Variáveis aleatórias discretas

2022-09-20

1. Exemplo 1 dos slides

Experiência: Lançamento de uma moeda equilibrada, três vezes.

1.1 Definição da variável aleatória teoricamente: X - número de faces em três lançamentos de uma moeda equilibrada

1.2. Definição da variável aleatória, para cálculos Como queremos contar o número de "Faces" vamos usar a codificação: 1 se "face", 0 se "coroa"

```
# Usar expand.grid para obter o espaço de resultados

result<-c(0,1)                  # Definição dos resultados individuais

S1<-expand.grid("toss1"=result,  # moeda 1
                "toss2"=result,  # moeda 2
                "toss3"=result)  # moeda 3

# e admitindo a moeda equilibrada

S1$prob<-rep(1/nrow(S1),times=nrow(S1)) # modelo equiprovável
```

O que pretendemos é a variável aleatória X acima definida e não apenas o que observamos em S1.

Começamos por adicionar uma nova coluna ao dataframe S1.

```
# Como o acontecimento elementar de interesse em cada lançamento
# está codificado com 1, e o outro possível, com 0, basta
# somar em cada linha os valores das três primeirs colunas

S1$n_faces<-rowSums(               # soma em linha
                    S1[,1:3])      # das 3 primeiras colunas
S1
```

```
##   toss1 toss2 toss3  prob n_faces
## 1     0     0     0 0.125       0
## 2     1     0     0 0.125       1
## 3     0     1     0 0.125       1
## 4     1     1     0 0.125       2
## 5     0     0     1 0.125       1
## 6     1     0     1 0.125       2
## 7     0     1     1 0.125       2
## 8     1     1     1 0.125       3
```

Mas várias linhas correspondem ao mesmo resultado, pelo que, para obter a **função de probabilidade** de **X**, temos de obter um dataframe simplificado.

```
X<-aggregate(prob~n_faces, # agregar a coluna prob, de acordo com n_faces
             S1,           # no dataframe S1
             sum)          # através da soma
X
```

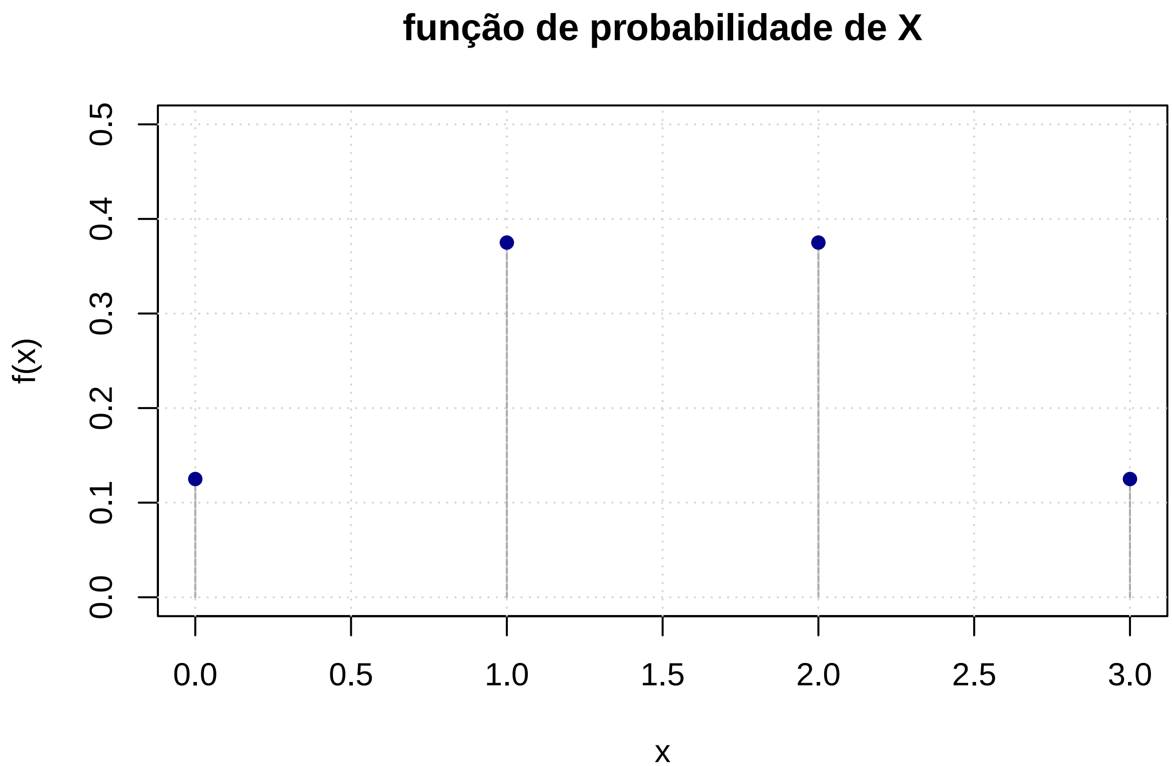
```
##   n_faces  prob
## 1       0 0.125
## 2       1 0.375
## 3       2 0.375
## 4       3 0.125
```

1.3. Representação gráfica da função de probabilidade Vamos representar a função de probabilidade recorrendo a **plot** e **points**.

```
plot (x=X$n_faces,              # os valores da variável
y=X$prob ,                      # os valores da função de probabilidade
type="h",                       # linhas verticais entre x e y
main="função de probabilidade de X",
ylim = c(0,0.5),                # min e max no eixo dos yy
xlab="x",                       # id do eixo x
ylab="f(x)",                    # id do eixo y
col="dark grey"                # cor das linhas
)

grid()

points(x=X$n_faces,
y=X$prob,
pch=16,                        # símbolo usado, 16 é círculo
col="dark blue"
)
```



1.4 Definição da função de distribuição

- Nos pontos x_i Calcular a probabilidade acumulada nos pontos de massa não nula

```
FX_pontos<-cumsum(X$prob)
FX_pontos
```

```
## [1] 0.125 0.500 0.875 1.000
```

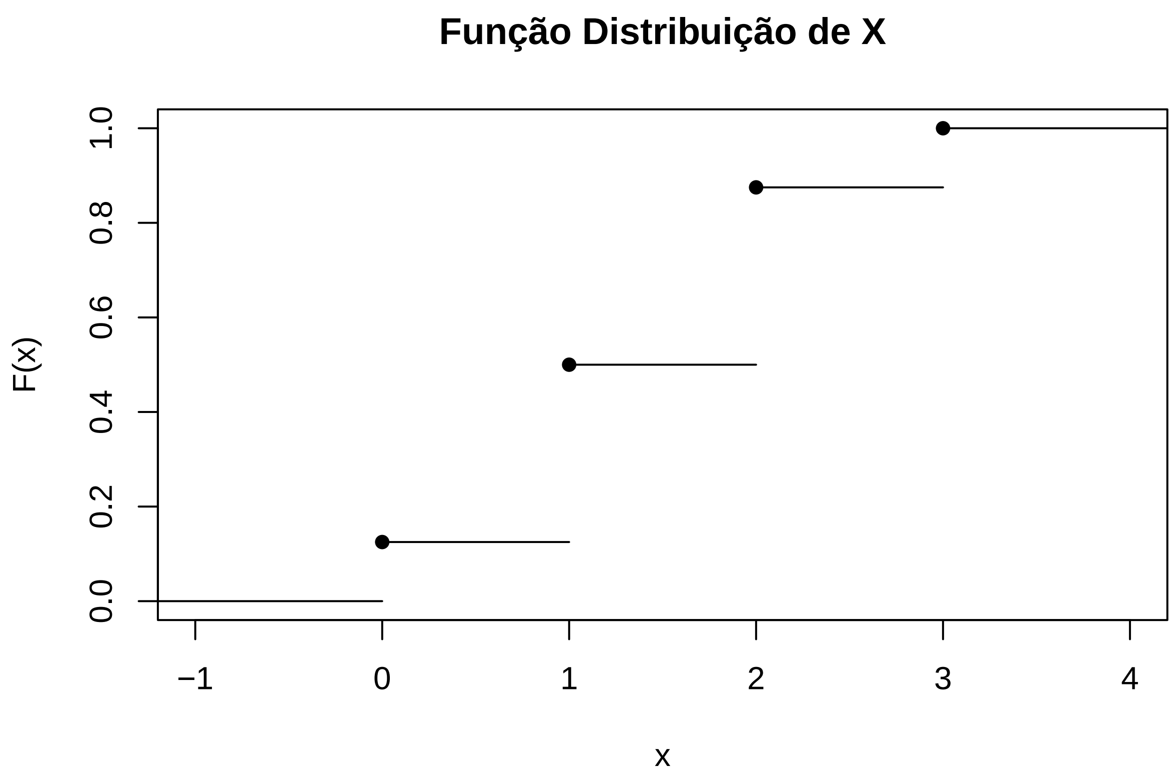
- Em $\mathbb{R}$ A função de distribuição será então

$$F(x) = \begin{cases} 0 & x < 0 \\ 0.125 & 0 \leq x < 1 \\ 0.500 & 1 \leq x < 2 \\ 0.875 & 2 \leq x < 3 \\ 1 & x \geq 3 \end{cases}$$

Representação gráfica de F(x) Para representar graficamente a função de distribuição temos de recorrer a **stepfun** e **plot.stepfun**

```
plot.stepfun(
stepfun(                # definir a função em patamares
X$n_faces,              # os valores de x a considerar
c(0,FX_pontos),         # os patamares a considerar,
                        # ponto adicional inicial 0
```

```
right=FALSE            # intervalos fechados à esquerda
),
verticals=FALSE,       # não colocar traços verticais nos pontos de salto
pch = 16,              # tipo de símbolo
#
# título e identificação dos eixos
#
main="Função Distribuição de X",
xlab="x",
ylab="F(x)")
```



Cálculo de probabilidades recorrendo à função de probabilidade ou à função distribuição

1.5. Pretende-se $P[X \leq 2]$

via f.prob Temos de somar prob na condição n_faces <=2

```
sum(X$prob[X$n_faces %in% 0:2])
```

```
## [1] 0.875
```

via f.distribuição Neste caso é diretamente F(2)

```
FX_pontos[which(X$n_faces==2)]
```

```
## [1] 0.875
```

1.6 Queremos $P[1 \leq X \leq 2]$

via f.prob Basta fazer $f(1) + f(2)$

```
sum(X$prob[X$n_faces %in% 1:2])
```

```
## [1] 0.75
```

via f.distribuição **Não podemos** fazer diretamente $F(2) - F(1)$.

Como X é **discreta**, ser ≥ 1 equivale dizer que é > 0, pelo que neste caso podemos calcular $P[X \leq 2] - P[X \leq 0] = F(2) - F(0)$

```
FX_pontos[which(X$n_faces==2)]-FX_pontos[which(X$n_faces==0)]
```

```
## [1] 0.75
```

1.7 Valor esperado e variância numa v.a. discreta Seguindo a fórmula

$$E[X] = \sum_{1}^{n} x_i f(x_i)$$

, basta calcular a soma dos produtos da primeira com a segunda colunas definidas:

```
miu_x<-sum(X$n_faces * X$prob)
miu_x
```

```
## [1] 1.5
```

Para a variância, podemos aplicar a fórmula simplificada (média dos quadrados menos o quadrado da média) ou a definição formal (média dos quadrados dos desvios face à média).

```
# média dos quadrados menos o quadrado da média
var_x<- sum((X$n_faces^2)*X$prob)-miu_x^2
round(var_x,4)
```

```
## [1] 0.75
```

```
# média dos quadrados dos desvios face à média
round(sum((X$n_faces-miu_x)^2*X$prob),4)
```

```
## [1] 0.75
```

E assim o desvio-padrão será $\sigma_X = 0.87$