Lista 2 - MAE 0399 - Análise de Dados e Simulação

Guilherme Ventura (11340293), Milton Leal (8973974), Richard Sousa (11810898)

25/05/2021

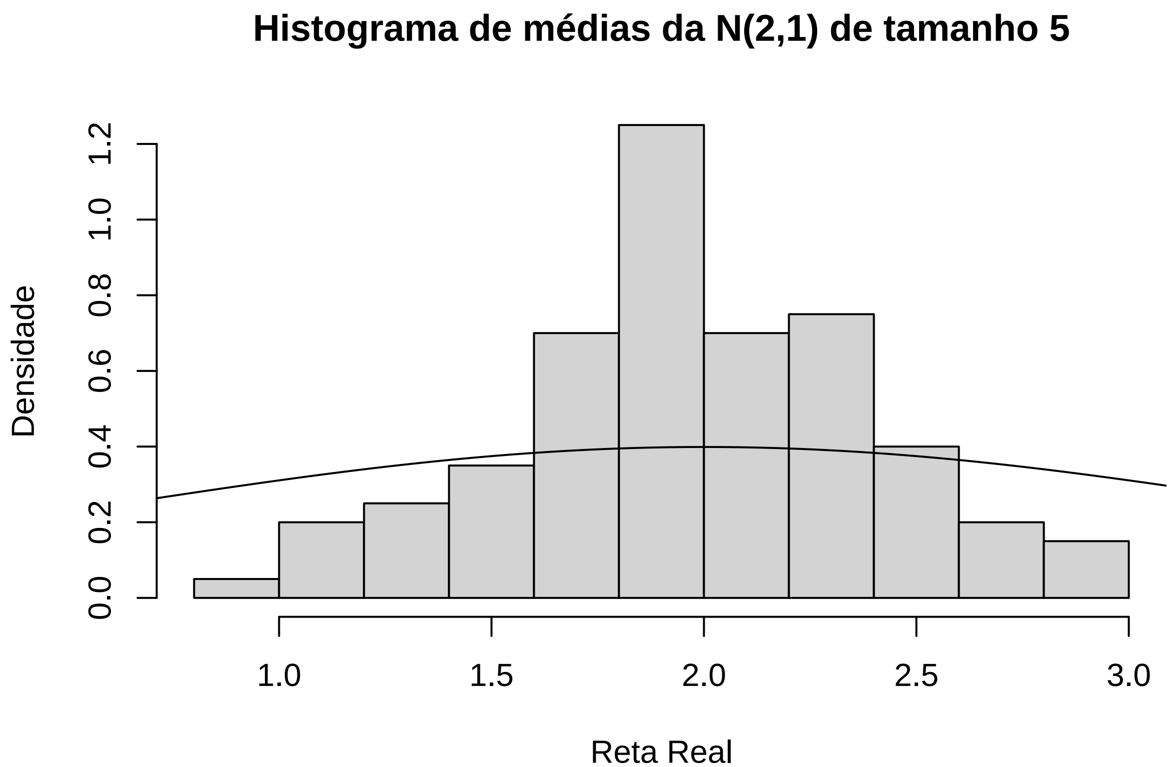
Exercício 2:

Para as distribuições N(2,1), Exp(2), U(1,3), faça 100 réplicas de amostras de tamanho n = 5, 10 e 50 e para cada réplica determine a média amostral. Apresente os histogramas das médias amostrais e compare as distribuições originais (desenhe as curvas das funções densidades de probabilidade).

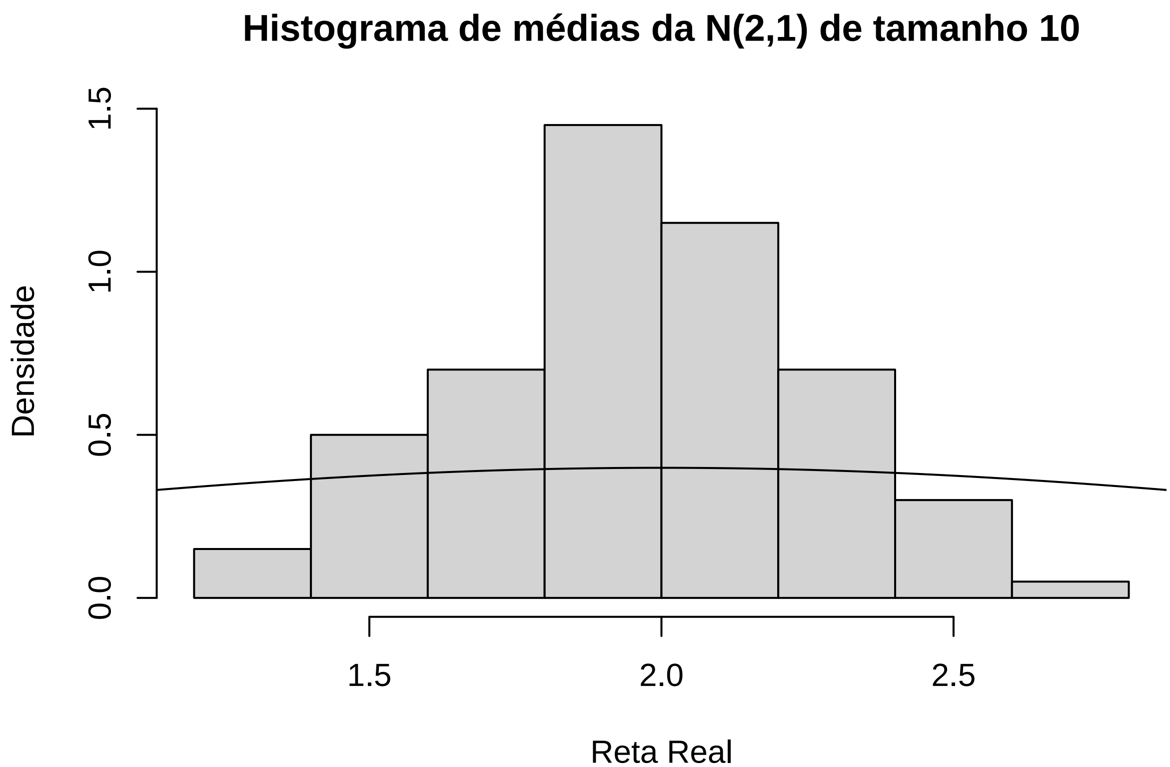
R:

N(2,1)

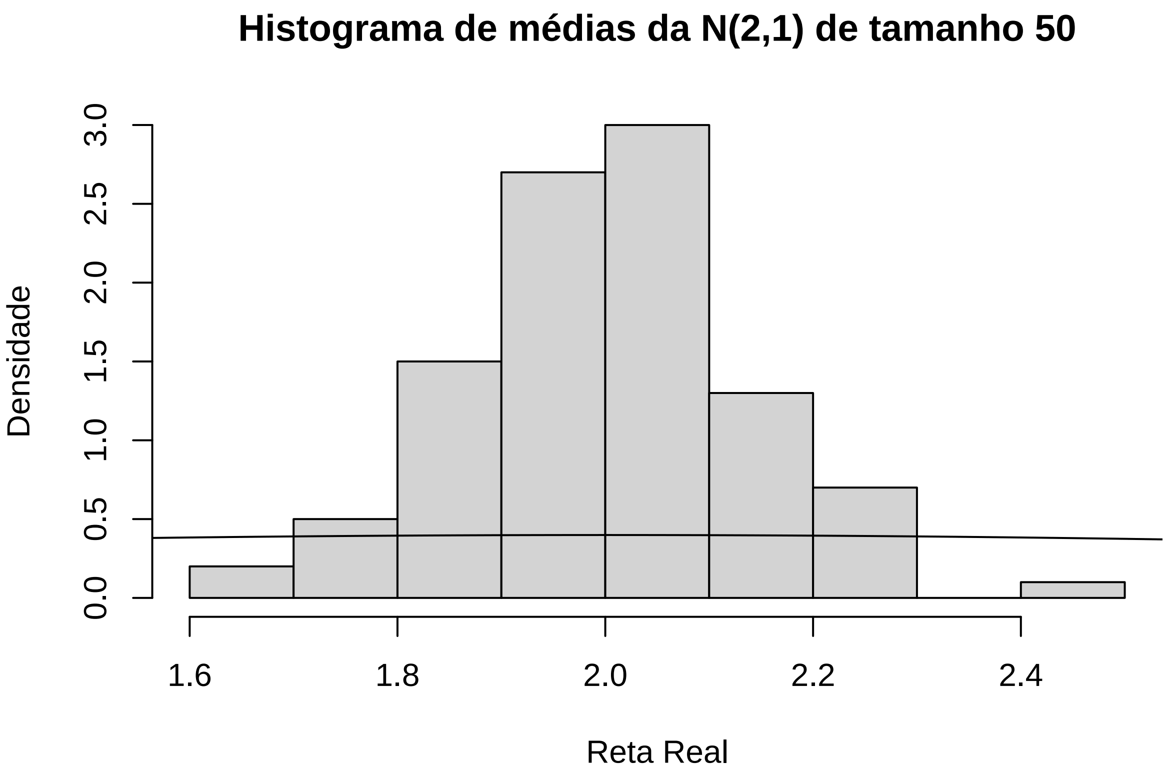
```r
set.seed(42)

#gera amostras de tamanhos 5, 10 e 50
normal_n5 = replicate(100, rnorm(5, 2,1))
normal_n10 = replicate(100, rnorm(10, 2,1))
normal_n50 = replicate(100, rnorm(50, 2,1))

#vetores com as médias das amostras
media_normal_n5 = c(colMeans(normal_n5))
media_normal_n10 = c(colMeans(normal_n10))
media_normal_n50 = c(colMeans(normal_n50))

#plota histograma dos dados simulados para amostra de tamanho 5
hist(media_normal_n5, probability = T,
     main = "Histograma de médias da N(2,1) de tamanho 5",
     ylab="Densidade",
     xlab="Reta Real")

#plota curva da função densidade
f = seq(-1,5, 0.001)
g = (1/sqrt(2*pi))*exp(-(1/2)*((f-2)^2)/2)
lines(f,g)
```




```r
#plota histograma dos dados simulados para amostra de tamanho 10
hist(media_normal_n10, probability = T,
     main = "Histograma de médias da N(2,1) de tamanho 10",
     ylab="Densidade",
     xlab="Reta Real")

f = seq(-1,5, 0.001)
g = (1/sqrt(2*pi))*exp(-(1/2)*((f-2)^2)/2)
lines(f,g)
```

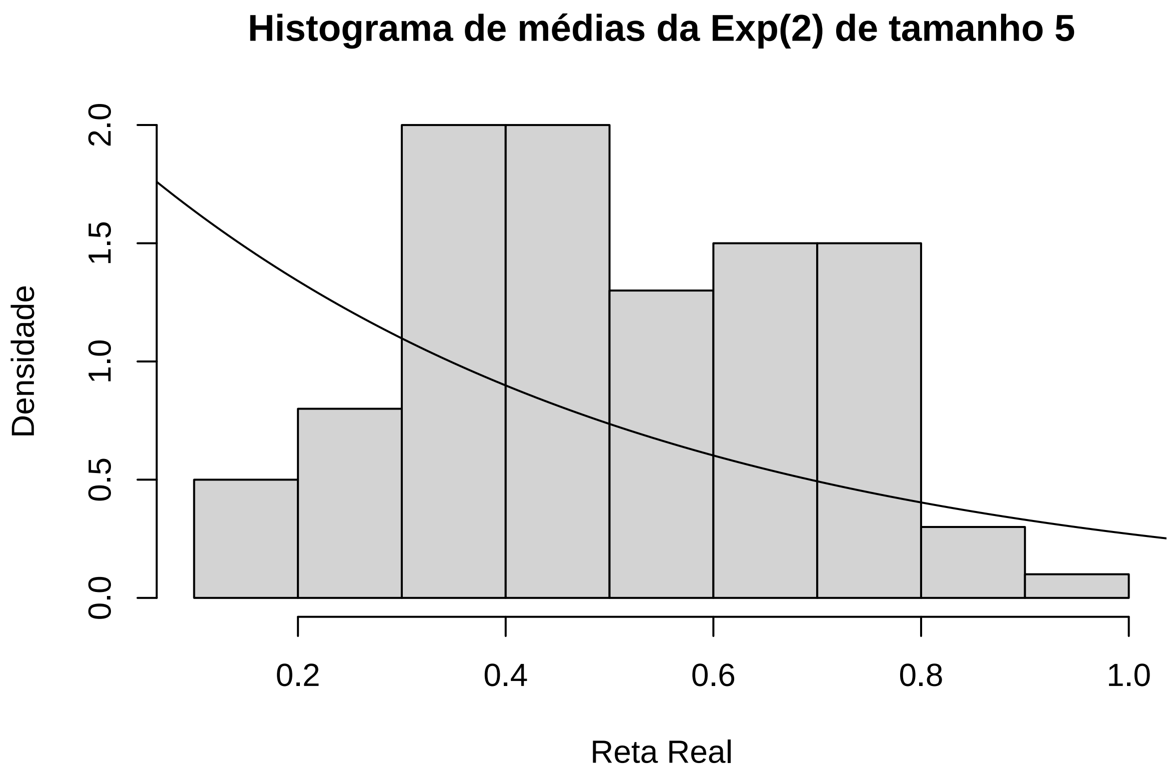



```r
#plota histograma dos dados simulados para amostra de tamanho 50
hist(media_normal_n50, probability = T,
     main = "Histograma de médias da N(2,1) de tamanho 50",
     ylab="Densidade",
     xlab="Reta Real")

f = seq(-1,5, 0.001)
g = (1/sqrt(2*pi))*exp(-(1/2)*((f-2)^2)/2)
lines(f,g)
```

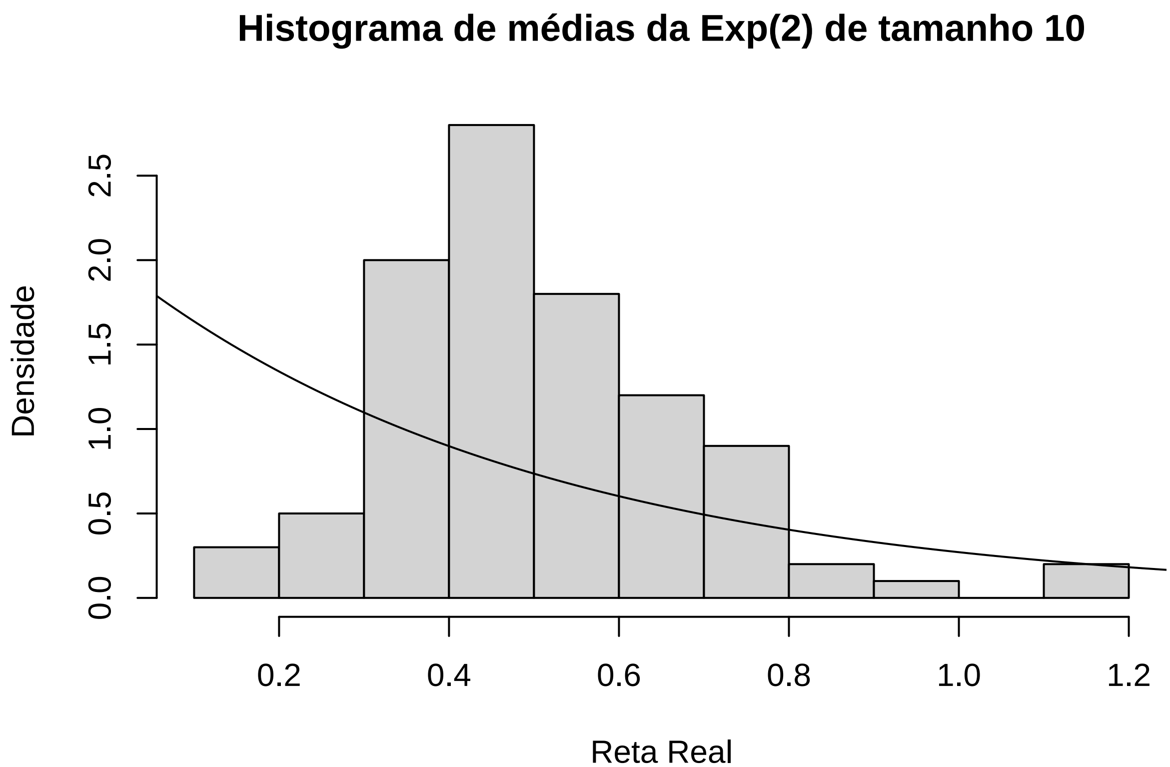


EXP(2)

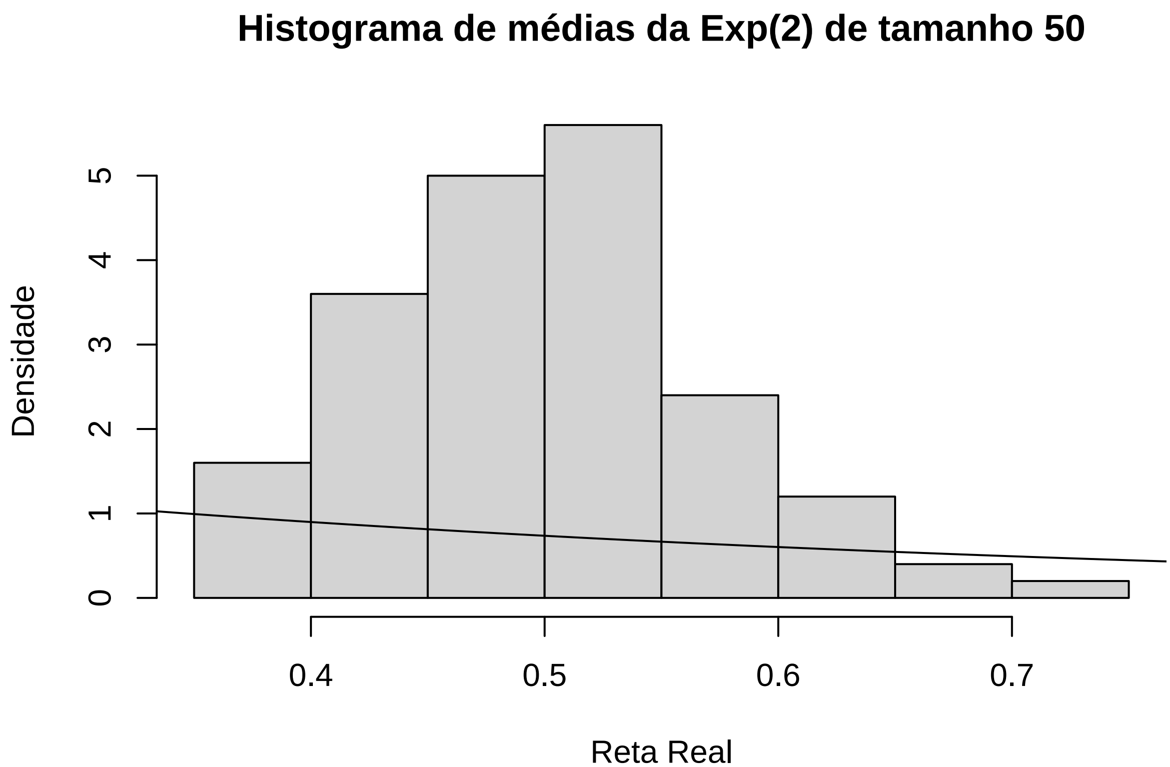
```
#gera amostras de tamanhos 5, 10 e 50
exp_n5 = replicate(100, rexp(5, 2))
exp_n10 = replicate(100, rexp(10, 2))
exp_n50 = replicate(100, rexp(50, 2))

#vetores com as médias das amostras
media_exp_n5 = c(colMeans(exp_n5))
media_exp_n10 = c(colMeans(exp_n10))
media_exp_n50 = c(colMeans(exp_n50))

#plota histograma dos dados simulados para amostra de tamanho 5
hist(media_exp_n5, probability = T,
     main = "Histograma de médias da Exp(2) de tamanho 5",
     ylab="Densidade",
     xlab="Reta Real")

#plota curva da função densidade
f = seq(0,5, 0.001)
g = 2*exp(-2*f)
lines(f,g)
```




```r
#plota histograma dos dados simulados para amostra de tamanho 10
hist(media_exp_n10, probability = T,
     main = "Histograma de médias da Exp(2) de tamanho 10",
     ylab="Densidade",
     xlab="Reta Real")

f = seq(0,5, 0.001)
g = 2*exp(-2*f)
lines(f,g)
```

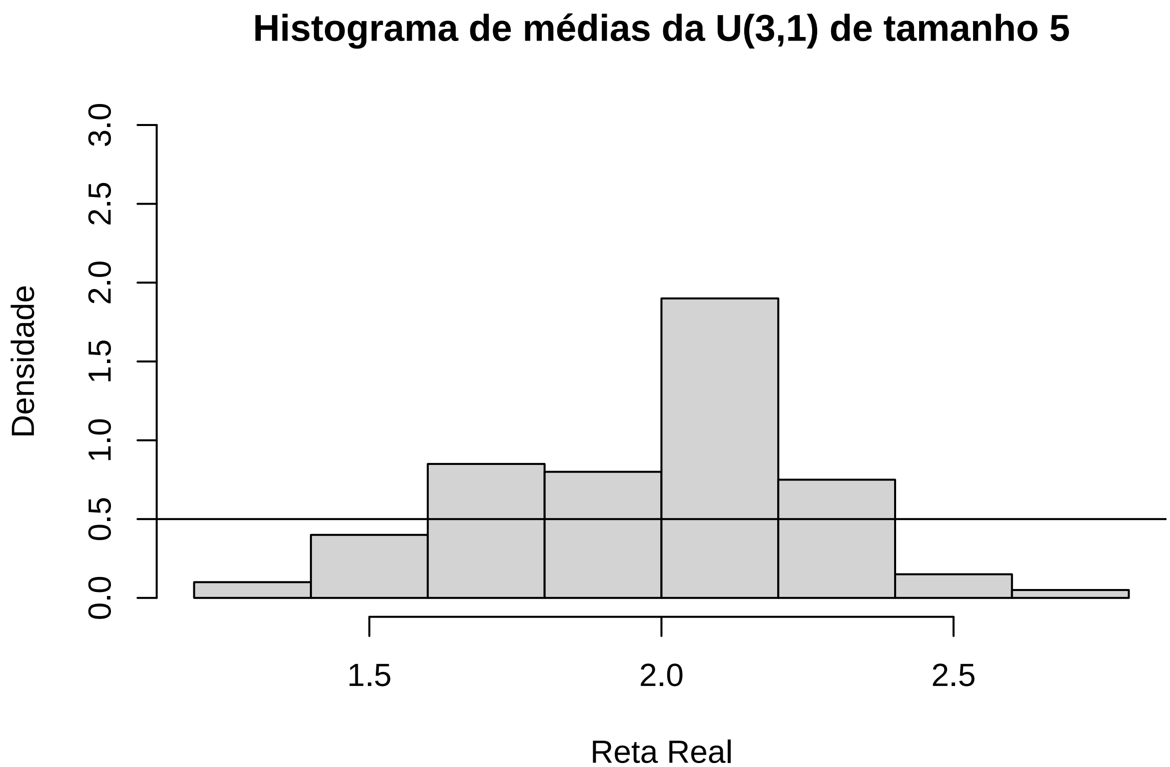



```r
#plota histograma dos dados simulados para amostra de tamanho 50
hist(media_exp_n50, probability = T,
     main = "Histograma de médias da Exp(2) de tamanho 50",
     ylab="Densidade",
     xlab="Reta Real")

f = seq(0,5, 0.001)
g = 2*exp(-2*f)
lines(f,g)
```

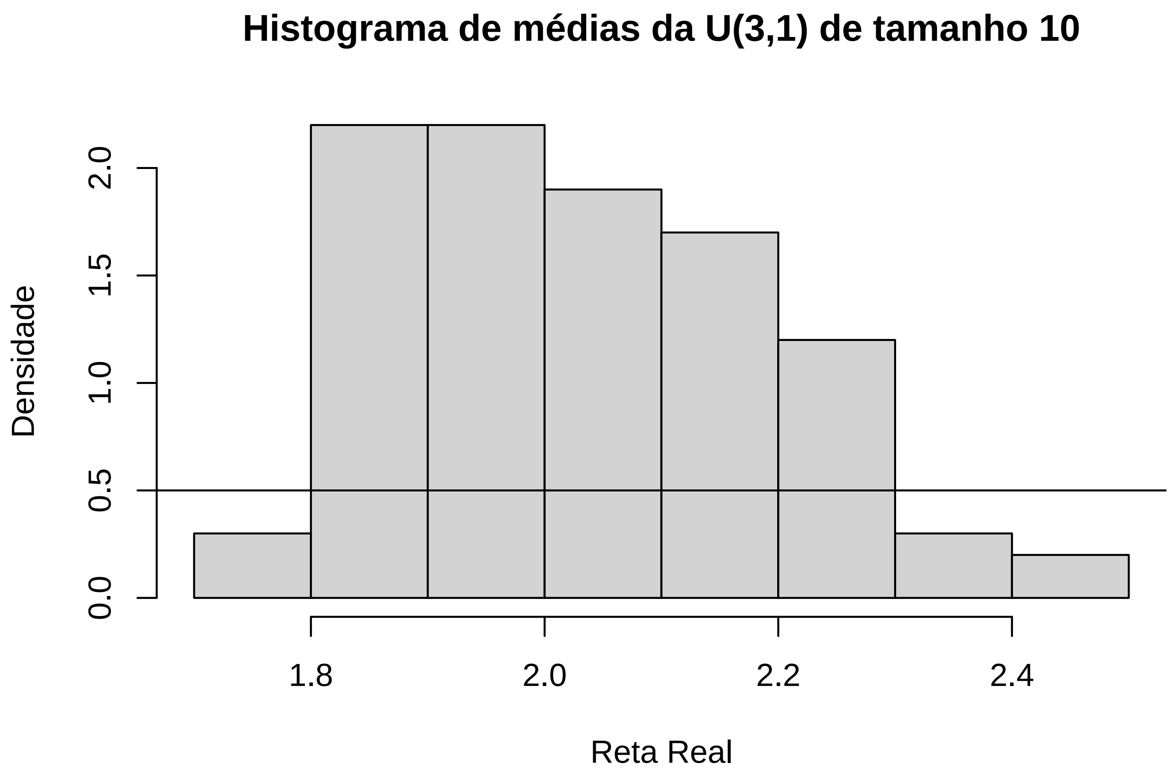


U(1,3)

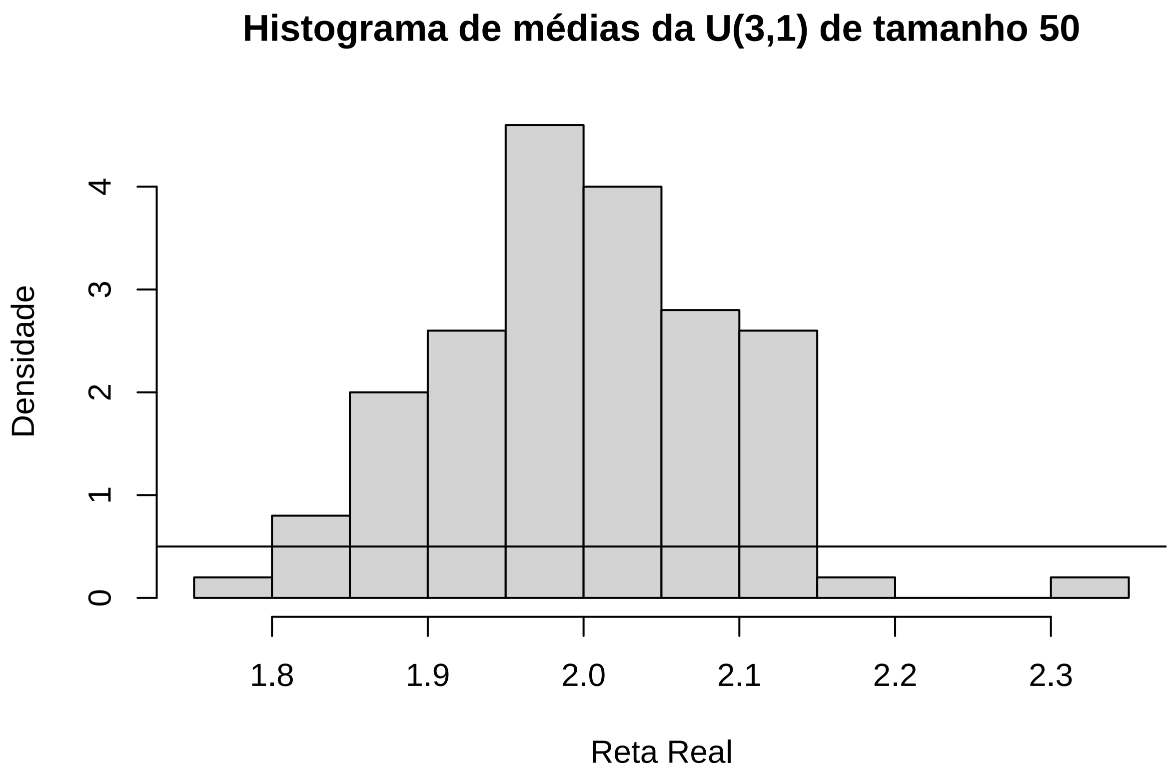
```r
#gera amostras de tamanhos 5, 10 e 50
u_n5 = replicate(100, runif(5,1,3))
u_n10 = replicate(100, runif(10,1,3))
u_n50 = replicate(100, runif(50,1,3))

#vetores com as médias das amostras
media_u_n5 = c(colMeans(u_n5))
media_u_n10 = c(colMeans(u_n10))
media_u_n50 = c(colMeans(u_n50))

#plota histograma dos dados simulados para amostra de tamanho 5
hist(media_u_n5, probability = T,
     main = "Histograma de médias da U(3,1) de tamanho 5",
     ylab="Densidade",
     xlab="Reta Real", ylim = c(0,3))

#plota curva densidade
abline(h=1/2)
```




```r
#plota histograma dos dados simulados para amostra de tamanho 10
hist(media_u_n10, probability = T,
     main = "Histograma de médias da U(3,1) de tamanho 10",
     ylab="Densidade",
     xlab="Reta Real")

abline(h=1/2)
```




```r
#plota histograma dos dados simulados para amostra de tamanho 50
hist(media_u_n50, probability = T,
     main = "Histograma de médias da U(3,1) de tamanho 50",
     ylab="Densidade",
     xlab="Reta Real")

abline(h=1/2)
```



Exercício 6:

É possível provar que se somarmos números aleatórios básicos até que a soma exceda 1, então o número esperado dessa soma é a constante e. Isto é,

$$N = min\left\{n : \sum_{i=1}^{n} U_i > 1\right\}$$

então $E[N] = e$.

a) Use este procedimento para estimar e. Faça 1000 simulações.

R:

```
set.seed(31415)

N <- c() #guarda quantidade de números básicos gerados por simulação

for (k in 1:1000){

        soma <- 0 #guarda a soma dos números básicos sorteados
        n <- 0 #conta a quantidade de números básicos gerados

        while (soma <= 1){ #verifica se a soma excedeu 1

                n <- n + 1 #acrescenta ao contador
                soma <- soma + runif(1) #acrescenta à soma e gera o número básico
```

```
        }
        N[k] <- n
}
E <- mean(N) #média das quantidades de números básicos gerados
E
```

```
## [1] 2.717
```

O valor simulado nessa execução foi $E[N] = 2.717$.

b) Estime a variância do estimador usado em (a) e apresente o intervalo de 95 % de confiança para e.

R:

Para essas simulações obtivemos a seguinte variância amostral:

```
variancia <- var(N)
variancia
```

```
## [1] 0.6855966
```

Com $\hat{e}$ sendo o estimador e s o desvio padrão amostral, o intervalo de confiança é dado por:

$$IC(\gamma; \hat{e}) = \left[\hat{e} - z_\gamma \frac{s}{\sqrt{n}};\ \hat{e} + z_\gamma \frac{s}{\sqrt{n}}\right]$$

Com $\gamma = 95\%$ tem-se $z_\gamma = 1.96$

```
s <- sqrt(variancia) #calcula desvio padrão

inf <- E - 1.96 * s / sqrt(length(N)) #calcula limite inferior
sup <- E + 1.96 * s / sqrt(length(N)) #calcula limite inferior

IC <- c(inf, sup)
IC
```

```
## [1] 2.66568 2.76832
```

Ou seja, com 95 de confiança, o intervalo aleatório entre 2.66568 e 2.76832 contém o verdadeiro valor de e.

Exercício 9:

Sejam $X_1, ..., X_n$ v.a. independentes e identicamente distribuídas com média μ. Para duas constantes tais que $a < b$, estamos interessados em estimar $p = P[a < \bar{X} - \mu < b]$.

a) Explique como usar o método de *bootstrap* para estimar p.

R:

Primeiramente, precisamos determinar a distribuição da variável aleatória $V = \bar{X} - \mu$.

Para isso, vamos considerar que $\bar{X}$ é uma proxy para μ e lançaremos mão do método de *bootstrap* para gerar B reamostragens. Calculemos a média de cada uma dessas reamostras e as denotermos por $\bar{X_i}^*$.

Dessa forma, a distribuição estimada de V será um vetor de tamanho B composto pelas diferenças $\bar{X_i}^* - \bar{X}$ para $i \in (1, ..., n)$.

Agora, se definirmos uma variável aleatória indicadora Y tal que para cada $\bar{X_i}^*$ seja:

$$Y_i = \begin{cases} 1, & \text{se } a < \bar{X_i}^* < b \\ 0, & \text{caso contrário,} \end{cases}$$

então p pode ser estimado como:

$$\hat{p} = \frac{\sum_i^n Y_i}{B}.$$

b) Considere $n = 10, a = -5$ e $b = 5$. Estime p considerando que os valores observados são: $56, 101, 78, 67, 93, 87, 64, 72, 80$ e 69.

R:

```r
set.seed(32)

dados = c(56,101,78,67,93,87,64,72,80,69) #amostra de dados
a = 5 #limite superior
b = -5 #Limite inferior

media_dados = mean(dados)

amostras_boot = 1000 #número B de amostras de Bootstrap

#calcula a média de 1000 amostras de Bootstrap com reposição
medias_boot = replicate(amostras_boot, mean(sample(dados, 10, replace = TRUE)))

#distribuição da v.a. V
V = medias_boot - media_dados

#soma a quantidade de TRUEs que verificiam a confição e divide pelo total
p_hat = sum(V > -5 & V < 5)/amostras_boot

p_hat
```

```
## [1] 0.756
```