Métodos Quantitativos II - Lista 3

Professor Manoel Galdino e Monitor Davi Veronese

September 14, 2023

Nesta lista, trabalharemos com simulação estatística. Explique detalhadamente suas simulações, comentando o código etapa por etapa.

```
# Material de apoio para esta lista:
# https://jonnyphillips.github.io/Analise_de_Dados_2022/
```

1

Rode ?rnorm no R e leia o help. Se preciso, consulte outras fontes. Explique o que a função faz. Verifique que essa função gera uma simulação de uma distribuição normal, em que você pode especificar a média e o desvio-padrão. Verifique que entendeu rodando função e gerando valores simulados da função rnorm. Pode usar média 0 e desvio-padrão 1, que são o default da função.

```
# ?rnorm
```

2

Rode $x < -rnorm(100, mean = 2, sd = 1)$. Por que a média da distribuição é diferente da média que você obtém rodando mean(x)?

```
x <- rnorm(100, mean=2, sd=1)
mean(x)

## [1] 2.150176
```

Os parâmetros estabelecidos definem a distribuição a partir da qual serão retirados valores aleatórios. A aleatoriedade faz com que a média desses valores não seja exatamente igual à média da distribuição.

3

Se você rodar de novo $x < -rnorm(100, mean = 2, sd = 1)$ e calcular a média de x, obterá um valor um pouco diferente da primeira vez. Por quê?

A aleatoriedade dos valores retirados faz com que a média desses valores não seja exatamente igual à média da distribuição. Como os valores retirados são aleatórios, há variabilidade entre simulações.

4

Uma forma de você armazenar as duas médias que você computou em um vetor é do seguinte modo:

```
#vetor_medias <- numeric()
#vetor_medias[1] <- mean(rnorm(100, mean=2, sd=1))
#vetor_medias[2] <- mean(rnorm(100, mean=2, sd=1))
```

Imprima o conteúdo de "vetor medias" e verifique que de fato armazenou duas médias.

5

Repita esse procedimento 30 vezes no total, armazenando as 30 médias no vetor vetor medias. Se possível, use um loop (laço) para fazer isso.

```
# vetor_medias <- numeric()

# for (i in 1:30) {

#  vetor_medias[i] <- mean(rnorm(100, mean=2, sd=1))

#}

#print(vetor_medias)
```

6

Plote o histograma (use a função geom histogram no ggplot) das médias. Para isso, crie um banco de dados (ggplot só aceita plotar variáveis de banco de dados) do seguinte modo:

```r
#df <- data.frame(medias = vetor_medias, sim_id = 1:30)
#hist(vetor_medias)
#mean(vetor_medias)
#var(vetor_medias)
```

Você reconhece a distribuição apresentada pelo histograma? Se sim, qual é ela? Consegue advinhar a média e desvio padrão da distribuição ou calculá-la?

Distribuição normal com média com média μ e variância $\frac{\sigma^2}{n}$.

7

Qual é a relação do histograma da questão anterior com o Teorema Central do Limite?

Pelo Teorema do Limite Central, para variáveis independentes e identicamente distribuídas, a distribuição amostral da média converge para a distribuição normal quando n aumenta, independentemente da distribuição populacional (distribuição da variável original).

8

Rode uma simulação de uma distribuição uniforme entre 0 e 10 e calcule a média. Repita o procedimento 30, 50 e 100 vezes e armazene as médias em um vetor. Faça um histograma dessas médias. Qual relação você estabelece com o Teorema Central do Limite?

Pelo Teorema do Limite Central, para variáveis independentes e identicamente distribuídas, a distribuição amostral da média converge para a distribuição normal quando n aumenta, independentemente da distribuição populacional (distribuição da variável original).

9

Realize uma simulação estatística para verificar a distribuição de probabilidade dos resultados do lançamento de uma moeda.

```r
library(tidyverse)
library(dplyr)
library(tidylog)

# Defina o número de lançamentos
```

```r
n <- 1000000

# Crie um vetor para armazenar os resultados das jogadas
X <- numeric()

# Suponha que "Cara" == "sair 1" & "Coroa" == "sair 2"

# Lance a moeda n vezes e armazene os resultados
set.seed(13492)
for (i in 1:n) {
  X[i] <- sample(1:2, size=1, replace = TRUE)
  }

X <- ifelse(X == 1, "Cara", "Coroa")

# Calcule as probabilidades
## Uma possibilidade
table(X)
```

```
## X
##   Cara  Coroa
## 500690 499310
```

```r
## Outra possibilidade:
sum(X=="Cara")/n # 50,069%
```

```
## [1] 0.50069
```

```r
sum(X=="Coroa")/n # 49,931%
```

```
## [1] 0.49931
```

10

Retire 10, 100, 1000 e 10000 valores de uma distribuição normal padrão e de uma distribuição binomial (n = 20, p = 0.7). Apresente os histogramas.

$$Z \sim \mathcal{N}(0,\, 1)$$

$$B \sim Bin(20,\, 0.7)$$

```r
# Defina o número de repetições
n_1 <- 10
n_2 <- 100
n_3 <- 1000
n_4 <- 10000

# Retiradas da normal
z1 <- rnorm(n_1, 0, 1)
z2 <- rnorm(n_2, 0, 1)
z3 <- rnorm(n_3, 0, 1)
z4 <- rnorm(n_4, 0, 1)

# Retiradas da binomial
b1 <- rbinom(n_1, 20, 0.7)
b2 <- rbinom(n_2, 20, 0.7)
b3 <- rbinom(n_3, 20, 0.7)
b4 <- rbinom(n_4, 20, 0.7)

# Apresente os histogramas
hist(z1, probability = TRUE)
lines(density(z1), col = 'blue')

hist(z2, probability = TRUE)
lines(density(z2), col = 'blue')

hist(z3, probability = TRUE)
lines(density(z3), col = 'blue')

hist(z4, probability = TRUE)
lines(density(z4), col = 'blue')
```

```r
hist(b1, probability = TRUE)

hist(b2, probability = TRUE)
```

Figure 1: Histogramas Z

(a) Z1 (b) Z2 (c) Z3 (d) Z4

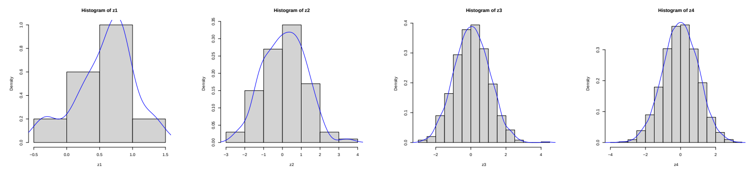
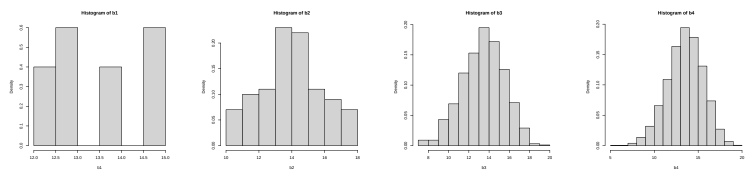


Figure 2: Histogramas B

(a) B1 (b) B2 (c) B3 (d) B4



```r
hist(b3, probability = TRUE)

hist(b4, probability = TRUE)
```

```r
# Limpe seu environment
rm(list=ls())
```

11

(OPCIONAL PARA A PÓS)

Considere o famoso Problema de Monty Hall:

Em um show de televisão, existem três portas, atrás das quais há duas cabras e um carro. Você escolhe uma das portas e ganha o que estiver atrás dela. Evidentemente, você deseja escolhar a porta que contém o carro. Uma vez que você tenha escolhido uma porta, o apresentador examina as outras duas e abre uma porta que contém uma cabra. Depois disso, restam duas portas. Você tem a oportunidade de mudar de porta.

Qual é a melhor estratégia para ganhar o carro? Faz diferença mudar ou não a porta escolhida? Avalie esse problema realizando uma simulação estatística. Explique o resultado.

```r
set.seed(11231)

# Defina o número de repetições
n <- 100000

# Defina as portas
portas <- c(1, 2, 3)

# Estabeleça o número inicial de vitórias (a ser atualizado ao longo da simulação)
n_vitorias <- 0

# Primeiro, considere o cenário em que você não muda a escolha da porta
for(i in 1:n) {

  vitoria <- sample(portas, 1, replace = TRUE) ## Porta com o carro

  escolha <-  sample(portas, 1, replace = TRUE) ## Pode ser qualquer uma das três portas

  if(escolha == vitoria)
  {
    porta_revelada <- sample(portas[-escolha],size = 1)
  } ## Se a porta escolhida é igual à porta vitoriosa, o apresentador retira uma das restan
  else
  {
    porta_revelada <- portas[-c(escolha, vitoria)]
  } ## Se a porta escolhida não é igual à vitoriosa, o apresentador retira uma porta não vi

  if(escolha == vitoria)
  {
    n_vitorias <- n_vitorias + 1
  } ## Se a porta escolhida é igual à vitoriosa, somo 1 no número de vitórias
}

## Para calcular a probabilidade de vitória, divida o número de vitórias pelo número de rep
n_vitorias/n # 33,275%

## [1] 0.33275
```

```r
# Agora considere o cenário em que você troca de porta após o apresentador abrir uma porta

## Defina uma nova variável para contar o número de vitórias
n_vitorias_2 <- 0

for(i in 1:n) {

  vitoria <- sample(portas, 1, replace = TRUE) ## Porta com o carro

  escolha <-  sample(portas, 1, replace = TRUE) ## Pode ser qualquer uma das três portas

  if(escolha == vitoria)
  {
    porta_revelada <- sample(portas[-escolha],size = 1)
  } ## Se a porta escolhida é igual à porta vitoriosa, o apresentador retira uma das restan
  else
  {
    porta_revelada <- portas[-c(escolha, vitoria)]
  } ## Se a porta escolhida não é igual à vitoriosa, o apresentador retira uma porta não vi

  nova_escolha <- portas[-c(escolha, porta_revelada)] # Mudo a porta escolhida após o apres

  if(nova_escolha == vitoria)
  {
    n_vitorias_2 <- n_vitorias_2 + 1
  } ## Se a nova porta escolhida é igual à vitoriosa, somo 1 no número de vitórias
}

## Para calcular a nova probabilidade de vitória, divida novamente o número de vitórias pel
n_vitorias_2/n # 66,752%

## [1] 0.66752

# Referências de soluções disponíveis na internet
# https://rpubs.com/nth-education/Monty_Hall_Simulation_R
# https://statisticsbyjim.com/fun/monty-hall-problem/
```

12

(OPCIONAL PARA A PÓS)

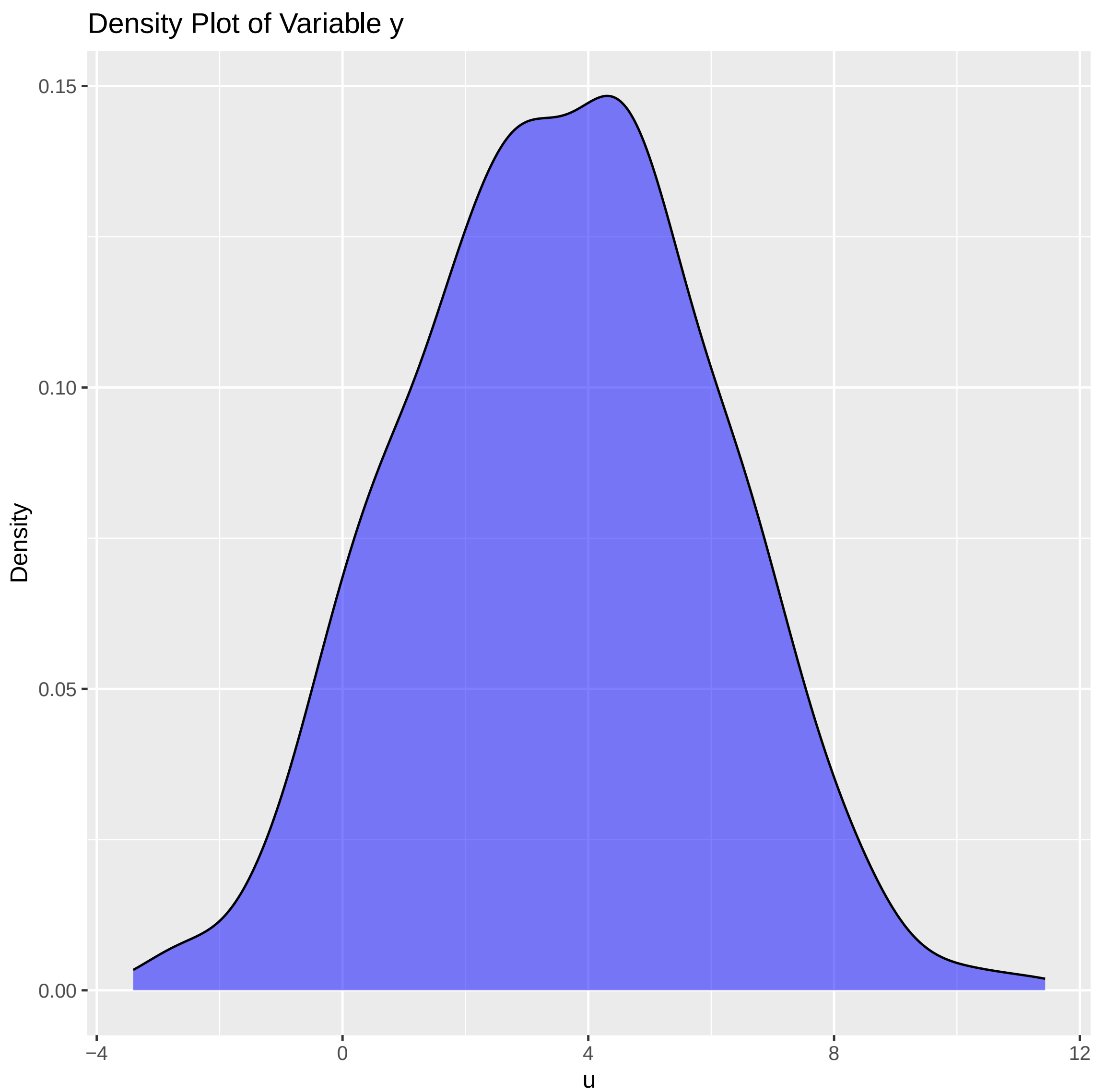
Identifique a distribuição da variável aleatória Y, considerando que:

$$Y = 0.75 \times X + u \tag{1}$$

$$X \sim \mathcal{N}(5, 3) \tag{2}$$

$$u \sim \mathcal{N}(0, 1) \tag{3}$$

```r
library(ggplot2)

set.seed(1238224)
# Defina o número de observações
n <- 1000

# Simule u e x (segue normal padrão)
u <- rnorm(n, mean = 0, sd = 1)
x <- rnorm(n, mean = 5, sd = 3)

# Escreva a equação
y <- 0.75*x + u

# Crie um banco de dados
df <- data.frame(y = y,
                 x = x,
                 u = u)

ggplot(data = df, aes(x = y)) +
  geom_density(fill = "blue", alpha = 0.5) +
  labs(title = "Density Plot of Variable y", x = "u", y = "Density")
```




```
shapiro.test(y) # não pode rejeitar a hipótese nula de normalidade

##
##  Shapiro-Wilk normality test
##
## data:  y
## W = 0.99802, p-value = 0.2932
```

A variável aleatória resultante da soma de variáveis que seguem distribuição normal também segue distribuição normal.

13

Apresente seus resultados em um arquivo PDF. Garanta que seu arquivo esteja limpo, contendo as respostas, os gráficos e as tabelas, mas não eventuais mensagens e erros. O arquivo PDF pode ser gerado diretamente a partir do R por meio do RMarkdown ou do RSweave. Para os alunos de graduação, isso é recomendado, mas não obrigatório. Adicionalmente, forneça o script para replicação.