Lista de Exercícios IPE #1

Taiguara Melo Tupinambás

Entrega: 04 de abril de 2017

As referências a seções, equações, figuras, exemplos e exercícios são do livro *Intuitive Probability and Random Processes using MATLAB* de Stephen M. Kay, Springer, 2006.

Exercício 1:

Foi considerado como "quaisquer duas pessoas da família estarem em casa simultaneamente" apenas duas pessoas quaisquer. O evento quaisquer três ou as quatro estarem em casa foram descartados.

Esse entendimento é semelhante ao que foi considerado na seção 1.3 do livro, quando é tratado a probabilidade de 3 pessoas estarem ao telefone, como apenas 3 (o evento 4 pessoas estarem ao telefone não fazia parte do cálculo da probabilidade)

a) O modelo probabilístico adotado para solução do problema é dado pela lei binomial de probabilidade, cuja fórmula é apresentado em (3.28). O cálculo da probabilidade é apresentado a seguir

```
Pk=factorial(4)/(factorial(4-2)*factorial(4-2))*0.4^2*0.6^2
print(Pk)
```

```
## [1] 0.3456
```

b) cálculo da probabilidade via simulação é apresentado a seguir:

```
A2<-0 #inicializa a variável de contagem dos eventos
n<-10000 #número de experimentos

for (i in 1:n) {
        Adultos<-c(0,0,0,0) #cria vetor de adultos com 0 (1: em casa, 0: fora)
        sim<-c(runif(1),runif(1),runif(1),runif(1)) #vetor de números aleatórios
        for (j in 1:length(sim))
                if (sim[j]>0.6) { #probabilidade de estar em casa
                        Adultos[j]<-1
                }
        if (sum(Adultos)==2) { #experimento com 2 adultos em casa
                A2<-A2+1
        }
}

print(A2/n) #calcula a frequência relativa do experimento
```

```
## [1] 0.3531
```

Exercício 2:

A função densidade de probabilidade da distribuição (pdf) gaussiana é dada por (10.7), conforme:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp[-\frac{1}{2}(\frac{x-\mu}{\sigma})^2], x \in [-\infty, \infty]$$

Para calcular a probabilidade de um intervalo em uma pdf, a solução é dada por (10.4), conforme equação:

$$\Pr(x \in (a,b)) = \int_a^b \rho(x)dx.$$

Uma aproximação numérica pode ser obtido pela fórmula a seguir (quanto menor o delta, melhor é a aproximação):

$$\sum_{n={}^{L}1/\Delta}^{{}^{L}2/\Delta} \frac{1}{\sqrt{2\pi\sigma^2}} \exp[-\frac{1}{2}(\frac{n\Delta-\mu}{\sigma})^2]\Delta$$

Para o problema em questão, com média 7, desvio padrão unitário e evento 5 <= T <= 6, a aproximação é dada por:

$$\sum_{n=\frac{5}{\Delta}}^{\frac{6}{\Delta}} \frac{1}{\sqrt{2\pi}} \exp[-\frac{1}{2}(n\Delta-7)^2]\Delta$$

A solução utilizando R para o cálculo numérico deste problema, com um delta=0.0001, é apresentado a seguir

```
d<-0.0001 #define o valor de delta
p<-0 #inicializa a variável probabilidade
L1<-5/d #calcula o limite inferior
L2<-6/d #calcula o limite superior

for (n in L1:L2) {
        p<-p+1/sqrt(2*pi)*exp(-0.5*(n*d-7)^2)*d #calcula a aproximação recursivamente
}

print(p)
```

```
## [1] 0.1359199
```

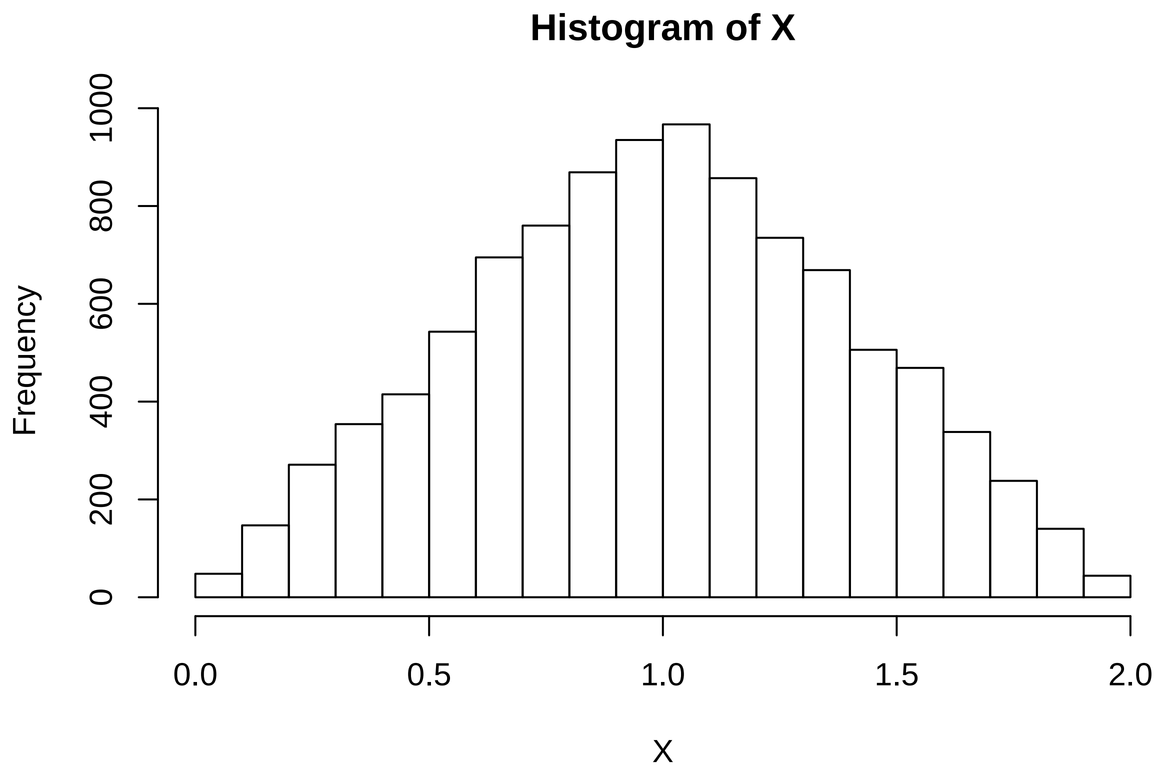
Exercício 3:

Para a solução do problema, foi criado um vetor com n somas de valores aleatórios entre 0,1 e calculado as frequências relativas, conforme código abaixo (para n=10000).

```
n=10000 #número de experimentos

X <- runif(n,0,1)+runif(n,0,1)

hist(X)
```




```
Xa <- sum(X<0.5)/length(X)
Xb <- sum(X>=0.5 & X<1)/length(X)
Xc <- sum(X>=1 & X<1.5)/length(X)
Xd <- sum(X>=1.5 & X<=2)/length(X)
Xe <- sum(X>2)/length(X)


cat("a)",Xa,"\nb)",Xb,"\nc)",Xc,"\nd)",Xd,"\ne)",Xe)
```

```
## a) 0.1235
## b) 0.3802
## c) 0.3734
## d) 0.1229
## e) 0
```

Exercício 4:

Para a solução do exercício 4 foi criado um código de forma diferente, com um *for* de 1 até 20 (quantidade de intervalos) em que era subtraído 0.5 do vetor Y a cada iteração e verificado a quantidade de experimentos que teria valor negativo, atribuindo o resultado a um vetor auxiliar.

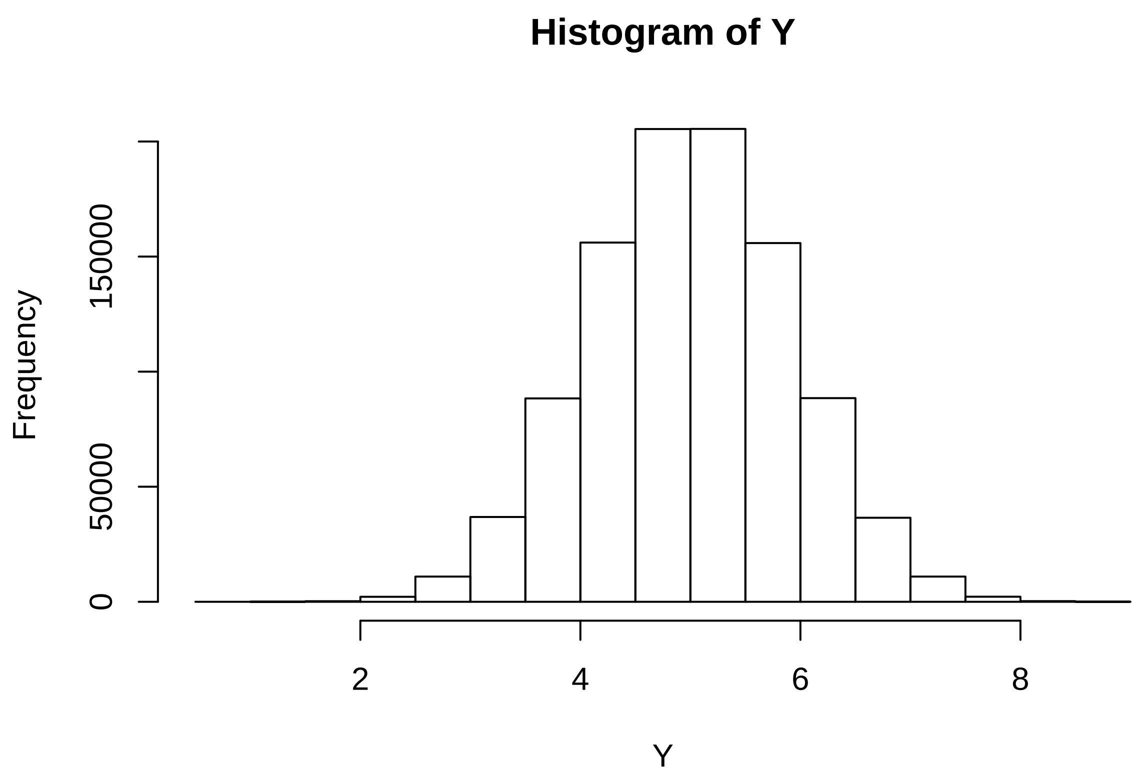
Cada elemento desse vetor contem a quantidade de experimentos com valor menor do que o limite superior de cada intervalo. **Ex.:** o elemento *Paux[5]* representa a quantidade de experimentos com valor menor do que 2.5, enquanto que o *Paux[4]* representa a quantidade de experimentos com valor menor do que 2. Para achar o vetor *Py*, com a frequência de cada intervalo, era só subtrair um elemento do vetor *Paux* de seu elemento anterior, para encontra o “saldo” daquele intervalo. Ao final, o vetor *Py* foi dividido pela quantidade de experimentos, para encontrar a frequência relativa de cada intervalo.

Interessante ressaltar a observação de que a pdf resultante da soma de distribuições uniformes tende para uma distribuição gaussiana. E que quanto maior a soma, mais próximo da gaussiana e menor o desvio padrão.

```
n<-1000000 #numero de experimentos
Y<-0

for (i in 1:10){
        Y<-Y+runif(n,0,1)
}

hist(Y)
```




```
Py<-c(1:20)-c(1:20) #inicializando o vetor das probabilidades
Paux<-Py

for (i in 1:length(Py)) {
        Y <- Y-0.5
        Paux[i]<-sum(Y<0) #experimentos < limite superior do intervalo
        if (i>1) {
                Py[i]<-(Paux[i]-Paux[i-1])/n #remove experimentos < limite inferior
        }
        else {
                Py[i]<-Paux[i]/n #calculando a frequência relativa
        }
}
```

```r
intervals<-c("[0 0.5)","[0.5 1)","[1 1.5)","[1.5 2)","[2 2.5)",
             "[2.5 3)","[3 3.5)","[3.5 4)","[4 4.5)","[4.5 5)",
             "[5 5.5)","[5.5 6)","[6 6.5)","[6.5 7)","[7 7.5)",
             "[7.5 8)","[8 8.5)","[8.5 9)","[9 9.5)","[9.5 10]")

result<-data.frame(intervals,Py)

print(result)
```

```
##    intervals       Py
## 1    [0 0.5) 0.000000
## 2    [0.5 1) 0.000001
## 3    [1 1.5) 0.000018
## 4    [1.5 2) 0.000259
## 5    [2 2.5) 0.002171
## 6    [2.5 3) 0.010945
## 7    [3 3.5) 0.036852
## 8    [3.5 4) 0.088383
## 9    [4 4.5) 0.156093
## 10   [4.5 5) 0.205434
## 11   [5 5.5) 0.205492
## 12   [5.5 6) 0.155877
## 13   [6 6.5) 0.088502
## 14   [6.5 7) 0.036517
## 15   [7 7.5) 0.010956
## 16   [7.5 8) 0.002206
## 17   [8 8.5) 0.000276
## 18   [8.5 9) 0.000018
## 19   [9 9.5) 0.000000
## 20  [9.5 10] 0.000000
```

Exercício 5:

Foram feitos cálculos para o número de Bell de acordo com as duas fórmulas:

```r
N_infty=100 #limite superior do somatório
N_seq=10 #quantidade de números de Bell a calcular

##Fórmula 1:
Bn <- c(1:N_seq)-c(1:N_seq)
Bn[1]<-1;
for (n in 1:(N_seq)) {
        for (k in 1:N_infty) {
                Bn[n+1]<-Bn[n+1]+k^n/factorial(k) #i+1 - índice 0 não existe no R
        }
        Bn[n+1]<-Bn[n+1]/exp(1)
}

print(toString(Bn[1:10]))
```

```
## [1] "1, 1, 2, 5, 15, 52, 203, 877, 4140, 21147"
```

```r
##Fórmula 2:
Bnn <- c(1:N_seq)-c(1:N_seq)
```

```r
aux<-Bnn

Bnn[1] <- 1 #inicializa os dois primeiros elementos do vetor / casos especiais
Bnn[2] <- 1

comb = function(n, x) { #função para análise combinatória
        return(factorial(n) / (factorial(x) * factorial(n-x)))
}


for (n in 2:(N_seq)) {
        aux[1]<-1
        for (k in 2:n) {
                aux[k]<-comb(n-1,k-1)*Bnn[k]
        }
        Bnn[n+1]<-sum(aux)
}

print(toString(Bnn[1:10])) #trunca os números e imprimie como string, para melhor visualização
```

```
## [1] "1, 1, 2, 5, 15, 52, 203, 877, 4140, 21147"
```

Realizando o cálculo para o espaço amostral igual a {1,2,3,4}, temos que:

```r
print(Bnn[4+1]) #considerando que os números começam para n=0
```

```
## [1] 15
```

Exercício 6:

Considerando que cada evento é um tipo de resultado diferente (A C B B) =! (B B), o código e o resultado das probabilidades de cada evento são apresentados a seguir:

```r
n=10000
eventos=0
for (i in 1:n) {
        x=0
        if (runif(1)<0.5) { #A ganha de B e joga com C
                x="A"
                if (runif(1)<0.5) { #A ganha de C e é o campeão
                        x=paste(x,"A")
                } else {
                        x=paste(x,"C") #C ganha de A e joga com B
                        if (runif(1)<0.5) {
                                x=paste(x,"C") #C ganha e é o campeão
                        }
                        else { #B ganha e joga com A
                                x=paste(x,"B")
                                if (runif(1)<0.5) { #B ganha e é o campeão
                                        x=paste(x,"B")
                                } else {
                                        x=paste(x,"A") #A ganha e é o campeão
                                }
                        }
                }
```

```
        } else { #B ganha de A e joga com C
                x="B"
                if (runif(1)<0.5) { #B ganha de C e é o campeão
                        x=paste(x,"B")
                } else {
                        x=paste(x,"C") #C ganha de B e joga com A
                        if (runif(1)<0.5) {
                                x=paste(x,"C") #C ganha e é o campeão
                        }
                        else { #A ganha e joga com B
                                x=paste(x,"A")
                                if (runif(1)<0.5) { #A ganha e é o campeão
                                        x=paste(x,"A")
                                } else {
                                        x=paste(x,"B") #B ganha e é o campeão
                                }
                        }
                }
        }
        eventos[i]=x
}
eventos<-data.frame(table(eventos))
eventos["prob"]<-eventos$Freq/n
print(eventos)
```

```
##   eventos Freq   prob
## 1     A A 2590 0.2590
## 2 A C B A  621 0.0621
## 3 A C B B  624 0.0624
## 4   A C C 1244 0.1244
## 5     B B 2477 0.2477
## 6 B C A A  595 0.0595
## 7 B C A B  634 0.0634
## 8   B C C 1215 0.1215
```

Exercício 7:

A seguir são resolvidos os problemas computacionais do capítulo 2 do livro do Kay.

Exercício 2.1:

O resultado (3/4 para 0 e 1/4 para 1) era esperado, pois há 3 combinações de resultados para os eventos cujo produto é 0 ({0,0},{0,1},{1,0}), enquanto há apenas 1 evento cujo produto é 1 ({1,1})

```
n<-100000

X1<-runif(n,0,1)
X2<-runif(n,0,1)

Y<-round(X1)*round(X2)

Prob1<-sum(Y==1)/length(Y)
Prob0<-sum(Y==0)/length(Y)
```

```
cat("Probability of Y=1",Prob1,"\nProbability of Y=0",Prob0)
```

```
## Probability of Y=1 0.2484
## Probability of Y=0 0.7516
```

Exercício 2.2:

De forma similar ao Exercício 2.1, há apenas um evento dos quatro possíveis que retorna *snake eyes* ({1,1}). Assim, era de se esperar que o resultado fosse 1/4

```
n<-100000

X1<-runif(n,0,1)
X2<-runif(n,0,1)

Y<-round(X1)+round(X2)

snake_eyes<-sum(Y==2)/length(Y)

cat("Probability of snake eyes is:", snake_eyes)
```

```
## Probability of snake eyes is: 0.25074
```

Exercício 2.3:

O valor verdadeiro para a probabilidade $-1 < X < 1$ é de 0.68268. Usando o mesmo código do Exercício 2, alterando os valores de mi e dos limites do somatórios, temos que:

```
d<-0.000001 #define o valor de delta
p<-0 #inicializa a variável probabilidade
L1<--1/d #calcula o limite inferior
L2<-1/d #calcula o limite superior

for (n in L1:L2) {
        p<-p+1/sqrt(2*pi)*exp(-0.5*(n*d)^2)*d
}

print(p)
```

```
## [1] 0.6826897
```

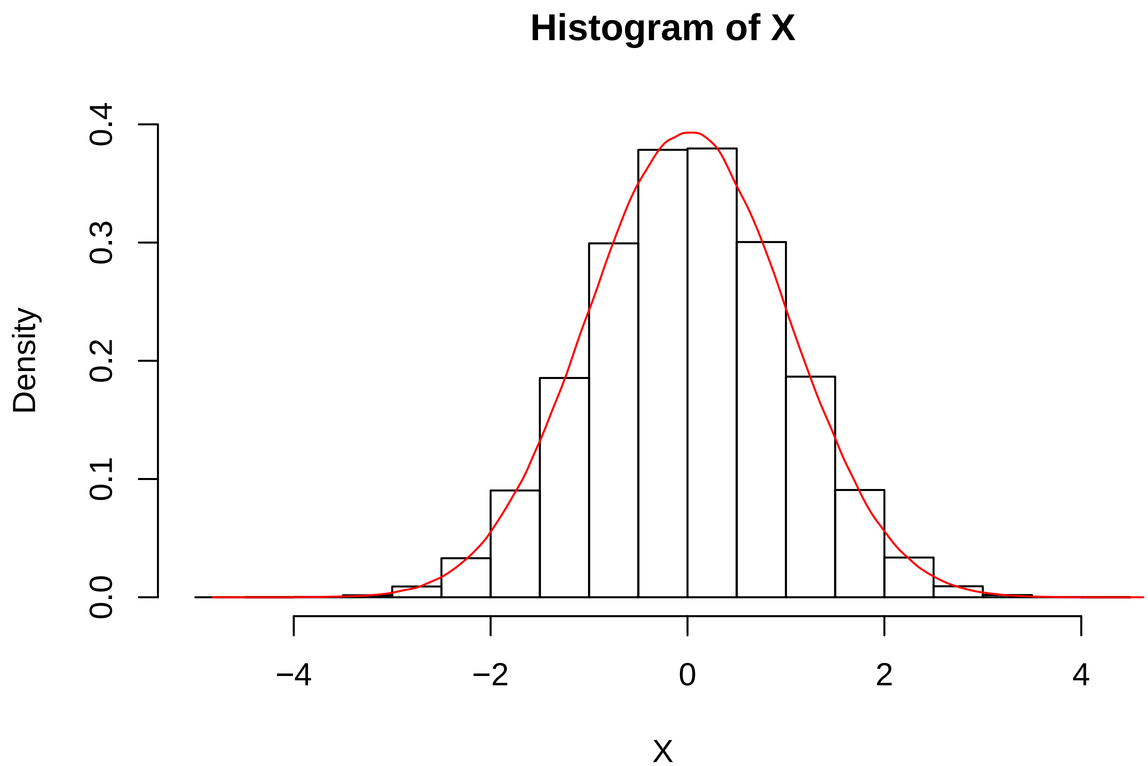
Exercício 2.4:

Ao comparar a estimativa de PDF para a VA X com a PDF da gaussiana de média 0 e desvio padrão unitário, nota-se que são muito semelhantes. Isso reforça a observação feita no Exercício 4, de que a soma de várias VAs de distribuição uniforme tendem a uma VA com distribuição gaussiana.

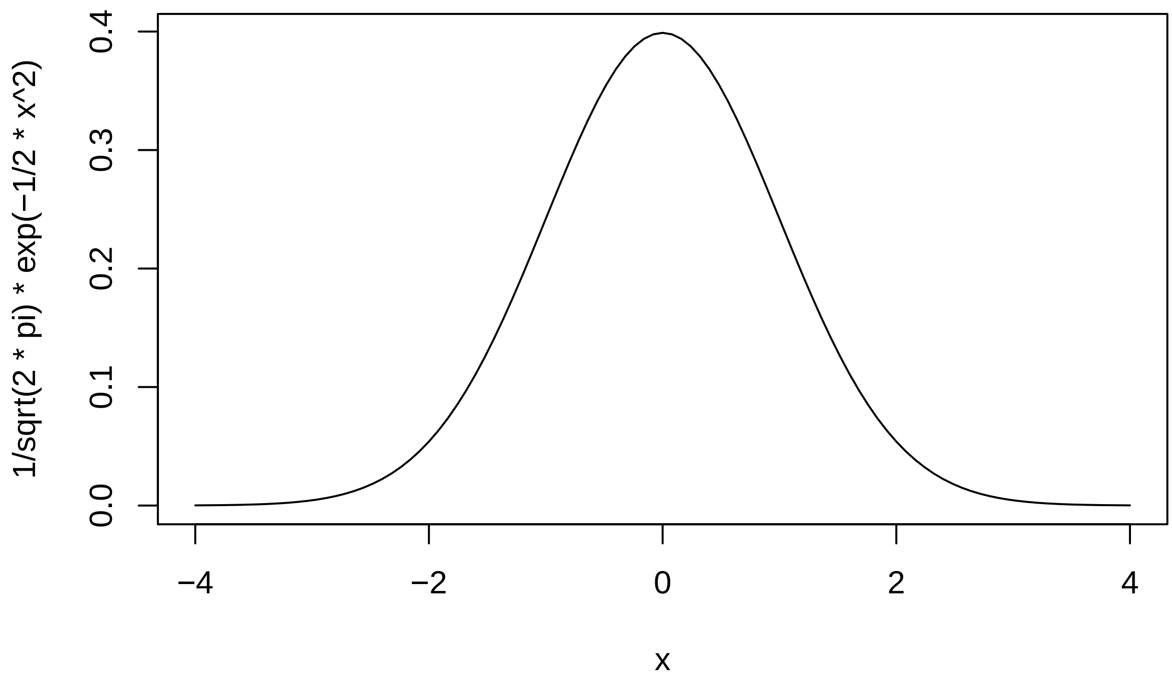
```
n<-1000000 #número de experimentos
X<-0

for (i in 1:12) {
        X<-X+(runif(n,0,1)-0.5)
}

hist(X, probability=TRUE, ylim=c(0,0.4))
lines(density(X),col="red")
```




```
curve(1/sqrt(2*pi)*exp(-1/2*x^2),from=-4, to=4)
```

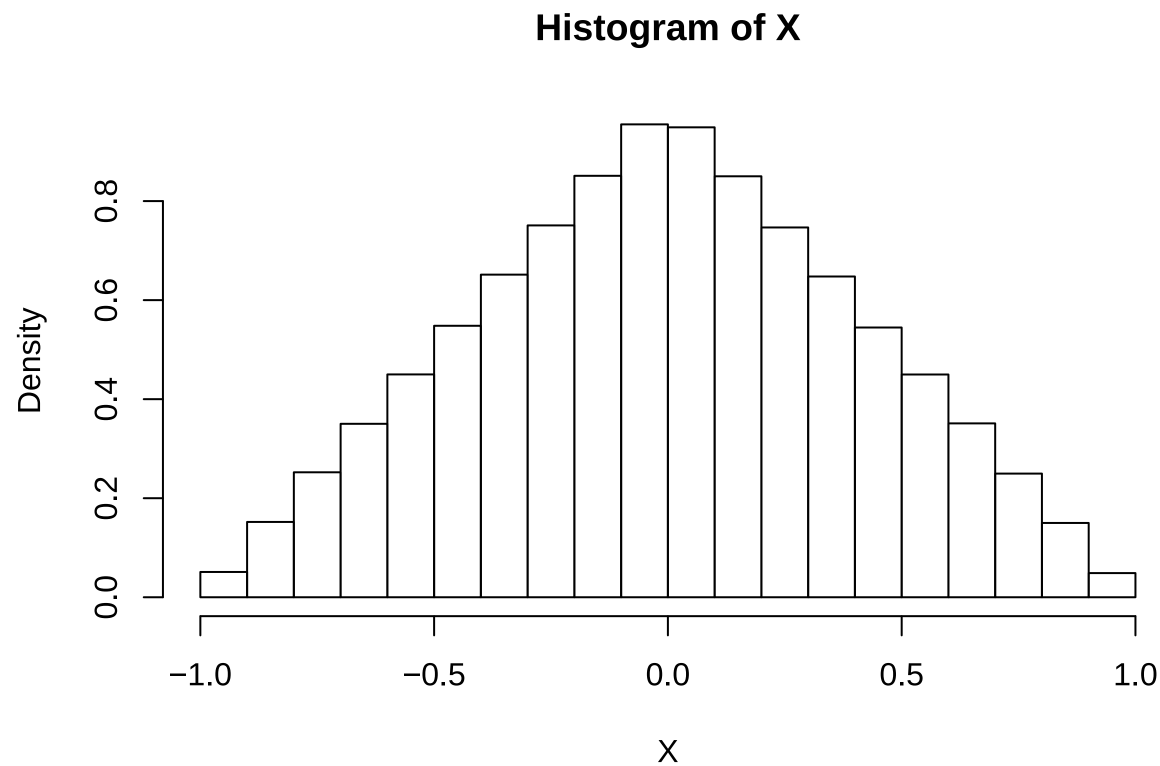


Exercício 2.5:

Os valores mais prováveis são os ao redor de zero (negativo e positivo):

```
n<-1000000 #número de experimentos
X<-runif(n,0,1)-runif(n,0,1)

hist(X,probability=TRUE)
```

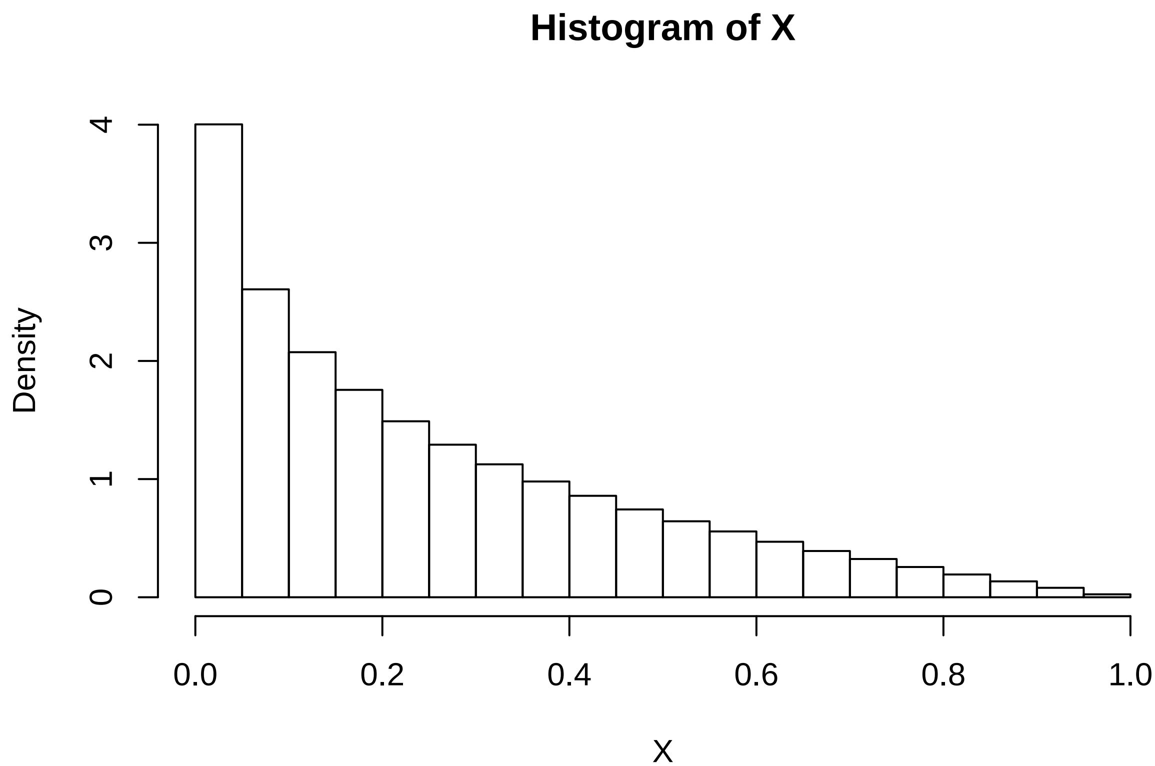


Exercício 2.6:

Os valores mais prováveis são os próximos a 0 (positivos):

```
n<-1000000 #número de experimentos
X<-runif(n,0,1)*runif(n,0,1)

hist(X,probability=TRUE)
```



Exercício 2.7:

O resultado, conforme esperado, é próximo a p1 para X1, p2 para X2 e p3 para X3.

```
n<-10000
r<-runif(n,0,1)
x<-0

for (i in 1:n) {
        if (r[i]<0.1) {
                x[i]<-1
        } else if (r[i]<0.3) {
                x[i]<-2
        } else {
                x[i]<-3
        }
}

probs<-c(sum(x==1)/n,sum(x==2)/n,sum(x==3)/n)
cat("Prob X=1 is",probs[1],"\nProb X=2 is",probs[2],"\nProb X=3 is",probs[3])
```

```
## Prob X=1 is 0.1073
## Prob X=2 is 0.2021
## Prob X=3 is 0.6906
```

Exercício 2.8:

Para uma distribuição uniforme, o valor verdadeiro é a média entre os limites da VA. No caso do problema, para uma VA entre 0 e 1, o valor verdadeiro é 0.5. O resultado da simulação foi muito próximo:

```
n<-10000
x<-runif(n,0,1)
print(mean(x))
```

```
## [1] 0.4984356
```

Exercício 2.9:

Considerando X uma VA gaussiana, com média 0 e desvio padrão unitátio, X + 1 terá um valor verdadeiro da média igual a 1. O resultado da simulação foi um valor muito próximo:

```
n<-10000
x<-rnorm(n,0,1)+1
print(mean(x))
```

```
## [1] 0.9954648
```

Exercício 2.10:

Considerando X uma VA gaussiana, com média 0 e desvio padrão unitátio, X^2 terá um valor médio igual a 1, sendo os valores próximos a 0 os mais prováveis e todos valores positivos.

```
n<-10000
x<-rnorm(n,0,1)^2
print(mean(x))
```

```
## [1] 0.9866529
```

Exercício 2.11:

Considerando X uma VA uniforme, entre 0 e 1, 2*x deverá ter média 1. O resultado da simulaçã é muito próximo disso

```
n<-10000
x<-runif(n,0,1)*2
print(mean(x))
```

```
## [1] 1.000166
```

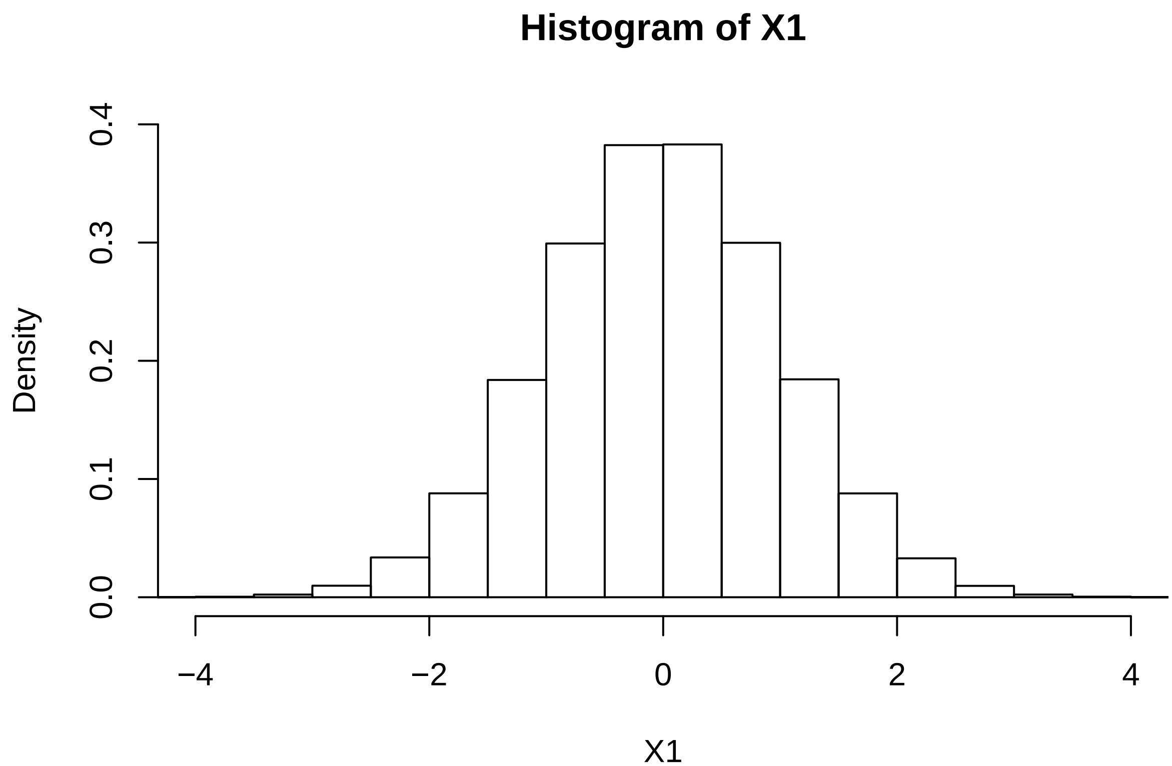
Exercício 2.12:

Se considerarmos o desvio padrão como uma medida para o tamanho do intervalo de valores prováveis, a conjectura do problema é falsa. O desvio padrão de Y é a raiz quadrada da soma dos quadrados dos desvios padrões de X1 e X2, conforme simulação. Além disso, pelos histogramas apresentados, percebe-se uma distribuição mais espalhada para Y.

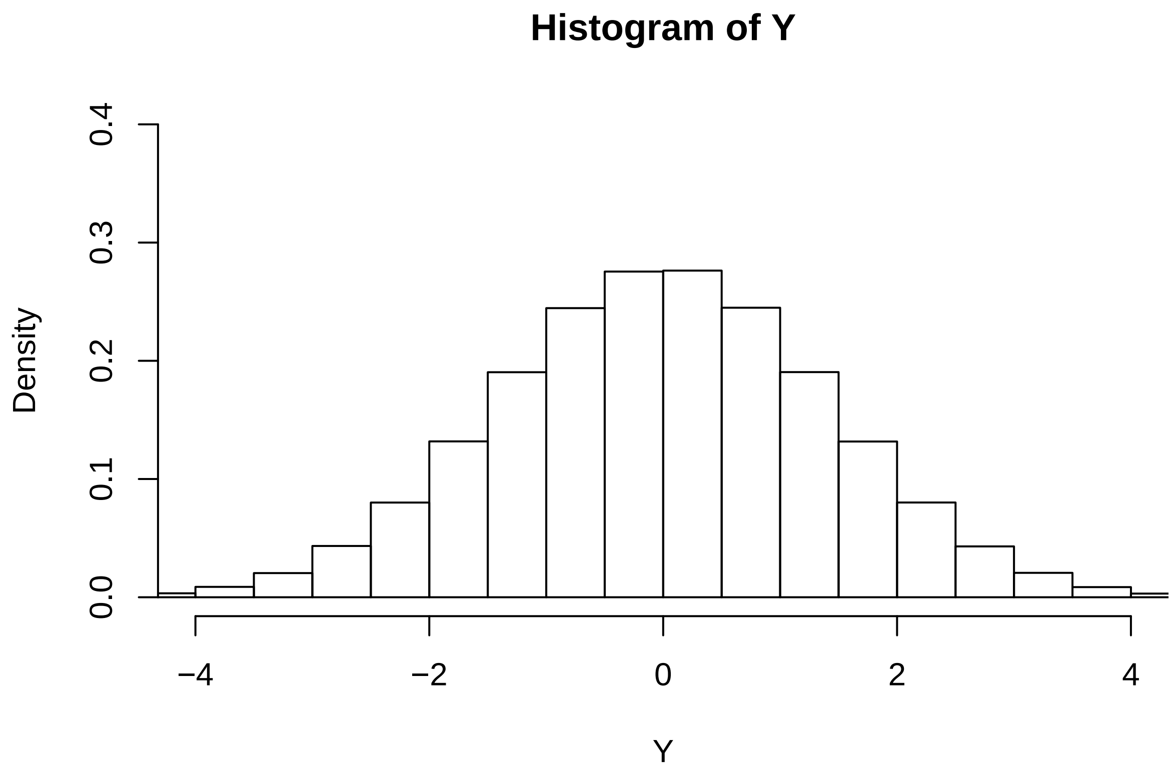
```
n<-1000000
X1<-rnorm(n,0,1)
X2<-rnorm(n,0,1)
Y<-X1-X2
cat("Desvio padrão de X1=X2 é", sd(X1), "\nDesvio padrão de Y é", sd(Y))
```

```
## Desvio padrão de X1=X2 é 1.000996
## Desvio padrão de Y é 1.414345
```

```
hist(X1,probability = TRUE,xlim=c(-4,4),ylim=c(0,0.4))
```




```
hist(Y,probability = TRUE,xlim=c(-4,4),ylim=c(0,0.4))
```



Exercício 2.13:

A distância média, calculada por simuluação, é de 1.25:

```
n<-100000
X<-rnorm(n,0,1)
Y<-rnorm(n,0,1)
dist<-sqrt(X^2+Y^2)
print(mean(dist))
```

```
## [1] 1.252969
```

Exercício 2.14:

Pelos resultados da simulação abaixo, a conjectura não é verdade.

```
n<-100000
U<-runif(n,0,1)
X<-sqrt(U)
cat("Mean of Sqrt(U) is", mean(X), "\nSqrt(mean(U)) is",sqrt(mean(U)))
```

```
## Mean of Sqrt(U) is 0.6678209
## Sqrt(mean(U)) is 0.7079836
```

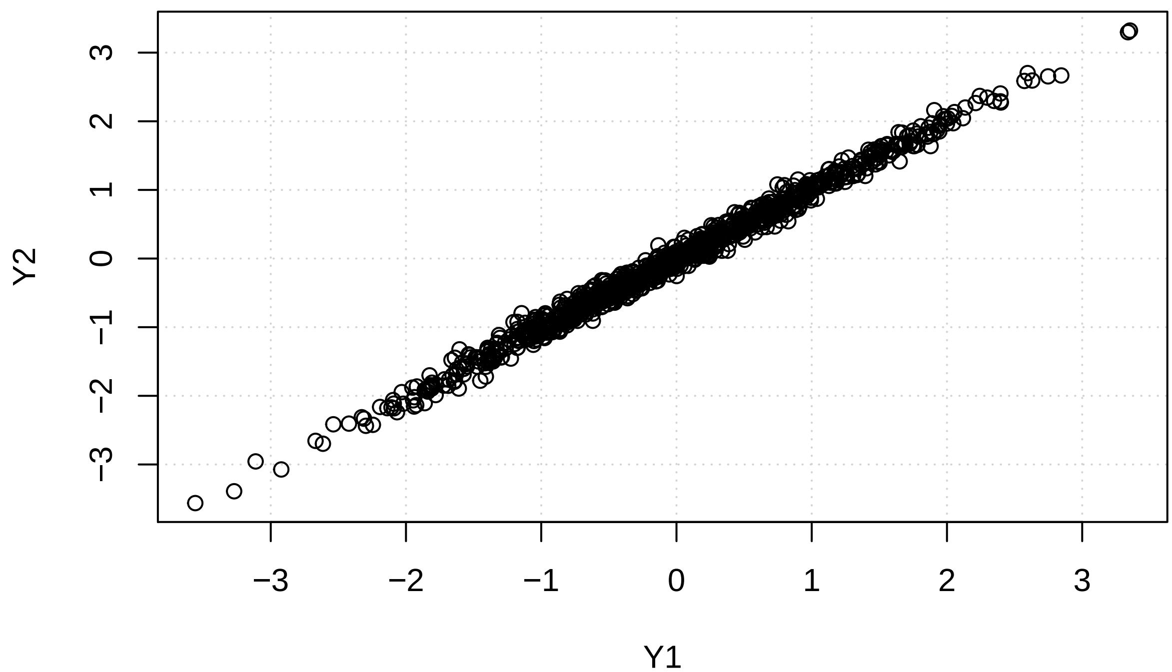
Exercício 2.15:

De acordo com o diagrama plotado, e com a correlação entre as variáveis Y1 e Y2, é possível determinar o valor aproximado de Y2 em função de Y1. Para o caso de Y1, pelo diagrama temos que Y2 é aproximadamente 1, caso Y1 também seja 1.

```r
n<-1000
X1<-rnorm(n,0,1)
X2<-rnorm(n,0,1)
Y1<-X1+0.1*X2
Y2<-X1+0.2*X2
cor(Y1,Y2)
```

```
## [1] 0.9949975
```

```r
plot(Y1,Y2,panel.first=grid())
```



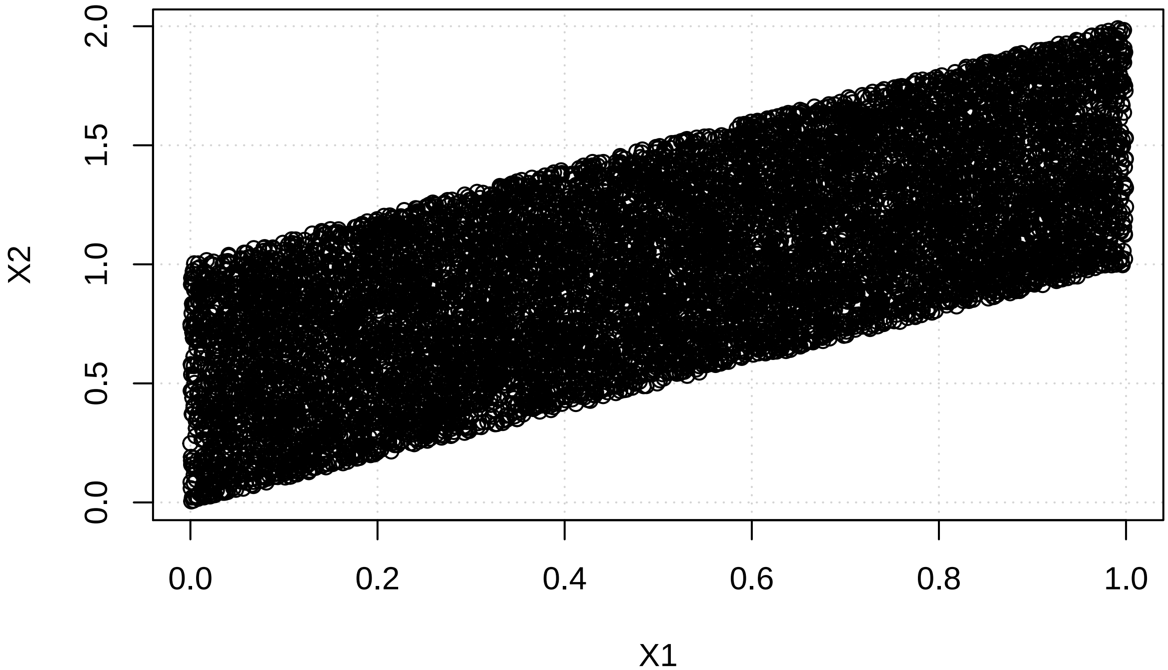
Exercício 2.16:

A combinação linear de X pode ser dada por:

$$X = U_1 e_1 + U_2 e_2$$

Os vetores e1 e e2 delimitam o paralelograma encontrado pelo diagrama de dispersão, explicando o formato do mesmo.

```r
n<-10000
U1<-runif(n,0,1)
U2<-runif(n,0,1)
X1<-U1
X2<-U1+U2
plot(X1,X2,panel.first=grid())
```


X2
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
X1