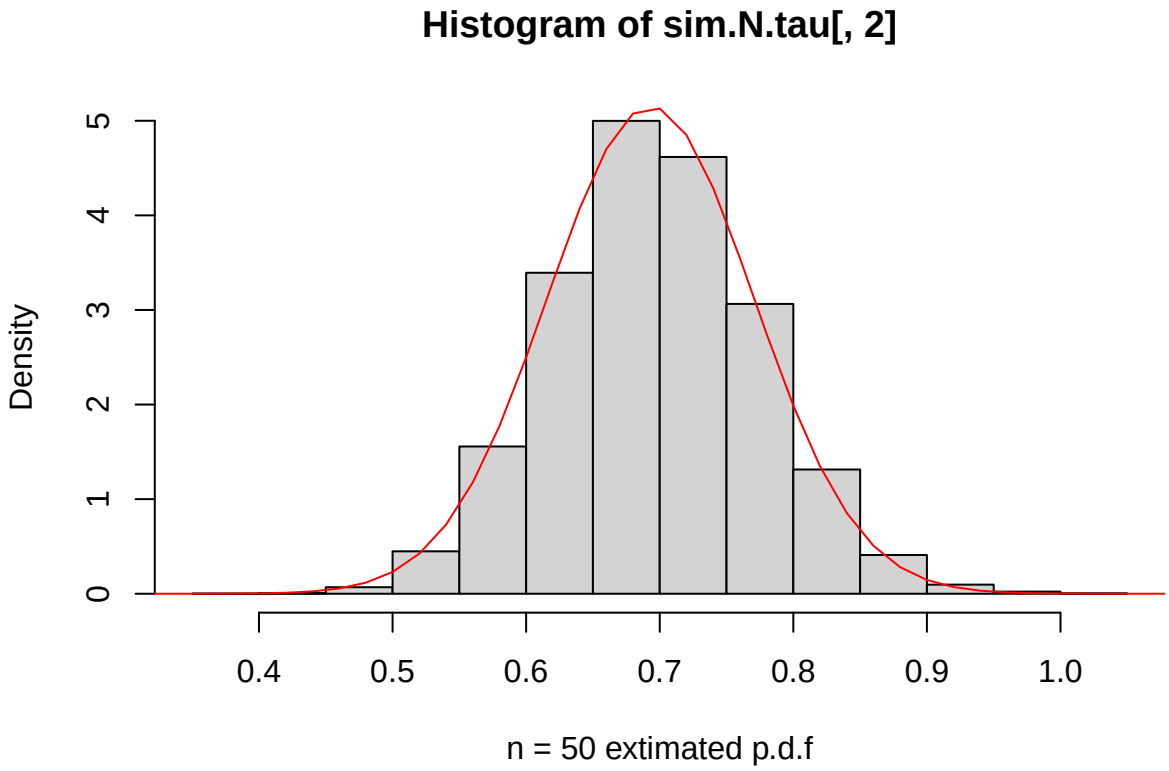
```
true_mean = (tau(theta0))
true_var10_o <- (theta0^2*(1-theta0))/(r*n[1])
true_var10 <- true_var10_o*(1/(theta0*(1-theta0)))^2
true_var50_o <- (theta0^2*(1-theta0))/(r*n[2])
true_var50 <- true_var50_o*(1/(theta0*(1-theta0)))^2
```

I've made a couple of plot to see if our theoretical results are supported by our simulations.

Plot for n = 10:



Plot for n = 50:



We can observe that the extimated distribution fits perfecly the exact p.d.f.. I've runned the simulation with just 10^5 trials but to obtain something that fits even better under the curve I could have runned with more samples with the perks of having to wait a large amount of time to execute the code.

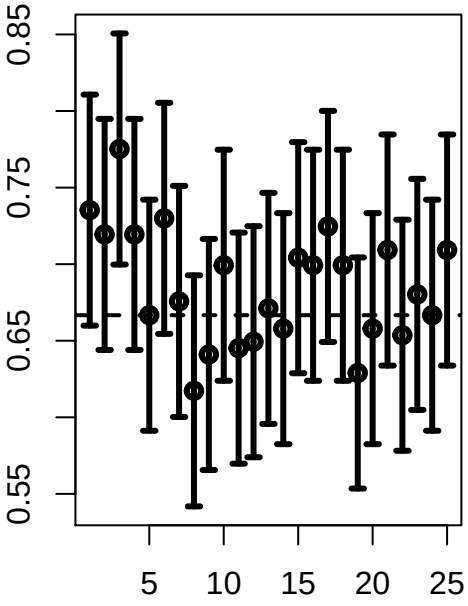
Coverage Probability, Wald C.I.

In the following lines I've made a simulation to study the coverage probability of the .95 Wald confidene interval for τ and θ

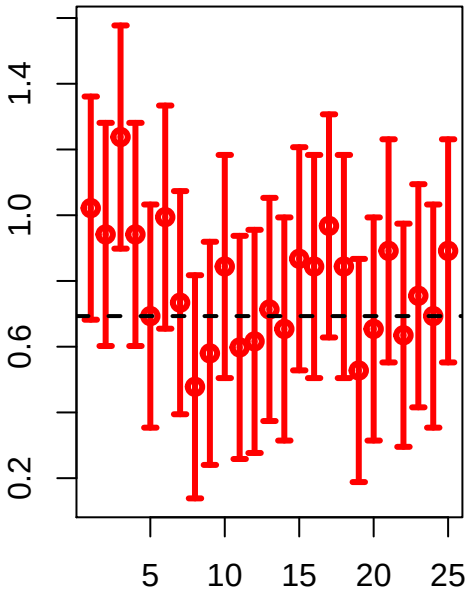
```
sim.N.CI_the <-matrix(NA, nrow = N, ncol = 4)
sim.N.CI_tau <-matrix(NA, nrow = N, ncol = 4)
for(i in 1:N){
    xnegbin10 <- rnbinom(n=n[1],size=r,prob=theta0)
    xnegbin50 <- rnbinom(n=n[2],size=r,prob=theta0)
    sim.N.CI_the[i,1:2] <- r*n[1]/(r*n[1]+sum(xnegbin10))
    sim.N.CI_the[i,1:2] <- sim.N.CI_the[i,1:2] + c(-1,1)*qnorm(p = alpha/2,
         lower.tail = FALSE)*sqrt(true_var10_o)
    sim.N.CI_the[i,3:4] <- r*n[2]/(r*n[2]+sum(xnegbin50))
    sim.N.CI_the[i,3:4] <- sim.N.CI_the[i,3:4] + c(-1,1)*qnorm(p = alpha/2,
         lower.tail = FALSE)*sqrt(true_var50_o)
    sim.N.CI_tau[i,1:2] <- tau(r*n[1]/(r*n[1]+sum(xnegbin10)))
    sim.N.CI_tau[i,1:2] <- sim.N.CI_tau[i,1:2] + c(-1,1)*qnorm(p = alpha/2,
        lower.tail = FALSE)*sqrt(true_var10)
    sim.N.CI_tau[i,3:4] <- tau(r*n[2]/(r*n[2]+sum(xnegbin50)))
    sim.N.CI_tau[i,3:4] <- sim.N.CI_tau[i,3:4] + c(-1,1)*qnorm(p = alpha/2,
        lower.tail = FALSE)*sqrt(true_var50)
}
theta10.inside <- apply(sim.N.CI_the[,1:2], MARGIN = 1,
    function(x)ifelse(theta0>=x[1]&theta0<=x[2],1,0))
theta50.inside <- apply(sim.N.CI_the[,3:4], MARGIN = 1,
    function(x)ifelse(theta0>=x[1]&theta0<=x[2],1,0))
tau10.inside <- apply(sim.N.CI_tau[,1:2], MARGIN = 1,
    function(x)ifelse(true_mean>=x[1]&true_mean<=x[2],1,0))
tau50.inside <- apply(sim.N.CI_tau[,3:4], MARGIN = 1,
    function(x)ifelse(true_mean>=x[1]&true_mean<=x[2],1,0))
```

In the plots below I've reported the first 25 confidence intervals for θ and τ

Confidence interval plot for θ and τ with n = 10:

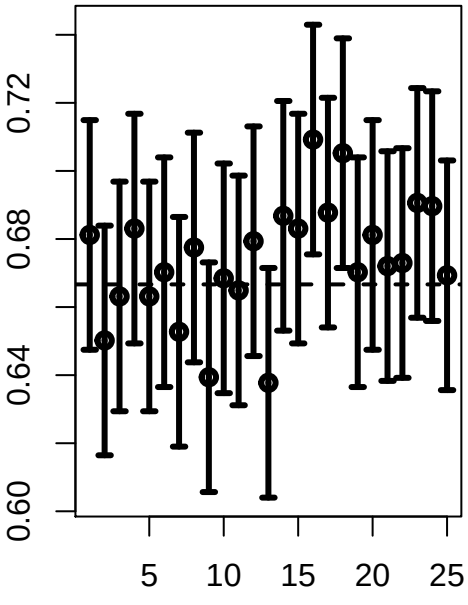


Observed C.I. for theta (n=10)

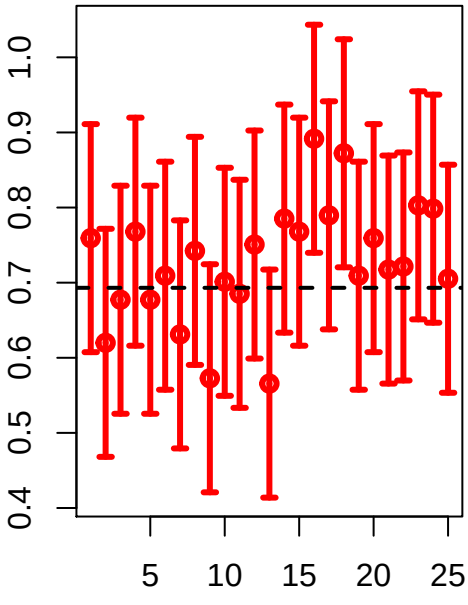


Observed C.I. for tau (n=10)

Confidence interval plot for θ and τ with n = 50:



Observed C.I. for theta (n=50)



Observed C.I. for tau (n=50)

Here's some results:

```
mean(theta10.inside) #Confidence probability for theta n=10
```

```
## [1] 0.95415
```

```
mean(theta50.inside)  #Confidence probability for theta n=50
```

```
## [1] 0.94976
```

```
mean(tau10.inside)  #Confidence probability for tau n=10
```

```
## [1] 0.94768
```

```
mean(tau50.inside)  #Confidence probability for tau n=50
```

```
## [1] 0.94857
```

We can notice that in every scenario the confidence probability close to .95 (even better some times). It's easy to state that our confidence interval for $\hat{\theta}$ and $\hat{\tau}$ have the same behaviour.

2. Third Assignment

I've defined some recurrent values and settings below:

```
set.seed(29)
n = c(10,15,10,15) #Number of elements for every col
var = c(1,1,1,1) #Variance for every col
mu = c(0,0,0,0) #Mean for every col
k=4 #Number of cols
N = 1e6 #Number of iterations for simulation
```

I've defined a vector with dimension N to store the value of F. This vector is going to be used in a following plot.

We also define SSR and SSE as:

$$SSR := \sum_{j=1}^{k}(\sum_{i=1}^{n_j}(\bar{Y}_{*j} - \bar{Y})^2)$$

$$SSE := \sum_{j=1}^{k}(\sum_{i=1}^{n_j}(Y_{ij} - \bar{Y}_{*j})^2)$$

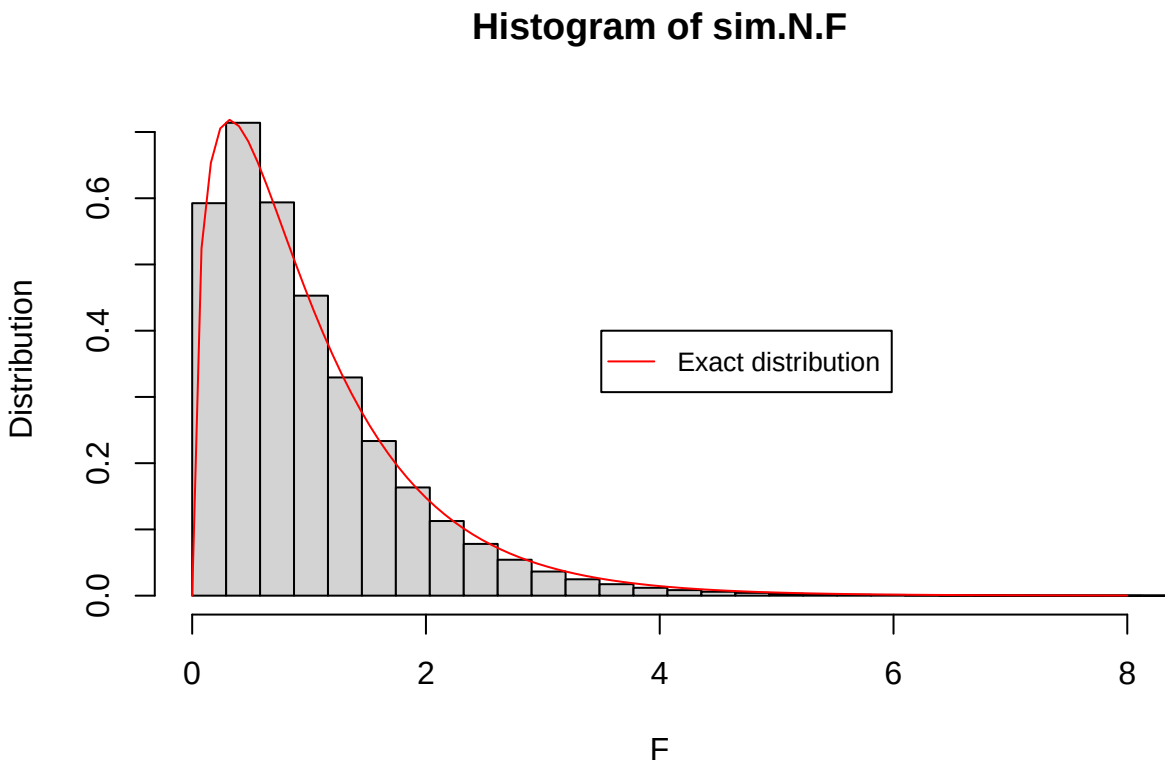
where $\bar{Y}_{*j}$ is the observed mean for treatment j, $\bar{Y}$ is the overall mean and Y_{ij} is the observed sample ij.

```
sim.N.F <- rep(0,N)
for(l in 1:N){
    y <- rnorm(sum(n), mean = 0, sd = sqrt(var[1]))#Same variance for every Yij
    #Treatment j mean
    estMuj <- c(sum(y[1:10])/n[1],
            sum(y[11:25])/n[2],
            sum(y[26:35])/n[3],
            sum(y[36:50])/n[4])
    #Overall mean
    estMu <- sum(y)/sum(n)
    SSR <- 0
    SSE <- 0
    for (i in 1:k){
        offset <- sum(n[1:(i-1)])
        for(j in 1:n[i]){
            SSR = SSR + (estMuj[i]-estMu)^2
            SSE = SSE + (y[offset+j]-estMuj[i])^2
        }
    }
    sim.N.F[l] <- (SSR/(k-1))/(SSE/(sum(n)-k)) #value of F observed (Fobs)
}
```

From theory we know that

$$\frac{\frac{SSR}{k-1}}{\frac{SSE}{n-k}} \sim \frac{{\chi_{\text{k-1}}}^2}{{\chi_{\text{n-k}}}^2}$$

This new r.v. should follow a F distribution with parameters k-1 and n-k. In the plot below there's a comparison between our simulated data and the real density of F. As we can see our simulation is coherent with the real p.d.f..



Now we are asked to change the value of the variance as follows:

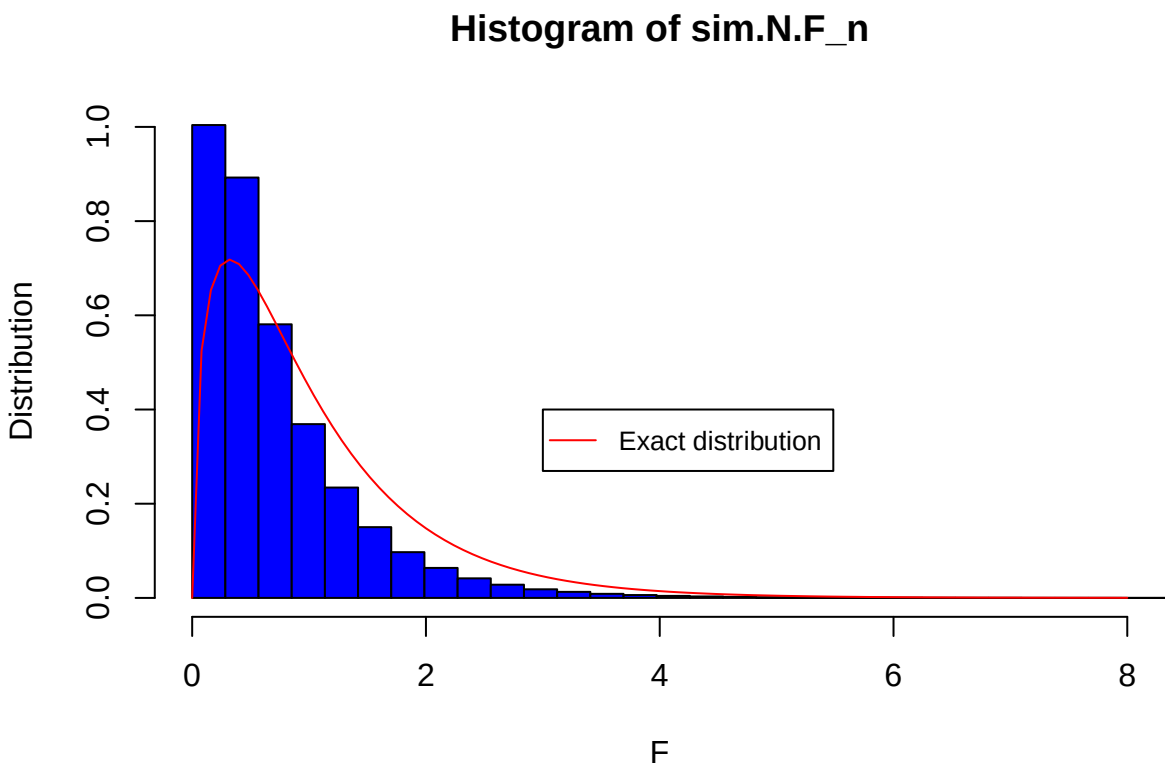
```
var <- c(1/2,2,1/2,2) #Vector of variances
```

In the next code lines I've re-executed the analysis with the changes I just made.

```
sim.N.F_n <- rep(0, N)
for(l in 1:N){
    offset <- 1
    y_n <- rep(0, 50)
    for (i in 1:k){
        y_n[offset:sum(n[1:i])]<-rnorm(n[i],
            mean = mu[i], sd = sqrt(var[i]))
        offset<- offset + n[i]
    }
    #Treatment j mean
    estMuj <- c(sum(y_n[1:10])/n[1],
            sum(y_n[11:25])/n[2],
            sum(y_n[26:35])/n[3],
```

```
            sum(y_n[36:50])/n[4])
    #Overall mean
    estMu <- sum(y_n)/sum(n)
    SSR <- 0
    SSE <- 0
    for (i in 1:k){
        offset <- sum(n[1:(i-1)])
        for(j in 1:n[i]){
            SSR = SSR + (estMuj[i]-estMu)^2
            SSE = SSE + (y_n[offset+j]-estMuj[i])^2
        }
    }
    #value of new F observed (Fobs)
    sim.N.F_n[l] <- (SSR/(k-1))/(SSE/(sum(n)-k))
}
```

Thi is the plot obtained with the data obtained in the latest analysis:



We can observe that, with this perturbation of the variance vector, our simulated data does not follow the exact p.d.f.. That's because ANOVA works for r.v.'s with the same variance and this is not the case.