Computational Stats

Tiago dos Santos

2018-11-18

Contents

To use the lab computers, the access credentials are:

usr: enc pwd: Ecom*2018

1 Lesson 1

```
x <- 3+5

ls()
```

```
## [1] "LatexOrOther" "datasetsDir"  "fig_basePath" "x"
```

1.1 Start by creating a vector

```
y <- c(2,5,9,8)

y[1:3]
```

```
## [1] 2 5 9
```

```
y[c(1,3)]
```

```
## [1] 2 9
```

1.1.0.1 Get the elements 1,2,3 from the vector

```r
y[1:3]
```

```
## [1] 2 5 9
```

1.1.0.2 Get the elements 1,3 from the vector

```r
y[c(1,3)]
```

```
## [1] 2 9
```

1.1.0.3 Get an array from 0 to 1, with a 0.001 step

```r
y <- 1:1000/1000
y <- seq(0,1,0.001)
```

1.1.0.4 Which values are lower than 0.008?

```r
isValueLowerThan <- y < 0.008
y[isValueLowerThan]
```

```
## [1] 0.000 0.001 0.002 0.003 0.004 0.005 0.006 0.007
```

```r
idxs <- which(y<0.08)
y[idxs]
```

```
##  [1] 0.000 0.001 0.002 0.003 0.004 0.005 0.006 0.007 0.008 0.009 0.010
## [12] 0.011 0.012 0.013 0.014 0.015 0.016 0.017 0.018 0.019 0.020 0.021
## [23] 0.022 0.023 0.024 0.025 0.026 0.027 0.028 0.029 0.030 0.031 0.032
## [34] 0.033 0.034 0.035 0.036 0.037 0.038 0.039 0.040 0.041 0.042 0.043
## [45] 0.044 0.045 0.046 0.047 0.048 0.049 0.050 0.051 0.052 0.053 0.054
## [56] 0.055 0.056 0.057 0.058 0.059 0.060 0.061 0.062 0.063 0.064 0.065
## [67] 0.066 0.067 0.068 0.069 0.070 0.071 0.072 0.073 0.074 0.075 0.076
## [78] 0.077 0.078 0.079
```

1.1.0.5 Creating objects by repetition

```r
colors <- c("amarelo","verde","vermelho","azul")

rep(colors, 5)
```

```
##  [1] "amarelo"  "verde"    "vermelho" "azul"     "amarelo"  "verde"
##  [7] "vermelho" "azul"     "amarelo"  "verde"    "vermelho" "azul"
## [13] "amarelo"  "verde"    "vermelho" "azul"     "amarelo"  "verde"
## [19] "vermelho" "azul"
```

```r
print("===")
```

```
## [1] "==="
```

```r
rep(10,5)
```

```
## [1] 10 10 10 10 10
```

1.2 Now a Matrix!

```
M <- matrix(1:9, ncol=3)
M
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

Transposing the Matrix

```
t(M)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

Accessing the Matrix

```
M[1,2]
```

```
## [1] 4
```

```
M[1,]
```

```
## [1] 1 4 7
```

```
M[,2]
```

```
## [1] 4 5 6
```

Matrix Operation

```
M2 <- t(M)

M+M2 # valuewise add
```

```
##      [,1] [,2] [,3]
## [1,]    2    6   10
## [2,]    6   10   14
## [3,]   10   14   18
```

```
M*M2 # valuewise multiplication
```

```
##      [,1] [,2] [,3]
## [1,]    1    8   21
## [2,]    8   25   48
## [3,]   21   48   81
```

```
M%*%M2 # Matricial Multiplication
```

```
##      [,1] [,2] [,3]
## [1,]   66   78   90
## [2,]   78   93  108
## [3,]   90  108  126
```

1.2.0.1 Joining Matrixes

Matrix Operation

```r
cbind(M,M2)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    4    7    1    2    3
## [2,]    2    5    8    4    5    6
## [3,]    3    6    9    7    8    9
```

```r
rbind(M,M2)
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
## [4,]    1    2    3
## [5,]    4    5    6
## [6,]    7    8    9
```

1.2.0.2 Inverting a matrix

```r
#solve(M) # M must not be singular
```

1.3 DataFrames

```r
y <- 1:10
y2 <- 11:20
y3 <- letters[1:10]

d1 <- data.frame(y,y2,y3)

d1
```

```
##     y y2 y3
## 1   1 11  a
## 2   2 12  b
## 3   3 13  c
## 4   4 14  d
## 5   5 15  e
## 6   6 16  f
## 7   7 17  g
## 8   8 18  h
## 9   9 19  i
## 10 10 20  j
```

1.4 Reading a Tab Separated File

```r
emp <- read.table(file.path(datasetsDir,"empresas.txt"), header=F)
knitr::kable(head(emp))
```

V1	V2	V3	V4	V5
Soflor	2	5	10	3
Florinha	3	10	22	7
Flora	5	30	55	18
Floflo	2	5	12	4
Fazflor	3	15	28	8
Comercflor	2	10	18	5

```r
dim(emp)
```

```
## [1] 40  5
```

```r
names(emp) <- c("nome","n.socios","c.social","vmm","n.emp")
knitr::kable(head(emp))
```

nome	n.socios	c.social	vmm	n.emp
Soflor	2	5	10	3
Florinha	3	10	22	7
Flora	5	30	55	18
Floflo	2	5	12	4
Fazflor	3	15	28	8
Comercflor	2	10	18	5

```r
emp$n.socios
```

```
##  [1] 2 3 5 2 3 2 3 4 6 5 2 3 2 3 2 3 3 2 5 2 2 3 3 2 2 2 2 4 4 3 2 2 4 2 2
## [36] 2 3 3 3 2
```

```r
emp[,2]
```

```
##  [1] 2 3 5 2 3 2 3 4 6 5 2 3 2 3 2 3 3 2 5 2 2 3 3 2 2 2 2 4 4 3 2 2 4 2 2
## [36] 2 3 3 3 2
```

1.5 Generating data

```r
set.seed(5)

emp$ant <- round(rnorm(dim(emp)[1],10,1))
```

1.6 Getting insights

```r
summary(emp)
```

```
##          nome       n.socios       c.social          vmm
##  Alecrim    : 1   Min.   :2.00   Min.   : 5.00   Min.   :  5.00
##  Beijaflor  : 1   1st Qu.:2.00   1st Qu.: 5.00   1st Qu.: 11.00
##  Caflor     : 1   Median :3.00   Median :10.00   Median : 19.00
##  Comercflor: 1   Mean   :2.85   Mean   :11.72   Mean   : 24.48
##  Cravinho   : 1   3rd Qu.:3.00   3rd Qu.:15.00   3rd Qu.: 31.00
##  Cravo      : 1   Max.   :6.00   Max.   :50.00   Max.   :100.00
##  (Other)    :34
##      n.emp              ant
##  Min.   : 2.000   Min.   : 8
##  1st Qu.: 3.000   1st Qu.: 9
```

```
##  Median : 5.500   Median :10
##  Mean   : 6.225   Mean   :10
##  3rd Qu.: 9.000   3rd Qu.:11
##  Max.   :18.000   Max.   :12
##
```

```r
mean(emp$n.socios)
```

```
## [1] 2.85
```

```r
sd(emp$n.socios)
```

```
## [1] 1.051251
```

```r
tapply(emp$vmm, emp$n.emp, mean) # vmm mean by number of employes
```

```
##          2          3          4          5          6          7
##   8.714286  10.875000  12.666667  18.000000  22.000000  23.000000
##          8          9         10         11         14         15
##  28.000000  32.250000  45.000000  61.000000  45.000000 100.000000
##         16         18
##  55.000000  55.000000
```

```r
tapply(emp$vmm, emp$n.emp, sd) # vmm sd by number of employes
```

```
##        2        3        4        5        6        7        8        9
## 2.627691 1.457738 1.154701 0.000000 1.414214 1.414214       NA 1.500000
##       10       11       14       15       16       18
##       NA 1.414214       NA       NA       NA       NA
```

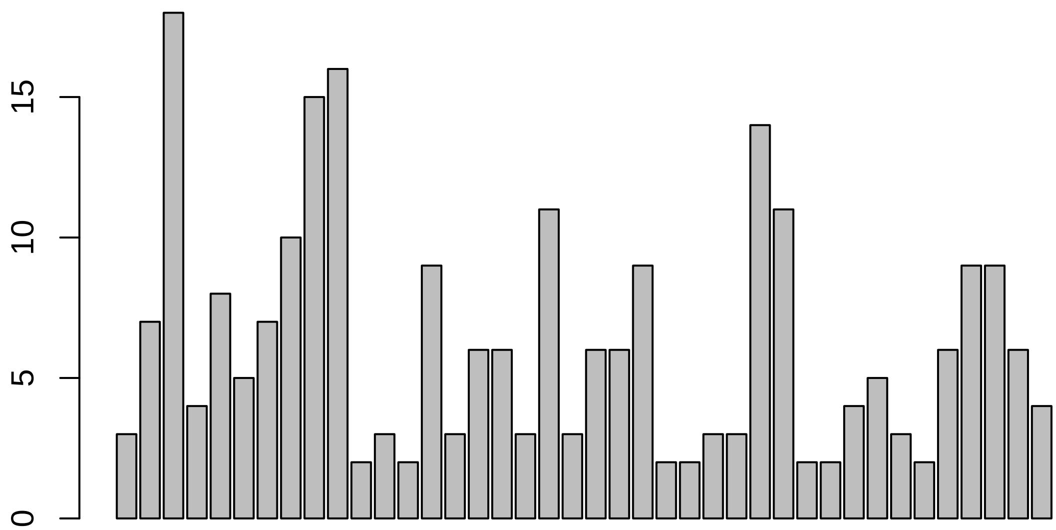
$$\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$$

$$S^2 = \frac{1}{N}\sum_{i=1}^{N}(X_i - \overline{X})^2 \tag{1}$$

```r
table(emp$n.emp) #first line are values, second line is frequency
```

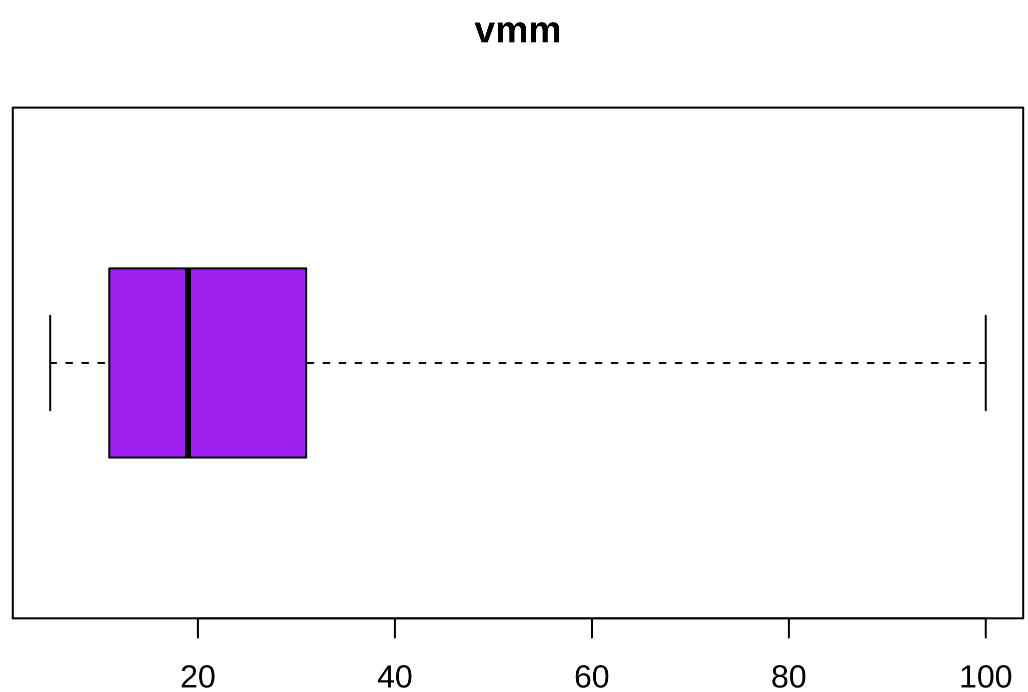
```
##
##  2  3  4  5  6  7  8  9 10 11 14 15 16 18
##  7  8  3  2  6  2  1  4  1  2  1  1  1  1
```

```r
barplot(emp$n.emp) # each company is a bin in x label, y is the number of employees
```

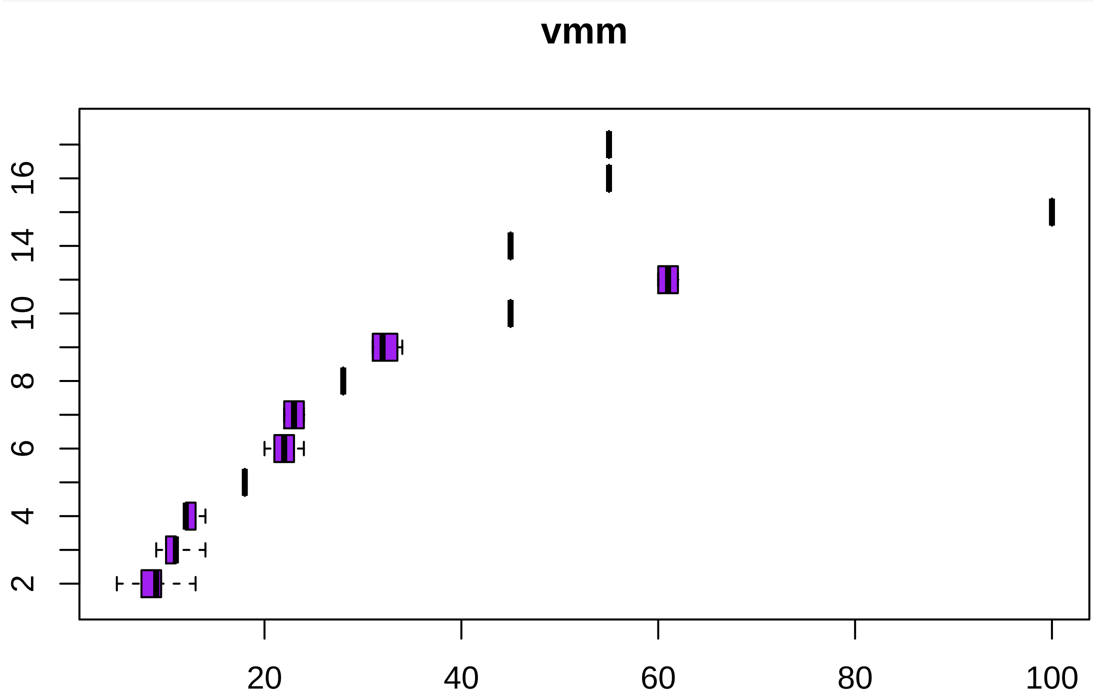



```r
barplot(table(emp$n.emp), xlab="#Employees", ylab="Frequecy", col="pink")
```

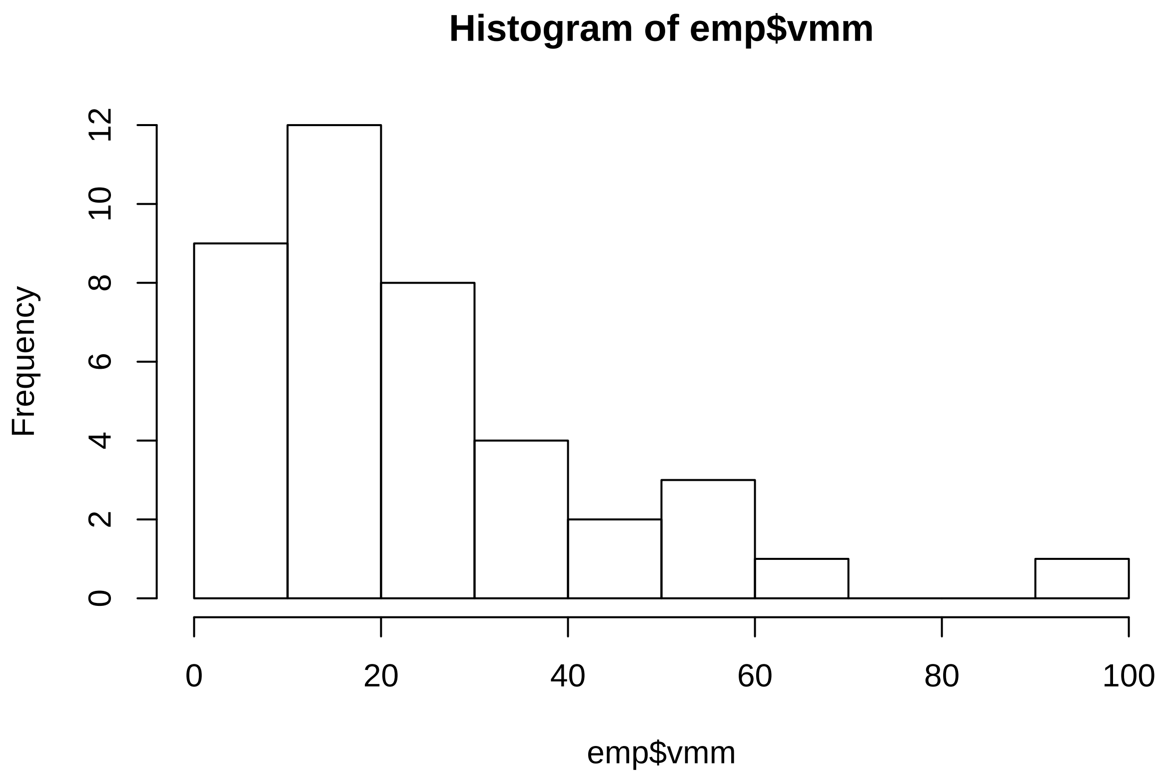

```r
boxplot(emp$vmm,range=0,col="purple",horizontal=T,main="vmm")
```



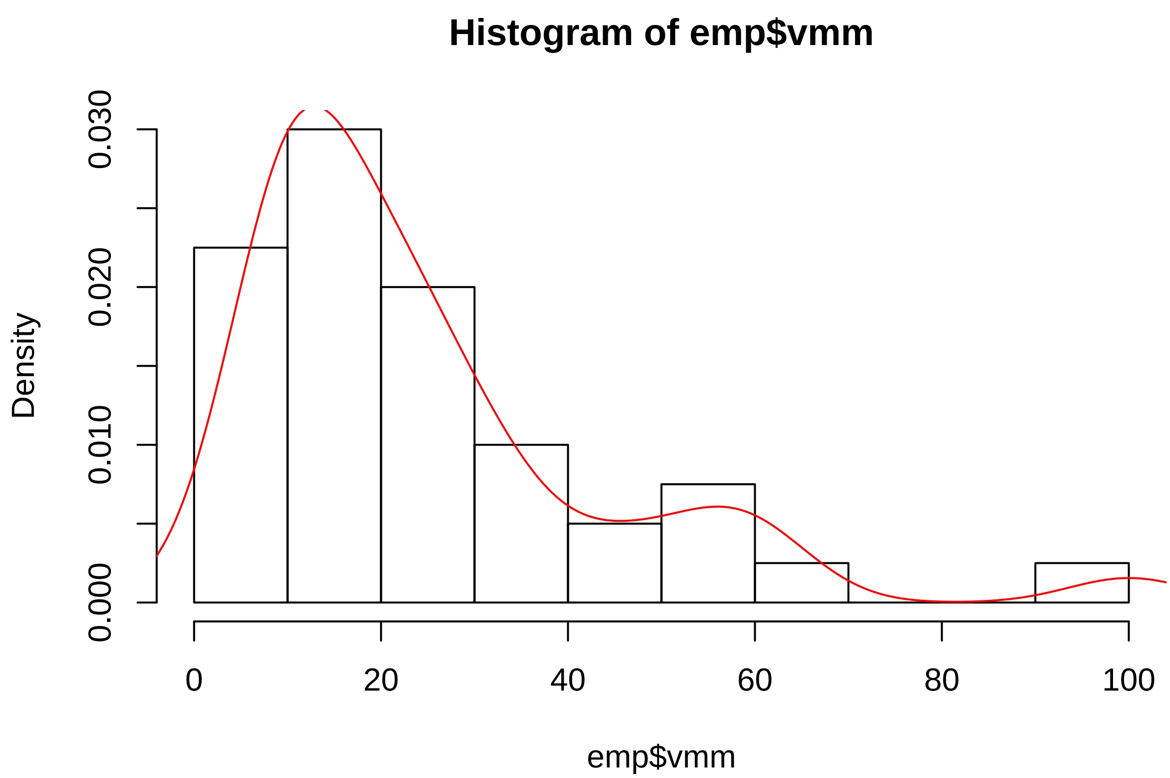

```
boxplot(emp$vmm ~ emp$n.emp,range=0,col="purple",horizontal=T,main="vmm")
```




```
hist(emp$vmm)
```

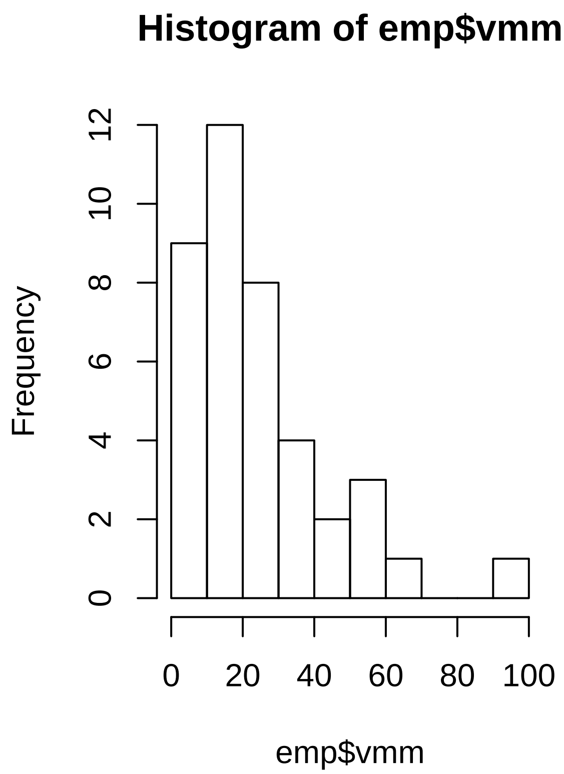


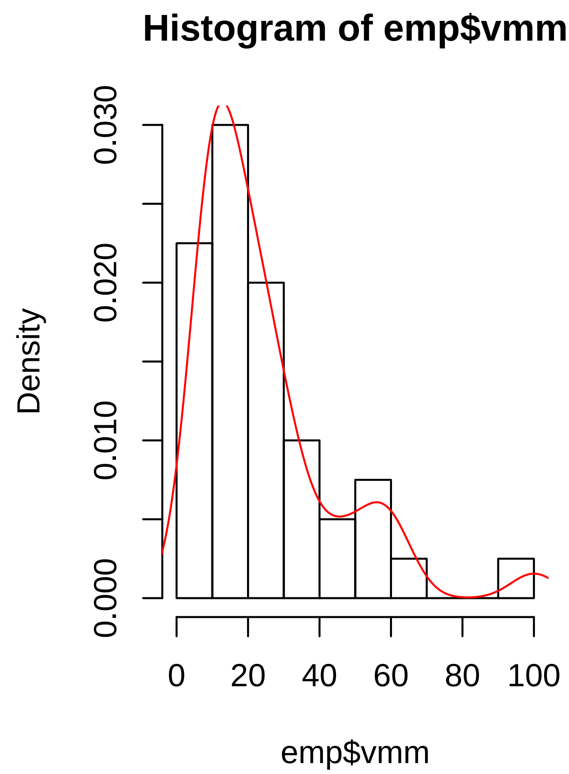

```r
hist(emp$vmm, freq=F)
lines(density(emp$vmm),col=2)
```




```r
par(mfrow=c(1,2))
hist(emp$vmm)
```

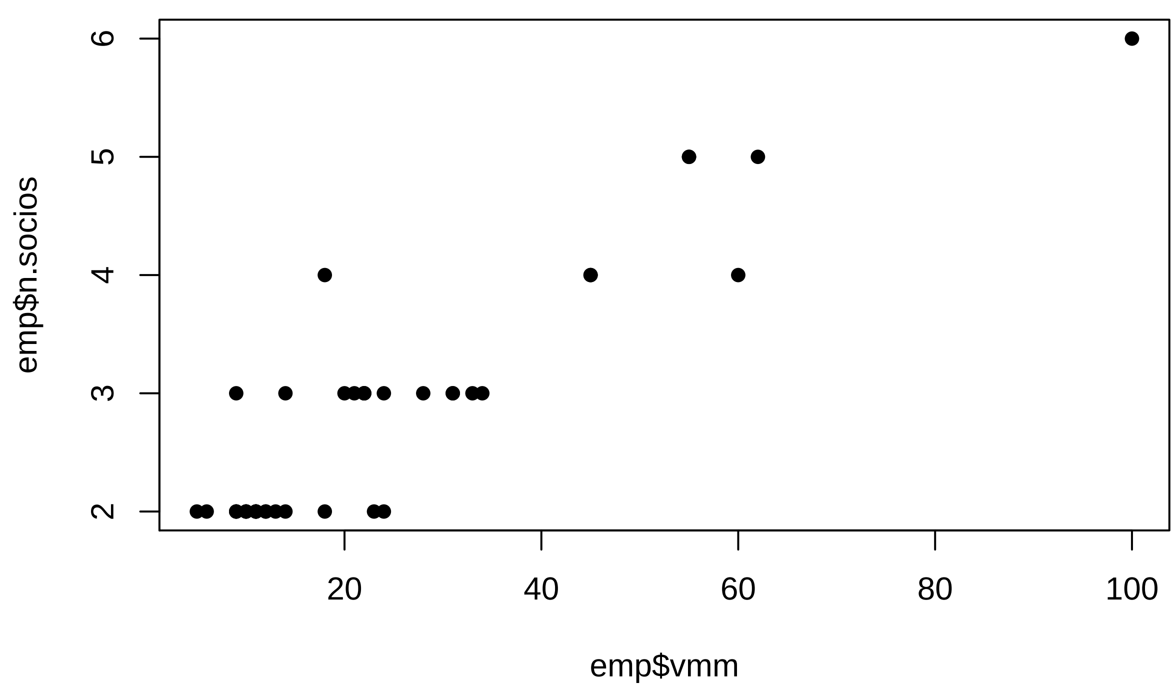
```r
hist(emp$vmm, freq=F)
lines(density(emp$vmm),col=2)
```



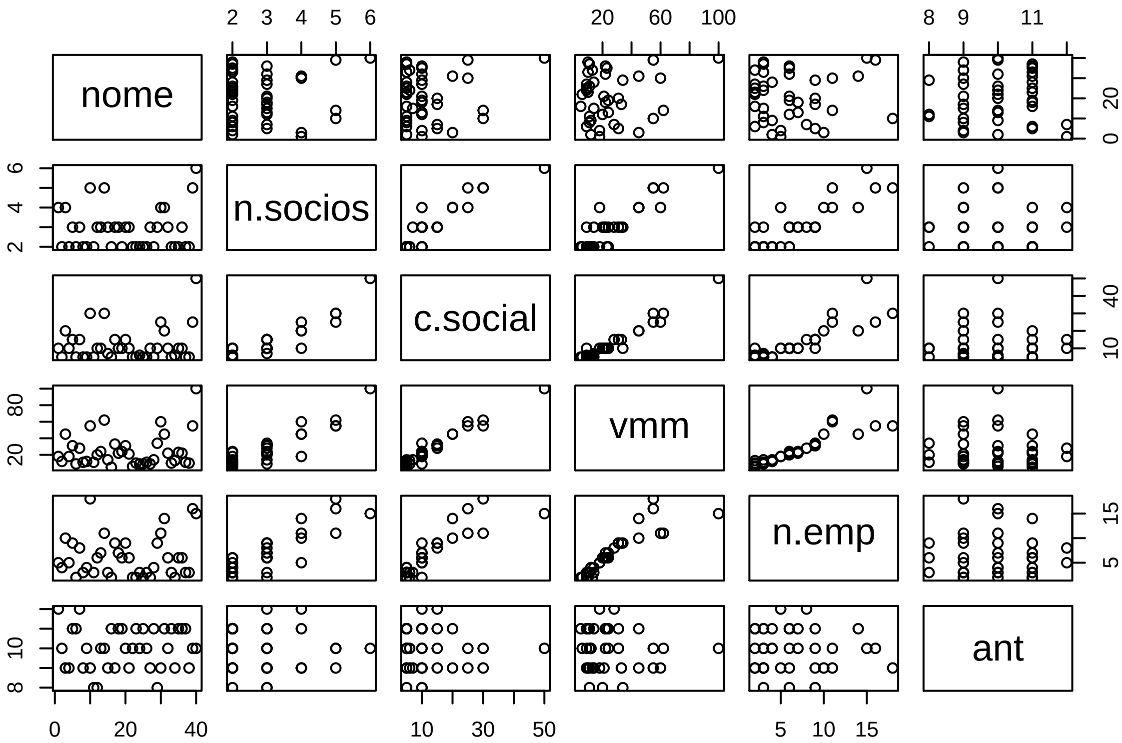



```r
par(mfrow=c(1,1))

plot(emp$vmm,emp$n.socios,pch=16)
```



```r
plot(emp)
```



1.7 Lists

```r
uma.lista <- list(
  um.vector=1:10,
  uma.palavra="olá",
  uma.matrix=M,
  outra.lista=list(
    a="flor",
    b=rep(3,5)
  )
)

uma.lista["um.vector"]
```

```
## $um.vector
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
uma.lista$um.vector
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
uma.lista[1]
```

```
## $um.vector
##  [1]  1  2  3  4  5  6  7  8  9 10
```

1.8 Functions

```r
desconto <- function(price, discount=25){
  #Discount is a number between 0 and 100
  #calcula o desconto de um preço
  newPrice <- price*(1-discount/100)
  discount <- price - newPrice
  list(
    novo.preco=newPrice,
    desconto=discount)
}

desconto(1000,20)
```

```
## $novo.preco
## [1] 800
##
## $desconto
## [1] 200
```

```r
desconto(1000,25)
```

```
## $novo.preco
## [1] 750
##
## $desconto
## [1] 250
```

This is how you function

2 Lessons 2

2.1 Random Variables and Vectors

2.1.1 Elements of probability

A random variable **X** is a function that takes an event space and return a value:

$$X : \Omega \to \mathbb{R}$$

2.1.2 Expected value

3 Lesson 3

```r
g <- function(x){
  exp(x^2)
}

#create sample from uniform distribution
sample <- runif(10000)
sample.length <- length(sample)
```

```r
mean(g(sample))
```

```
## [1] 1.46525
```

3.0.1 Estimating pi

```r
g <- function(x){
  sqrt(1-x^2)
}

#create sample from uniform distribution
sample <- runif(100000000)

mean(g(sample))*4
```

```
## [1] 3.141693
```

```r
gIndicatriz <- function(x,y){
  ifelse((x^2 + y^2) <= 1, 1, 0)
}

sampleX <- runif(1000000)
sampleY <- runif(1000000)
mean(gIndicatriz(sampleX,sampleY))*4
```

```
## [1] 3.141924
```

3.1 Hypothesis Testing

A statistical hypothesis is some conjecture about the distribution of one or more random variables. For each hypothesis designated by null hypothesis and denoted by $H0$, there is always an alternative hypothesis denoted by $H1$. We start the test by believing that $H0$ is true, and during the test we can discard that hypothesis only if the data points there.

Moreover, we can see these hypothesis testing as:

- A statistical hypothesis is some statement about the parameters of one or more populations (parametric tests) or about the distribution of the population (non-parametric tests).
- The goal of a test is to use the information of a data sample to decide (reject or no reject) about a conjecture over unknown aspects of a given population.

3.1.1 Types of error while infering through hypothesis testing

There are always some risk associated to statistical inference:

- *** Type 1 error ***: reject $H0$ when $H0$ is true (rejecting error, aka False Negative in ML nomenclature)
- *** Type 2 error ***: accept $H0$ when $H0$ is false (no rejecting error, aka False Positive in ML nomenclature)

3.1.2 Defining α to reduce a type of error

$$\alpha = P(Type1Err) = P(RejectingH0|H0istrue)$$

So, we define α as being the probability that we want for the Type 1 error - or how much are we willing to be prone to this type of error.

Therefone, α is called ***significance level*** of the test (a test that is very prone to errors is not very significant, right?)

In general, we assign a very small value to the probability of type I error (0.05 ou 0.01).

On the other end of the error spectrum, we define β as

$$\beta = P(Type2Err) = P(AcceptingH0|H0isfalse)$$

where $1 - \beta$ is called power of the test. The insight here is that the lower the β, the more "power" this test have.

3.1.3 Procedure to make a test using $p - value$

3.1.3.1 Wait, what is p-value?

(WIP)

3.1.4 Estimating test stats

The hypothesis being tested is the following:

We have a sample (the variable `popSample` below) of independent observations from a random variable that we know follows an exponencial distribution, with unknown parameter λ.

We want to test if $\lambda = 3$.

```
popSample <- c(0.2,1.2,2.9,1.2,0.1,0.1,0.4,0.1,0.7,0.1,0.9,0.3,0.6,0.1,0.2,0.1,0.4,0.1,0.3,1.4)

lambdaEstimator <- function(sample){
  1/mean(sample)
}


parameter <- 3

testStatsEstimator <- function(sample,hypothesisLambda, estimatedLambda){
  sampleMean <- mean(sample)
  sampleLength <- length(sample)
  return(
    (
      1/((sampleMean*hypothesisLambda)^sampleLength))
        *
      (exp(
        sampleLength*
        (
          hypothesisLambda*sampleMean -1)
        )
     )
  )
}

tobs <- testStatsEstimator(popSample,parameter,lambdaEstimator(popSample))
```

Here, we will do the following 1000 times:

- we get a random sample from an exponential with $\lambda = 3$
- we obtain the estimated test statistic for this sample

By the end of this process, we will get 1000 values that represente possible values of the Test Statistic Function

```
empiricDistTestStats <- sapply(1:1000,function(idx){
  sampleTest <- rexp(length(popSample),parameter)
  testStatsEstimator(sampleTest,parameter,lambdaEstimator(sampleTest))
})

empiricDistTestStats <- c(empiricDistTestStats,tobs)
empiricDistTestStats.df <- as.data.frame(empiricDistTestStats)
names(empiricDistTestStats.df) <- c("values")

empiricFrequency <- empiricDistTestStats.df %>% dplyr::group_by(values) %>% dplyr::summarise(n=n())

p_value_estimated <- sum(empiricFrequency[empiricFrequency$values >= tobs,]$n)/sum(empiricFrequency$n)


a <- list(
  text = paste0("P value estimated: " , round(p_value_estimated,5)),
  x = tobs,
  y = 0.3,
  xref = "x",
  yref = "y",
  ax = 50
)

plotly::plot_ly(
  x = empiricDistTestStats
  , type="histogram"
  , histnorm = "probability"
  , name = "Empiric Frequency") %>%
plotly::add_segments(
  x = tobs, xend = tobs, y = 0, yend = 0.3, name = "T obs"
) %>% plotly::layout(annotations=a)
```

4 Deliverables

5 Trabalho 1

5.1 Exerc?cio 1

1. Considere que uma variável continua X com a seguinte função de densidade:

$$f(x) = \begin{cases} \frac{4}{3}(x^3 + x) & 0 < x < 1 \\ 0, & \text{for all others } x \text{ values} \end{cases}$$

Agora considerando a variavel aliatória $Y = g(X)$, em que $g(x) = log(x^2+4)$. Estime o valor $P(1.3 < Y < 1.5)$ usando o método de monte carlo e estime o valor do desvio padrão da estimativa

---

$$P(1.3 < Y < 1.5) \quad = \quad P(1.3 < g(x) < 1.5)$$

$$D(g(x)) = D(log(x^2+4)) = [min(log(x^2+4), +\infty[\quad , \quad min(log(x^2+4)) = log(4) = 1.386294 > 1.3$$

Onde $D(g(x))$ é o dominio de $g(x)$. Uma vez que $D(g(x))$ está definido no intervalo $[min(log(x^2+4), +\infty[$ afimar o seguinte:

$$P(1.3 < g(x) < log(4)) = 0 \quad \Longrightarrow \quad P(1.3 < g(x) < 1.5) \equiv P(log(4) < g(X) < 1.5)$$

Que se pode desenvolver :

$$P(log(4) < log(x^2+4) < 1.5) \quad = \quad P(4 < x^2+4 < e^{1.5}) \quad = \quad P(0 < x^2 < e^{1.5}-4) \quad = \quad P(0 < x < \sqrt{e^{1.5}-4})$$

Agora, sabemos que a probabilidade que queriamos calcular pode ser obtida através do seguinte integral:

$$\int_0^{\sqrt{e^{1.5}-4}} \frac{4}{3}(x^3+x)dx$$

O qual puderá ser escrito com a seguinte mudança de variável:

5.1.1 Mudança de variável

$$z(x) = xc \quad , \quad z(\sqrt{e^{1.5}-4}) = 1 \quad \Longrightarrow c = \frac{1}{\sqrt{e^{1.5}-4}}$$

$$x = z\sqrt{e^{1.5}-4} \quad \Longrightarrow \quad x' = \sqrt{e^{1.5}-4}$$

No que resulta no seguinte integral:

$$\int_0^{\sqrt{e^{1.5}-4}} \frac{4}{3}(x^3+x)dx \quad \equiv \quad \frac{4}{3}\int_0^1 ((z\sqrt{e^{1.5}-4})^3+z\sqrt{e^{1.5}-4}).\sqrt{e^{1.5}-4}\,dz \equiv \frac{4}{3}(e^{1.5}-4)\int_0^1 (z^3(e^{1.5}-4)+z).1\,dx$$

Onde pudemos usar o método de monte carlo para estimar

$$\int_0^1 (z^3(e^{1.5}-1)+z).1\,dx$$

, assumindo que z segue uma distribuição $U(0,1)$.

$$\int_0^1 (z^3(e^{1.5}-4)+z).1\,dx \quad \approx \quad \hat{\theta} = \frac{\sum_{i=1}^{n}(z_i^3.(e^{1.5}-4)+z_i)}{n}$$

Calculo do estimador $\hat{\theta}$ do valor esperado do integral anterior.

```
int_func <- function(z){
  res=(z^3)*(exp(1.5)-4)+z
}

#z follows an uniform

sample <- runif(1000)
int_est <- mean(int_func(sample))
prob_value <- (4/3)*(exp(1.5)-4)*int_est
```

Assim sendo:

$P(1.3 < x < 1.5) \quad = \quad 0.3937988$

5.1.2 Cálculo do desvio padrão do estimador da probabilidade

$$var(P(1.3 < Y < 1.5)) \quad = \quad \left(\frac{4}{3}(e^{1.5}-4)\right)^2 .var(\theta)$$

```
varEstimator <- (1/(length(sample)^2))*sum(((4/3)*(exp(1.5)-4)*int_func(sample)-prob_value)^2)
df <- data.frame(
  probEstimated = prob_value,
  stdMC = sqrt(varEstimator)
)

knitr::kable(df)
```

probEstimated	stdMC
0.3937988	0.0085771

5.2 Exercise 2

5.2.1 2.1

$$E(e^{x+y}) = E(e^x + e^y)$$

Sendo:

$$E(X) = \int_D \mathrm{x.f(x)}\,\mathrm{d}x$$

, onde X é uma variável aletória e f(x) a sua função densidade de probabilidade.

Temos:

$$\int_0^\infty \int_0^\infty \mathrm{e}^{x+y}.\frac{2}{\sqrt{2\pi}}.\mathrm{e}^{\frac{-x^2}{2}}.\frac{2}{\sqrt{2\pi}}.\mathrm{e}^{\frac{-y^2}{2}}\,\mathrm{d}x\mathrm{d}y \quad = \quad \frac{2}{\pi}\int_0^\infty \int_0^\infty \mathrm{e}^x.\mathrm{e}^{\frac{-x^2}{2}}.\mathrm{e}^y.\mathrm{e}^{\frac{-y^2}{2}}\,\mathrm{d}x\mathrm{d}y$$

Fazendo as seguintes mudanças de variável:

$$\alpha = \mathrm{e}^{-x} \implies x = -log(\alpha)$$
$$\beta = \mathrm{e}^{-y} \implies y = -log(\beta)$$

Ficamos com os seguintes limites de integração para α:

$$\lim_{x \to \infty} \mathrm{e}^{-x} = 0$$
$$\lim_{x \to 0} \mathrm{e}^{-x} = 1$$

e para β:

$$\lim_{x \to \infty} \mathrm{e}^{-y} = 0$$
$$\lim_{x \to 0} \mathrm{e}^{-y} = 1$$

Substituindo na equação, temos:

$$\begin{aligned}
&\frac{2}{\pi} \int_1^0 \int_1^0 \mathrm{e}^{-log(\alpha)}.\mathrm{e}^{\frac{-(-log^2(\alpha))}{2}}.(-\frac{1}{\alpha}).\mathrm{e}^{-log(\beta)}.\mathrm{e}^{\frac{-(-log^2(\beta))}{2}}.(-\frac{1}{\beta})\mathrm{d}\alpha\mathrm{d}\beta \\
&= \frac{2}{\pi}.(-\int_0^1 \mathrm{e}^{-log(\alpha)}.\mathrm{e}^{\frac{-(-log^2(\alpha))}{2}}.\frac{1}{\alpha}\mathrm{d}\alpha).(-\int_0^1 \mathrm{e}^{-log(\beta)}.\mathrm{e}^{\frac{-(-log^2(\beta))}{2}}.\frac{1}{\beta}\mathrm{d}\beta) \\
&= \frac{2}{\pi}.\int_0^1 \mathrm{e}^{-log(\alpha)}.\mathrm{e}^{\frac{-log^2(\alpha)}{2}}.\frac{1}{\alpha}\mathrm{d}\alpha \int_0^1 \mathrm{e}^{-log(\beta)}.\mathrm{e}^{\frac{-log^2(\beta)}{2}}.\frac{1}{\beta}\mathrm{d}\beta \\
&= \frac{2}{\pi}.\int_0^1 \mathrm{e}^{-log(\alpha).(1+\frac{1}{2}log(\alpha))}.\frac{1}{\alpha}\mathrm{d}\alpha.\int_0^1 \mathrm{e}^{-log(\beta).(1+\frac{1}{2}log(\beta))}.\frac{1}{\beta}\mathrm{d}\beta \\
&= \frac{2}{\pi}.\int_0^1 \mathrm{e}^{-log(\alpha).(1+\frac{1}{2}log(\alpha))}.\frac{1}{\alpha}.1\mathrm{d}\alpha.\int_0^1 \mathrm{e}^{-log(\beta).(1+\frac{1}{2}log(\beta))}.\frac{1}{\beta}.1\mathrm{d}\beta \\
&= \frac{2}{\pi}.\int_0^1 \mathrm{e}^{-log(\alpha).(1+\frac{1}{2}log(\alpha))}.\frac{1}{\alpha}.\frac{1}{1-0}\mathrm{d}\alpha.\int_0^1 \mathrm{e}^{-log(\beta).(1+\frac{1}{2}log(\beta))}.\frac{1}{\beta}.\frac{1}{1-0}\mathrm{d}\beta
\end{aligned}$$

Sendo:

$$h_1(\alpha) = \mathrm{e}^{-log(\alpha).(1+\frac{1}{2}log(\alpha))}.\frac{1}{\alpha}.\frac{1}{1-0}$$
$$g_1(\alpha) = \mathrm{e}^{-log(\alpha).(1+\frac{1}{2}log(\alpha))}.\frac{1}{\alpha}$$
$$f_1(\alpha) = \frac{1}{1-0}$$
$$h_2(\beta) = \mathrm{e}^{-log(\beta).(1+\frac{1}{2}log(\beta))}.\frac{1}{\beta}.\frac{1}{1-0}.$$
$$g_2(\beta) = \mathrm{e}^{-log(\beta).(1+\frac{1}{2}log(\beta))}.\frac{1}{\beta}$$
$$f_2(\beta) = \frac{1}{1-0}$$

Onde:

$$f_1(\alpha), f_2(\beta) \sim \mathcal{U}(1,0)$$

e,

$$f_1(\alpha), f_2(\beta) \geq 0$$

Estamos em condições de aplicar Monte Carlo:

$$\begin{cases} \theta_1 = \int_D h_1(\alpha)\mathrm{d}\alpha = \int_D g_1(\alpha).f_1(\alpha)\mathrm{d}\alpha = E(g_1(X)) \\ \theta_2 = \int_D h_2(\beta)\mathrm{d}\beta = \int_D g_2(\beta).f_2(\beta)\mathrm{d}\beta = E(g_2(Y)) \end{cases}$$

Se tivermos uma amostra aleatória $x_1, ..., x_n$ da variavél aleatória X com densidade f, um estimador θ é:

$$\begin{cases} \hat{\theta_1} = \sum\limits_{i=1}^{n} \frac{g_1(x_i)}{n} \\ \hat{\theta_2} = \sum\limits_{i=1}^{n} \frac{g_2(y_i)}{n} \end{cases}$$

Finalmente:
$E(\hat{e^{x+y}}) = \hat{\theta} = \frac{2}{\pi}.\hat{\theta_1}.\hat{\theta_2}$

A variância de $\hat{\theta}$ será:

$v = Var(\frac{2}{\pi}.\hat{\theta_1}.\hat{\theta_2}) = \frac{4}{\pi^2}.Var(\hat{\theta_1}).Var(\hat{\theta_2})$

Aplicando o método de Monte Carlo, ficamos com:

$$\begin{cases} \hat{V}ar(\hat{\theta_1}) = \frac{1}{n^2} \sum\limits_{i=1}^{n} (g_1(x_i) - \hat{\theta_1})^2 \\ \hat{V}ar(\hat{\theta_2}) = \frac{1}{n^2} \sum\limits_{i=1}^{n} (g_2(y_i) - \hat{\theta_2})^2 \end{cases}$$

Substituindo na equação inicial:

$$\hat{v} = \frac{4}{\pi^2}.\sum_{i=1}^{n} \frac{(g_1(x_i) - \theta)^2}{n}.\sum_{i=1}^{n} \frac{(g_2(y_i) - \theta)^2}{n}$$

5.2.2 Implementação do método:

5.2.2.1 Determinação do estimador e do estimador da variância

```r
set.seed(1)
n<-1000
u1<-runif(n)
u2<-runif(n)
g1<-function(x){exp(-log(x)*(1+(1/2)*log(x)))*(1/x)}
g2<-function(y){exp(-log(y)*(1+(1/2)*log(y)))*(1/y)}
teta<-(2/pi)*mean(g1(u1))*mean(g2(u2))
teta1<-mean(g1(u1))
teta2<-mean(g2(u2))

v<-(4/(pi^2))*(mean((g1(u1)-teta1)^2)/n)*(mean((g2(u2)-teta2)^2)/n)
df <- data.frame(
  probEstimated = teta,
  varianceMC = v
)

knitr::kable(df)
```

probEstimated	varianceMC
7.874321	8.3e-06

5.2.3 2.2

Dado que:

$$X, Y \text{ random variables with p.d.f.:}$$

$$f(x) = \frac{2}{sqrt(2\pi)} e^{-\frac{x^2}{2}} \quad ,0 < x < +\infty$$

precisamos de estimar o par?metro θ com o método de Monte Carlo utilizando uma variável que não seja Uniforme, onde o parâmetro θ é definido como:

$$\theta = E(e^{X+Y})$$

5.2.4 Trabalhar o estimador

$$\theta = E(e^{X+Y}) \quad = E(e^X \times e^Y) \quad = E(e^X) \times E(e^Y)$$

$$= \int_0^{+\infty} e^x \frac{2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx \times \int_0^{+\infty} e^y \frac{2}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} dy$$

Este integral n?o as condições para o método de Monte Carlo ser aplicado, e portanto é necessário trabalhar o integral de maneira a que seja possível aplicar o método.

5.2.5 Mudança de variável

Ao aplicar a seguinte mudança de variável:

$$x = \varphi(t) = \sqrt{t}$$

$$t = \varphi^{-1}(x) = x^2$$

$$\varphi'(t) = (\sqrt{t})' = (t^{\frac{1}{2}})' = \frac{1}{2}t^{-\frac{1}{2}}$$

$$\lim_{t \to +\infty} \sqrt{t} = +\infty$$

$$\lim_{t \to 0} \sqrt{t} = 0$$

podemos reorganizar o integral da seguinte maneira:

$$x = \varphi(t) = t_x \quad , y = \varphi(t) = t_y$$

$$\int_0^{+\infty} e^{\sqrt{t_x}} \frac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}t_x} \frac{1}{2} t_x^{-\frac{1}{2}} dt_x \times \int_0^{+\infty} e^{\sqrt{t_y}} \frac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}t_y} \frac{1}{2} t_y^{-\frac{1}{2}} dt_y$$

Re-ordenando a equação, estamos agora em condições de aplicar o método de Monte Carlo uma vez que o integral é definido pela multiplicação de:

1. uma p.d.f. $f(x)$ conhecida
2. uma outra função $g(x)$

$$\int_0^{+\infty} e^{\sqrt{t_x}} \frac{2}{\sqrt{2\pi}} t_x^{-\frac{1}{2}} \frac{1}{2} e^{-\frac{1}{2}t_x} dt_x \times \int_0^{+\infty} e^{\sqrt{t_y}} \frac{2}{\sqrt{2\pi}} t_y^{-\frac{1}{2}} \frac{1}{2} e^{-\frac{1}{2}t_y} dt_y$$

onde:

$$g(x) = e^{\sqrt{x}} \frac{2}{\sqrt{2\pi}} x^{-\frac{1}{2}} \quad \text{and} \quad f(x) = \frac{1}{2} e^{-\frac{1}{2}x}$$

onde $f(x)$ é a função distribuição densidade de uma variável Exponencial com $\lambda = \frac{1}{2}$:

$$X \sim Exp(\frac{1}{2})$$

Portanto, é agora necessário gerar amostras aleatórias de $X \sim Exp(\frac{1}{2})$

5.2.6 Gerar uma variável com distribuição Exponencial utilizando o método da Transformação Inversa

Comecemos com a função distribui??o cumulativa de uma variável Exponencial, que é definida por:

$$F(X) = 1 - e^{-\lambda x}, \quad x \in \mathbb{R}$$

Tendo em consideração que o resultado de $F(X)$ é um número real entre 0 e 1, e que:

1. $F(X)$? uma função monótona não decrescente
2. $F(X)$? uma função cont?nua

? sabido que $F(X)$? invert?vel.

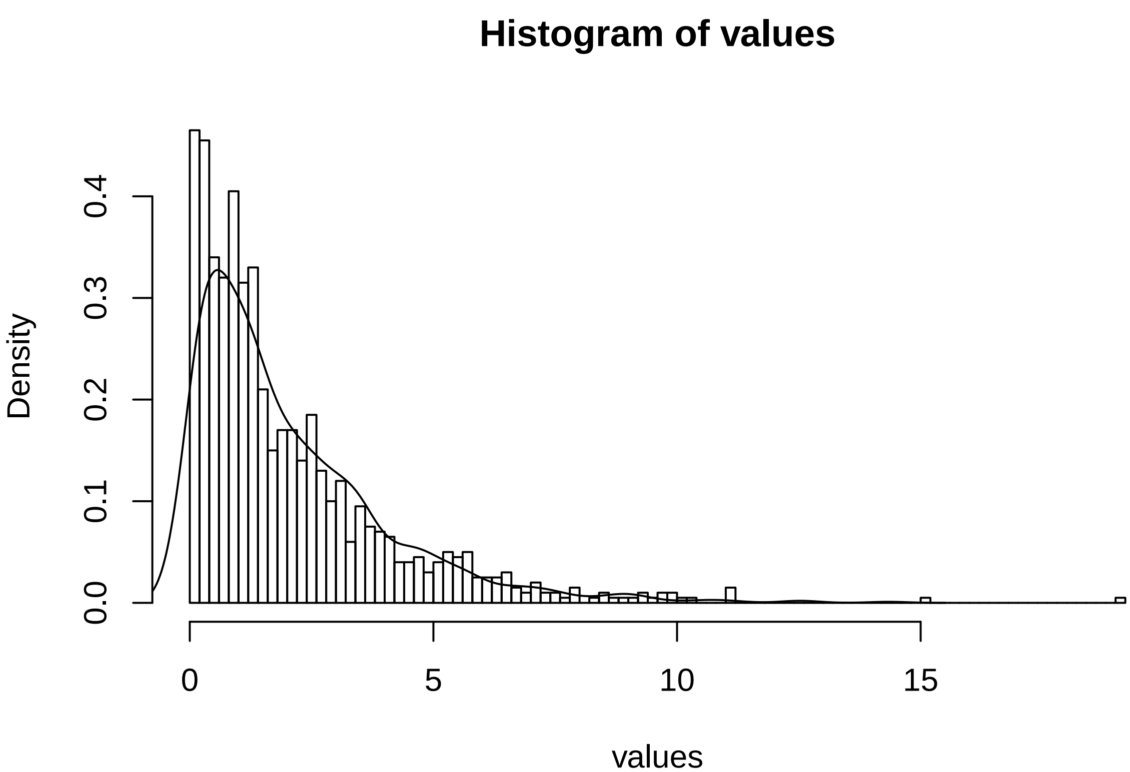
5.2.7 Implementação do método

```
set.seed(1)
lambda <- 0.5
N <- 1000
samples <- runif(N)


inverseExp <- function(u, lambda){
  -(1/lambda)*log(1-u)
}

values <- inverseExp(samples, lambda)

hist(values, breaks=100, freq = F)
lines(density(rexp(1000,0.5)))
```



```r
g <- function(x){
  exp(sqrt(x))*(2/(sqrt(2*pi)))*x^(-1/2)
}

X <- runif(N)
Y <- runif(N)

EX <- mean(g(inverseExp(X, lambda)))
EY <- mean(g(inverseExp(Y, lambda)))

theta2 <- EX*EY

vEX <- (1/(N^2))*sum((g(inverseExp(X, lambda))-EX)^2)
vEY <- (1/(N^2))*sum((g(inverseExp(Y, lambda))-EY)^2)

vtheta <-vEX*vEY

df <- data.frame(
  probEstimated = theta2,
  varianceMC = vtheta
)

knitr::kable(df)
```

probEstimated	varianceMC
7.901322	2.4e-06

5.2.8 2.3

$\hat{\theta} = E(e^{x+y})$ $var(\hat{\theta}) = \frac{4}{\pi^2}.\sum_{i=1}^{n}\frac{(g_1(x_i)-\hat{\theta})^2}{n}.\sum_{i=1}^{n}\frac{(g_2(y_i)-\hat{\theta})^2}{n}$

novo estimador:

$\hat{\theta}_c \quad = \quad \hat{\theta} - \beta.(c-\mu)$

$E(C) \quad = \quad \mu \; var(\hat{\theta}_c) = var(\hat{\theta}) + \beta^2.var(C) - 2\beta cov(\hat{\theta}, C)$

Queremos minimazar a variância, minimizando a variável β: para tal derivamos $var(\hat{\theta}_c)$ em ordem a β o que resulta na expressão: $var(\hat{\theta}_c)' = 2\beta var(C) - 2cov(\hat{\theta}, C)$

Com $var(\hat{\theta}_c)' = 0$ iremos obter os extremos. $\beta = \frac{-2cov(\hat{\theta},C)}{2var(C)}$

Calculo auxiliares:

$$b) \qquad var(\hat{\theta}) = Var\left(\frac{2}{\pi}.(\hat{\theta_1}.\hat{\theta_2})\right) = \frac{4}{\pi^2}.Var(\hat{\theta_1}).Var(\hat{\theta_2})$$

$$E(C) = E\left(\frac{1}{n}.\sum_{i=0}^{n} u_i\, v_i\right) = \frac{1}{n}\sum(E(u).E(v)) = E(u).E(v) = \int_0^1 u\, du.\int_0^1 v\, dv$$

$$var(C) = \frac{1}{n}.\left(\int_0^1 f_c(x)^2 - E(C)^2)\right)$$

$$Cov(\hat{\theta}, C) = cov\left(\frac{1}{n}\sum_{i=1}^{n} g(U_i, V_i), \frac{1}{n}\sum_{i=1}^{n} U_i, V_i\right) =$$

$$= \frac{1}{n^2}\sum_{i=1}^{n} cov\left(g(U_i,V_i),U_i,V_i\right) - \sum_{i=1}^{n}\sum_{j=1}^{n} cov\left(g(U_i,V_i),U_jV_j\right) =$$

Como os indices $i \neq j$ então$g(U_i,V_i)$ será independente de U_iV_i e a sua covariância será Zero.

$$= \frac{1}{n^2}\sum_{i=1}^{n} cov\left(g(U_i,V_i),U_i,V_i\right) - 0 =$$

$$= \frac{1}{n}(E(g(U,V))UV)) - \frac{\theta}{4n}$$

$$E(g(U,V))UV) = \int_0^1\int_0^1 uv.g(u,v)\,du\,dv$$

que será estimado em r pelo método de monte carlo

```r
kk=1
res_final <- data.frame(integer(), double(), double(), double(), double())
n <- 10

while(kk<6) {
    if(kk!=1) {
      if(kk%%2!=0)
        n<-n*2
      else
        n<-n*5
    }
    set.seed(1)
    u1<-runif(n)
    u2<-runif(n)
    g1<-function(x){exp(-log(x)*(1+(1/2)*log(x)))*(1/x)}
    g2<-function(y){exp(-log(y)*(1+(1/2)*log(y)))*(1/y)}
    teta1<-mean(g1(u1))
    teta2<-mean(g2(u2))
    teta<-(2/pi)*mean(g1(u1))*mean(g2(u2))
    teta_var<-(4/(pi^2))*((mean((g1(u1)-teta1)^2))/n)*((mean((g2(u2)-teta2)^2))/n)
    df <- data.frame(
      probEstimated = teta,
      varianceMC = teta_var
    )

    g1_b<-function(x){exp(-log(x)*(1+(1/2)*log(x)))}
    g2_b<-function(y){exp(-log(y)*(1+(1/2)*log(y)))}
    covar_tc<-((1/n^2)*mean(g1_b(u1)*g2_b(u2)*u1*u2))-teta/(4*n^2)

    c_var<-(7/(n*144))
    beta<-covar_tc/(c_var)
    tetac<-(teta  -   (beta*((1/4)  -   mean(u1*u2))))
    tetac_var<-(teta_var+(beta^2)*c_var-2*beta*covar_tc)

    res<-data.frame(n, teta, teta_var, tetac, tetac_var)
    res_final<-rbind(res_final,res)
    kk<-kk+1
}
knitr::kable(res_final)
```

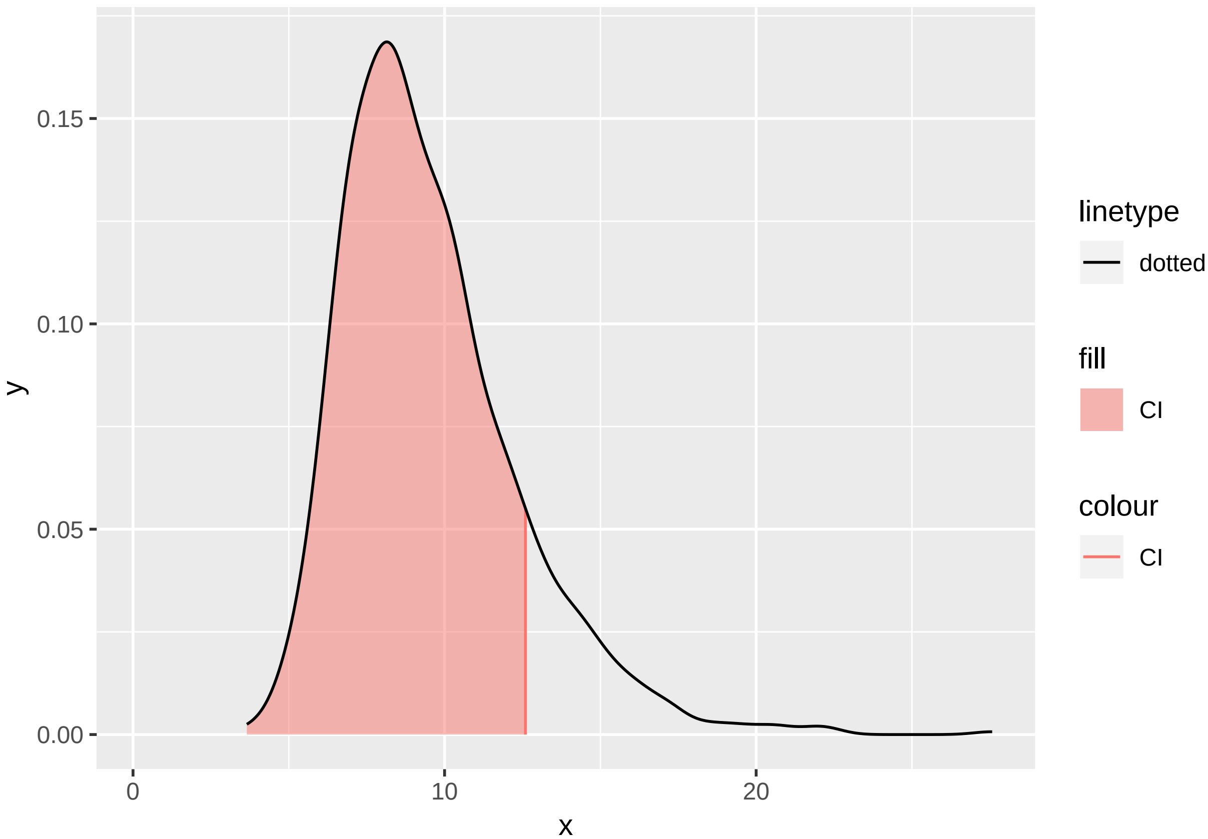
n	teta	teta_var	tetac	tetac_var
10	7.114578	0.0728507	6.962646	0.0401382
50	7.444414	0.0024827	7.432106	0.0021721
100	7.604431	0.0006777	7.598763	0.0006353
500	7.839631	0.0000307	7.839400	0.0000303
1000	7.874321	0.0000083	7.874559	0.0000083

```
knitr::knit_child("Works/work2/Deliverable2.Rmd")
```

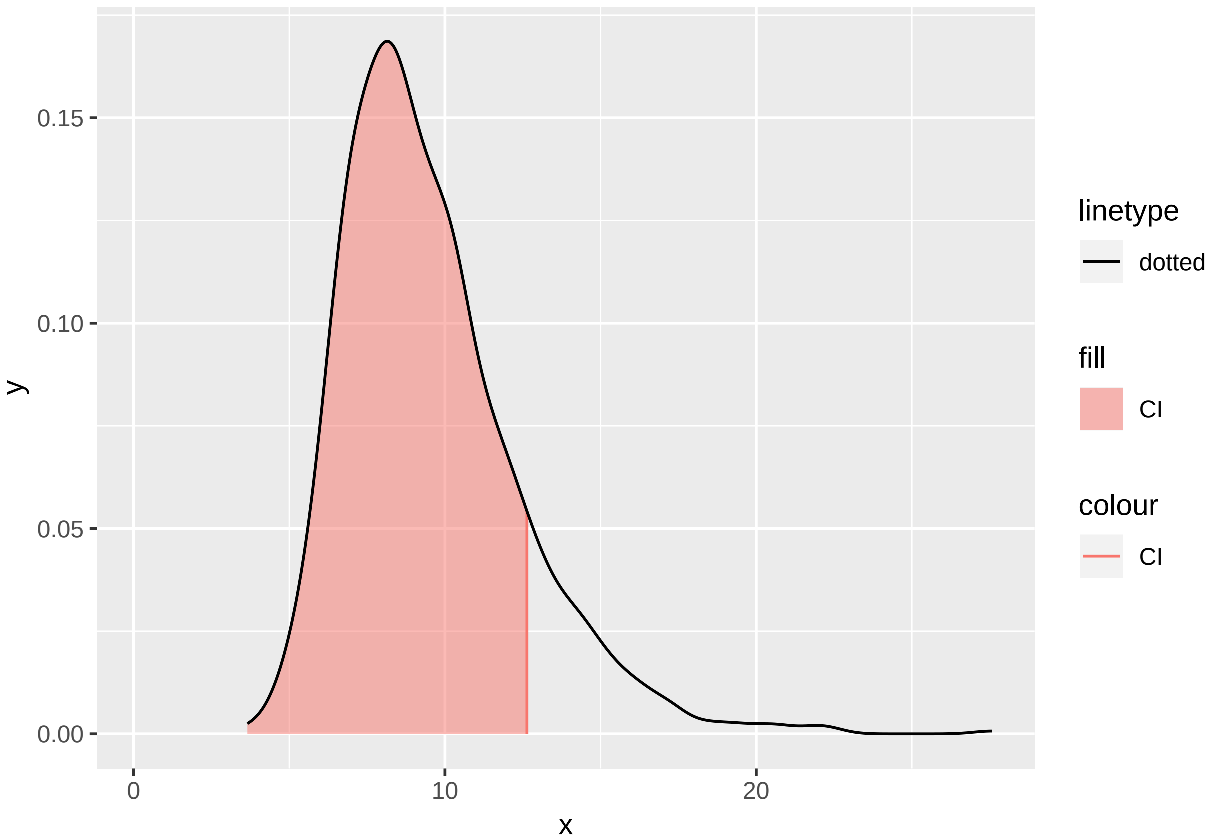
```
## Warning: Ignoring unknown aesthetics: label
```

```
## Warning: Removed 1 rows containing missing values (geom_segment).
```

```
## Warning: Ignoring unknown aesthetics: label
```




```
## Warning: Removed 1 rows containing missing values (geom_segment).
```




```
## [1] "\n\n\n\n# Problem 1\n\nConsider the following sample:\n\n```r\nc(7.0,3.5,11.9,8.9,10.1,1.2,1.1,
```