Análise Exploratória de Dados

Gabriela Scarpini e Victor Fossaluza

2025-08-19

Contents

Chapter 1

Introdução

1.1 Apresentação

Esse material está em fase inicial de desenvolvimento e será utilizado para apoiar as aulas de MAE0111 - Análise Exploratória de Dados.

O objetivo principal da apostila é funcionar como um resumo e auxiliar os estudantes na compreensão dos conceitos teóricos, com capítulos curtos e por meio de exercícios ao final de cada capítulo.

Comentários e correções podem ser enviadas para gabi.scarpini@usp.br.

1.2 Programa

1. A profissão de Estatística. A Estatística como metodologia de todas as ciências experimentais. O mercado de trabalho. O perfil profissional do Estatístico. A Estatística acadêmica: pós-graduação e pesquisa.
2. Apresentação de problemas reais analisados no CEA – Centro de Estatística Aplicada da USP, com ênfase na análise descritiva dos dados. Conclusões dos estudos.
3. Estatística descritiva e inferência estatística, tipos de dados, bancos de dados, ordem de grandeza, precisão e arredondamento de dados quantitativos, proporções e porcentagens, taxas e números índices, sugestões para construção e apresentação de gráficos e tabelas.
4. Representação gráfica e tabular da distribuição de dados: tabelas de frequências, gráficos de barras e do tipo "torta", histogramas, densidade suavizada e função de distribuição empírica.

5. Medidas-resumo: medidas de posição, de dispersão, de assimetria e curtose, gráficos do tipo boxplot.
6. Modelos para distribuições de frequências: gráficos de probabilidade.
7. Associação entre variáveis qualitativas: tabelas de contingência de dupla entrada, coeficientes de associação, sensibilidade e especificidade, risco relativo, razão de chances, tabelas de contingência de múltiplas entradas.
8. Associação entre variáveis quantitativas: gráficos de dispersão, covariância, correlação linear, matriz de covariâncias, matriz de correlações.
9. Associação entre uma variável quantitativa e uma variável qualitativa: homogeneidade de distribuições, gráficos de médias, gráficos de perfis.
10. Outros tópicos: elaboração de relatórios técnicos, uso do aplicativo R, dashboards.

1.3 Bibliografia

- Damiani, A., Milz, B., Lente, C., Falbel, D., Correa, F., Trecenti, J., Luduvice, N., Lacerda, T., Amorim, W. Ciência de Dados em R, Curso-R link
- Peng, R.D. Exploratory Data Analysis with R, Leanpub. link
- Mayer, F.P, Bonat, W.H., Zeviani, W.M., Krainski, E.T., Ribeiro Jr, P.J. Estatística Computacional com R. link
- Grolemund, G. Wickham, H. R for Data Science. link (versão em português: link)
- Chang, W. R Graphics Cookbook, 2nd edition. link
- Grolemund, G. Hands-On Programming with R. link

1.4 Bibliografia Jupiterweb

- Morettin, P. A., Bussab, W. O. (2017). Estatística Básica. 9a edição. Saraiva Educação SA.
- Wickham, H., Grolemund, G. (2017). R for data science: import, tidy, transform, visualize, and model data, O'Reilly Media, Inc. link
- Tukey,J. W. (1977). Exploratory Data Analysis. Reading: Addison Wesley.

- Cairo, A. (2016). The truthful art: Data, charts, and maps for communication. New Riders.
- Tufte, E. R. (1983). The Visual Display of Quantitative Information, Cheshire: Graphics Press.
- Few, S. (2012). Show Me the Numbers: Designing Tables and Graphs to Enlighten, 2a ed. Analytics Press.

1.5 Bibliografia Complementar

- Relatórios do CEA – Centro de Estatística Aplicada – USP.
- Magalhães, M. N, de Lima, A. C. P. (2015). Noções de Probabilidade e Estatística. 7a edição. Editora da Universidade de São Paulo.
- Murteira, B. F. J., Black, G. H. J.. (1983). Análise Exploratória de Dados - Estatística Descritiva, Lisboa: McGraw Hill.
- Wexler, S., Shaffer, J., & Cotgreave, A. (2017). The big book of dashboards: visualizing your data using real-world business scenarios. John Wiley & Sons.
- ChambersJ. M., ClevelandW. S., Tukey, P. A.. (1983). Graphical Methods for Data Analysis. Boston: Duxbury Press.
- W. M. Cleveland. (1993). Visualizing Data, Summit, New Jersey: Hobart Press.
- W. M. Cleveland. (1994). The Elements of Graphing Data, Summit: Hobart Press.

Chapter 2

Conceitos básicos de Análise Exploratória de Dados

2.1 O que significa algo ser "aleatório"?

- **Aleatório** *(Google - Oxford Languages)*
  (adjetivo)

1. que depende das circunstâncias, do acaso; casual, fortuito, contingente.

2. (física) referente a fenômenos físicos para os quais as variáveis tomam valores segundo uma determinada lei de probabilidade (p.ex., o movimento browniano).

- **Experimento Aleatório** *(The Concise Encyclopedia of Statistics, pp 430–433)*

Um experimento em que o resultado não é previsível com antecedência é chamado de experimento aleatório. Um experimento aleatório pode ser caracterizado da seguinte forma:
1. É possível descrever o conjunto de todos os resultados possíveis (chamado espaço amostral do experimento aleatório).
2. Não é possível prever o resultado com certeza.
3. É possível associar cada resultado possível a uma probabilidade de ocorrência.

2.2 O que é Probabilidade?

- **Probabilidade** *(Google - Oxford Languages)*
  (substantivo feminino)

1. perspectiva favorável de que algo venha a ocorrer; possibilidade, chance. "há pouca probabilidade de chuva"

2. grau de segurança com que se pode esperar a realização de um evento, determinado pela frequência relativa dos eventos do mesmo tipo numa série de tentativas.

- **Probabilidade** *(The Concise Encyclopedia of Statistics, pp 430–433)*

Podemos definir a probabilidade de um evento usando as frequências relativas ou por meio de uma abordagem axiomática.

Na primeira abordagem, supomos que um experimento aleatório é repetido muitas vezes nas mesmas condições. Para cada evento A definido no espaço amostral Ω, definimos n_A como o número de vezes que o evento A ocorreu durante as primeiras n repetições do experimento. Neste caso, a probabilidade do evento A, denotada por $P(A)$, é definido por:

$$P(A) = \lim_{n\to\infty} \frac{n_A}{n} \ ,$$

o que significa que $P(A)$ é definido como o limite relativo ao número de vezes que o evento A ocorreu relativo ao número total de repetições.

Na segunda abordagem, para cada evento A, aceitamos que existe uma probabilidade de A, $P(A)$, satisfazendo os três axiomas a seguir:
1. $0 \leq P(A) \leq 1$,
2. $P(\Omega) = 1$,
3. Para cada sequência de eventos mutuamente exclusivos $A_1, A_2, \ldots$ (isto é, eventos tais que $A_i \cap A_j = \emptyset$ se $i \neq j$):

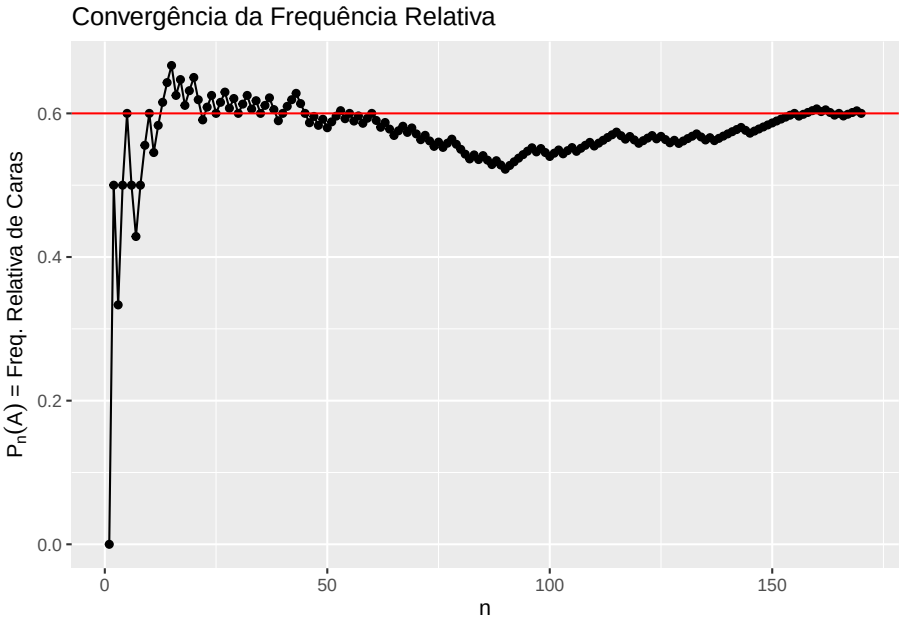
$$P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(Ai) \ .$$

—

2.2.1 Interpretações de Probabilidade

- **Interpretação Clássica** (De Moivre, Laplace)
  - baseia-se na equiprobabilidade dos resultados;
  - $P(A) = \frac{|A|}{|\Omega|}$.
  - **Exemplo:** um lançamento de moeda, $A =$ "cara", $P(A) = \frac{1}{2}$.

- **Interpretação Frequentista** (Venn, von Mises, Reichenbach, etc.)
  - quase unânime na primeira metade do século XX e ainda é a mais aceita;
  - baseia-se na regularidade das frequências relativas (lei dos grandes números);
  - $P(A) = lim\frac{A_n}{n}$, onde A_n é o número de ocorrências de A em n realizações *idênticas e independentes* do experimento;
  - Supõe que é possível repetir indefinidamente o experimento nas mesmas circustâncias.
  - **Exemplo:** um lançamento de moeda, $A =$ "cara".



- **Interpretação Lógica** (Keynes, Jeffreys, Carnap, etc.)

  - medida de "vínculo parcial" entre uma evidência e uma hipótese;
  - baseia-se em relações objetivas entre proposições.
  - **Exemplo:** considere duas proposições: "até agora todos os lançamentos resultaram em cara" e "será realizado um novo lançamento". Pode-se afirmar que "provavelmente o resultado do novo lançamento será cara".

- **Interpretação Subjetivista** (Ramsey, de Finetti, Savage, etc)
  - probabilidade como medida subjetiva de crença;
  - baseada na experiência de cada indivíduo, portanto única.
  - **Exemplo:** suponha que Bruno lançou uma moeda 3 vezes e todos os resultados foram cara. Esse indivíduo, em posse dessa informação, pode acreditar que o resultado cara é mais provável que coroa. Contudo, quando pergunta sobre a probabilidade de cara ao seu colega Olavo, ignorante com relação a moeda, ele responde que é 1/2.

2.2.2 Comentários sobre Probabilidade e Aleatoriedade

- Exemplo da moeda. Aleatoriedade é uma característica (física) do lançamento da moeda?

- Exemplo das bolas na Urna. A "aleatoriedade" está em "chacoalhar" a urna? E se eu "embrulhar" as bolas e colocá-las em fila sobre a mesa? O experimento ainda é "aleatório"? Qual a "probabilidade" de selecionar uma bola verde?

2.3 O que é Estatística?

- **Estatística** *(Google - Oxford Languages)*
(substantivo feminino)

1. ramo da matemática que trata da coleta, da análise, da interpretação e da apresentação de massas de dados numéricos.

2. qualquer coleta de dados quantitativos.

- **Estatística** *(The Concise Encyclopedia of Statistics, pp 518–520)*

A palavra estatística, derivada do latim, refere-se à noção de estado (status): "que é relativo ao estado". Os governos têm uma grande necessidade de contar e medir numerosos eventos e atividades, como mudanças demográficas, nascimentos, tendências de imigração e emigração, mudanças nas taxas de emprego, negócios, etc.

Nessa perspectiva, o termo "estatística" é usado para indicar um conjunto de dados disponíveis sobre um determinado fenômeno (por exemplo, estatísticas de desemprego).

No sentido mais moderno e preciso da palavra, "estatística" é considerada uma disciplina que se preocupa com dados quantitativos. É constituído por um conjunto de técnicas de obtenção de conhecimento a partir de dados incompletos, de um rigoroso sistema científico de gestão de coleta de dados, da sua organização, análise e interpretação, quando é possível apresentá-los de forma numérica.

Numa população de indivíduos, pode ser de interesse saber, em termos de teoria estatística, se um determinado indivíduo tem carro ou se fuma. Por outro lado, também pode ser de interesse saber quantos indivíduos têm automóvel e são fumantes, e se existe relação entre possuir automóvel e hábitos de tabagismo na população estudada.

Gostaríamos de conhecer as características da população globalmente, sem nos preocuparmos com cada pessoa ou cada objeto da população.

Distinguimos dois subconjuntos de técnicas: (1) aquelas que envolvem estatísticas descritivas e (2) aquelas que envolvem estatísticas inferenciais. O objetivo essencial da estatística descritiva é representar a informação em um formato compreensível e útil. A estatística inferencial, por outro lado, visa facilitar a generalização dessas informações ou, mais especificamente, fazer inferências (relativas a populações) com base em amostras dessas populações.

- **Estatístico** *(segundo prof. Carlos Alberto de Bragança Pereira)*

"The Statistician is the Wizard who makes"scientific" statements about invisible states and quantities. However, contrary to the real wishes (or witches), he attaches uncertainties to his statements."

2.4 Método Científico

1. Formulação de uma questão, teoria ou hipótese.

2. Coleta de informações: planejamento de um experimento para obtenção de dados ou apenas a observação de um fenômeno ou variáveis de interesse.

3. Conclusões (por vezes, parciais) baseadas nos dados obtidos anteriormente.

4. Se necessário, repetir (2) e (3) ou formular novas hipóteses.

2.5 População X Amostra, Probabilidade X Estatística

- *População* é o conjunto de todos os elementos ou resultados possíveis.

- *Amostra* é um subconjunto da população.

- *Experimento* é "tornar visível o que antes era invisível", por exemplo, observar uma amostra da população.

- *Probabilidade* é uma descrição matemática da incerteza, é bem especificada quando a população é conhecida.

- *Estatística* estuda a distribuição de probabilidades quando esta não está bem especificada (é desconhecida, ao menos parcialmente).

  - **Modelo Probabilístico:** $(\Omega, \mathcal{F}, \boldsymbol{P})$, onde Ω é o espaço amostral, $\mathcal{F}$ é uma coleção (σ-álgebra) de subconjuntos de omega e $\boldsymbol{P}$ é uma medida de probabilidade (conhecida, fixada)

  - **Modelo Estatístico:** simplificadamente, um modelo estatístico é uma forma probabilística de relacionar uma quantidade desconhecida de interesse (*parâmetro*) com os dados observados.

  - $(\Omega, \mathcal{F}, \mathcal{P})$, onde $\mathcal{P}$ é uma família de distribuições de probabilidade. Na estatística, o objetivo é fazer afirmações sobre essa família.

2.6 Estatística Descritiva

- Conjunto de técnicas para visualização (redução) dos dados.

- Análises e conclusões preliminares.
- Fornece informações que auxiliam na especificações de um *modelo estatístico*.
- Também utilizada para
  - Avaliação de modelos;
  - Interpretação de modelos complexos;
  - Comunicação dos resultados.
- Exemplo: medidas resumo, gráficos e tabelas.

2.7 Inferência Estatística

- Generalizar resultados observados em uma amostra para a população de interesse.
- Normalmente baseada em algumas suposições que são traduzidas em um *modelo estatístico*.
- Principais objetivos: concluir se há relações entre variáveis, estimativas pontuais e intervalares e testes de hipóteses.
- Como o objetivo é generalizar conclusões para a população, é usual ter uma grande preocupação com a verificação das suposições do modelo estatístico adotado.
- Por vezes chamada de "análise confirmatória".

2.8 Aprendizado Estatístico

- Similarmente à inferência estatística, estuda relação entre variáveis mas tem como objetivo fazer predições para novas observações.

- Como o objetivo é fazer predições, o foco é obter um modelo que "acerte" mais ou, em outras palavras, que minimize algum tipo de função de perda, dando menos atenção à suposições sobre o modelo probabilístico utilizado.

- Na prática, o conjunto de dados é dividido em um *conjunto de treinamento* e um *conjunto de teste*. O primeiro é usado para a obtenção de modelos e o segundo para a sua avaliação. O modelo escolhido é aquele que "acerta mais" no conjunto de teste.

- Também pode ser pensada como uma forma reproduzir o mecanismo gerador dos dados.

Chapter 3

Apresentação à linguagem R

O R é uma linguagem de programação amplamente utilizada para análise de dados, cálculo estatísticos e visualização de dados. Além disso, o R tem uma enorme coleção de pacotes que ampliam suas funcionalidades, usaremos alguns durante o nosso estudo. O RStudio é uma interface gráfica para a linguagem de programação R. Ele torna o R mais fácil de usar e fornece algumas funcionalidades úteis.

3.1 Como instalar

Para instalar o R e o R-Studio basta seguir as instruções aqui e aqui

---

3.2 Comparação com outras linguagens estatísticas

Diferente de outras linguagens de programação, o R foi desenvolvido especificamente para a área de estatística, sendo usado na análise de dados e modelagem. Por isso, essa linguagem possui uma enorme quantidade de pacotes voltados para visualização e análise de dados (alguns exemplos são o ggplot2, dplyr, caret).

3.3 Introdução ao RMarkdown

Na disciplina de Análise Exploratória de Dados, usaremos somente o RMarkdown, uma ferramenta que nos permite escrever textos e executar códigos em R.

Para criar um arquivo em RMarrkdown clique em *File > New File > R Markdown*. Preencha as informações iniciais, como o título do documento e o autor e por fim escolha o tipo de saída, como PDF, HTML ou Word.

- **Estrutura de um Arquivo RMarkdown**

Um arquivo .Rmd tem a seguinte estrutura:

1. Cabeçalho YAML, que define as configurações do documento.

```
title: "Introdução ao RMarkdown"
author: "Seu Nome"
date: "18/03/2025"
output: html_document
```

2. Texto formatado, que usa a sintaxe Markdown para formatação de texto, por exemplo:

- "# Título 1" para criar títulos de nível 1.
- Para destacar: ‘ ”Texto em itálico ” ou ‘’Texto em negrito’’.
- Para criar listas não ordenadas (com marcadores): use - ou * antes dos itens. Já para listas ordenadas (numeradas): use números seguidos de ponto.

3. Blocos de códigos (chunks), que são delimitados por três crases (“‘) e é onde você pode escrever e executar os comandos em R. Eles permitem integrar o código diretamente ao documento, gerando tabelas, gráficos, etc. Para cria-lo basta escrever as três crases Um atalho para cria-lo é com CTRL+ALT+I.

```
# seu código R.
```

- **Equações Matemáticas no RMarkdown:**

É possível fazer equações matemáticas em R Markdown usando a sintaxe do LaTeX. Existem duas formas principais:

1. Equações em linha: use cifrões simples `$...$` para inserir fórmulas no meio do texto.
2. Equações destacadas (em bloco): use dois cifrões `$$...$$` para centralizar a equação.

Para mais informações sobre o R Markdown, acesse a cheatsheet oficial diretamente pelo RStudio. Basta ir em Help > Cheatsheets > R Markdown Cheat Sheet. Lá você encontra um resumo com os principais comandos de formatação, código, tabelas, gráficos e equações.

3.4 Operadores Básicos

- **Operadores de atribuição:**

Em R usamos os operadores de atribuição para atribuir valores a variáveis. Para fazer isso podemos usar <-, -> e =.

```
a <- 5 # armazena o valor 5 dentro da variável "a"
a
```

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

```
10 -> b # armazena o valor 10 dentro da variável "b"
b
```

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

```
c = 7 # armazena o valor 7 dentro da variável "c"
c
```

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

Mesmo sendo possível usar diferentes operadores de atribuição, o padrão costuma ser feito com <-. Para isso, podemos usar o atalho ALT+-.

- **Operadores aritméticos:**

Para realizar operações matemáticas básicas, usamos:

```
# Adição: "+"
resultado1 <- 7 + 3

# Subtração: "-"
resultado2 <- 50 - 5

# Multiplicação: "*"
resultado3 <- 6 * 7  # 42

# Divisão: "/"
resultado4 <- 17 / 3

# Exponenciação: "^" ou "**"
resultado5 <- 2 ^ 4
resultado6 <- 3 ** 2

# Resto da divisão: "%%"
resultado7 <- 10 %% 3

# Divisão inteira: "%/%"
resultado8 <- 10 %/% 3
```

- **Operadores de comparação:**

Usamos os operadores de comparação para comparar dois valores, sendo retorado um valor lógico: TRUE ou FALSE, dependendo do resultado da comparação.

```
# Igualdade: "=="
3 == 3 # retorna TRUE
```

```
## [1] TRUE
```

```
3 == 4 # retorna FALSE
```

```
## [1] FALSE
```

```
# Desigualdade: "!="
8 != 7 # retorna TRUE
```

```
## [1] TRUE
```

```
9 != 9 # retorna FALSE
```

```
## [1] FALSE
```

```
# Maior que: ">"
7 > 3 # retorna TRUE
```

```
## [1] TRUE
```

```
# Menor que: "<"
7 < 3 # retorna FALSE
```

```
## [1] FALSE
```

```
# Maior ou igual: ">="
8 >= 10 # retorna FALSE
```

```
## [1] FALSE
```

```
# Menor ou igua: "<="
9 <= 9 # retorna TRUE
```

```
## [1] TRUE
```

- **Valores especiais**

Os valores especiais são usados para representar situações atípicas ou condições especiais nos dados, como valores ausentes, infinitos ou indefinidos.

1. NA (Not Available): O valor NA representa dados ausentes ou não disponíveis.

```
a <- NA

b <- c(1, 2, 3)
b[4] # valor fora dos límites de um vetor também é NA
```

```
## [1] NA
```

```
# Verifica se o valor é NA
is.na(a)
```

```
## [1] TRUE
```

```
is.na(b)
```

```
## [1] FALSE FALSE FALSE
```

2. NaN (Not a Number): um tipo especial de NA, usado para representar resultados indefinidos de operações matemáticas, como divisões por zero.

```
c <- 0/0
d <- log(-1)
```

```
## Warning in log(-1): NaNs produzidos
```

```
# Verifica se o valor é NaN
is.nan(c)
```

```
## [1] TRUE
```

```
is.nan(d)
```

```
## [1] TRUE
```

3. Inf e -Inf (Infinito): Inf e -Inf representam valores infinitos.

```
f <- 2 / 0
g <- -1 / 0

# Verifica se o valor é Inf
is.infinite(f)
```

```
## [1] TRUE
```

4. NULL: representa a ausência de um valor ou objeto. Diferente do NA, que representa um valor ausente dentro do de um vetor ou lista, NULL indica que o objeto não existe. Usamos normalmente para iniciar variáveis ou para remover elementos de listas.

```
h <- NULL

# Verifica se o valor é null
is.null(h)
```

```
## [1] TRUE
```

3.5 Estrutura de Dados em R

- **Vetores**

Os vetores são uma estrutura de dados que armazena uma sequência de elementos, sendo todos do mesmo tipo (número, caracteres ou valores lógicos).

- Vetor numérico:

```
n <- c(11, 22, 33, 44)

n[2] # Valor no índice 2 do vetor
```

```
## [1] 22
```

```
# Valor do vetor em um subconjunto de índices
n[c(1,3)]
```

```
## [1] 11 33
```

```
n[c(3,1,4,2)]
```

```
## [1] 33 11 44 22
```

```
n[1:3] # valores do vetor do índice 1 ao índice 3
```

```
## [1] 11 22 33
```

```
# Multiplicação por escalar (multiplica cada elemento)
2*n
```

```
## [1] 22 44 66 88
```

```
# Soma com escalar (soma cada elemento)
v <- n+1

# Soma de Vetores (soma cada termo de n com o elemento de mesmo índice de v)
n+v
```

```
## [1] 23 45 67 89
```

```
# Produto de Vetores (termo a termo)
n*v
```

```
## [1]  132  506 1122 1980
```

```
# Produto Escalar (multiplica os dois vetores da mesma posição e soma os resultados de
n%*%v
```

```
##      [,1]
## [1,] 3740
```

```
# Quando somamos vetores de tamanhos diferentes repetimos os elementos até o tamanho d
n - c(1,2)
```

```
## [1] 10 20 32 42
```

```
# Caso o tamanho de um deles naõ seja multiplo do outro, o R solta uma mensagem de err
c(1,2,3) - n
```

```
## Warning in c(1, 2, 3) - n: comprimento do objeto maior não é múltiplo do
## comprimento do objeto menor
```

```
## [1] -10 -20 -30 -43
```

- Vetor de Caracteres:

```
Nomes = c("Letícia","Mariana","Guilherme","Viviane",
          "Ana","Otávio","Eduardo")


# Identificando o tipo de vetor
typeof(Nomes)
```

```
## [1] "character"
```

```
class(Nomes)
```

```
## [1] "character"
```

- Vetor de valores lógicos:

```
Feminino = c(TRUE,TRUE,FALSE,TRUE, TRUE, FALSE, FALSE)

# Identificando o tipo de vetor
class(Feminino)
```

```
## [1] "logical"
```

```
# TRUE e FALSE são tratados como 1 e 0
sum(Feminino)
```

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

- **Matrizes**

Matrizes é uma estrutura de dados que permite armazenar os dados de uma forma bidimensional, com linhas e colunas. Todos os elementos precisam ser do mesmo tipo.

```
M <- matrix(c(1, 0, 0, 2, 1, 0, 3, 4, 1), ncol=3)

# Dimensção da Matriz M
dim(M) # linha, coluna
```

```
## [1] 3 3
```

```r
# Acessando um elemento da matriz
M[1, 2] # elemento da linha 1 e da coluna 2
```

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

```r
M[1,] # primeira linha (vetor)
```

```
## [1] 1 2 3
```

```r
M[,3] # última coluna (vetor)
```

```
## [1] 3 4 1
```

```r
# Determinante
det(M)
```

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

```r
# Matriz Inversa
IM <- solve(M)

# Produto de Matrizes
M%*%IM
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

- **Listas**

Diferente dos vetores, as listas podem armazenar objetos de tipos diferentes, sendo muito mais flexíveis.

```r
lista <- list(
  nome = "Letícia",
  notas = c(8, 10, 9, 7, 6, 8),
  matriz = matrix(1:4, nrow = 2)
)

# Acessando elementos de uma lista
lista[[3]] # usando o índice
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

```
lista$notas # usando o nome
```

```
## [1]  8 10  9  7  6  8
```

```
lista[c("nome", "notas")]
```

```
## $nome
## [1] "Letícia"
##
## $notas
## [1]  8 10  9  7  6  8
```

```
# Modificando elementos
lista$notas <- c(8, 9)
```

- **Fatores**

No R, a estrutura de dados fatores podem ser usadas para representar os dados categóricos, ou seja, variáveis que assumem valores diferentes, como níveis, podendo ser ordenados ou não.

```
nivel_educacional <- factor(c("Fundamental", "Médio", "Superior"))

satisfacao <- factor(c("baixa", "alta", "média"),
              levels = c("baixa", "média", "alta"), # indica os níveis
              ordered = TRUE) # indica que a ordem dos níveis importa
```

Para trabalhar com fatores podemos usar o pacote forcats, que nos fornece funções para criar, modificar e organizar fatores de forma eficiente, sendo possível reordenar níveis com base em valores associados, agrupar categorias raras, ordenar níveis com base em suas frequências e trabalhar com dados categóricos de forma mais eficaz. Ainda nesse capítulo, falaremos mais sobre a instalação e o uso de pacotes dentro do RMarkdown.

- **Data Frames**

Os Data Frames são uma das estruturas mais usadas para armazenar os dados em formato de tabela. Cada coluna é um vetor de mesmo tamanho e pode ter tipos diferentes (números, caracteres, fatores, etc.).

```
df <- data.frame(
  nome = c("Mariana", "Juliana", "Isabela"),
  idade = c(19, 8, 14),
  altura = c(1.68, 1.50, 1.55)
)

# Acessando colunas
df$nome
```

```
## [1] "Mariana" "Juliana" "Isabela"
```

```
# Acessando elementos usando colchetes:
# o primeiro indíce refere-se as linhas
# e o segundo as colunas
df[, "idade"]
```

```
## [1] 19  8 14
```

```
df[1, ]
```

```
##      nome idade altura
## 1 Mariana    19   1.68
```

```
df[1, "altura"]
```

```
## [1] 1.68
```

Por mais que possa parecer com a estrutura de listas, o data frame é uma tabela de dados em formato de linhas e colunas. Diferente de uma lista, as suas colunas devem ter o mesmo número de elementos.

Também podemos manipular os data frames no R usando banco de dados.

```
# Carregando o banco de dados 'iris'
data("iris")

# Vendo o conteúdo do banco de dados
# iris # tudo
head(iris) # primeiros elementos
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
tail(iris) # últimos elementos
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 145          6.7         3.3          5.7         2.5 virginica
## 146          6.7         3.0          5.2         2.3 virginica
## 147          6.3         2.5          5.0         1.9 virginica
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
```

```
# Dimensões do banco de bados
dim(iris) # primeiro as linhas, depois as colunas
```

```
## [1] 150   5
```

```
nrow(iris) # número de linhas
```

```
## [1] 150
```

```
ncol(iris) # número de colunas
```

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

```
# Nome das variáveis (colunas)
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
colnames(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```r
# Nome das variáveis (linhas)
rownames(iris)
```

```
##   [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"  "12"
##  [13] "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"  "23"  "24"
##  [25] "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"  "34"  "35"  "36"
##  [37] "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"  "45"  "46"  "47"  "48"
##  [49] "49"  "50"  "51"  "52"  "53"  "54"  "55"  "56"  "57"  "58"  "59"  "60"
##  [61] "61"  "62"  "63"  "64"  "65"  "66"  "67"  "68"  "69"  "70"  "71"  "72"
##  [73] "73"  "74"  "75"  "76"  "77"  "78"  "79"  "80"  "81"  "82"  "83"  "84"
##  [85] "85"  "86"  "87"  "88"  "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"
##  [97] "97"  "98"  "99"  "100" "101" "102" "103" "104" "105" "106" "107" "108"
## [109] "109" "110" "111" "112" "113" "114" "115" "116" "117" "118" "119" "120"
## [121] "121" "122" "123" "124" "125" "126" "127" "128" "129" "130" "131" "132"
## [133] "133" "134" "135" "136" "137" "138" "139" "140" "141" "142" "143" "144"
## [145] "145" "146" "147" "148" "149" "150"
```

3.6 Estruturas de controle

As estruturas de controles são "instruções" que nos permiter criar lógica nos programas e determinar como as operações e funções serão excecutadas.

- **Condicionais**

As estruturas condicionais permitem executar blocos de código com base em condições lógicas. Elas verificam se determinada condição é verdadeira ou falsa e excecutam diferentes blocos do código com base no resultado.

O comando *if* avalia se uma condição é verdadeira (TRUE), caso seja o bloco de código associado é excecutado.

```r
# if (condição) {
  # Código executado se a condição for TRUE
# }

# Exemplo
a <- 10
if (a > 5) {
  print("a é maior que 5")
}
```

```
## [1] "a é maior que 5"
```

Quando é necessário tratar tanto o caso que a condição é verdadeira, quanto o caso em que é falsa, utilizamos *if - else*.

```
# if (condição) {
  # Código executado se a condição for TRUE
# } else {
  # Código executado se a condição for FALSE
# }

# Exemplo

b <- 3
if (b > 5) {
  print("b é maior que 5")
} else {
  print("b é menor ou igual a 5")
}
```

```
## [1] "b é menor ou igual a 5"
```

Quando temos mais de uma condição usamos *if - else if - else*.

```
# if (condição1) {
  # Código executado se condição1 for TRUE
# } else if (condição2) {
  # Código executado se condição1 for falsa e condição2 for TRUE
# } else {
  # Código executado se nenhuma das condições anteriores for TRUE
# }

# Exemplo
c <- 0
if (c > 0) {
  print("c é positivo")
} else if (c < 0) {
  print("c é negativo")
} else {
  print("c é zero")
}
```

```
## [1] "c é zero"
```

Quando se trata de analisar vetores, podemos usar *ifelse*, que avalia cada elemento de vetor de maneira individual.

```
# ifelse(condição, valor_se_verdadeiro, valor_se_falso)

# Exemplo
d <- c(-5, 0, 4, 9)
resposta <- ifelse(d > 0, "positivo", "não positivo")
print(resposta)
```

```
## [1] "não positivo" "não positivo" "positivo"     "positivo"
```

- **Laços de repetição**

Os laços de repetição (ou loops) em R nos permitem executar blocos de códigos múltipla vezes. Temos dois principais tipos: *for* e *while*.

O *for* é um laço de iteração, usado para iterar sobre uma sequência, como vetores, listas, etc.

```
# for (variável in sequência) {
  # Código a ser executado
# }

# Exemplo
for (i in 1:5) {
  print(paste("O quadrado de", i, "é", i^2))
}
```

```
## [1] "O quadrado de 1 é 1"
## [1] "O quadrado de 2 é 4"
## [1] "O quadrado de 3 é 9"
## [1] "O quadrado de 4 é 16"
## [1] "O quadrado de 5 é 25"
```

No exemplo acima a variável "i" assume o valor de cada número da sequência em cada iteração, ou seja, i assume a sequência de números de 1 a 5.

O *while* é um laço condicional, ou seja, executa o bloco de código enquanto a condição determinada for verdadeira.

```
# while (condição) {
  # Código a ser executado
# }

# Exemplo
a <- 1
while (a <= 5) {
  print(a)
  a <- a + 1
}
```

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

O exemplo acima realiza uma contagem de 1 até 5. O valor da variável "a" começa em 1, a condição a <= 5 é verificada e enquanto ela for verdadeira o código dentro do laço é executado. A cada iteração o valor de x é impresso e incrementado 1. O loop finaliza assim que a se torna 6, ou seja, o momento em que a condição x<= 5 se torna falsa.

É preciso tomar cuidado para que o *while* não se torne um loop infinito, ou seja, é necessário garantir que a condição imposta no *while* se torne falsa em algum momento.

3.7 Funções

As funções são blocos de código reutilizáveis que nos permitem executar tarefas específicas. A estrutura básica de uma função é: - Nome da função; - Argumentos (entrada que a função espera receber); - Corpo da função (especifica o que a função faz); - Valor de retorno (o que a função retorna após a execução).

```
soma <- function(a, b){
  resultado <- a + b
  return(resultado)
  }

# Testando a função 'soma'
soma(7, 3)
```

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

Podemos criar funções utilizando estruturas de controle, para determinar o fluxo de execução do código.

```
# Função que calcula o fatorial de um inteiro não negativo
fatorial <- function(n){
  if(n<0){
    print("Número negativo! Digite um inteiro positivo!")
    return()
  }
  f <- 1
  while(n>1){
    f <- f*n
    n <- n-1
  }
  return(f)
}

# Testa a função 'fatorial'
fatorial(4)
```

```
## [1] 24
```

```
# Teste para um número negativo
fatorial(-1)
```

```
## [1] "Número negativo! Digite um inteiro positivo!"
```

```
## NULL
```

Podemos criar também função com valores padrões, atribuindo valores aos argumentos. Dessa forma, se o usuário não fornecer um valor ao chamar a função, o valor padrão é utilizado.

```
subtracao <- function(a, b = 1) {
  resultado <- a - b
  return(resultado)
}

# Testando a função 'subtracao'
subtracao(7, 2) # com os dois argumentos
```

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

```r
subtracao(5) # usando o argumento padrão para "b"
```

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

3.8 Funções Básicas e Pacotes

As funções básicas são aquelas já incorporadas na linguagem, que não requerem pacotes adicionais. Alguns exemplos são: - 'mean()' para calcular a média de um vetor - 'sd()' para o desvio padrão. - 'sum()' para somar os elementos de um vetor. - 'prod()' – Multiplica todos os elementos de um vetor.

Já os pacotes são coleções de funções que estendem a funcionalidade do R, permitindo a realização de análises mais específicas. Os pacotes que usaremos nessa apostila são: ggplot2, dplyr, tdyr, forcats, tibble

Pacotes podem ser instalados usando `install.packages("nome_do_pacote")` e carregados com `library(nome_do_pacote)`.

3.9 Exercícios

1) Crie dois vetores de números inteiros: vetor1 e vetor2, ambos com a mesma quantidade de elementos. Depois realize as seguintes operações:

a. some os dois vetores.
b. subtraia o vetor 1 do vetor2.
c. multiplique os dois vetores, elemento por elemento.

2) Crie duas matrizes de dimensões 3x3. Depois realize as seguintes operações:

a. some as duas matrizes.
b. subtraia a segunda matriz da primeira.
c. multiplique as duas matrizes.

3. Crie um data.frame chamado alunos com três colunas: Nome, Idade e Nota. Preencha o data.frame com pelo menos 5 registros. Em seguida, crie uma nova coluna chamada Aprovado, que será TRUE se a Nota for maior ou igual a 5 e FALSE caso contrário.

4) Escreva uma função que retorna uma matriz “E” em que cada elemento é o valor “b” elevado a cada elemento da matriz M. Como padrão, b=e.

5) Escreva uma função que recebe dois parâmetros: base e altura, e retorna a área de um triângulo.

6) Escreva uma que recebe um vetor de números e retorna uma lista com a quantidade de números positivos e a quantidade de números negativos no vetor.

Chapter 4

Dados

4.1 Processos de obtenção, importação, organização e transformação

- **Obtenção**: experimentos controlados, estudos observacionais, etc.
- **Importação**: armazenar (ou importar) os dados em um formato compatível com software utilizado, aqui utilizaremos o R.
- **Organização**: colocar os dados em uma estrutura consistente. Normalmente, cada linha é a uma observação e cada coluna é uma variável.
- **Transformação**: criar novas variáveis como função das variáveis existentes, restringir observações de interesse, calcular medidas resumo, etc.

Ordem Lançamento	Filme	Data	Duração	Bilheteria (Milhões)	Gênero
1	Toy Story	1995-11-22	81	373	Aventura
2	Vida de inseto	1998-11-25	95	363	Comédia
3	Toy Story 2	1999-11-24	92	497	Aventura
4	Monstros S. A.	2001-11-02	92	632	Aventura
5	Procurando Nemo	2003-05-30	100	871	Comédia
6	Os Incríveis	2004-11-05	115	631	Ação
7	Carros	2006-06-09	117	461	Esporte
8	Ratatouille	2007-06-29	111	623	Aventura
9	WALL-E	2008-06-27	98	521	Ficção Científica
10	Up Altas Aventuras	2009-05-29	96	735	Drama

A tabela acima é uma versão reduzida do banco de dados "filmes_pixar", disponível para download **aqui**. Este banco de dados foi elaborado para servir de base na construção de tabelas e gráficos nesse e nos próximos capítulos. Já a versão reduzida será usada para facilitar os cálculos e a resolução de exemplos na lousa, durante a aula.

Formalmente, uma amostra é uma coleção de vetores aleatórios, $X_1, X_2, \cdots, X_n$, independente e indenticamente distribuida ($i.i.d$), com $X_i = X_{i1}, X_{i2}, \cdots, X_{ik}$,

em que o primeiro índice se refere à unidade amostral (ou seja, a linha do banco de dados) e o segundo índice se refere a uma característica (variável) da unidade amostral (ou seja, a coluna do banco de dados).

Denotaremos aqui os valores observados por letras minusculas, por exemplo, $x_1, x_2, \cdots, x_n$, são os valores observados em uma particular amostra de $X_1, X_2, \cdots, X_n$.

4.2 Tipos de Variáveis

1. **Qualitativas:** atributos não numéricos
   - *Nominal*
     - Nomes ou rótulos, sem uma relação de ordem
     - Exemplos: Sexo, Religião, Cor dos Olhos, Time de Futebol
   - *Ordinal*
     - As diferentes categorias podem ser colocados em ordem
     - Exemplos: Faixa Etária, Escolaridade, Classe Social
2. **Quantitativas:** atributos numéricos
   - *Discretas*
     - Assume uma quantidade enumerável de valores
     - Exemplos: Número de Filhos, Quantidade de Erros na Prova, Número de Livros Lidos em 2023
   - *Contínuas*
     - Assume uma quantidade não enumerável de valores
     - Exemplos: Altura, Pressão, Tempo

4.3 Tabelas de Frequências

- Tabela contendo frequências absolutas e/ou relativas de cada categoria de uma *variável qualitativa.*

Gênero	Freq	FreqRel
Aventura	8	0.348
Ação	2	0.087
Comédia	4	0.174
Drama	3	0.130
Esporte	3	0.130
Ficção Científica	1	0.043
Musical	2	0.087

Pode-se afirmar que nesta amostra, o gênero predominante é aventura (34,7% dos filmes).

- Para *variáveis qualitativas ordinais*, pode-se também considerar as frequências relativas acumuladas.
- Também é possível fazer tabela de frequências para *variáveis quantitativas discretas*. Para algumas variáveis, como a duração do filme, poucos valores se repetem. Nesses casos, é comum agrupar os valores dessas variáveis em classes e calcular a frequência de cada classe.

Faixas_duração	Freq	FreqRel
80 |– 90	1	0.043
90 |– 100	12	0.522
101 |– 110	6	0.261
111 |– 120	4	0.174

- Por fim, para *variáveis quantitativas contínuas*, também podemos usar

Bilheteria_Mundial	Freq	FreqRel	FreqAcum
1.1 bilhão |– 1.3 bilhão	1	0.043	0.043
300 milhões |– 500 milhões	8	0.348	0.391
500 milhões |– 700 milhões	6	0.261	0.652
700 milhões |– 900 milhões	5	0.217	0.869
900 milhões |– 1.1 bilhão	3	0.130	1.000

- A quantidade e o tamanho das faixas é arbitrário. Contudo, um número muito pequeno de classes pode ocasionar perda de informação, enquanto um número muito grande de classes pode prejudicar o objetivo de resumir os dados.

- Por fim, as faixas podem ter tamanhos diferentes. No entanto, a análise dessas classes deve ser feito com cuidado. A escolha de classes com tamanhos diferentes normalmente só é feita quando há poucas observações em algum intervalo.

4.4 Manipulação de Dados usando o tidyverse

A manipulação de dados no R tem como objetivo organizar, filtrar e transformar um banco de dados. O *tidyverse* é um conjunto de pacotes que compartilham uma filosofia de design e uma gramática comum, tornando a manipulação de dados mais intuitiva e fluida. Os pacotes principais são:

Tibble

O tibble é uma estrutura de dados do R, sendo uma versão atualizada do data.frame. Ele é mais completo, mais legível e menos propenso a erros do que o data.frame, uma vez que não converte automaticamente strings em fatores e permite colunas com tipos de dados mais complexos (como listas e funções). Além disso, o tibble se encaixa devidamente com outros pacotes do tidyverse.

Para transformar um data.frame já existente em tibble:

```
library(tibble)
exemplo_tibble <- as_tibble(meu_dataframe)
```

Ou então, basta criá-lo diretamente

```
meu_tibble <- tibble(
  x = 1:4,
  y = c("a", "b", "c", "d"),
  z = x^2
)
```

O operador pipe

O pipe (%>%) é um operador do pacote *magrittr* que encadeia funções, ou seja, ele pega o resultado de uma expressão e passa como argumento para a próxima função. A ideia é que ao invés de escrever o código de dentro pra fora, escrevemos passo a passo, numa sequência lógica. Por exemplo

```
# As duas linhas representam a mesma coisa
f(x, y)
x %>% f(y)
```

```
# Raiz quadrada sem o pipe
x <- c(5, 3, 1, 0, 4, 2, 1)
sqrt(sum(x))
```

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

```
# Raiz quadrada com o pipe
library(magrittr)
x <- c(5, 3, 1, 0, 4, 2, 1)
x %>% sum() %>% sqrt()
```

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

O pacote dplyr

O dplyr é o pacote mais comum para manipulação de dados no R. Suas principais funções são:

- select() - seleciona colunas
- arrange() - ordena a base
- filter() - filtra linhas
- mutate() - cria/modifica colunas
- group_by() - agrupa a base
- summarise() - sumariza a base

Todas essas funçõees tem como entrada e como saída uma tibble. Além disso, o dplyr facilida o uso do operador *pipe*.

Exemplo:

```
library(dbplyr)
dados %>%
  filter(idade > 18) %>%
  group_by(genero) %>%
  summarise(media_salario = mean(salario))
```

O exemplo acima resulta em uma tabela mostrando, para cada gênero, a média de salário das pessoas maiores de 18 anos. Ou seja, primeiro filtramos (*filter*) apenas as pessoas com idade acima de 18, agrupamos (*group_by*) os dados por gênero e resume (*summarise*) calculando a média dos salários de cada grupo.

Outra coisa que o pacote dplyr permite fazer é combinar duas tabelas com base em uma ou mais colunas em comum, o nome disso é *join*. Os principais tipos de join são:

- inner_join() - mantém apenas as linhas que aparecem nas duas tabelas.
- left_join() → mantém todas as linhas da tabela da esquerda, preenchendo com NA o que não casar na direita.
- right_join - mantém todas as linhas da tabela da direita.
- full_join - mantém todas as linhas das duas tabelas.

Um exemplo:

```
library(dplyr)

clientes <- tibble(cliente_id = c(1, 2, 3),
                   nome = c("Victor", "Letícia", "Mariana"))

compras <- tibble(cliente_id = c(1, 2, 4),
                  valor = c(10, 20, 30))

# Juntar apenas quem aparece nos dois
inner_join(clientes, compras, by = "cliente_id")
```

```
## # A tibble: 2 x 3
##   cliente_id nome    valor
##        <dbl> <chr>   <dbl>
## 1          1 Victor     10
## 2          2 Letícia    20
```

O pacote Stringr

O pacote *stringr* é um dos pacotes do *tidyverse* especializado em manipulação de strings (textos). Seu objetivo é tornar o trabalho com strings mais simples, eficiente e legível. Suas principais funções são:

- str_detect() - Verifica se um padrão existe em uma string (retorna TRUE/FALSE).
- str_subset() - Filtra strings que contêm determinado padrão.
- str_split() - Divide strings com base em um delimitador.
- str_length() - Retorna o número de caracteres da string.
- str_trim() Remove espaços em branco no início e no fim.
- str_to_lower() / str_to_upper() Converte texto para minúsculas / maiúsculas.

Exemplo:

```
library(tidyverse)
dados <- tibble(nome = c("Letícia", "Mariana", "Mateus", "Guilherme"))
dados %>%
  mutate(comeca_com_B = str_detect(nome, "^G"))
```

```
## # A tibble: 4 x 2
##   nome      comeca_com_B
##   <chr>     <lgl>
## 1 Letícia   FALSE
## 2 Mariana   FALSE
## 3 Mateus    FALSE
## 4 Guilherme TRUE
```

O exemplo abaxio cria uma coluna que retorna *TRUE* se o nome começa com B e *FALSE* caso contrário.

O pacote lubridate

Trabalhar com datas e horários na base do R pode ser complicado, uma vez que, pode ser cheio de erros e formatos diferentes, então o pacote *lubridade*, do *tidyverse* foi criado para deixar tudo isso mais fácil. Suas principais funções são:

- second() - extrai os segundos.
- minute() - extrai os minutos.
- hour() - extrai a hora.
- wday() - extrai o dia da semana.
- mday() - extrai o dia do mês.
- month() - extrai o mês.
- year() - extrai o ano.

```
# Data e hora (editados pela última vez)
library(lubridate)
today()
```

```
## [1] "2025-05-07"
```

```
hour(now())
```

```
## [1] 21
```

O pacote Forcats

Embora o R base já tenha suporte a fatores, manipular níveis, ordenar ou recodificar fatores pode ser meio confuso e gerar muito error. O pacote *forcats* (FOR Categorial VariableS) é um pacote do *tidyverse* foi criado para facilitar esse trabalhar e tornar a manipulação de fatores mais simples.

O *forcats* criar fatores de forma controlada, renomeia e recodifica níveis, reordena níveis com base em frêquencia ou valores, lida com fatores não utilizados. Algumas de suas funções são:

- fct_relevel() - reordena níveis manualmente, movendo um ou mais para frente.
- fct_reorder() - reordena níveis de um fator com base em outra variável .
- fct_infreq() - ordena os níveis do fator pela frequência (do mais comum ao menos comum).
- fct_recode() - renomeia os níveis.
- fct_count() - conta as observações por nível.
- fct_lump() - Junta os níveis menos frequentes em "Outro".

Exemplo:

```
library(forcats)
nivel <- factor(c("alto", "baixo", "médio", "baixo", "alto", "baixo"))
fct_infreq(nivel)
```

```
## [1] alto  baixo médio baixo alto  baixo
## Levels: baixo alto médio
```

O código acima reordena os fatores por frequência.

4.5 Exercícios

1. Usando o conjunto de pacotes *tidyverse* e o banco de dados do R mtcars, faça seguinte:

(a) Filtre apenas os carros que têm 6 cilindros.
(b) Selecione apenas as colunas: mpg, hp e wt.
(c) Ordene o resultado do maior para o menor consumo de combustível (mpg).

2. Primeiro, crie um vetor com o nome e sobrenome de 5 pessoas, depois, usando o pacote *stringr* separe o nome e o sobrenome de cada pessoa.

3. Crie o fator:

```
cores <- factor(c("rosa", "vermelho", "azul", "amarelo", "rosa",
                  "verde", "azul", "rosa"))
```

A partir disso: (a) Reordene os níveis para que a cor mais frequente venha primeiro. (b) Agrupe todas as cores menos frequentes que "azul" em um novo nível chamado "Outro".

Chapter 5

Medidas de uma variável

5.1 Medidas de posição ou de Tendência Central

- Utilidadas para resumir variáveis quantitativas.
- Dão a ideia do lugar da reta estão concentrados os valores de uma variável.

5.1.1 Média (Aritmética)

- A medida mais utilizada é a **média**, definida por

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + \ldots + x_n}{n} \ .$$

*Exemplo - Duração dos filmes da pixar:

$$\bar{x} = \frac{81 + 95 + 92 + 92 + 100 + 115 + 117 + 111 + 98 + 96 + 103 + 106 + 93 + 104 + 95 + 93 + 97 + 102 + 105 +}{23}$$

- Alternativamente, quando há empates (isto é, n_1 observações do valor x_1, n_2 observações do valor x_2, e assim por diante), podemos calcular a média por

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{k} n_i \ x_i = \sum_{i=1}^{k} f_i \ x_i = \frac{n_1 \ x_1 + n_2 \ x_2 + \ldots + n_k \ x_k}{n} \ ,$$

em que $k \leq n$ é a quantidade de valores diferentes assumidos pela variável X e $\sum n_i = n$. Ainda pode-se considerar para o cálculo da média a frequência relativa $f_i = n_i/n$.

- Quando os dados estão agrupados em classes, podemos calcular um valor aproximado da média fazendo

$$\bar{x} \approx \sum_{i=1}^{k} f_i \; \bar{x}_i \; ,$$

em que $\bar{x}_i$ é o valor médio da i-ésima classe e f_i é a frequência relativa daquela classe.

5.1.2 Mediana

- A **mediana** é o valor central dos dados. Pode ser calculada por

$$md(x) = \begin{cases} x_{(\frac{n+1}{2})}, & \text{se } n \text{ é ímpar} \\ \dfrac{x_{(\frac{n}{2})} + x_{(\frac{n}{2}+1)}}{2}, & \text{se } n \text{ é par} \end{cases}$$

em que $x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(n)}$, $x_{(1)}$ é o menor valor observado na amostra, $x_{(2)}$ é o segundo menor valor, ..., $x_{(n)}$ é o maior valor na amostra. $x_{(1)}, x_{(2)}, \ldots, x_{(n)}$ são chamados *estatísticas de ordem.*
* Para a duração do filme: $md(x) = 1.725$

- **Observação:** a mediana é uma medida mais robusta que a média pois é menos afetada por valores extremos.

5.1.3 Moda

- A **moda** é o valor que mais frequente na amostra.

- Em alguns casos, pode existir mais de uma moda.
  - Para a duração do filme: $mo(x) = 100$

5.2 Medidas de Dispersão

- Na maioria dos casos, somente as medidas de posição trazem pouca informação sobre a variável de interesse.

- Por exemplo, considere as seguintes amostras de tamanho 3: $(1, 5, 9)$, $(4, 5, 6)$ e $(5, 5, 5)$. Em todas elas, $\bar{x} = md(x) = 5$. Contudo, na primeira os valores estão mais "espalhados", enquanto na última os valores estão mais "concentrados".

- Para descrever melhor esta diferença, podemos usar medidas que nos informem o quanto os dados estão "espalhados", ou como é a dispersão dos dados.

5.2.1 Variância

- A variância é a média dos desvios ao quadrado das observações com relação à média, dada por

$$var(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 .$$

  - No exemplo acima:
    1. para a Amostra 1: $\frac{(1-5)^2 + (5-5)^2 + (9-5)^2}{3} = \frac{16+0+16}{3} =$ 10.666;
    2. para a Amostra 2: $\frac{(4-5)^2 + (5-5)^2 + (6-5)^2}{3} = \frac{1+0+1}{3} =$ 0.666;
    3. para a Amostra 3: $\frac{(5-5)^2 + (5-5)^2 + (5-5)^2}{3} = 0.$
  - No exemplo da duração do filme: $var(x) = 73$

5.2.2 Desvio padrão

- A variância está em uma escala diferente da variável observada. Uma forma de contornar isso é calcular sua raiz. A medida resultante é chamada de desvio padrão: $dp(x) = \sqrt{var(x)}$

  - No exemplo acima:
    1. para a Amostra 1: $\sqrt{\frac{(1-5)^2 + (5-5)^2 + (9-5)^2}{3}} = \sqrt{32/3} =$ 3.265986;
    2. para a Amostra 2: $\sqrt{\frac{(4-5)^2 + (5-5)^2 + (6-5)^2}{3}} = \sqrt{2/3} =$ 0.8164966;
    3. para a Amostra 3: $\sqrt{\frac{(5-5)^2 + (5-5)^2 + (5-5)^2}{3}} = 0.$
  - No exemplo da duração do filme: $dp(x) = \sqrt{73} = 8.54\ m$

5.2.3 Desvio médio ou absoluto

- Outra medida de dispersão é o desvio médio (ou absoluto):

$$dm(x) = \frac{1}{n}\sum_{i=1}^{n} |x_i - \bar{x}| .$$

  - No exemplo acima:
    1. para a Amostra 1: $\frac{|1-5|+|5-5|+|9-5|}{3} = \frac{8}{3} \approx 2.666$;
    2. para a Amostra 2: $\frac{|4-5|+|5-5|+|6-5|}{3} = \frac{2}{3} \approx 0.666$;
    3. para a Amostra 3: $\frac{|5-5|+|5-5|+|5-5|}{3} = 0$.

- O desvio médio, assim como o desvio padrão, representam "erros" médios ao aproximar os valores observados pela média.

5.3 Medidas de ordem

5.3.1 Quartis e quantis

- Como vimos, a mediana é o valor que "divide ao meio" a amostra.

- De forma similar, podemos dividir a amostra em partes menores. Por exemplo, pode ser de interesse considerar os valores que dividem a amostra nos 5% menores valores, 10% menores, 20% e assim por diante.

- O *quantil* de ordem p ou p-quantil, $0 < p < 1$, é o valor $Q(p)$ tal que $100 \cdot p$ % das observações sejam menores do que $Q(p)$.

- Há diversas formas de definir os quantis amostrais (veja, por exemplo, a ajuda do R para a função *quantile*). Aqui, por simplicidade, vamos considerar a definição a seguir.

$$Q(p) = \begin{cases} x_{(i)} & \text{se} \quad p = p_i = \frac{i-0.5}{n}, \quad i = 1, \ldots, n \\ (1-f_i)Q(p_i) + f_i Q(p_{i+1}) & \text{se} \quad p_i < p < p_{i+1} \\ x_{(1)} & \text{se} \quad 0 < p < p_1 \\ x_{(n)} & \text{se} \quad p_n < p < 1 \end{cases}$$

Em que

$$f_i = \frac{p - p_i}{p_{i+1} - p_i}$$

- Quando houver empates (valores iguais) na amostra, vamos considerar o maior p_i entre as observações empatadas.

- Outros quantis podem ser calculados, como por exemplo,

  - $Q(0.5) = \left(1 - \frac{0.5-0.45}{0.55-0.45}\right) Q(0.45) + \left(\frac{0.5-0.45}{0.55-0.45}\right) Q(0.55) = \frac{1}{2} \cdot 1.70 + \frac{1}{2} \cdot 1.75 = 1.725 = md(x)$
  - $Q(0.83) = \left(1 - \frac{0.83-0.75}{0.85-0.75}\right) Q(0.75) + \left(\frac{0.83-0.75}{0.85-0.75}\right) Q(0.85) = 0.2 \cdot 1.79 + 0.8 \cdot 1.81 = 1.806$
  - $Q(0.13) = \left(1 - \frac{0.13-0.05}{0.25-0.05}\right) Q(0.05) + \left(\frac{0.13-0.05}{0.25-0.05}\right) Q(0.25) = 0.6 \cdot 1.50 + 0.4 \cdot 1.60 = 1.54$

- Os quantis $Q(0.25)$, $Q(0.50)$ e $Q(0.75)$ são chamados de primeiro, segundo e terceiro **quartis** e são denotados por q_1, q_2 e q_3, respectivamente. Como já foi dito, $q_2 = md(x)$.

5.3.2 Distância (ou Amplitude) Interquartis

- Outra medida de dispersão bastante utilizada é a *distância interquartis*, definida por

$$d_Q = q_3 - q_1 \ .$$

- A distância interquartis é a amplitude do intervalo que concentra 50% das observações centrais.

5.3.3 Desvio Mediano Absoluto

- Como a mediana é uma medida mais robusta que a média, é possível estabelecer também uma medida de dispersão em termos de desvios em relação à mediana. Assim, defina o *desvio mediano absoluto* como

$$dma(x) = md(|x_i - md(x)|) \ .$$

5.3.4 Amplitude

- Distância entre o maior e o menos valor observado.

$$\Delta = x_{(n)} - x_{(1)}$$

5.4 Calculo de medidas no R

- Média: função mean()

```
media <- mean(c(10, 20, 30, 40, 50, 55, 60))
```

- Mediana: função median()

```
mediana <- median(c(10, 20, 30, 40, 50, 68))
```

- Moda: o R não possui uma função embutida para moda, porém podemos calcular dessa forma:

```
dados <- c(1, 2, 2, 3, 3, 3, 4)

frequencias <- table(dados)
moda <- as.numeric(names(frequencias[frequencias == max(frequencias)]))
```

- Quartis: função quantile()

```
quartis <- quantile(c(10, 20, 30, 40, 50))
```

- Variância: função var()

```
dados <- c(10, 12, 23, 23, 16, 23, 21, 16)
variancia <- var(dados)
```

- Desvio padrão: função sd()

```
dados <- c(10, 12, 23, 23, 16, 23, 21, 16)
desvio_padrao <- sd(dados)
```

- Desvio mediano absoluto: função´mad()

```
dados <- c(10, 12, 23, 23, 16, 23, 21, 16)
desvio_mediano_absoluto <- mad(dados)
```

5.5 Exercícios

1. Considere uma amostra de tamanho n, $x_1, \ldots, x_n$. Mostre que

$$\sum_{i=1}^{n}(x_i - \bar{x}) = 0$$

2. Uma empresa está realizando um levantamento sobre o tempo médio de espera de seus clientes em uma fila de atendimento. Para isso, foi coletado o número de minutos que os 50 últimos clientes ficaram aguardando na fila. Os dados, já ordenados, são os seguintes:

1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 6, 7, 7, 8, 15.

Calcule a média, moda, mediana e quartis.

3. Prove que $\sum(x_i - \bar{x})^2 = \sum(x_i - \mu)^2 - n(\bar{x} - \mu)^2$

Chapter 6

Modelos Gráficos

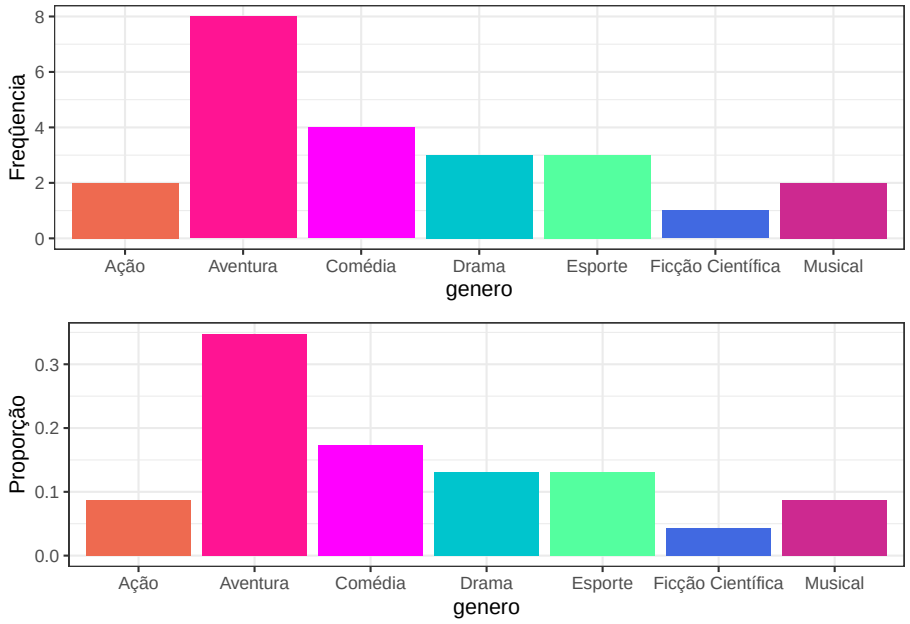
6.1 Gráfico de Barras

- O gráfico de barras é adequado para variáveis qualitativas (nominais ou ordinais) e também para variáveis quantitativas discretas.
- **Exemplo 1: Variável Qualitativa Nominal** (Gênero de filme)

Gênero	Freq	FreqRel
Aventura	8	0.3478261
Ação	2	0.0869565
Comédia	4	0.1739130
Drama	3	0.1304348
Esporte	3	0.1304348
Ficção Científica	1	0.0434783
Musical	2	0.0869565

```
graf_genero_f <- filmes_pixar %>%
  ggplot() + theme_bw() + xlab("genero") + ylab("Freqûencia") +
  geom_bar(aes(x = genero, fill= genero)) +
  scale_fill_manual("genero", values = c("Aventura" = "deeppink",
              "Comédia" = "magenta", "Ação" = "coral2",
              "Esporte" = "seagreen1", "Drama" = "turquoise3",
              "Musical" = "maroon3", "Ficção Científica" = "royalblue"))

graf_genero_p <- tab_genero %>%
  ggplot() + theme_bw() + xlab("genero") + ylab("Proporção") +
  geom_bar(aes(x= Gênero, fill= Gênero, y=FreqRel), stat="identity") +
  scale_fill_manual("genero", values = c("Aventura" = "deeppink",
```

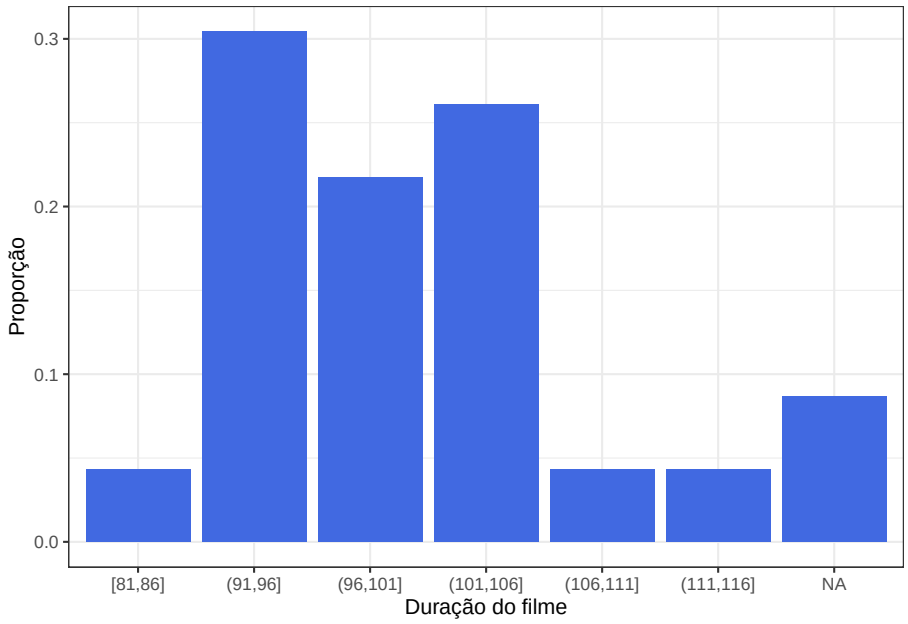
```
                "Comédia" = "magenta", "Ação" = "coral2",
                "Esporte" = "seagreen1", "Drama" = "turquoise3",
                "Musical" = "maroon3", "Ficção Científica" = "royalblue"))

# ggpubr::ggarrange(graf_genero_f,graf_genero_p, legend="none")
ggpubr::ggarrange(graf_genero_f, graf_genero_p, legend = "none", ncol = 1)
```



- **Exemplo 2: Variável Quantitativa** (duração do filme)

Faixa_duracao	Freq	FreqRel
[81,86]	1	0.0434783
(91,96]	7	0.3043478
(96,101]	5	0.2173913
(101,106]	6	0.2608696
(106,111]	1	0.0434783
(111,116]	1	0.0434783
NA	2	0.0869565

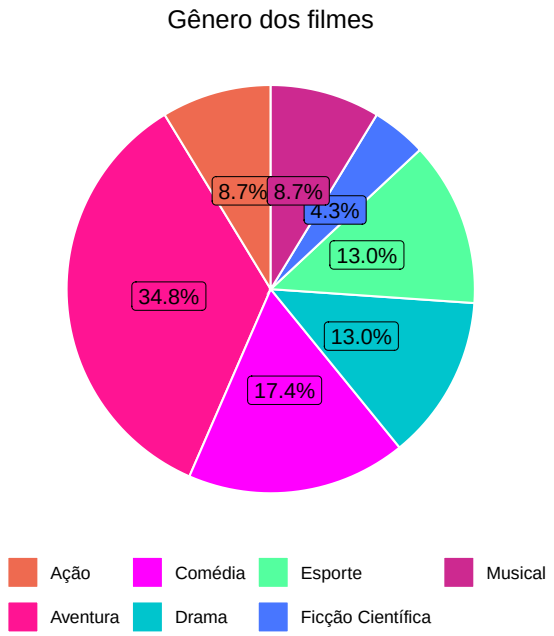
```
tab_duracao %>%
  ggplot() + theme_bw() + xlab("Duração do filme") + ylab("Proporção") +
  geom_bar(aes(x= Faixa_duracao, y= FreqRel), fill="royalblue", stat="identity")
```



6.2 Gráfico de Setores (Pizza)

- Pode ser utilizado para variáveis qualitativas.

```
pizza_genero <- tab_genero %>%
  mutate(Porc = scales::percent(FreqRel)) %>%
  ggplot(aes(x = "", y = FreqRel, fill = Gênero)) +
  geom_col(color = "white") +
  geom_label(aes(label = Porc), #color = c(1, "white", "white"),
        position = position_stack(vjust = 0.5),
        show.legend = FALSE) + guides(fill = guide_legend(title = "")) +
        scale_fill_manual("Gênero dos filmes", values =c("Aventura" = "deeppink",
                          "Comédia" = "magenta", "Ação" = "coral2",
                          "Esporte" = "seagreen1", "Drama" = "turquoise3",
                          "Musical" = "maroon3", "Ficção Científica" = "royalblue1")) +
  coord_polar(theta = "y") + ggtitle("Gênero dos filmes") +
  theme_void() + theme(legend.position="bottom",
                       plot.title = element_text(hjust = 0.5))

ggpubr::ggarrange(pizza_genero)
```



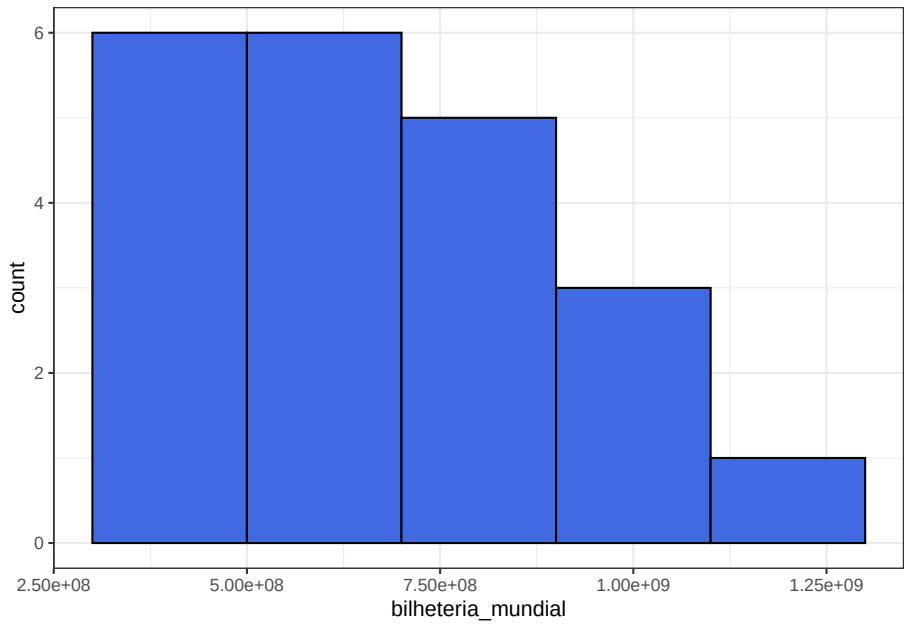
- Por que deve ser evitado?
  - Quando as frequências são muito pequenas (abaixo de 5%, por exemplo), as fatias se tornam de difícil visualização.
  - Dependem do uso de cores. Isso pode dificultar a escolha de cores que sejam suficientemente contrastantes para uma melhor visualização. Isso pode ser ainda mais prejudicado dependendo do dispotitivo que for visualizar o gráfico (se a impressão ou o monitor for de baixa qualidade, por exemplo). Por fim, isso pode dificultar a visualização por pessoas que tem dificuldades em enxergar cores (cerca de 8% da população masculina é daltônica, por exemplo).
  - A comparação direta entre dois gráficos de pizza é bem mais difícil que em gráficos de barras. No segundo é bem mais facil visualizar diferenças, se esse for seu objetivo.
  - Ainda assim, podem ser utilizados em casos específicos onde os problemas anteriores não ocorrem (quando há poucas categorias, nenhuma delas com frequencias muito baixas e as diferenças são muito evidentes ou o objetivo não é fazer comparações).

6.3 Histograma

- Adequado para variáveis quantitativas (contínuas).
- Apesar de ser parecido com o gráfico de barras, no histograma as bases dos retângulos são proporcionais aos intervalos das classes e as áreas de cada retângulo devem ser proporcionais às frequências de cada classe.

Bilheteria_Mundial	Freq	FreqRel
1.1 bilhão |– 1.3 bilhão	1	0.0434783
300 milhões |– 500 milhões	8	0.3478261
500 milhões |– 700 milhões	6	0.2608696
700 milhões |– 900 milhões	5	0.2173913
900 milhões |– 1.1 bilhão	3	0.1304348

```
filmes_pixar %>% ggplot() + theme_bw() +
  geom_histogram(aes(bilheteria_mundial), color="black", fill="royalblue",
                 breaks= c(300000000, 500000000, 700000000, 900000000,
                           1100000000, 1300000000))
```



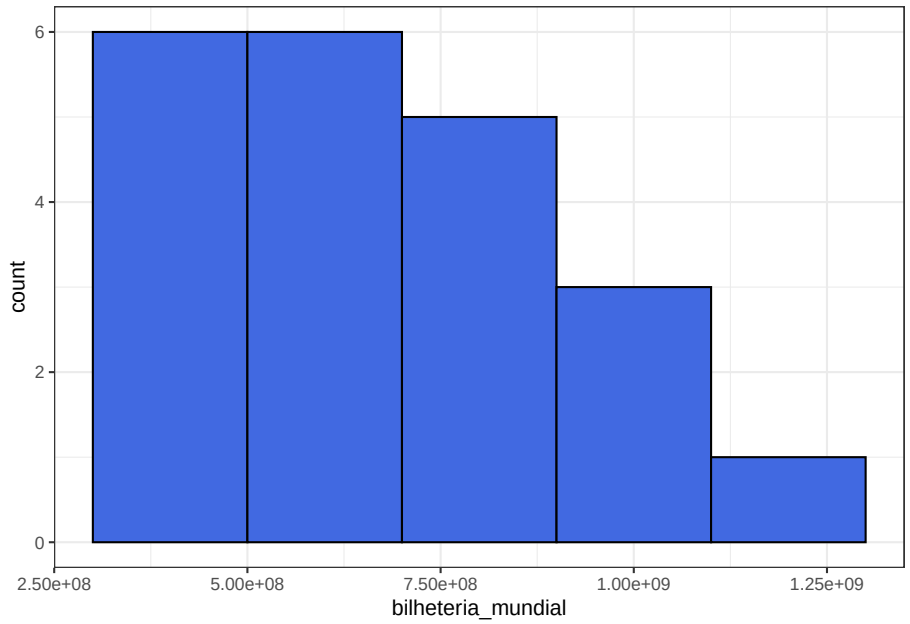
- Quando as faixas tem tamanhos diferentes, não é adequado usar as frequências absolutas ou relativas no eixo y pois a área do gráfico

correspondente pode dar a impressão de que as frequências são maiores do que efetivamente foi observado. É possível ver isso no gráfico a seguir.

- Neste caso, o ideal é utilizar a *densidade de frequência* no eixo y, dada por $d_i = \frac{f_i}{\delta_i}$, onde δ_i é o comprimento da faixa.

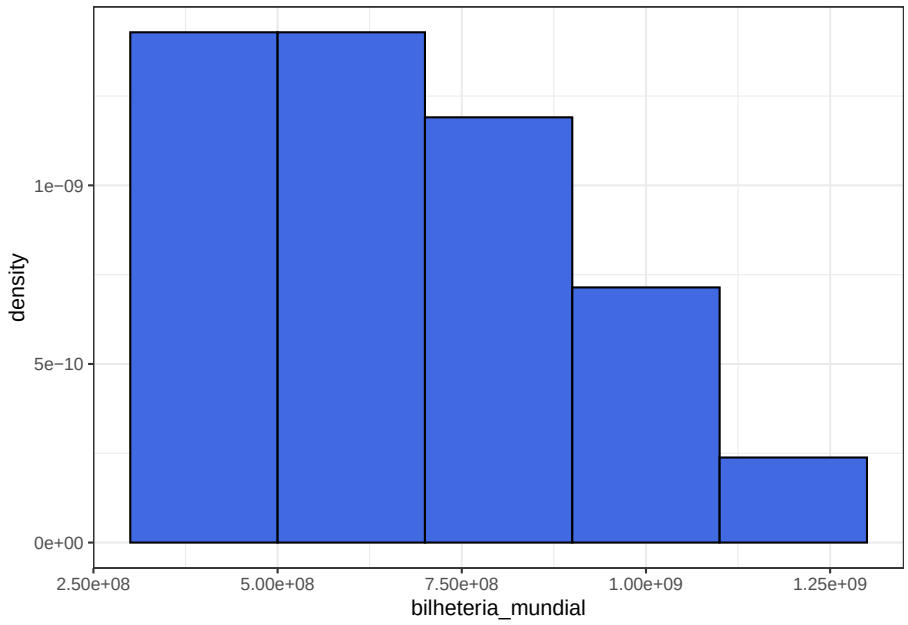
Bilheteria_Mundial	Freq	FreqRel	delta	Dens
1.1 bilhão |– 1.3 bilhão	1	0.0434783	0.2	0.2173913
300 milhões |– 500 milhões	8	0.3478261	0.1	3.4782609
500 milhões |– 700 milhões	6	0.2608696	0.1	2.6086957
700 milhões |– 900 milhões	5	0.2173913	0.1	2.1739130
900 milhões |– 1.1 bilhão	3	0.1304348	0.1	1.3043478

```
#hist_bilheteria_c <-
filmes_pixar %>% ggplot() + theme_bw() +
  geom_histogram(aes(bilheteria_mundial), color="black", fill="royalblue",
                 breaks= c(300000000, 500000000, 700000000, 900000000,
                           1100000000, 1300000000))
```




```
#hist_bilheteria_m <-
filmes_pixar %>% ggplot() + theme_bw() +
  geom_histogram(aes(bilheteria_mundial,after_stat(density)),
                 color="black", fill="royalblue",
```

```
                breaks= c(300000000, 500000000, 700000000, 900000000,
                          1100000000, 1300000000))
```




```
#ggpubr::ggarrange(hist_bilheteria_c, hist_bilheteria_m, ncol=1)
```

- Note que desta forma, a área total do histograma é igual a 1.

6.3.1 Números de faixas e largura

Para construir um histograma, não existe um número correto de faixas, e diferentes larguras podem revelar diferentes aspectos dos dados. Faixas mais largas ajudam a reduzir o ruído onde há poucos dados, enquanto faixas mais estreitas aumentam a precisão onde há muitos dados. Existem diferentes métodos para fazer essa escolha, alguns exemplos são:

Fórmula de Sturges:

Uma opção simples é usar a *Fórmula de Sturges*, que calcula o número de faixas a partir do tamanho da amostra:

$$k = \lceil \log_2 n \rceil + 1$$

Esse é o método padrão usado pelo R base. Como esse método calcula o número de faixas com base no tamanho da amostra n, ela pode ter um desempenho

ruim quando $n < 30$, pois gera poucas faixas, o que dificulta a visualização de tendências. Para grandes conjunto de dados, ela pode superestimar a largura das faixas, gerando um histogramas execivamente suavizado. Além disso, pode não funcionar bem para dados que não seguem uma distribuição simétrica.

Regra de Referência Normal de Scoot:

Outra método é a *Regra de Referência Normal de Scott*, que busca minimizar o erro na estimativa da densidade. Ela define a largura h das faixas como:

$$h = \frac{3,49 \cdot \text{desvio padrão}}{\sqrt[3]{n}}$$

Essa abordagem é melhor para dados com variabilidade semelhante à de uma distribuição simétrica.

Regra de Freedman-Diaconis:

Outro exemplo é a regra de Freedman–Diaconis, que é mais robusta a dados assimétricos ou com outliers, usando o intervalo interquartil (IQR) no lugar do desvio padrão, ele define a largura h das faixas como:

$$h = 2 \cdot \frac{\text{IQR}(x)}{\sqrt[3]{n}}$$

Ela se adapta melhor a distribuições que não são tão bem comportadas.

Exemplo no R:

Para criar um histograma no R usando o número de faixas (ou largura das faixas) usando a fórmula desejada, é possível calcular o valor manualmente e depois passá-lo para o argumento breaks da função *hist()*.

```r
# Exemplo usando a fórmula de Sturges:
n <- length(dados)  # Tamanho da amostra
k_sturges <- 1 + log2(n)  # Fórmula de Sturges

hist(dados, breaks = k_sturges)
```

Além disso, também é possível usar um número de faixas diretamente, colocando o valor de breaks como um número inteiro.

Assim, dependendo do tipo de dado e do objetivo da análise, podemos escolher o método mais adequado para definir o número de faixas no histograma. Para encontrar outros métodos e fórmulas, basta acessar o site **aqui**.

6.4 Ramos e Folhas

- Similar a um histograma mas com menos perda de informação.

duracao
81
92
92
93
93
95
95
96
97
98
100
100
100
102
102
103
104
105
106
111
115
117
118

```
stem(filmes_pixar$duracao)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    8 | 1
##    9 | 223355678
##   10 | 000223456
##   11 | 1578
```

```
stem(filmes_pixar$bilheteria_mundial / 1e9)
```

```
##
##   The decimal point is at the |
##
```

Bilheteria_Mundial	Freq	FreqRel	delta	Dens
1.1 bilhão |– 1.3 bilhão	1	0.0434783	0.1	0.4347826
300 milhões |– 500 milhões	8	0.3478261	0.1	3.4782609
500 milhões |– 700 milhões	6	0.2608696	0.1	2.6086957
700 milhões |– 900 milhões	5	0.2173913	0.1	2.1739130
900 milhões |– 1.1 bilhão	3	0.1304348	0.1	1.3043478

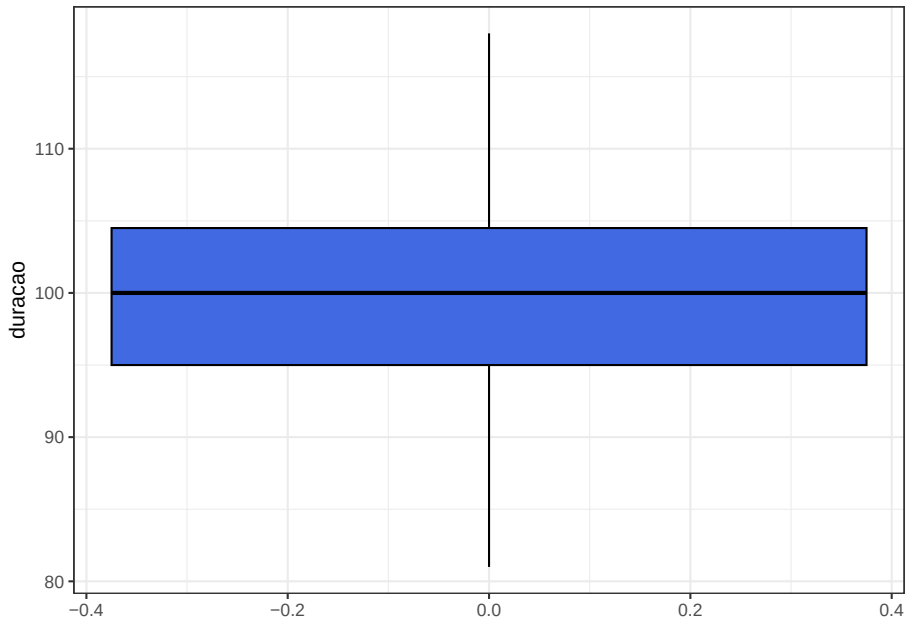
```
##   0 | 113444
##   0 | 5555666677899
##   1 | 0112
```

- Não é adequado quando temos grandes bancos de dados e não tem o mesmo efeito visual de um boxplot.

6.5 Box-Plot

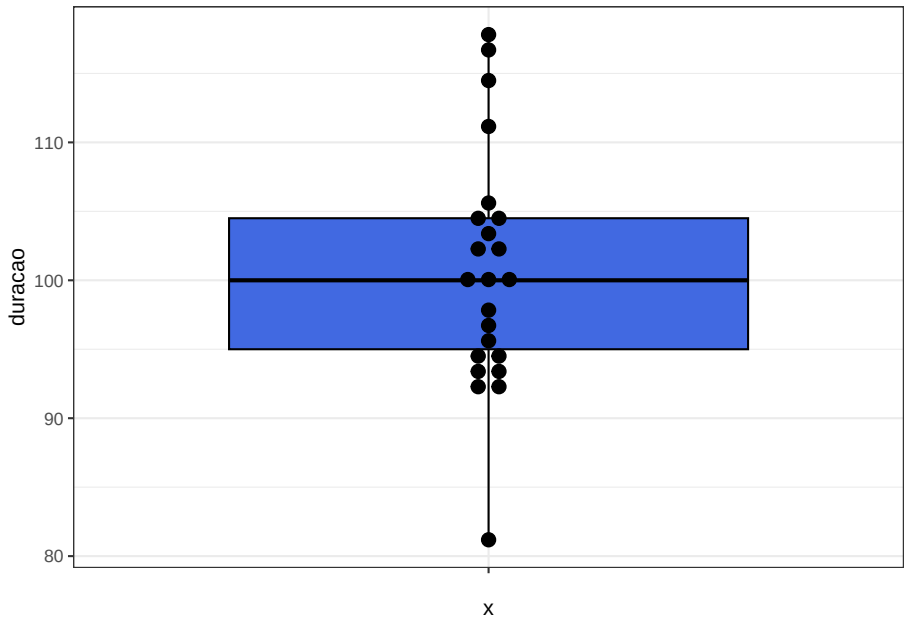
- Utilizado para representar graficamente os quartis, além dos valores mínimo e máximo.

```
filmes_pixar %>% ggplot() + theme_bw() +
  geom_boxplot(aes(y=duracao), color="black", fill="royalblue")
```



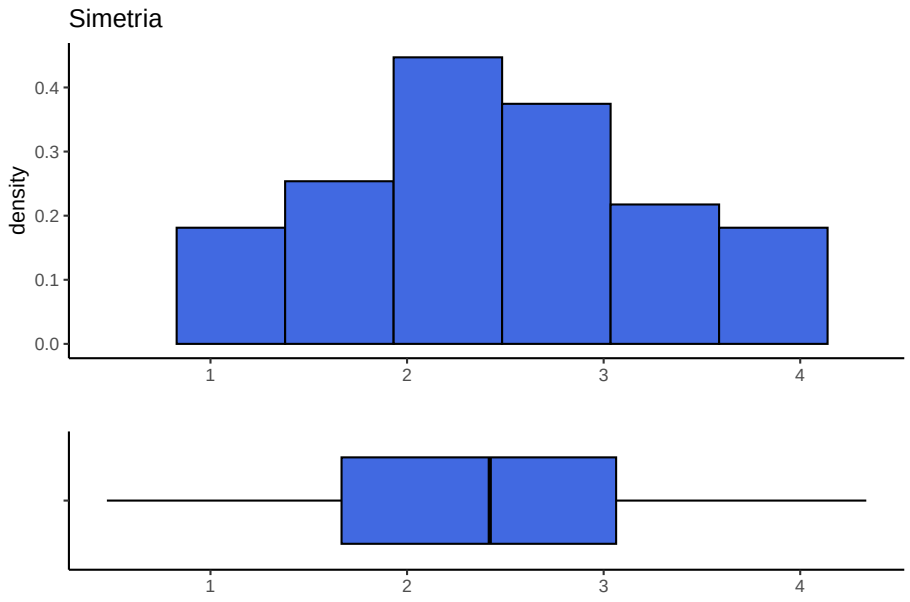
- No retângulo estão representados os quartis q_1, q_2 e q_3.
- A reta acima do retângulo se estende até o valor máximo observado, desde que esse não seja maior que $q_3 + 1.5 \cdot d_q$.
- Do mesmo modo, a reta abaixo do retângulo do retângulo se estende até o mínimo, desde que esse não seja menor que $q_1 - 1.5 \cdot d_q$.
- Se houver valores que excedam os limites acima propostos, a reta acima (abaixo) do retângulo vai até o maior (menor) valor menor (maior) que $q_3 + 1.5 \cdot d_q$ ($q_1 - 1.5 \cdot d_q$).
- Os valores fora destes limites serão representados por asteriscos e são chamados de *outliers* (ou *valores atípicos*)
- É possivel incluir os pontos observados no boxplot para não ter perda de informação.

```
filmes_pixar %>% ggplot(aes(x="",y=duracao)) + theme_bw() +
  geom_boxplot(color="black", fill="royalblue") +
  ggbeeswarm::geom_beeswarm(cex=3,size=3,method = "center")
```



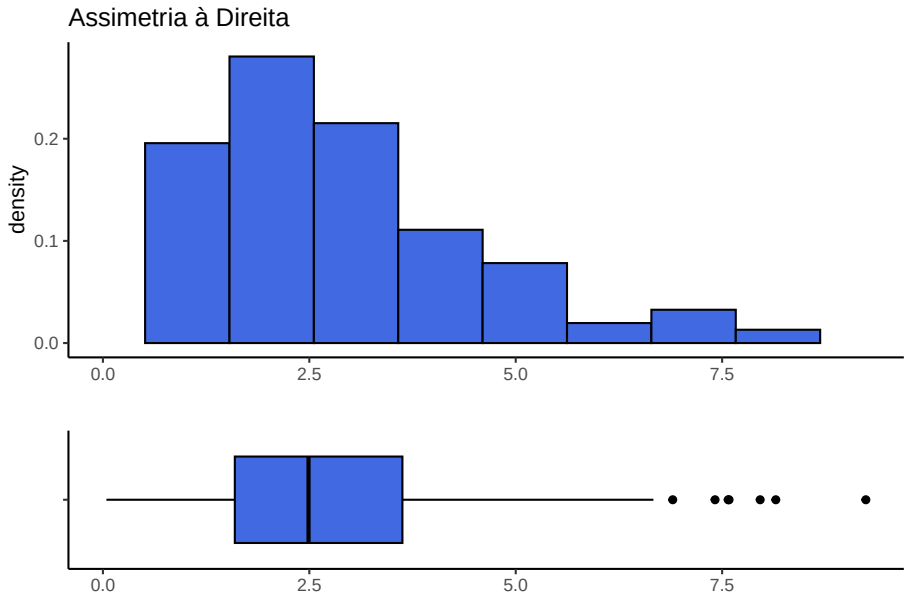
6.6 Gráficos e simetria

```
set.seed(13)

simul <- tibble(y = rnorm(150,2.5,1))
lim = c(min(simul$y),max(simul$y))

hist <- simul %>% ggplot() + theme_classic() + xlab("") + xlim(lim[1],lim[2]) +
  ggtitle("Simetria") +
  geom_histogram(aes(y,after_stat(density)),
                 color="black", fill="royalblue", bins=8)

box <- simul %>% ggplot(aes(x="",y=y)) +
  theme_classic() + coord_flip() + xlab("") + ylab("") + ylim(lim[1],lim[2]) +
  geom_boxplot(color="black", fill="royalblue")
  #ggbeeswarm::geom_beeswarm(cex=1,size=1,method = "center")

ggpubr::ggarrange(hist, box, heights = c(2, 1), nrow=2, align = "v")
```



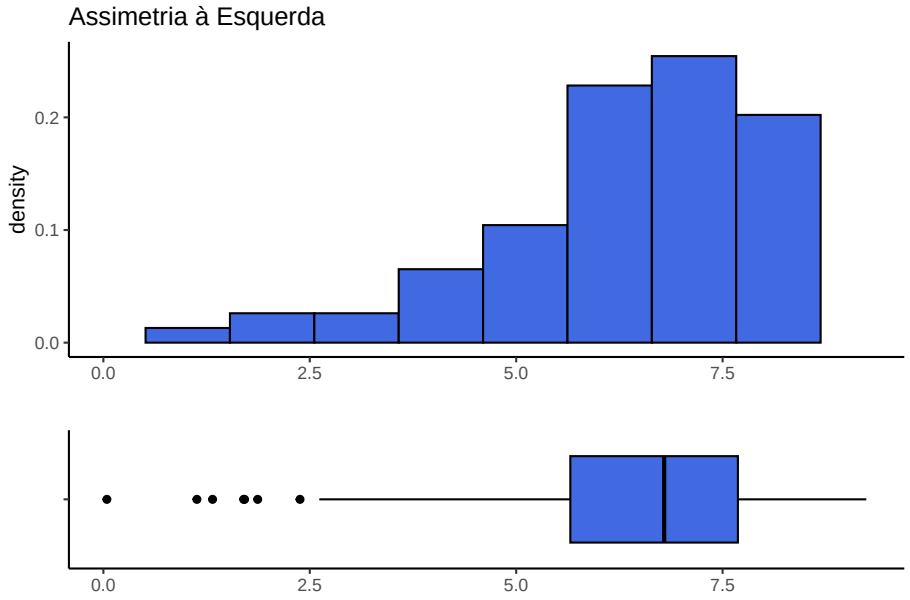

```
set.seed(13)

simul <- tibble(y = rgamma(150,2.5,1))
lim = c(min(simul$y),max(simul$y))
```

```
hist <- simul %>% ggplot() + theme_classic() + xlab("") + xlim(lim[1],lim[2]) +
  ggtitle("Assimetria à Direita") +
  geom_histogram(aes(y,after_stat(density)),
                 color="black", fill="royalblue", bins=10)

box <- simul %>% ggplot(aes(x="",y=y)) +
  theme_classic() + coord_flip() + xlab("") + ylab("") + ylim(lim[1],lim[2]) +
  geom_boxplot(color="black", fill="royalblue")
  #ggbeeswarm::geom_beeswarm(cex=1,size=1,method = "center")

ggpubr::ggarrange(hist, box, heights = c(2, 1), nrow=2, align = "v")
```




```
simul <- lim[2]-simul+lim[1]
lim = c(min(simul$y),max(simul$y))

hist <- simul %>% ggplot() + theme_classic() + xlab("") + xlim(lim[1],lim[2]) +
  ggtitle("Assimetria à Esquerda") +
  geom_histogram(aes(y,after_stat(density)),
                 color="black", fill="royalblue", bins=10)

box <- simul %>% ggplot(aes(x="",y=y)) +
  theme_classic() + coord_flip() + xlab("") + ylab("") + ylim(lim[1],lim[2]) +
  geom_boxplot(color="black", fill="royalblue")
```

```
  #ggbeeswarm::geom_beeswarm(cex=1,size=1,method = "center")

ggpubr::ggarrange(hist, box, heights = c(2, 1), nrow=2, align = "v")
```



- Os quartis são medidas de posição que auxiliam na avaliação da simetria dos dados. Para uma distribuição aproximadamente **simétrica**, espera-se que

  - $q_2 - x(1) \approx x(n) - q_2$,

  - $q_2 - q_1 \approx q_3 - q_2$,

  - $q_1 - x(1) \approx x(n) - q_3$.

- A distribiuição dos dados é dita **assimétrica à direita** se as diferenças entre os quantis situados a direita da mediana e a mediana são maiores que as diferenças entre a mediana e os quantis situados à esquerda da mediana. Se o contrário ocorre, dizemos que a distribuição é **assimétrica à esquerda**.

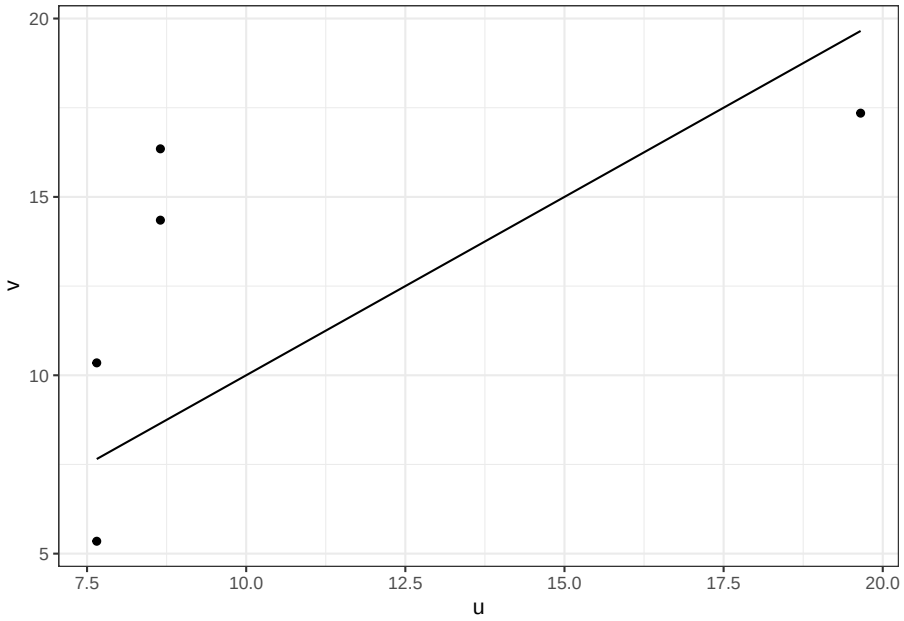
- Além disso, se uma distribuição é aproximadamente simétrica,

  - $q_2 - x_{(i)} \approx x_{(n+1-i)} - q_2$, $i = 1, \ldots, \lfloor (n+1)/2 \rfloor$, em que $\lfloor y \rfloor$ é o maior inteiro menor ou igual a y.

- Assim, defina $u_i = q_2 - x_{(i)}$ e $v_i = x_{(n+1-i)} - q_2$, para $i = 1, \ldots, \lfloor (n+1)/2 \rfloor$. Então,
  - Se a distribuição é simetrica, espera-se que $u_i \approx v_i$;
  - Se a distribuição é assimetrica à direira, espera-se que $u_i \leq v_i$;
  - Se a distribuição é assimetrica à esquerda, espera-se que $u_i \geq v_i$.

- Uma forma de fazer essa avaliação é fazer um gráfico dos pares (u_i, v_i).

```
tibble(u = 100.6522 - sort(filmes_pixar$duracao)[1:5],
       v = sort(filmes_pixar$duracao,decreasing=TRUE)[1:5] - 100.6522) %>%
  ggplot() + theme_bw() +
  geom_point(aes(x=u,y=v)) +
  geom_line(aes(x=u,y=u))
```



6.7 Medidas de assimetria

Outra forma de avaliar e descrever a simetria (ou a falta dela) são com medidas especificas de assimetria, que descrevem a inclinação ou formato da distribuição.

6.7.1 Coeficiente de Assimetria de Bowley

O coeficiente de assimetria de Bowley usa os quartis para medir a assimetria.

$$B = \frac{(q_3 - q_2) - (q_2 - q_1)}{q_3 - q_1} = \frac{(q_3 - q_2) - (q_2 - q_1)}{(q_3 - q_2) + (q_2 - q_1)} = \frac{q_3 + q_1 - 2q_2}{q_3 - q_1}$$

Onde: - q_1: primeiro quartil - q_2: mediana - q_3: terceiro quartil

Se o resultado for um valor próximo de 0, a distribuição é simétrica; se for maior que zero, a distribuição é assimétrica à direita; e, se for menor que zero, é assimétrica à esquerda.

6.7.2 Coeficiente de Assimetria de Pearson 1

O coeficiente de assimetria de Pearson 1 compara a média com a moda e é útil quando a moda é bem definida.

$$Sk_1 = \frac{\bar{x} - moda(x)}{\sqrt{\frac{n}{n-1}}\ dp(x)}$$

Onde: - $\bar{x}$ é a média amostral - $moda(x)$ é a moda da amostra - n: tamanho da amostra - $dp(x)$: desvio padrão

Se o resultado for um valor próximo de 0, a distribuição é simétrica; se for maior que zero, a distribuição é assimétrica à direita; e, se for menor que zero, é assimétrica à esquerda.

6.7.3 Coeficiente de Assimetria de Pearson 2

O coeficiente de Assimetria de Pearson 2 é mais estável que o 1, uma vez que compara a média com a mediana ($md(x)$).

$$Sk_2 = 3 \cdot \frac{\bar{x} - md(x)}{\sqrt{\frac{n}{n-1}}\ dp(x)}$$

Se o resultado for um valor próximo de 0, a distribuição é simétrica; se for maior que zero, a distribuição é assimétrica à direita; e, se for menor que zero, é assimétrica à esquerda.

6.7.4 Coeficiente de Assimetria de Fisher-Pearson

- Considere o ***k*-ésimo momento (central) amostral**, definido por

$$m_k = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^k$$

A fórmula do coeficiente de Assimetria de Fisher-Pearson é:

$$g_1 = \frac{m_3}{m_2^{3/2}}$$

Onde: - m_3: terceiro momento central - m_2: segundo momento central (variância)

Se o resultado for um valor próximo de 0, a distribuição é simétrica; se for maior que zero, a distribuição é assimétrica à direita; e, se for menor que zero, é assimétrica à esquerda.

6.7.5 Coeficiente de Assimetria de Fisher-Pearson ajustado

O coeficiente de Assimetria de Fisher-Pearson ajustado tenta corrigir o viés do estimador de Fisher-Pearson em amostras pequenas.

$$g_2 = \frac{n\sqrt{n(n-1)}}{n-1}g1$$

Se o resultado for um valor próximo de 0, a distribuição é simétrica; se for maior que zero, a distribuição é assimétrica à direita; e, se for menor que zero, é assimétrica à esquerda.

6.8 Função de distribuição empírica (FDE)

Seja $x_1, x_2, \cdots, x_n$ valores observados de uma amostra. A função de distribuição empírica é definida como:

$$\widetilde{F}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(x_i \leq x)$$

Onde,

$$\mathbb{I}_A(x) = I(x \in A) = \begin{cases} 1, & \text{se } x \in A \\ 0, & \text{se } x \notin A \end{cases}$$

Ou seja, para qualquer valor de x, $\widetilde{F}(x)$ nos diz a proporção de dados que é menor ou igual a x, obtendo uma estimativa empírica da função de distribuição acumulada verdadeira $F(x)$.

Exemplo:

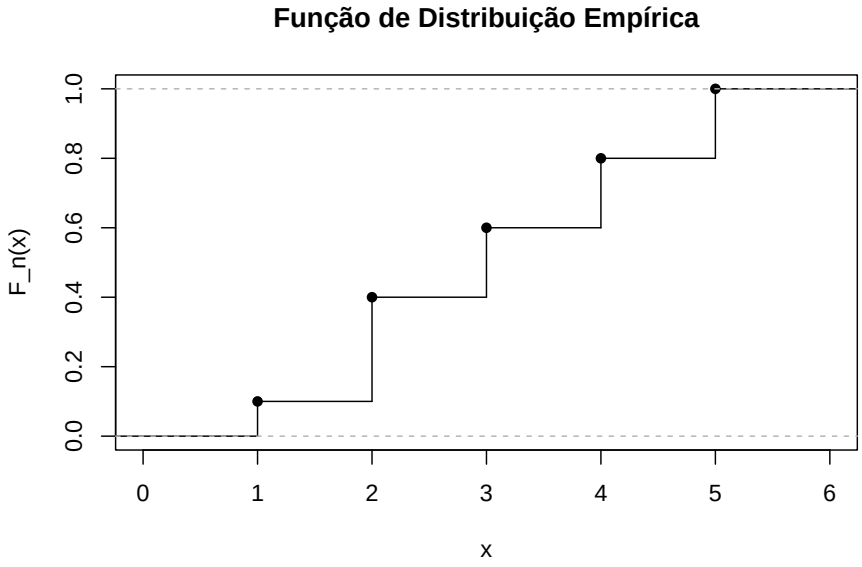
Suponha a amostra **{3,5,2,4,3,2,5,1,4,2}**, para calcular a sua função empírica vamos primeiro reordenar os valores: **{1,2,2,2,3,3,4,4,5,5}**

Calculando a *FDE*:

x	FDE
1	0.1
2	0.4
3	0.6
4	0.8
5	1.0

Representação gráfica

```
amostra <- c(3, 5, 2, 4, 3, 2, 5, 1, 4, 2)

# Criar a função de distribuição empírica
fde <- ecdf(amostra)

# Plotar a FDE
plot(fde, verticals = TRUE, do.points = TRUE, pch = 16,
     main = "Função de Distribuição Empírica",
     xlab = "x", ylab = "F_n(x)",
     col = "black")
```



6.9 Exercícios

1. Quinze pacientes de uma clínica de ortopedia foram entrevistados quanto ao *número de meses previstos de fisioterapia*, se há *expectativa de sequelas* (S) ou não (N) após o tratamento e o *graus de complexidade da cirurgia* realizada: alto (A), médio (M), ou baixo (B). Os dados estão apresentados na tabela a seguir.

(a) Classifique cada uma das variáveis.
(b) Para cada variável, construa a tabela de frequência e faça uma representação gráfica.
(c) Para o grupo de pacientes que não ficaram com sequelas, faça um gráfico de barras para a variável Fisioterapia. Você acha que essa variável se comporta de modo diferente nesse grupo quando comparado com a amostra total?

Obs.: Para o item (b) e (c) faça à mão e depois repita o exercício no R.

2. Usando o banco de dados do R *mtcars*, analise as características dos carros com motor de 6 cilindros versus os de motor de 8 cilindros. Para isso,

Paciente	Fisioterapia (em meses)	Sequelas	Cirurgia
1	7	S	A
2	8	S	M
3	5	N	A
4	6	N	M
5	4	N	M
6	5	S	B
7	7	S	A
8	7	N	M
9	6	N	B
10	8	S	M
11	6	S	B
12	5	N	B
13	5	S	M
14	4	N	M
15	5	N	A

calcule a média e o desvio padrão da potência do motor (hp) para os carros onde a variável vs (motor V/S) é igual a 0 (motor de 6 cilindros) e igual a 1 (motor de 8 cilindros). A partir disso, crie um histograma.

3. Usando os dados e os calculos feitos no *exercício 2* do *capítulo 5*, construa a mão um boxplot.

4. Considere a amostra **{3, 7, 4, 2, 7}**

a) Organize os dados em ordem crescente.

b) Construa a tabela da FDE $\widetilde{F}(x)$, para todos os valores de x da amostra.

c) Faça o gráfico da *FDE* calculada acima.

Chapter 7

Medidas de duas variáveis

Considere que o interesse agora é estudar a relação entre variáveis. Se não há associação entre duas ou mais variáveis, dizemos que elas são independentes.

Exemplo: suponha que deseja-se estudar se a cor da roupa (1: clara, 0: escura) está associado com gostar de basquete (1: sim, 0: não). Perguntei para 10 alunos da turma e os dados estão apresentados abaixo

```
dados_cap7 <- tibble(Camisa=c(1,1,1,1,1,0,0,0,0,0),
                     Basquete=c(1,0,1,0,1,0,0,0,1,0))
```

7.1 Tabela de Contingência (de Frequências)

- Para variáveis categóricas, vamos primeiramente considerar tabelas de frequências.

```
# R base
tab1 <- table(dados_cap7$Camisa,dados_cap7$Basquete)
tab1
```

```
##
##     0 1
##   0 4 1
##   1 2 3
```

- Lembre-se que podemos pensam em independência com relação à distribuição conjunta ou à distribuição condicional.

	Basquete		
	0	1	**Total**
Camisa			
0	4 (40%)	1 (10%)	5 (50%)
1	2 (20%)	3 (30%)	5 (50%)
Total	6 (60%)	4 (40%)	10 (100%)

	Basquete		
	0	1	**Total**
Camisa			
0	4 (80%)	1 (20%)	5 (100%)
1	2 (40%)	3 (60%)	5 (100%)
Total	6 (60%)	4 (40%)	10 (100%)

- Se o objetivo é estudar a distribuição conjunta, podemos considerar as frequências relativas ao tamanho total da amostra observada.

```
# Tidyverse
require(gtsummary)
dados_cap7 %>% tbl_cross(Camisa,Basquete,percent = "cell") %>%
  bold_labels()
```

- Se o objetivo é estudar a distribuição condicional, podemos considerar as frequências relativas ao total das linhas ou das colunas. Pelo desenho de nosso estudo, eu fixei o total de cada cor da camisa, então a tabela abaixo é construída com relação ao total das linhas.

```
# Tidyverse com porcentagens das linhas
dados_cap7 %>% tbl_cross(Camisa,Basquete,percent = "row") %>%
  bold_labels()
```

7.2 Qui-Quadrado de Pearson

- Sejam o_{ij} as frequências observadas na i-ésima linha e j-ésima coluna da tabela, $o_{i.}$ o total observado na linha i e o_j o total observado na coluna j.

	Basquete		
	0	1	**Total**
Camisa			
0	4 (40%)	1 (10%)	5 (50%)
1	2 (20%)	3 (30%)	5 (50%)
Total	6 (60%)	4 (40%)	10 (100%)

Sob a hipótese de independência, espera que o valor observado em cada casela da tabela seja $e_{ij} = \frac{o_{i.}\ o_{.j}}{n}$.

- A estatística de Qui-Quadrado é dada por:

$$Q^2 = \sum_i \sum_j \frac{(o_{ij} - e_{ij})^2}{e_{ij}} \ .$$

- No exemplo:

```
chi2 = chisq.test(tab1)$statistic
chi2
```

```
## X-squared
## 0.4166667
```

7.3 Medidas de Associação baseadas no Qui-Quadrado

```
require(gtsummary)
dados_cap7 %>% tbl_cross(Camisa, Basquete, percent = "cell") %>%
  bold_labels()
```

7.3.1 Coeficiente de Contingência de Pearson

$$C = \sqrt{\frac{Q^2}{Q^2 + n}} \qquad 0 \leq C \leq 1$$

No exemplo:

$$C = \sqrt{\frac{\frac{10}{6}}{\frac{10}{6} + 10}} = 0.791$$

O Coeficiente de Contingência de Pearson é muito influenciado pelo número de linhas (l) e número de colunas (c).

7.3.2 Coeficiente de Tcschupov

$$T = \sqrt{\frac{Q^2/n}{(l-1)(c-1)}} \qquad 0 \leq T \leq 1$$

No exemplo:

$$T = \sqrt{\frac{\frac{1.66}{10}}{(2-1)(2-1)}} = 0.41$$

7.4 Outras Medidas de Associação

- Considere um contexto em que deseja-se avalia a presença de um desfecho (ter um determinado cancer, gostar de basquete, etc) na presença de um fator de risco (fumar, usar roupa clara).

```
require(kableExtra)
tibble('Fator de Risco' = c("Não", "Sim"),
       'Sem o Desfecho' = c("(1-q)", "(1-p)"),
       'Com o Desfecho' = c("q", "p")) %>%
  kbl(align = 'c', format = "html", booktabs = TRUE)%>%
  kable_styling(
     bootstrap_options = c("striped","hover", "bordered", "condensed"),
     latex_options = c("striped"))
```

Fator de Risco

Sem o Desfecho

Com o Desfecho

Não

(1-q)

q

Sim

(1-p)

p

7.4.1 Risco Atribuível

- $RA = p - q$: é a diferença entre as probabilidades de ter a doença dado a presença do fator de risco e de ter a doença sem fator de risco.

- No exemplo:

$$RA = \frac{3}{5} - \frac{1}{5} = \frac{2}{5}$$

7.4.2 Risco Relativo

- $RR = p/q$: é quantas vezes é mais provavél ter a doença tendo o fator de risco em relação a quem não tem.

- No exemplo:

$$RR = \frac{3/5}{1/5} = 3$$

7.4.3 Razão de Chances ("Odds Ratio")

- Os termos probabilidade e chance são sinônimos mas, por convenção, usaremos a notação $3:2$ " = " $\frac{3}{5}$, sendo que $3:2 = \frac{3/5}{2/5}$ denotará a chance e $\frac{3}{5}$ a probabilidade.

$$OR \ = \ \frac{p}{(1-p)} \div \frac{q}{(1-q)} \ = \ \frac{p(1-q)}{q(1-q)}$$

- No exemplo:

$$OR \ = \ \frac{3/5}{2/5} \div \frac{1/5}{4/5} \ = \ \frac{3}{2} \div \frac{1}{4} \ = \ \frac{3}{2} \cdot \frac{4}{1} = 6$$

	Doente		
	0	1	**Total**
Teste			
0	4 (40%)	1 (10%)	5 (50%)
1	2 (20%)	3 (30%)	5 (50%)
Total	6 (60%)	4 (40%)	10 (100%)

7.5 Medidas para Testes de Diagnóstico

Considere um teste para uma determinada doença, de modo que o resultado do teste pode ser 1: *Positivo* e 0: *Negativo* e o indivíduos podem estar 1: *Doente* ou 0: *Não Doente*.

```
require(gtsummary)

dados <- tibble(Teste = c(1,1,1,1,1,0,0,0,0,0),
                Doente = c(1,0,1,0,1,0,0,0,1,0))

dados %>% tbl_cross(Teste, Doente, percent = "cell") %>%
  bold_labels()
```

As medidas a seguir são bastante utilizadas no contexto de testes de diagnósticos:

7.5.1 Sensibilidade

$S = P(Teste = 1 \mid Doente = 1)$
* estimativa: $s = \frac{o_{22}}{o_{\bullet 2}}$
* no exemplo: $s = \frac{3}{4}$

7.5.2 Especificidade

$E = P(Teste = 0 \mid Doente = 0)$
- estimativa: $e = \frac{o_{11}}{o_{\bullet 1}}$
- no exemplo: $e = \frac{4}{6}$

7.5.3 Falso Positivo

$FP = P(Teste = 1 \mid Doente = 0)$
- estimativa: $fp = \dfrac{o_{21}}{o_{\bullet 1}}$
- no exemplo: $fp = \dfrac{2}{6}$

7.5.4 Falso Negativo

$FN = P(Teste = 0 \mid Doente = 1)$
- estimativa: $fn = \dfrac{o_{12}}{o_{\bullet 2}}$
- no exemplo: $fn = \dfrac{1}{4}$

7.5.5 Valor Preditivo Positivo

$VPP = P(Doente = 1 \mid Teste = 1)$
- estimativa: $vpp = \dfrac{o_{22}}{o_{2\bullet}}$
- no exemplo: $vpp = \dfrac{3}{5}$

7.5.6 Valor Preditivo Negativo

$VPN = P(Doente = 0 \mid Teste = 0)$
- estimativa: $vpn = \dfrac{o_{11}}{o_{1\bullet}}$
- no exemplo: $vpn = \dfrac{4}{5}$

7.5.7 Acurácia

$AC = P\left[(\text{Teste} = 0, \text{Doente} = 0) \cup (\text{Teste} = 1, \text{Doente} = 1)\right]$ - estimativa: $ac = \dfrac{o_{11} + o_{22}}{n}$
- no exemplo: $ac = \dfrac{3+4}{10} = \dfrac{7}{10}$

7.6 Correlação amostral

- Podemos estimar a $E[xY]$ como $\frac{1}{n}\sum_{i=1}^{n} x_i\ y_i$.
- Assim, a $COV(X,Y)$ pode ser estimada por $cov = \frac{1}{n}\sum_{i=1}^{n} x_i\ y_i - \bar{x}\ \bar{y}$.
- Analogamente, a correção amostral é

$$cor = \frac{cov}{\sqrt{var(x)var(y)}} = \frac{\sum_{i=1}^{n} x_i\ y_i - n\ \bar{x}\ \bar{y}}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{j=1}^{n}(y_j-\bar{y})^2}}\ .$$

- No exemplo:

```
cor(dados_cap7$Camisa, dados_cap7$Basquete)
```

```
## [1] 0.4082483
```

7.7 Exercícios

1. Um novo teste está sendo desenvolvido para a identificação do HIV. Das 200 pessoas estudadas, metade tem HIV e a outra metade não tem. O teste deu positivo para 75 pessoas e negativo para 125, sendo 25 falsos-positivos e 50 falsos-negativos.

(a) Construa uma tabela de dupla entrada com as informações do enunciado
(b) Encontre as medidas de sensibilidade e especificidade do teste.
(c) Calcule os valores preditivos positivo e negativo.
(d) Qual é a acúracia do teste?

Chapter 8

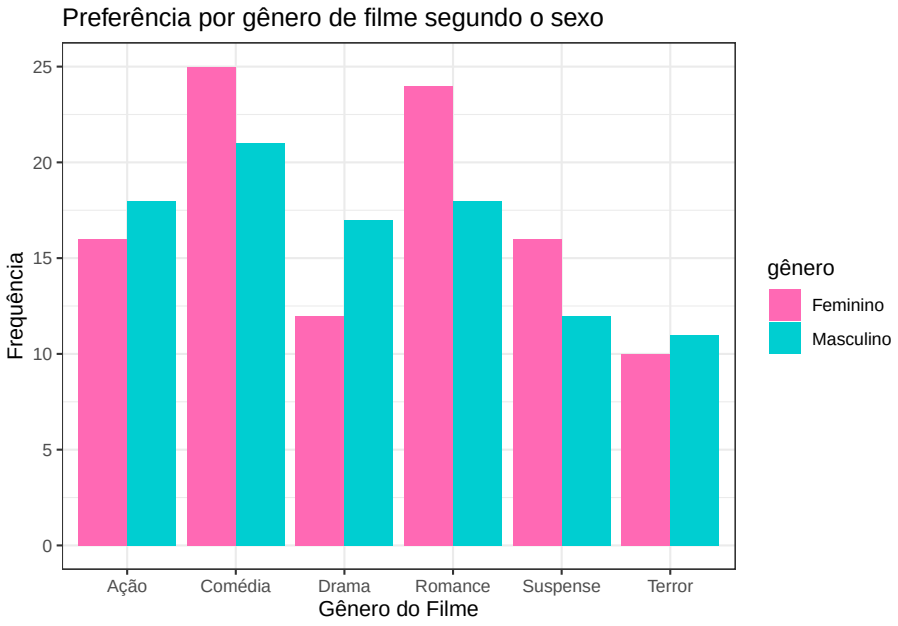
Modelos Gráficos de Associação entre Duas Variáveis

Além das medidas númericas que quantificam a associação entre duas variáveis, como as vistas no capitulo anterior, também é possível analisar essa relação de forma mais visual, através de gráficos.

8.1 Gráfico de barras

- É adequado quando ambas variáveis são categóricas (qualitativas nominais ou ordinais). Trás um ótimo comparativo entre as frequências absolutas.
- **Exemplo:** explorar a relação entre gênero e preferencia por um tipo de filme.

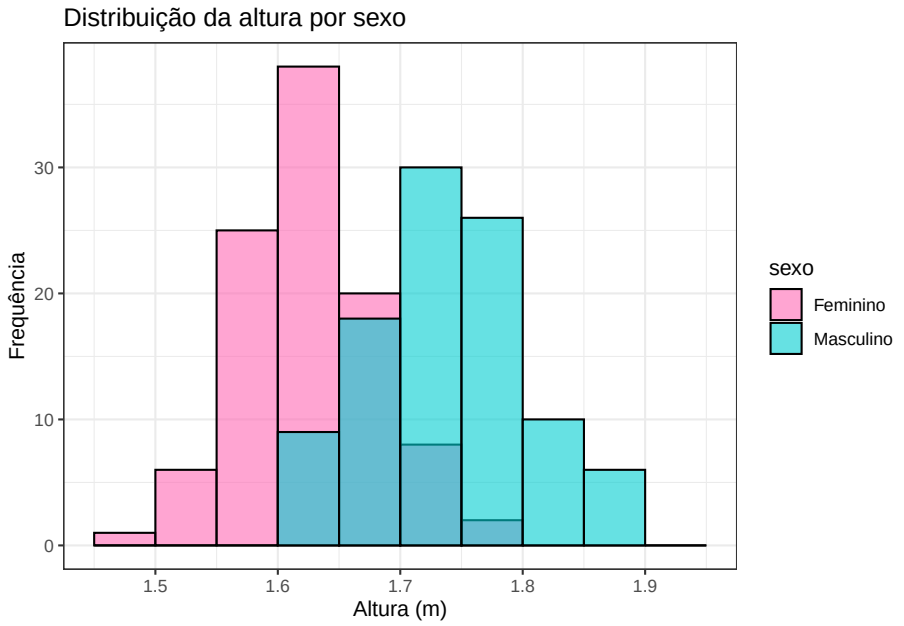
```
ggplot(dados1, aes(x = filme, fill = gênero)) +
  geom_bar(position = "dodge") +
  scale_fill_manual(values = c("Feminino" = "hotpink",
                               "Masculino" = "darkturquoise")) +
  labs(title = "Preferência por gênero de filme segundo o sexo",
       x = "Gênero do Filme", y = "Frequência") +
  theme_bw()
```



8.2 Histograma

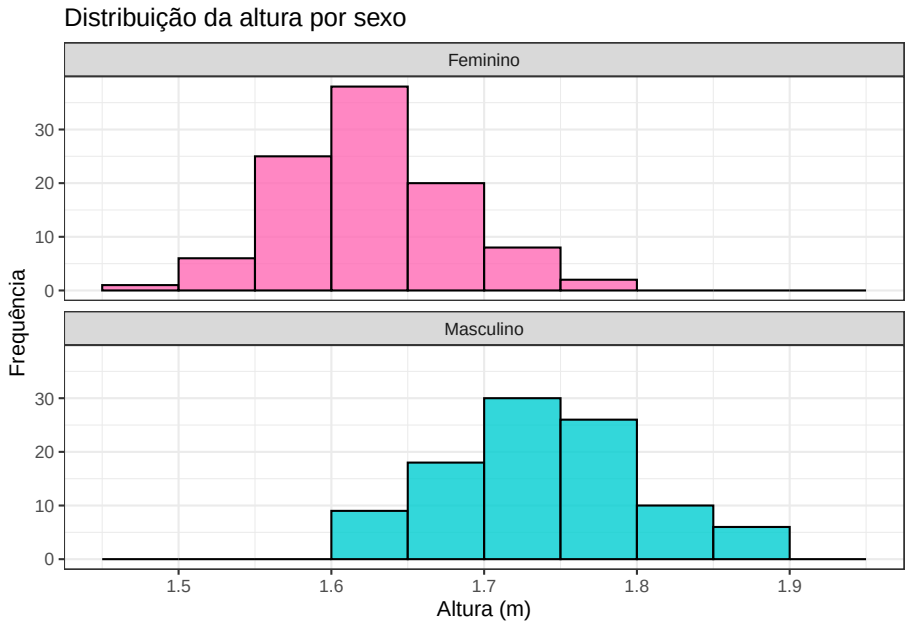
- Outra opção para comparar distribuições de uma variável quantitativa por grupo categórico são os histogramas sobrepostos ou lado a lado.
- Adequado para uma variável que é quantitativa e queremos observar a distribuição da variável contínua separada por grupos definidos por uma variável categórica.
- **Exemplo:** Relação de altura (quantitativo) entre de homens e mulheres (qualitativo).

```
# Definindo intervalos personalizados para o eixo x
intervalos <- seq(1.45, 1.95, by = 0.05)

ggplot(dados2, aes(x = altura, fill = sexo)) +
  geom_histogram(position = "identity", alpha = 0.6, color = "black",
                 breaks = intervalos) +
  scale_fill_manual(values = c("Feminino" = "hotpink",
                               "Masculino" = "darkturquoise")) +
  labs(title = "Distribuição da altura por sexo", x = "Altura (m)", y = "Frequência") +
  theme_bw()
```



Com o mesmo exemplo, mas sem ser sobreposto:

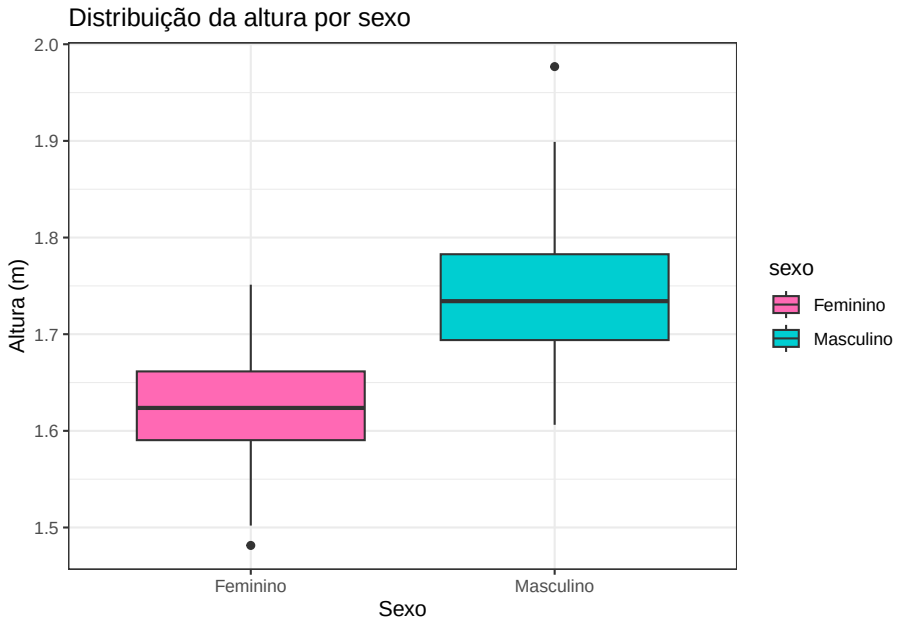
```
ggplot(dados2, aes(x = altura, fill = sexo)) +
  geom_histogram(alpha = 0.8, color = "black", breaks = intervalos) +
  scale_fill_manual(values = c("Feminino" = "hotpink",
                               "Masculino" = "darkturquoise")) +
  facet_wrap(~sexo, ncol = 1) +
  labs(title = "Distribuição da altura por sexo",
       x = "Altura (m)", y = "Frequência") +
  theme_bw() +
  theme(legend.position = "none")
```



8.3 Boxplot

- Útil quando se deseja comparar a distribuição de uma variável quantitativa entre diferentes grupos definidos por uma variável categórica.
- Ele mostra a mediana (linha central da caixa), o primeiro e terceiro quartis (bordas da caixa), valores mínimos e máximos (sem considerar outliers), outliers (pontos individuais fora dos limites).
- **Exemplo:** Relação de altura (quantitativo) entre de homens e mulheres (qualitativo).

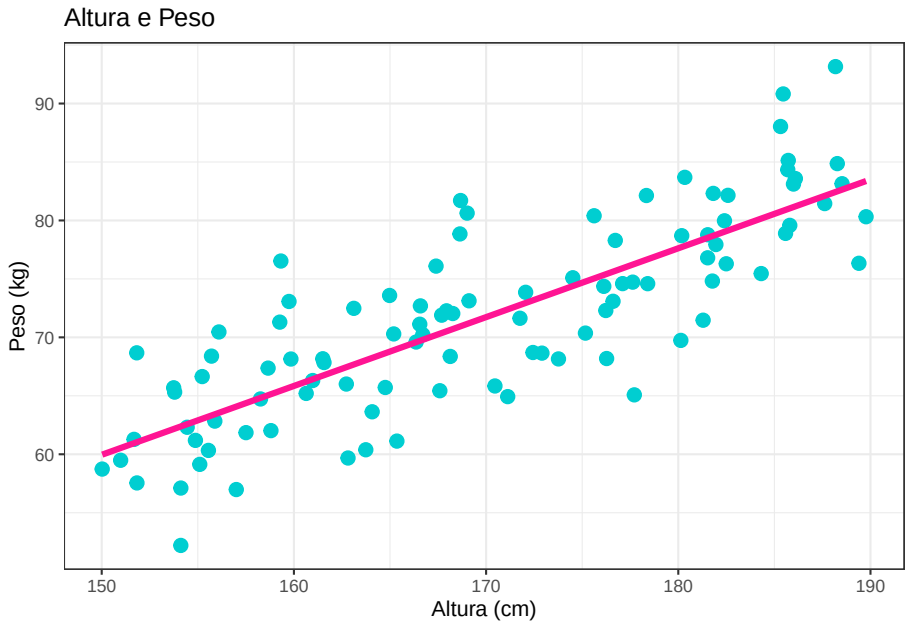
```
ggplot(dados3, aes(x = sexo, y = altura, fill = sexo)) +
  geom_boxplot() +
  scale_fill_manual(values = c("Feminino" = "hotpink",
                               "Masculino" = "darkturquoise")) +
  labs(title = "Distribuição da altura por sexo",
       x = "Sexo", y = "Altura (m)") +
  theme_bw()
```



8.4 Gráfico de dispersão

- É útil quando ambas variáveis são quantitativas.
- Esse gráfico permite identificar padrões de correlação linear ou não linear (que será explicado posteriormente), além de possíveis agrupamentos por categoria.
- **Exemplo 1:** Com relação linear clara entre altura e peso.

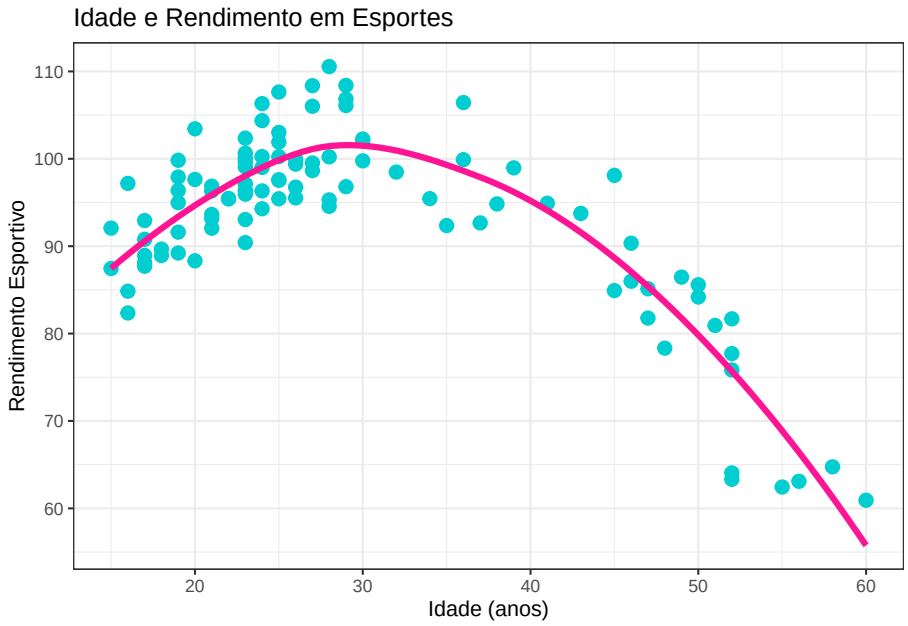
```
ggplot(dados_linear, aes(x = altura, y = peso)) +
  geom_point(color = "darkturquoise", size = 3) +
  geom_smooth(method = "lm", color = "deeppink", se = FALSE, size = 1.5) +
  labs(title = "Altura e Peso",
       x = "Altura (cm)",
       y = "Peso (kg)") +
  theme_bw()
```



O gráfico acima mostra uma relação de linear positiva entre a altura e o peso. Isso indica que à medida que a altura aumenta, o peso também tende a aumentar proporcionalmente, com isso, podemos concluir que a uma correlação forte e direta entre as duas variáveis.

- **Exemplo 2:** Com relação não linear, entre rendimento esportivo e idade.

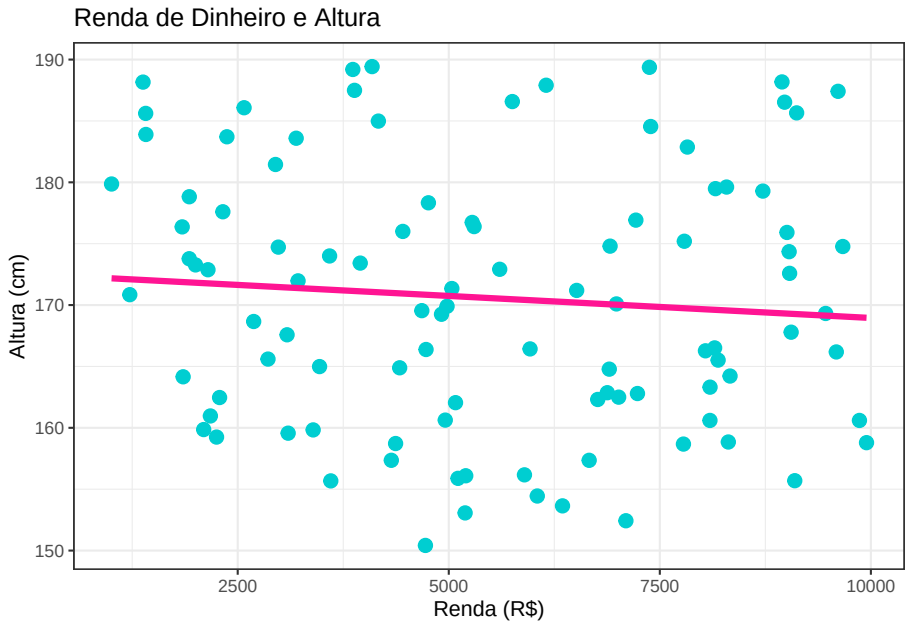
```
ggplot(dados_naolinear, aes(x = idade, y = rendimento)) +
  geom_point(color = "darkturquoise", size = 3) +
  geom_smooth(method = "loess", color = "deeppink", se = FALSE, size = 1.5) +
  labs(title = "Idade e Rendimento em Esportes",
       x = "Idade (anos)",
       y = "Rendimento Esportivo") +
  theme_bw()
```



O gráfico acima mostra uma relação não linear entre a idade e o rendimento esportivo. No início, o rendimento (eixo Y) tende a aumentar com a idade (eixo X), mas após um certo ponto (o pico), o rendimento começa a diminuir à medida que a idade avança.

- **Exemplo 3:** Sem relação linear, entre renda (em reais) e altura (em cm).

```
ggplot(dados_semrelacao, aes(x = renda, y = altura)) +
  geom_point(color = "darkturquoise", size = 3) +
  geom_smooth(method = "lm", color = "deeppink", se = FALSE, size = 1.5) +
  labs(title = "Renda de Dinheiro e Altura",
       x = "Renda (R$)",
       y = "Altura (cm)") +
  theme_bw()
```



O gráfico acima mostra a falta de relação entre renda e altura de uma pessoa. Podemos observar que os pontos estão dispersos aleatóriamente pelo gráfico, sem qualquer padrão aparente. A linha de tendência ajustada é praticamente horizontal, indicando que a altura de uma pessoa não é um bom preditor de sua renda, e vice-versa. Ou seja, não há uma correlação linear entre as duas variáveis.

8.5 Matriz de gráficos (ggally)

Quando lidamos com análises mais abrangenres, especialmente para casos que temos muitas variáveis e queremos visualizar todas as associações par a par, podemos usar o pacote *ggally* do R, que é uma extensão do *ggplot2*. O pacote cria matrizes de gráficos, combinando diferentes tipos (histogramas, boxplots, densidades, etc.) dentro de uma única matriz onde cada cécula da matriz representa uma visualização da associação entre duas variáveis.

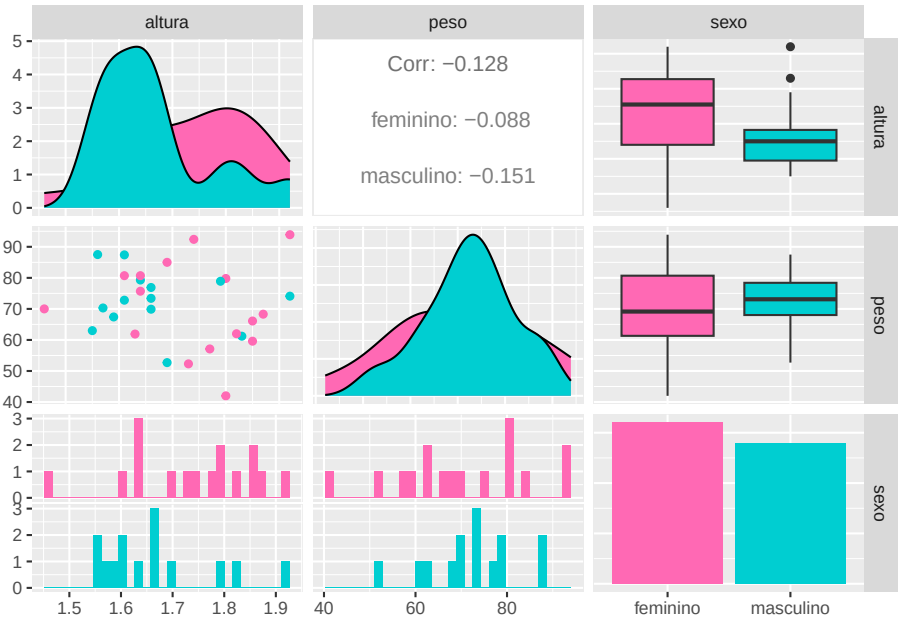
Função ggpairs():

A função *ggpairs()* é a principal do pacote. Ela gera uma matriz de gráficos de dispersão generalizada, onde cada célula da matriz é um gráfico que representa a relação entre duas variáveis. A matriz gerada por *ggpairs()* é dividida em três regiões principais:

- Diagonal: Mostra a distribuição de cada variável igualmente. Dependendo do tipo de variável a funçao apresenta diferentes tipos de gráficos.
- Triângulo inferior: Mostra as relações entre os pares de variáveis, através de graficos de dispersão, boxplot, graficos de frequência, entre outros, dependendo do tipo de variável.
- Triângulo superior: Mostra medidas de associação númerica, como as vistas no capítulo anterior.

O exemplo abaixo usa o banco de dados disponível para download **aqui**.

```
library(GGally)
library(readr)

ggpairs(dados4, aes(color = sexo))
```



Embora *ggpairs()* seja a principal, o *ggally* oferece outras funções para visualizações mais específicas, como *ggcorr()*, *ggsurv()*, etc.

8.6 Exercícios

1. Usando o banco de dados *mtcars* do R, crie um boxplot para entender a associação entre o número de cilindros de um carro (*cyl*) e sua eficiência

de combustível (*mpg*). Usa a vcariável cyl como eixo X e mpg como eixo Y. Interprete os resultados.

2. Usando o banco de dados *iris* do R, crie um histograma (sem ser sobreposto, como no segundo exemplo da seção de histograma desse capítulo), que associe a largura da sépala (*Sepal.Width*) para cada espécie (*Species*). Interprete os resultados. Dica: Use *facet_wrap(~ Species)* para criar um painel separado para cada espécie, permitindo uma comparação lado a lado.

Chapter 9

Simulação

A *simulação computacional* é uma ferramenta que permite o estudo de fenômenos complexos, que seriam difíceis ou inviáveis de analisar em contextos reais. Por meio dela, criamos modelos matemáticos que imitam o comportamento de sistemas verdadeiros.

Ou seja, em vez de observar o mundo real diretamente (o que pode ser caro, demorado ou impossível), geramos artificialmente observações que seguem uma determinada distribuição de probabilidade e que se comportam de forma semelhante aos dados reais. Dessa forma, podemos estudar suas propriedades, prever resultados ou testar hipóteses.

9.1 Lei dos Grandes Números

A *Lei dos Grandes Números* é a base teórica para a simulação ser algo tão confiável. Em termos simples, ela afirma que, à medida que o número de repetições de um experimento aleatório aumenta, a *média* dos resultados *observados* tende a se aproximar cada vez mais do *valor esperado* (média teórica) desse experimento. Ou seja, essa convergência garante que, com um número suficientemente grande de simulações, podemos calcular estimativas precisas para probabilidades e valores médios.

Por exemplo, a Lei dos Grandes Números nos diz que ao lançarmos um dado muitas vezes e calcularmos a média dos resultados, essa média se aproximará de 3,5 (o valor esperado de um dado uniforme discreta de 1 a 6).

De forma semelhante, podemos usar a simulação para observar o que acontece com uma moeda honesta. Se lançarmos 10 vezes, pode dar 7 caras e 3 coroas,

mas ao repetir o experimento 10 mil vezes, a proporção de caras de aproxima de 0,5.

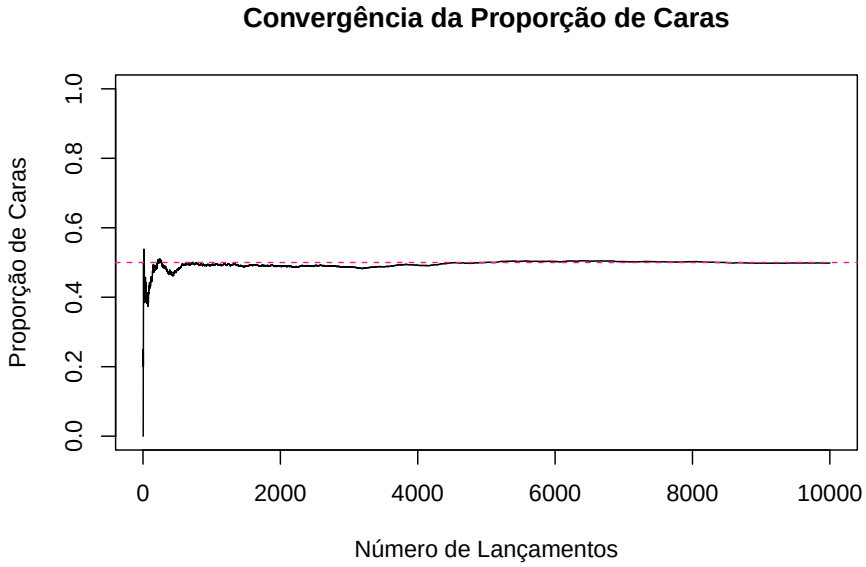
```r
set.seed(123)

# Número de lançamentos
n_lancamentos <- 10000

# Simular lançamentos de moeda (0 = coroa, 1 = cara)
resultados <- sample(c(0, 1), size = n_lancamentos, replace = TRUE)

# Calcular a proporção de caras
proporcao_caras <- sum(resultados) / n_lancamentos

cat("Número de lançamentos:", n_lancamentos, "\n")
```

```
## Número de lançamentos: 10000
```

```r
cat("Proporção de caras simulada:", proporcao_caras, "\n")
```

```
## Proporção de caras simulada: 0.4983
```

```r
# Para ver a convergência ao longo do tempo (Lei dos Grandes Números)
proporcoes_acumuladas <- cumsum(resultados) / (1:n_lancamentos)

plot(proporcoes_acumuladas, type = "l",
     main = "Convergência da Proporção de Caras",
     xlab = "Número de Lançamentos",
     ylab = "Proporção de Caras",
     ylim = c(0, 1))
abline(h = 0.5, col = "deeppink", lty = 2)
```



No código acima, primeiro definimos a semente do gerador de números aleatórios *set.seed(123)*, para garantir que os resultados da simulação sejam possíveis de reproduzir, ou seja, toda vez que o código for executado com essa seed, os mesmos números aleatórios serão gerados.

Depois, simulamos os lançamentos da moeda, com 10 mil lançamentos. A função *sample()* escolhe aleatoriamente entre os valores 0 (coroa) e 1 (cara), com reposição (*replace = TRUE*). Sendo assim, cada lançamento é independente dos anteriores. O *resultados* é um vetor com 10 mil resultados de zeros e uns.

Com esses dados, criamos um gráfico da proporção acumulada de caras ao longo dos lançamentos. E por fim, adicionamos uma linha horizontal pontilhada no valor de 0,5 (o valor esperado de caras de uma moeda justa). Dessa forma, conseguimos visualizar como a média de caras observadas se aproximam do valor esperado (0,5).

Observe que no início, com poucos lançamentos, a proporção de caras varia bastante, mas conforme o número de lançamentos aumenta, a linha da proporção acumulada se aproxima cada vez mais de 0,5, confirmando o que a Lei dos Grandes Números nos diz.

9.2 Método de Monte Carlo

O *Método de Monte Carlo* é uma técnica de simulação, baseada na amostragem aleatória para obter resultados numéricos. Esse método funciona gerando um grande número de amostras aleatórias de uma distribuição de probabilidade e, com isso, estimando uma quantidade de interesse.

Como exemplo, imagine que queremos estimar a área de uma forma irregular. Podemos fazer isso gerando pontos aleatórios dentro de um quadrado que a contém e contar a proporção de pontos que caem dentro da forma irregular.

Utilizando essa ideia, aplicaremos o método de Monte Carlo para estimar a área de um círculo, a ideia é "jogar" pontos aleatoriamente dentro de um quadrado conhecido e ver quantos desses pontos caem dentro do círculo inscrito nesse quadrado.

Pense em um círculo centrado na origem (ponto (0, 0)) e de raio 1. Ou seja, ele ocupa a região: $x^2 + y^2 \leq 1$. Esse círculo está inteiramente contido dentro de um quadrado que vai de [-1, 1] tanto no eixo x, quanto no eixo y, isto é, o quadrado tem lado 2.

Sabemos que a área real de um círculo é:

$$\text{Área real} = \pi \cdot r^2 = \pi \cdot 1^2 = \pi \approx 3,1416$$

Estimando isso pelo metódo de Monte Carlo, vamos gerar aleatoriamente pontos dentro do quadrado e contar quantos desses pontos caem dentro do círculo (satisfazem $x^2 + y^2 \leq 1$). A fração de pontos dentro do círculo multiplicada pela área do quadrado (que é 4) nos dá uma estimativa para π.

```r
# Define a semente para garantir reprodutibilidade dos resultados
set.seed(123)

# Definir o número de pontos
n_pontos <- 10000

# Gerar coordenadas X e Y aleatórias para os pontos
# dentro do quadrado [-1, 1] x [-1, 1]
x_coords <- runif(n_pontos, min = -1, max = 1)
y_coords <- runif(n_pontos, min = -1, max = 1)

# Calcular a distância ao quadrado até a origem para cada ponto
distancia_ao_quadrado <- x_coords^2 + y_coords^2

# Contar quantos pontos caem dentro do círculo de raio 1
pontos_dentro_circulo <- sum(distancia_ao_quadrado <= 1)
```

```r
# Calcular a proporção de pontos dentro do círculo
proporcao_no_circulo <- pontos_dentro_circulo / n_pontos

# Estimar a área do círculo (área do quadrado é 4)
area_circulo_estimada <- proporcao_no_circulo * 4

# Mostrar resultados
cat("Total de pontos jogados:", n_pontos, "\n")
```

```
## Total de pontos jogados: 10000
```

```r
cat("Pontos que caíram dentro do círculo:", pontos_dentro_circulo, "\n")
```

```
## Pontos que caíram dentro do círculo: 7894
```

```r
cat("Proporção de pontos dentro do círculo:", proporcao_no_circulo, "\n")
```

```
## Proporção de pontos dentro do círculo: 0.7894
```

```r
cat("Estimativa da Área do Círculo (ou Pi):", area_circulo_estimada, "\n")
```

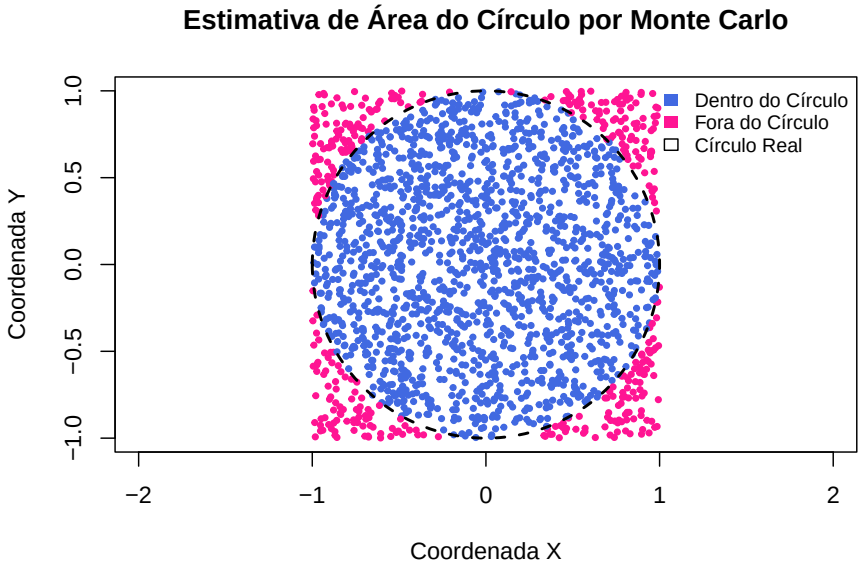
```
## Estimativa da Área do Círculo (ou Pi): 3.1576
```

```r
cat("Valor real de Pi (Área do Círculo de raio 1):", pi, "\n")
```

```
## Valor real de Pi (Área do Círculo de raio 1): 3.141593
```

No código acima, geramos dois vetores, com coordenadas x e y uniformemente distribuídas dentro do quadrado que vai de -1 a 1 nos dois eixos. Depois cointamos quantos pontos estão dentro do círculo (a soma dos quadrados das coordenadas seja menor ou igual a 1). E por fim, multiplicamos a proporção de pontos dentro do círculo pela área total do quadrado para obter a estimativa da área do círculo.

Vendo isso de forma visual:



Nesse gráfico, temos o círculo, com os pontos azuis (dentro) e vermelhos (fora), além do contorno do círculo verdadeiro para comparação visual.

9.3 Simulando no R

Vimos como a simulação computacional pode ser usada para estudar fenômenos aleatórios e visualizar resultados teóricos. Agora veremos como simular usando o R.

Primeiro, como já citamos anteriormente, sempre comece as suas simulações com *set.seed()*, isso garante que, se você ou outra pessoa executar o mesmo código com a *mesma semente*, os resultados aleatórios serão *idênticos*, tornando suas simulações reproduzíveis.

```
set.seed(123) # escolha qualquer número inteiro
```

9.3.1 Principais funções para simulação no R

O R possui um conjunto de funções para trabalhar com uma determinada distribuição de probabilidade (como a binomial, uniforme, poisson, etc.). Para

cada distribuição, existem geralmente quatro funções, prefixadas por letras que indicam sua finalidade.

- r (random): Geração de números aleatórios, a mais utilizada em simulações.
- d (density): função de massa de probabilidade, no caso discreto, ou seja, $P(X = x)$ e função densidade de probabilidade, no caso contínuo.
- p (probability): Função de distribuição acumulada, ou seja, $P(X \leq x)$.
- q (quantile): Função quantil, recebe um valor p e fornece o valor q que satisfaz $P(X \leq q) = p$.

Exemplos de funções random:

- *rbinom(n, size, prob)*: distribuição binomial, onde n é o número de simulações (dados gerados), *size* é a quantidade de ensaios (m da binomial(m, p)) e p a probabilidade de sucesso de cada ensaio. Note que, para *size=1* temos uma distribuição de Bernoulli.
- *sample(x, size, replace, prob)*: amostragem discreta aleatória, onde x é o conjunto de números de onde se quer sortear dados, *size* é a qunatidade de tentativas, *replace* é se tem repetição (=TRUE) ou não (=FALSE), ou seja, se tem reposição ou não e *prob* é um vetor opcional, que corresponde a probabilidade de sucesso de cada elemento do vetor x.

Para distribuições contínuas temos *rnorm()*, *runif()*, etc.

Exemplo prático, utilizando a distribuição Binomial

Sabemos que a distribuição binomial modela o número de sucessos em n tentativas independentes, com probabilidade de sucesso p.

Vamos pegar como exemplo o caso em que jogamos uma moeda honesta 10 vezes e contamos quantas vezes deu cara. Ou seja, temos uma binomial com $n = 10$ e $p = 0,5$. Repetiremos o experimento 100 vezes.

$$X \sim \text{Binomial}(n = 10, p = 0,5)$$

```
# Definindo parâmetros
n_tentativas <- 10
prob_sucesso <- 0.5
n_experimentos <- 100
```

Agora, geraremos os valores aleatórios usando *rbinom()*:

```r
# Simula 100 experimentos da binomial(n = 10, p = 0.5)
resultados <- rbinom(n_experimentos, size = n_tentativas, prob = prob_sucesso)
```

Para saber a probabilidade de termos exatamente x sucessos, usamos a *dbinom()*, no caso de $P(X = 5)$:

```r
# Probabilidade de sair exatamente 5 caras em 10 lançamentos
dbinom(5, size = 10, prob = 0.5)
```

```
## [1] 0.2460938
```

Para saber a probabilidade acumulada, usamos *pbinom()*, no caso de $P(X \leq 5)$:

```r
# Probabilidade de sair 5 ou menos caras em 10 lançamentos
pbinom(5, size = 10, prob = 0.5)
```

```
## [1] 0.6230469
```

Para saber o quantil, usando *qbinom()*, no caso de $P(X \leq x) = 0,8$

```r
# Qual o menor n de sucessos tal que a prob acumulada seja ao menos 0.8?
qbinom(0.8, size = 10, prob = 0.5)
```

```
## [1] 6
```

9.4 Teorema Central do Limite (TCL)

O Teorema Central do Limite afirma que a soma (ou média) de um grande número de variáveis aleatórias independentes e identicamente distribuidas (i.i.d), com média e variância finitas, tende a seguir uma distribuição normal, independente da distribuição original dessas variáveis.

Sendo assim, seja $X_1, X_2, \cdots, X_m$ uma sequência de variáveis aleatórias independentes e identicamente distribuídas (i.i.d.), com $E[X_1] = \mu$ e $Var(X_1) = \sigma^2 < +\infty$. Considere $\bar{X}_m = \frac{1}{m}\sum_{i=1}^{m} X_i$. Então,

$$\frac{\bar{X}_m - \mu}{\sigma/\sqrt{m}} \quad \xrightarrow[m\to\infty]{\mathcal{D}} \quad Normal(0,1) \ .$$

Usamos essa formula para a padronização da média amostral, ou seja, transformamos a variável $\bar{X}_m$ para que tenha média 0 e variância 1, permitindo compará-la com a normal padrão. E afirmamos que a distribuição da expressão à esquerda se aproxima da distribuição normal padrão à medida que m se aproxima de ∞.

Então, quando m é "grande", pode-se dizer que $\bar{X}_m$ tem distribuição aproximadamente $Normal\left(\mu, \frac{\sigma^2}{m}\right)$.

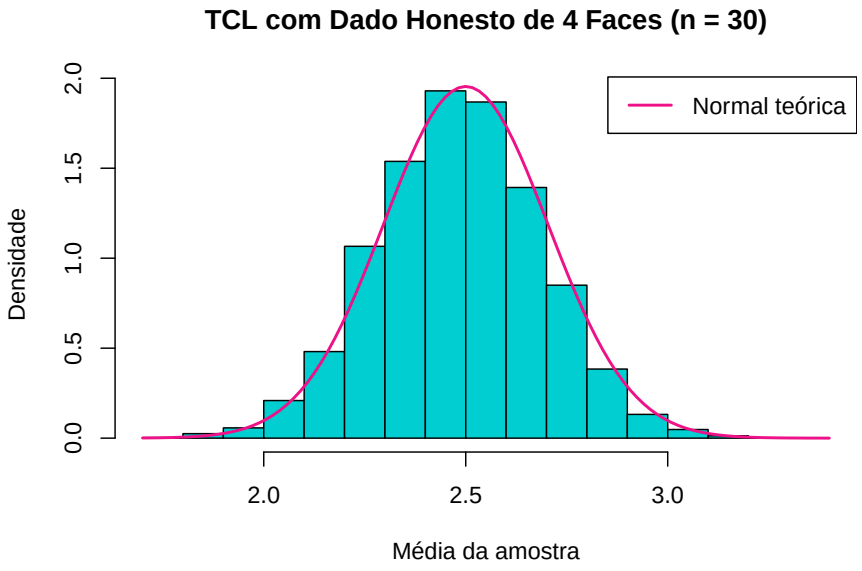
O TCL pode ser visualizado claramente por meio de simulações computacionais, a ideia é repetir um experimento aleatório várias vezes e observar o comportamento da média das amostras.

Para exemplificar, pense em um tetraedo (um dado de 4 faces) honesto, com valores de 1 a 4, com mesma probabilidade. A distribuição original é uniforme discreta, e não se parece com uma normal.

Vamos denotar X como o valor da face do dado, ou seja, $X \sim Unif\{1, 2, 3, 4\}$, com esperança $= \mu = 2,5$, variância $= \sigma^2 = 1,25$ e desvio padrão $\sigma = \sqrt{1,25}$.

Sortearemos variás amostras aleatórias de tamanho 30 ($n = 30$) e simularemos esse experimento 10.000 vezes. Depois, calcularemos a média de cada amostra e por fim faremos um histograma das médias, junto com a curva de distribuição normal esperada, usando a média e o desvio padrão teóricos.

```r
set.seed(123)

n <- 30      # tamanho da amostra
m <- 10000   # número de repetições

# Gerar N médias amostrais de amostras de tamanho n com valores de 1 a 4
medias <- replicate(m, mean(sample(1:4, n, replace = TRUE)))

# Parâmetros teóricos
media_teorica <- 2.5
desvio_padrao <- sqrt(1.25 / n)

hist(medias, freq = FALSE, col = "darkturquoise",
     main = "TCL com Dado Honesto de 4 Faces (n = 30)",
     xlab = "Média da amostra", ylab = "Densidade")

# Adicionando a curva normal teórica
curve(dnorm(x, mean = media_teorica, sd = desvio_padrao),
      col = "deeppink2", lwd = 2, add = TRUE)

legend("topright", legend = "Normal teórica", col = "deeppink2", lwd = 2)
```



O histograma acima mostra a distribuição das médias amostrais do nosso experimento simulado. A linha rosa (curva normal) mostra a aproximação prevista pelo Teorema Central do Limite. Perceba que mesmo com uma distribuição discreta e com poucos valores possíveis, a média se aproxima de uma distribuição normal, como afirma o TCL.

Apesar de ainda não termos visto a distribuição normal na graduação, saiba que ela é a mais importante em toda a estatística; um dos motivos para isso é o resultado que vimos acima do TCL.

Além dele, a normal é uma distribuição com dois parâmetros: sua média e variância. Ela é simétrica em torno da sua média e com duas caudas (iguais, por conta da simetria), muito conhecida e com boas propriedades. E isso tudo facilita a inferência (triar conclusões sobre os parâmetros) nela.

9.5 Reamostragem

Usado para estimar a distribuição amostral de estatísticas de interesse, a técnica de reamostragem nos ajuda a inferir sobre as propriedades de uma população a partir de uma única amostra observada.

Imagine que temos um conjuntos de dados e desejamos entender a variabilidade de alguma estatística calculada a partir dessa amostra, como a média ou variância. Para que não seja necessário coletar múltiplas amostras da população (o que nem sempre é viável), podemos usar a reamostragem.

A reamostragem consiste em gerar, a partir da nossa amostra original de tamanho m, um grande número M de novas amostras, também de tamanho m, com reposição. Com isso, conseguimos estimar a distribuição amostral de estatísticas de interesse, uma vez que, à medida que o tamanho da amostra original aumenta, a distribuição obtida por reamostragem se aproxima da distribuição teórica da estatística.

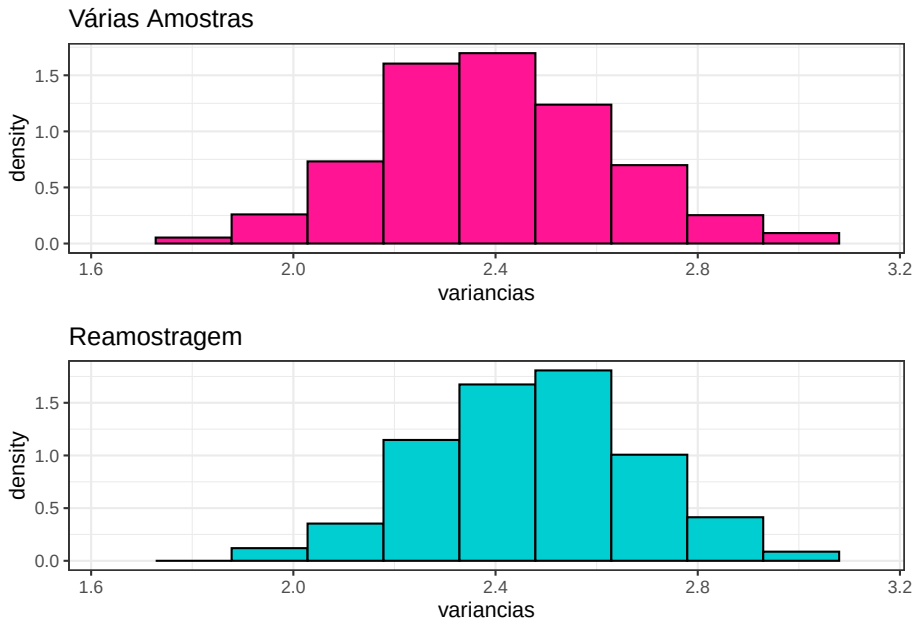
- **Exemplo:**

Pense que queremos entender como a variância de uma variável aleatória binomial se comporta em diferentes amostras, usando a reamostragem para produzir esse comportamento, mesmo com uma única amostra disponível.

Vamos analisar o número de sucessos em 10 tentativas de um experimento com probabilidade de sucesso $p = 0,6$. E o nosso interesse está em estudar como a variância do número de acertos se comporta em diferentes amostras. Para isso:

```
set.seed(666)

# Parâmetros da simulação
m = 200      # tamanho da amostra original
p = 0.6      # probabilidade de sucesso
n = 10       # número de tentativas em cada experimento
M = 1000     # número de repetições da simulação

# Simulando várias amostras independentes da distribuição binomial
S=matrix(rbinom(m*M,n,p),ncol=M)
variancias = apply(S,2,var)
b=c(min(variancias),max(variancias),nclass.Sturges(variancias))

hist1 <- tibble(variancias) %>% ggplot() + theme_bw() +
  ggtitle("Várias Amostras") +  xlim(b[1],b[2]) +
  geom_histogram(aes(x=variancias,y=..density..),
      bins=b[3],col="black",fill="deeppink")

# Fazendo reamostragem da primeira amostra
B <- apply(matrix(rep(1,M)),1,
           function(y){sample(S[,1],length(S[,1]),replace=TRUE)})
variancias = apply(B,2,var)

hist2 <- tibble(variancias) %>% ggplot() + theme_bw() +
```

```
  ggtitle("Reamostragem") + xlim(b[1],b[2]) +
  geom_histogram(aes(x=variancias,y=..density..),
      bins=b[3],col="black",fill="darkturquoise")

ggpubr::ggarrange(hist1,hist2,ncol=1)
```



Mesmo usando apenas uma única amostra, com o método de reamostragem conseguimos gerar uma distribuição da estatística (variância) muito próxima àquela obtida por várias amostras independentes.

9.6 Uma aplicação de simulação em inferência estatística

Uma das aplicações mais importantes da estatística é testar se duas variáveis são ou não independentes. Diferente da probabilidade, na inferência estatística, apenas os dados observados não são o suficiente pra afirmar independência. Além disso, variáveis aparentemente não correlacionadas têm correlação.

Por isso, tradicionalmente usamos testes formais, como o teste do Qui-quadrado (visto no capítulo 7) e comparamos o valor calculado com os valores críticos teóricos, obtidos da ditribuição Qui-quadrado.

No entanto, como ainda não vimos esse tipo de cálculo no curso, vamos simular esse comportamento do teste sob a hipotese de independência com um exemplo prático.

Imgine que você possui dois dados honestos, um de 4 faces e outro com 6 faces. Para cada rodada, você escolhe qual dado vai jogar (ambos com igual probabilidade) e anota o valor da face que saiu. Esse processo é repetido 23 vezes.

Esse experimento gera dois tipos de observações, que chamaremos de -X : o tipo de dado usado em cada jogada, $x = 4, 6$. -Y : o valor da face observada, $y = 1, 2, \cdots, x$.

A tabela a seguir mostra os resultados obtidos ao final das 23 jogadas, com o número de vezes que cada face apareceu, para cada tipo de dado:

X / Y	**Face 1**	**Face 2**	**Face 3**	**Face 4**	**Face 5**	**Face 6**	**Total**
Dado de 4 faces	2	3	3	4	0	0	12
Dado de 6 faces	2	1	2	3	2	1	11
Total	4	4	5	7	2	1	23

Após o experimento, queremos responder à seguinte pergunta: O valor da face (Y) depende do tipo de dado escolhido (X)? Ou seja, X e Y são variáveis independentes? Se X e Y forem independentes, a face sorteada não influência o tipo de dado que foi lançado.

A maneira clássica de testar isso é usando o teste Qui-quadrado, que compara os valores observados com os valores esperados sob suposição de independência.

Faremos uma tabela com os valores esperados, que representa a conjunta de X, Y, caso eles fossem independêntes. Calculamos esses valores esperados usando as proporções totais de cada linha e coluna. Ou seja, o valor esperado da linha i e coluna j é:

$$e_{ij} = \frac{(\text{soma da linha i}) \cdot (\text{soma da linha i})}{n}$$

Onde n é o total de valores observados (23).

Por exemplo, para o valor observado na primeira linha ($x = 4$) e primeira coluna ($Y = 1$), temos:

$$e_{11} = \frac{(\text{soma da linha 1}) \cdot (\text{soma da linha 1})}{n} = \frac{12 \cdot 4}{23} \approx 2,087$$

Agora, conseguimos calcular o Qui-quadrado,

$$Q^2 = \sum_i \sum_j \frac{(o_{ij} - e_{ij})^2}{e_{ij}}$$

X / Y	Face 1	Face 2	Face 3	Face 4	Face 5	Face 6	Total
Dado de 4 faces	2.09	2.09	2.61	3.65	1.04	0.52	12
Dado de 6 faces	1.91	1.91	2.39	3.35	0.96	0.48	11
Total	4.00	4.00	5.00	7.00	2.00	1.00	23

Onde -o_{ij} é o valor observado na linha i e coluna j. -e_{ij} é o valor esperado na linha i e coluna j.

Calculando pelo R, com:

```
qui_obs <- sum((observado - esperado)^2 / esperado)
qui_obs
```

```
## [1] 4.279942
```

Temos então que o Qui-quadrado observado (Q^2_{obs}) é aproximadamente 4,28. Precisamos descobrir se esse valor representa variáveis independentes ou não.

Em vez de apenas comparar o Qui-quadrado observado, vamos construir nossa própria "distribuição" de Qui-quadrados, simulando amostras sob a hipótese de independência e e verificando se o valor observado é compatível com o que esperaríamos por acaso.

Vamos gerar 100 amostras de 23 pares (X_i, Y_i), com $X_i \sim U\{4, 6\}$ e $Y_i \sim U\{1, 2, \cdots, X_i\}$, sendo U a distribuição uniforme discreta.

Para cada uma dessas 100 simulações, calculamos um Qui-quadrado. Ao final, teremos 100 valores de Qui-quadrado esperados sob independência e ordenaremos todos, caso o nosso Qui-quadrado *observado* for muito alto (maior que o Qui-quadrado simulado de posição 95), teremos fortes indícios de dependência.

```
set.seed(123)

simular_qui <- function(n_sim = 100, n_obs = 23) {
  qui_sim <- numeric(n_sim)

  for (i in 1:n_sim) {
    # Gerando X ~ unif{4, 6}
    x <- sample(c(4, 6), size = n_obs, replace = TRUE)

    # Para cada xi, gerar y ~ unif{1, ..., xi}
    y <- mapply(function(xi) sample(1:xi, 1), xi = x)

    # Tabela com valores simulados
    tabela_sim <- table(factor(x, levels = c(4, 6)), factor(y, levels = 1:6))
```

```
    # Totais marginais e esperados (para o cálculo de valor esperado)
    linha_totais <- rowSums(tabela_sim)
    coluna_totais <- colSums(tabela_sim)
    total <- sum(tabela_sim)

    esperado_sim <- outer(linha_totais, coluna_totais, FUN = function(r, c) r * c / total)

    # Calcular Q² apenas nas células com esperado > 0
    valido <- esperado_sim > 0
    Q2 <- sum((tabela_sim[valido] - esperado_sim[valido])^2 / esperado_sim[valido])

    qui_sim[i] <- Q2}

  return(qui_sim)}

qui_simulados <- simular_qui()
```

No código acima criamos uma função *simular_qui()*, com 100 simulações e 23 observações para cada simulação. Criamos também um vetor vazio *qui_sim* com $n\_sim = 100$ posições, para armazenarmos nossos qui-quadrados.

Em seguida, iniciamos um looping com 100 iterações. Para cada simulação, geramos um vetor x com $n\_obs = 23$ valores escolhidos aleatoriamente no conjunto $\{4, 6\}$ (representando os dois tipos de dado). A escolha é feita com reposição, e a distribuição é uniforme (probabilidade 0,5 para cada elemento). Depois, para cada valor de x_i, geramos outro vetor y, que sorteia uma face aleatoriamente de 1 até x_i.

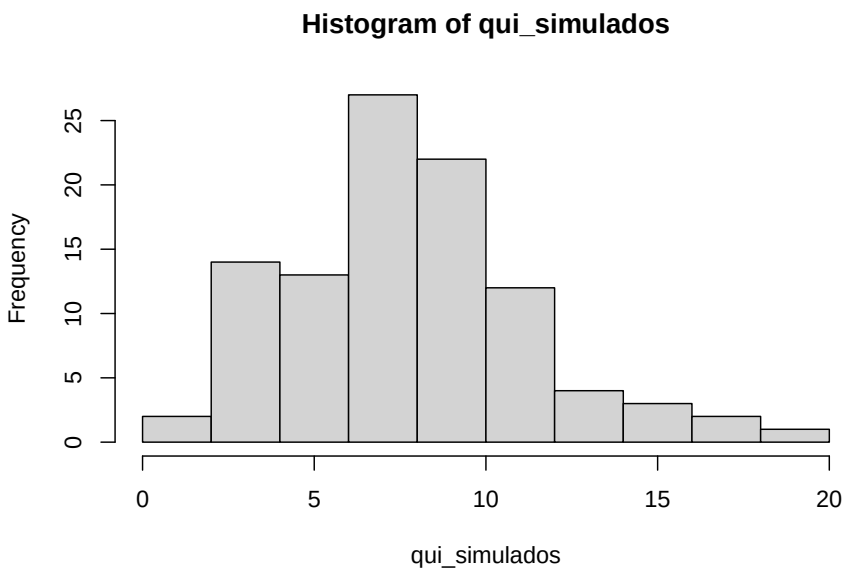
Criamos uma tabela de contingência X/Y, com os nossos valores simulados e geramos uma matriz *esperado_sim* com cada valor esperado e_{ij}. Com isso, aplicamos a fórmula do teste Qui-quadrado apenas nas células com valor esperado > 0 (evita divisões por zero).

Logo depois, armazenamos o valor do teste da simulação atual no vetor *qui_sim* e retornamos esse vetor. Por fim, chamamos a função *simular_qui()* para o novo vetor *qui_simulados*, que conterá 100 valores de Qui-quadrado simulados sob a hipótese de independência.

```
quantil_95 <- quantile(qui_simulados, 0.95)
quantil_95
```

```
##      95%
## 14.60973
```

Com todos os 100 Qui-quadrados simulados, calculamos o quantil 95% da distribuição empírica. É comumente usado o quantil 95% nesse tipo de análise, por isso, ele foi o escolhido. Vendo essa distribuição simulada, de forma visual:



Finalmente, comparando:

```
qui_obs > quantil_95
```

```
##   95%
## FALSE
```

A estatística Q^2 pode ser vista como uma distância entre os valores observados e esperados, sob hipótese de independencia. Então, para valores grandes de Q^2, tem-se mais indícios de independencia nos dados observados. Logo, comparamos o valor Q^2_{obs} com o quantil 95% da distribuição empírica ($q_{(95)}$): se o $Q^2_{obs} \leq q_{(95)}$, então, temos indícios de independência; caso contrário, não temos.

Nesse sentido, anteriormente, calculamos o Qui-quadrado dos nossos valores observados, tendo $Q^2_{obs} = 4,28$. Comparando esse valor com a distribuição empírica de Qui-quadrados gerada a partir de 100 simulações sob hipótese de independência entre as variáveis X e Y, ou então, comparando com o valor $q_{(95)} = 14,6$ gerado, vemos que $Q^2_{obs} q_{(95)}$.

Sendo assim, como nosso Qui-quadrado observado é *menor* do que o quantil obtido, através das simulações concluímos que os dados não fornecem evidência contra a hipótese de independência. Ou seja, valores de Qui-quadrado maiores que 14,6 seriam considerados incompatíveis com a independência.

9.7 Exercícios

Obs. Caso queira comparar seu resultado com o gabarito (capítulo 10) use *set.seed(123)*.

1. Simule o lançamento de um dado honesto (com faces de 1 a 6), um grande número de vezes. A cada novo lançamento, calcule a média dos valores obtidos. Em seguida, faça um gráfico que mostre como essa média se aproxima do valor esperado.

(a) Simule 10 mil lançamentos de um dado honesto.
(b) Calcule a média acumulada depois de cada lançamento simulado.
(c) Faça um gráfico com a média acumulada em função do número de lançamentos. Por fim, adicione uma linha horizontal indicando o valor esperado de um dado uniforme.

2. Caso um aluno chute aleatoriamente as respostas de uma prova com 10 questões, cada uma com 4 alternativas. Estime, por simulação, a probabilidade dele acertar pelo menos 4 questões.

(a) Simule esse experimento 10.000 vezes. E em cada simulação, gere 10 respostas com probabilidade de acerto = 0,25.
(b) Conte em quantas simulações o número de acertos foi maior ou igual a 4 e estime essa probabilidade.
(c) calcule o resultado teórico usando a função *pbinom()*.

3. Suponha que você tenha dois dados justos de seis faces. Qual é a probabilidade de que a soma dos números nos dois dados seja maior que 8 ou igual a 5?

(a) Simule 10 mil lançamentos. Em cada simulação gere dois números aleatórios inteiros entre 1 e 6.
(b) Calcule a soma de cada par de lançamentos e conte quantas vezes a soma é maior que 8 ou igual a 5.
(c) Calcule a probabilidade estimada.

Chapter 10

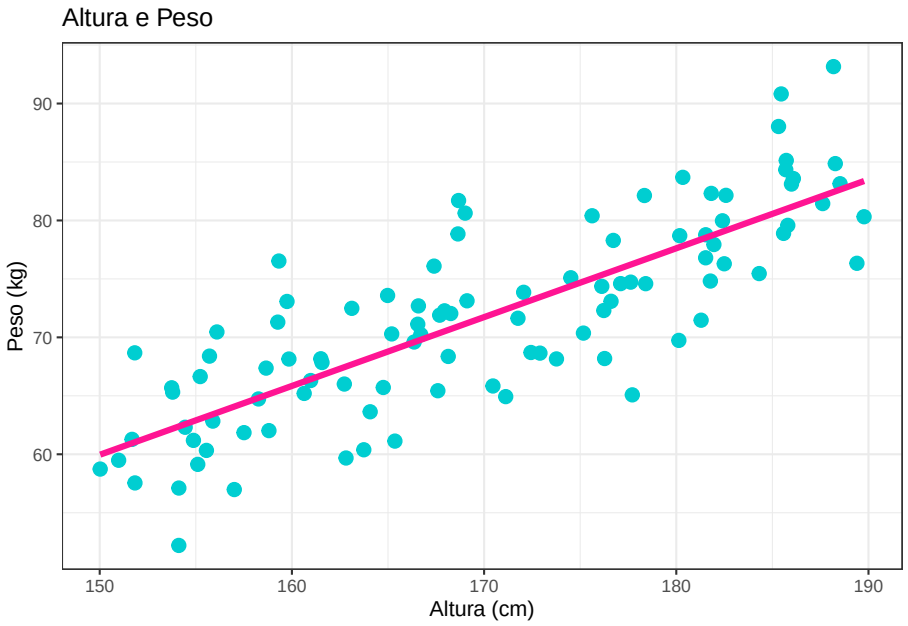
Regressão linear

A regressão linear modela a relação entre duas variáveis quantitativas, onde uma variável é considerada dependente e a outra é independente. O objetivo principal é estabelecer uma equação linear que descreva como a variável dependente Y se comporta em função da variável independente X.

A equação da regressão linear simples é $Y = a + bX + \varepsilon$.

Onde: - Y é a variável resposta (dependente) - X é a variável explicativa (independente) - a é o intercepto (valor de Y quando $X = 0$) - b é o coeficiente angular (indica a variação de Y para cada unidade de X) - εé um erro aleatório.

Suponha que temos interesse em estudar o valor esperado do peso de uma pessoa (Y, em kg), com base em sua altura (X, em cm). Para isso, coletamos dados de várias pessoas e ajustamos um modelo de regressão linear.



Temos então, que a esperança de Y, dado um valor X é,

$$E[Y \mid X = x] = a + bx$$

Agora, para conseguir estimar esse valor esperado do peso de uma pessoa, com base em sua altura, precisamos obter o valor a e b.

10.1 Estimar a e b

Para estimar os coeficientes usamos o método dos mínimos quadrados:

$$S = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - (a + bx_i))^2 = \sum_{i=1}^{n} (y_i - a - bx_i)^2$$

$$\begin{cases} \frac{\partial S(a,b)}{\partial a} = 0 \\ \frac{\partial S(a,b)}{\partial b} = 0 \end{cases} \implies \begin{cases} \frac{\partial S(a,b)}{\partial a} = -\sum_{i=1}^{n} 2(y_i - a - bx_i) = 0 \\ \frac{\partial S(a,b)}{\partial b} = -\sum_{i=1}^{n} x_i \cdot 2(y_i - a - bx_i) = 0 \end{cases}$$

Isolando a:

$$\implies -\sum_{i=1}^{n} y_i + na + b\sum_{i=1}^{n} x_i = 0 \implies a = \frac{\sum_{i=1}^{n} y_i}{n} - b\frac{\sum_{i=1}^{n} x_i}{n}$$

$$\implies a = \bar{y} - b\bar{x}$$

Isolando b:

$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Como os valores de a e b calculados acima são estimados, chamaremos de $\hat{a}$ e $\hat{b}$.

Para o nosso exemplo de altura e peso, vamos estimar a e b usando a função *ln()*:

```
regressao <- lm(dados_linear)
regressao$coefficients
```

```
## (Intercept)        peso
##   89.623253    1.120262
```

Com isso, temos que $\hat{a} \approx 89,62$ e $\hat{b} \approx 1,12$.

10.2 Resíduos

Em regressão linear, os resíduos são as diferenças entre os valores observados e os valores estimados pela reta de regressão. Ou seja:

$$\varepsilon = y - \hat{y} = y - (a + bx)$$

Obtendo os resíduos do exemplo de altura e peso:

```
residuos <- residuals(regressao)
```

Quando fazemos uma análise dos resíduos, nosso objetivo é verificar se os resíduos (diferenças entre valores observados e valores previstos pelo modelo) se aproximam de uma distribuição normal. Temos duas formas mais comuns de fazer isso, sendo através de um histograma ou de um QQ-plot.

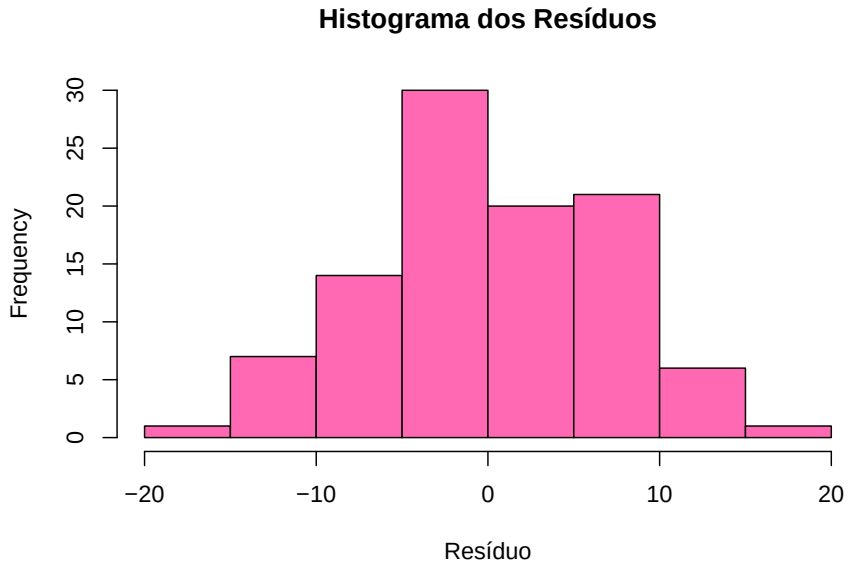
10.2.1 Analisando resíduos através de um histograma

O histograma mostra a distribuição dos resíduos. Com ele, podemos verificar se os resíduos têm uma distribuição aproximadamente simétrica com um formato próximo da distribuição normal.

Se o histograma mostrar uma distribuição assimétrica, com caudas muito longas ou picos incomuns, pode indicar que os resíduos não são normais.

Fazendo o histograma com o exemplo de altura e peso:

```
hist(residuos,
     main = "Histograma dos Resíduos",
     xlab = "Resíduo",
     col = "hotpink",
     border = "black")
```



A forma do histograma acima se assemelha ao de uma distribuição normal, é unimodal (tem apenas um pico) e é razoavelmente simétrica. Então, embora não seja uma curva perfeita (o que é raro em dados reais), a distribuição não apresenta uma assimetria severa ou múltiplos picos. Sendo assim, nosso modelo de regressão linear parece ser adequado.

10.2.2 Analisando resíduos através de um QQ-plot

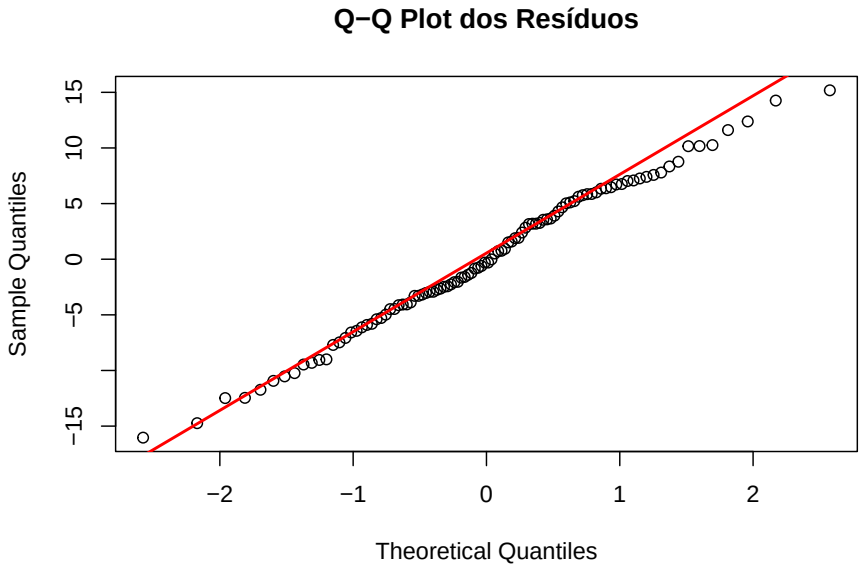
O QQ-plot compara os quantis dos resíduos com os quantis teóricos de uma distribuição normal. Se os pontos do gráfico ficarem aproximadamente numa linha reta, significa que os resíduos seguem bem a distribuição normal. Se os pontos se afastam da linha (curvando para cima ou para baixo, ou formando um "S"), indica desvio da normalidade.

Neste gráfico, ao invés das probabilidades acumuladas da Normal, são plotados os resíduos e os quantis teóricos $(x_{(i)}\ ,\ Q(p_i))$, em que $x_{(i)}$ são os valores observados ordenados, $Q(p)$ é o quantil teórico de ordem p e as probabilidades p_i usualmente são calculadas como:

$$p_i = \frac{i - 0,5}{n} \quad \text{ou} \quad p_i = \frac{i}{n-1}$$

Fazendo um QQ-plot com o exemplo de altura e peso:

```
qqnorm(residuos, main = "Q-Q Plot dos Resíduos")
qqline(residuos, col = "red", lwd = 2)
```



No gráfico acima, a linha vermelha representa a situação ideal, onde os seus resíduos seriam perfeitamente normais. Os pontos pretos (círculos) são os seus dados de resíduos.

O fato de os pontos estarem muito próximos da linha vermelha ao longo de quase toda a sua extensão é um excelente sinal. Isso indica que a distribuição dos seus resíduos se alinha de forma muito consistente com uma distribuição normal.

Chapter 11

Respostas dos exercícios

11.1 Capítulo 3

- **Exercício 1**

```
vetor1 <- c(10, 9, 8, 7, 6, 5, 4, 3, 2, 1)
vetor2 <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

soma <- vetor1 + vetor2
subtracao <- vetor1 - vetor2
multiplicacao <- vetor1 * vetor2
```

- **Exercício 2**

```
matriz1 <- matrix(1:4, nrow = 2, ncol = 2)
matriz2 <- matrix(5:8, nrow = 2, ncol = 2)

soma <- matriz1 + matriz2
subtracao <- matriz1 - matriz2
multiplicacao <- matriz1 %*% matriz2
```

- **Exercício 3**

```r
alunos <- data.frame(
  Nome = c("Letícia", "Mariana", "Ana", "Otávio", "Ricardo"),
  Idade = c(17, 18, 16, 17, 19),
  Nota = c(8.5, 6.2, 4.3, 2.0, 5.5)
)

alunos$Aprovado <- alunos$Nota >= 6
print(alunos)
```

```
##      Nome Idade Nota Aprovado
## 1 Letícia    17  8.5     TRUE
## 2 Mariana    18  6.2     TRUE
## 3     Ana    16  4.3    FALSE
## 4  Otávio    17  2.0    FALSE
## 5 Ricardo    19  5.5    FALSE
```

- **Exercício 4**

```r
exponencial <- function(M, b=exp(1)) {
  lc <- dim(M) # vetor com número de linhas e colunas de M
  E <- M # inicializa a matriz E que será retornada pela função
  i <- 1 # Inicializa i que irá percorrer as linhas
  while(i<=lc[1]){
    j <- 1 # inicializa j que irá percorrer as colunas
    while(j<=lc[2]){
      E[i,j] <- b^M[i,j] # calcula os elementos da matriz E
      j <- j + 1 # atualiza j
    }
    i <- i + 1 # atualiza i
  }
  return(E)
}

# Testa a função 'exponencial'
M
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    0    1    4
## [3,]    0    0    1
```

```r
exponencial(M,2)
```

```
##      [,1] [,2] [,3]
## [1,]    2    4    8
## [2,]    1    2   16
## [3,]    1    1    2
```

```
exponencial(M)
```

```
##          [,1]     [,2]      [,3]
## [1,] 2.718282 7.389056 20.085537
## [2,] 1.000000 2.718282 54.598150
## [3,] 1.000000 1.000000  2.718282
```

- **Exercício 5**

```
calcula_area <- function(base, altura) {
  return((base * altura) / 2)
}

# Testa a função 'calcula_area'
calcula_area(15, 3)
```

```
## [1] 22.5
```

- **Exercício 6**

```
conta_pos_neg <- function(vetor) {
  pos <- sum(vetor > 0)
  neg <- sum(vetor < 0)
  return(list(positivos = pos, negativos = neg))
}

# Testa a função 'conta_pos_neg'
vetor <- c(-10, 7, 4, -8, -15, 3, -5, 7, 0, 1, -2)
resultado <- conta_pos_neg(vetor)
print(resultado)
```

```
## $positivos
## [1] 5
##
## $negativos
## [1] 5
```

11.2 Capítulo 4

- **Exercício 1**

```
library(tidyverse) # Carregue o pacote

mtcars %>%
  filter(cyl == 6) %>% # Item a
  select(mpg, hp, wt) %>% # Item b
  arrange(desc(mpg)) # Item c
```

```
##                 mpg  hp    wt
## Hornet 4 Drive 21.4 110 3.215
## Mazda RX4      21.0 110 2.620
## Mazda RX4 Wag  21.0 110 2.875
## Ferrari Dino   19.7 175 2.770
## Merc 280       19.2 123 3.440
## Valiant        18.1 105 3.460
## Merc 280C      17.8 123 3.440
```

- **Exercício 2**

```
library(stringr)

nomes <- c("Mariana Silva", "Mateus Souza", "Letícia Dias", "Guilherme Almeida",
           "Yasmin Santos")

# Separando o nome e sobrenome
nomes_sobrenomes <- str_split(nomes, " ")

print(nomes_sobrenomes)
```

```
## [[1]]
## [1] "Mariana" "Silva"
##
## [[2]]
## [1] "Mateus" "Souza"
##
## [[3]]
## [1] "Letícia" "Dias"
##
## [[4]]
## [1] "Guilherme" "Almeida"
##
```

```
## [[5]]
## [1] "Yasmin" "Santos"
```

- **Exercício 3:**

```
library(forcats)

# Criando o fator
cores <- factor(c("rosa", "vermelho", "azul", "amarelo", "rosa", "verde", "azul", "rosa"))

# (a) Reordenar os níveis para que a cor mais frequente venha primeiro
cores_reordenadas <- fct_infreq(cores)
levels(cores_reordenadas)
```

```
## [1] "rosa"     "azul"     "amarelo"  "verde"    "vermelho"
```

```
# (b) Agrupar todas as cores menos frequentes que "azul" em "Outro"
cores_agrupadas <- fct_lump(cores, n = 2, other_level = "Outro")
table(cores_agrupadas)
```

```
## cores_agrupadas
##  azul  rosa Outro
##     2     3     3
```

11.3 Capítulo 5

- **Exercício 1:**

Dada uma amostra de tamanho n, $x_1, \ldots, x_n$, queremos mostrar que

$$\sum_{i=1}^{n}(x_i - \bar{x}) = 0$$

Onde $\bar{x}$ é a *média amostral*, dada por

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Primeiro, vamos expandir a soma

$$\sum_{i=1}^{n}(x_i - \bar{x}) = \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \bar{x}$$

Como $\bar{x}$ é uma constante, ou seja, não depende de i, podemos reescrever:

$$\sum_{i=1}^{n} \bar{x} = n\bar{x}$$

Mas pela definição de média amostral:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \Rightarrow n\bar{x} = \frac{1}{n}\sum_{i=1}^{n} nx_i = \sum_{i=1}^{n} x_i$$

Logo,

$$\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} x_i = 0 \blacksquare$$

- **Exercício 2**
- Média:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\bar{x} = \frac{(1 \times 3) + (2 \times 11) + (3 \times 16) + (4 \times 9) + (5 \times 6) + (6 \times 1) + (7 \times 2) + (8 \times 1) + (15 \times 1)}{50}$$

$$\bar{x} = \frac{182}{50} = 3,64$$

- Moda: 3
- Mediana:

$$\frac{x_{25} + x_{26}}{2} = \frac{3+3}{2} = 3$$

- Quartis $q(0,25) = x_{13} = 2$ e $q(0,75) = x_{38} = 4$

11.4 Capítulo 6

- **Exercício 1:**

Item (a)

- Fisioterapia: variável qualitativa discreta.
- Sequelas: variável qualitativa nominal.
- Cirurgia: variável qualitativa ordinal.

Item (b)

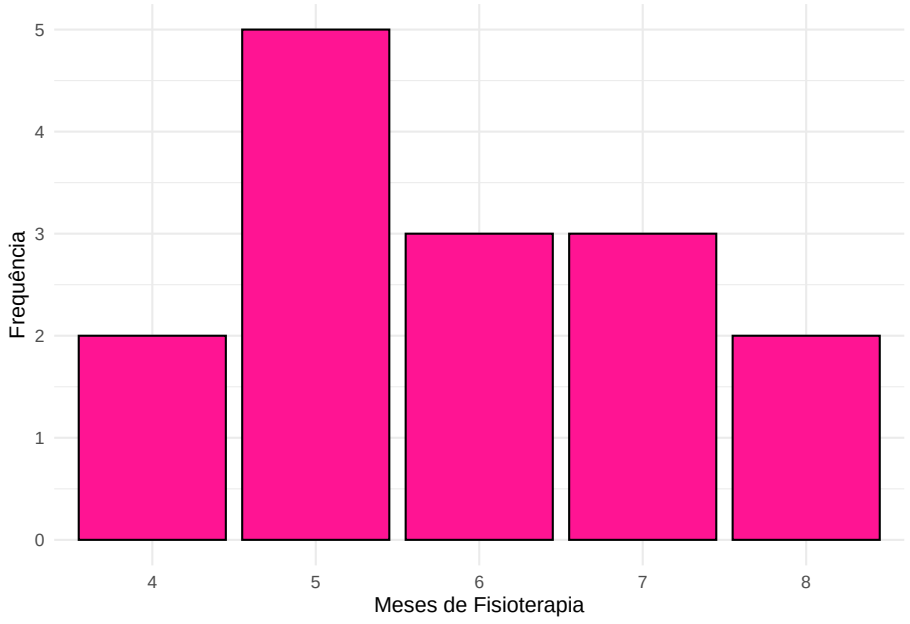
```
tab_fisio <- read.csv("tab_fisio.csv")
```

- Para *Fisioterapia (em meses)*:

```
freq_fisio <- tab_fisio %>%
  count(`Fisioterapia (em meses)`) %>%
  mutate(
    FreqRel = round(n / sum(n), 2)
  ) %>%
  rename(Frequencia = n)

print(freq_fisio)
```

```
## # A tibble: 5 x 3
##   `Fisioterapia (em meses)` Frequencia FreqRel
##                       <dbl>      <int>   <dbl>
## 1                         4          2    0.13
## 2                         5          5    0.33
## 3                         6          3    0.2
## 4                         7          3    0.2
## 5                         8          2    0.13
```

```
ggplot(tab_fisio, aes(x = factor(`Fisioterapia (em meses)`))) +
  geom_bar(fill = "deeppink", color = "black") +
  labs(x = "Meses de Fisioterapia", y = "Frequência") +
  theme_minimal()
```

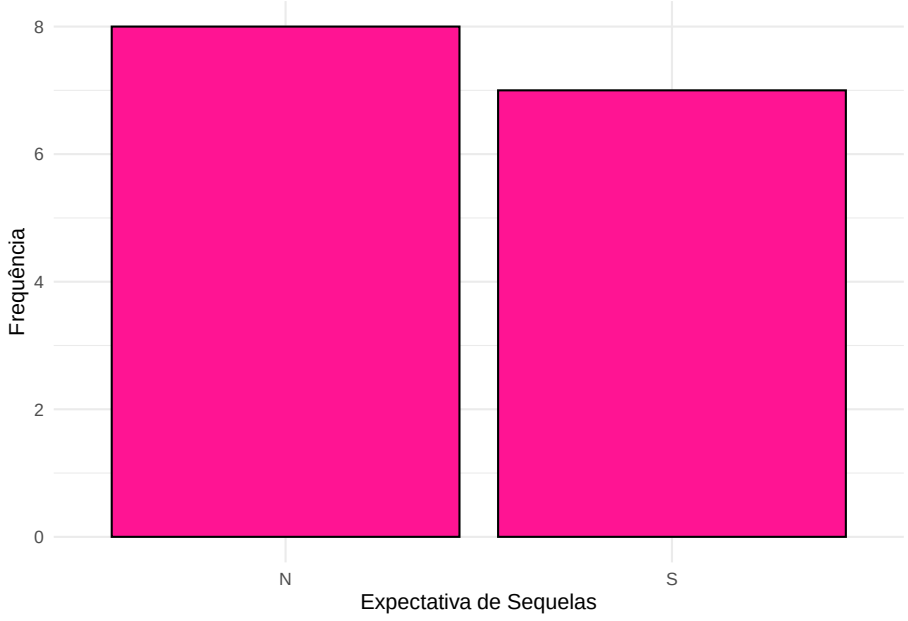


- Para *Sequêlas*:

```
freq_sequelas <- tab_fisio %>%
  count(Sequelas) %>%
  mutate(
    FreqRel = round(n / sum(n), 2)
  ) %>%
  rename(Frequencia = n)

print(freq_sequelas)
```

```
## # A tibble: 2 x 3
##   Sequelas Frequencia FreqRel
##   <chr>         <int>   <dbl>
## 1 N                 8    0.53
## 2 S                 7    0.47
```

```
ggplot(tab_fisio, aes(x = Sequelas)) +
  geom_bar(fill = "deeppink", color = "black") +
  labs(x = "Expectativa de Sequelas", y = "Frequência") +
  theme_minimal()
```

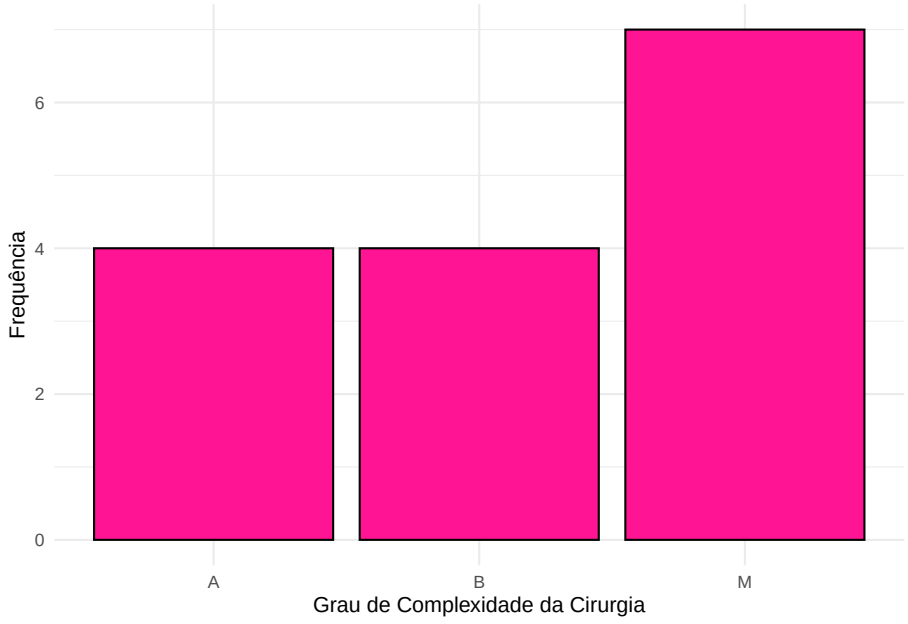


- Para *Cirurgia*:

```
freq_cirurgia <- tab_fisio %>%
  count(Cirurgia) %>%
  mutate(
    FreqRel = round(n / sum(n), 2)
  ) %>%
  rename(Frequencia = n)

print(freq_cirurgia)
```

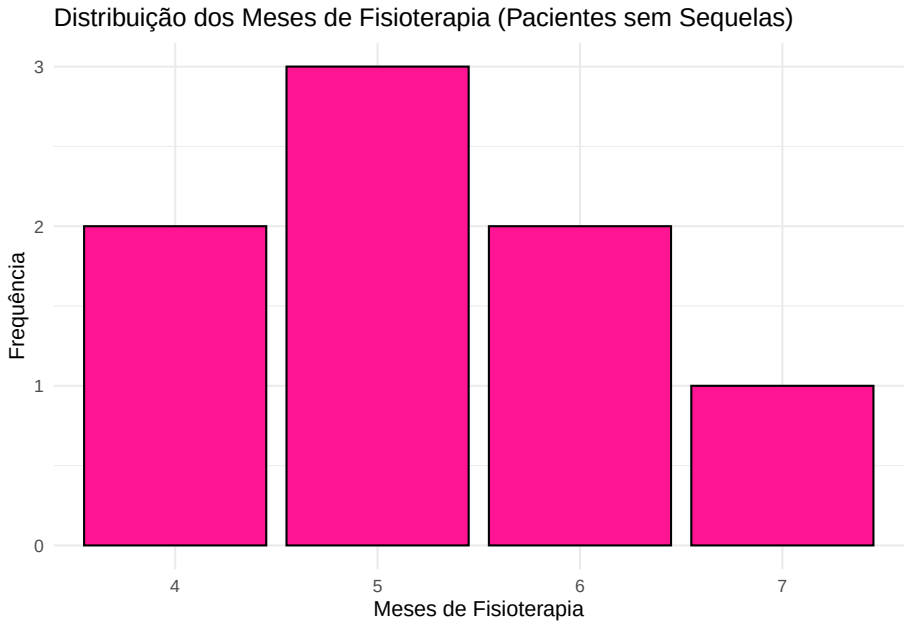
```
## # A tibble: 3 x 3
##   Cirurgia Frequencia FreqRel
##   <chr>         <int>   <dbl>
## 1 A                 4    0.27
## 2 B                 4    0.27
## 3 M                 7    0.47
```

```
ggplot(tab_fisio, aes(x = Cirurgia)) +
  geom_bar(fill = "deeppink", color = "black") +
  labs(x = "Grau de Complexidade da Cirurgia", y = "Frequência") +
  theme_minimal()
```



Item (c)

```
n_sequelas <- tab_fisio %>%
  filter(Sequelas == "N")

ggplot(n_sequelas, aes(x = factor(`Fisioterapia (em meses)`))) +
  geom_bar(fill = "deeppink", color = "black") +
  labs(
    title = "Distribuição dos Meses de Fisioterapia (Pacientes sem Sequelas)",
    x = "Meses de Fisioterapia",
    y = "Frequência"
  ) +
  theme_minimal()
```



- **Exercício 2:**

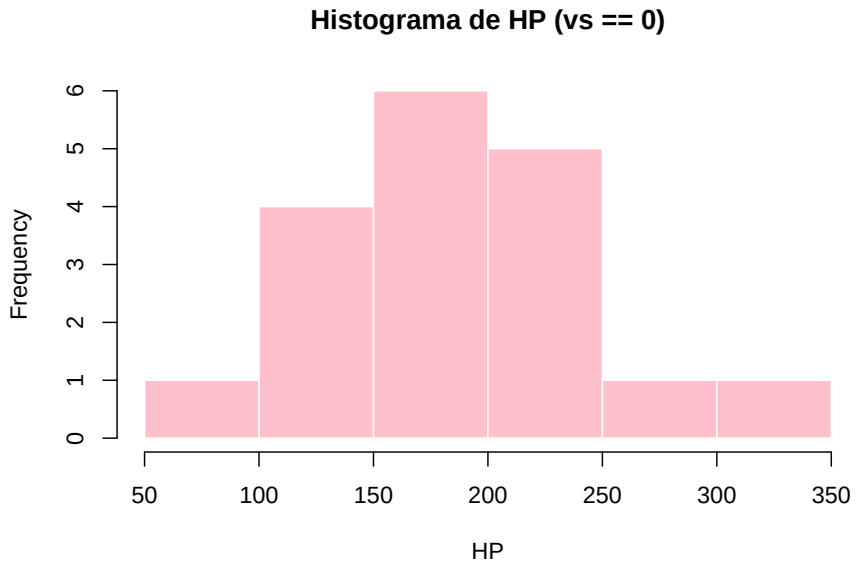
```
mtcars %>% filter(vs==0) %>% summarise(media = mean(hp), dp = sd(hp))
```

```
##      media      dp
## 1 189.7222 60.2815
```

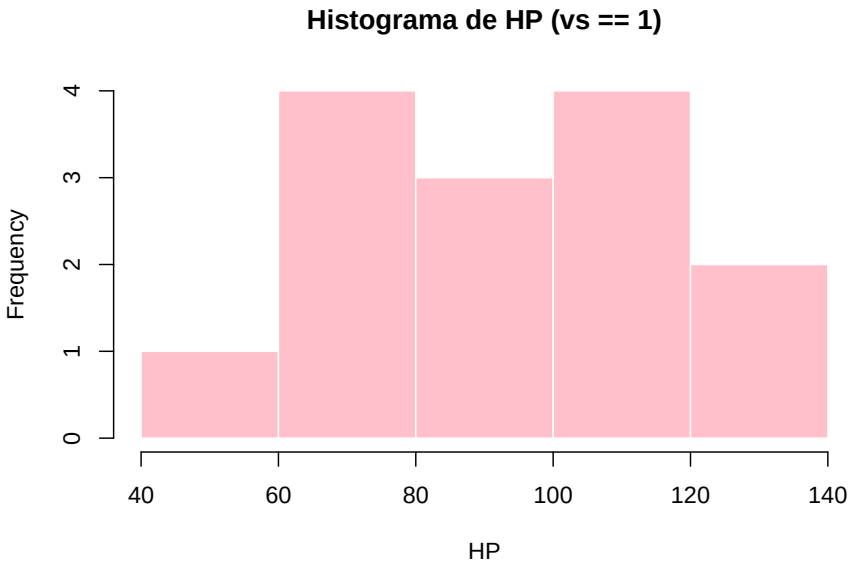
```
mtcars %>% filter(vs==1) %>% summarise(media = mean(hp), dp = sd(hp))
```

```
##      media       dp
## 1 91.35714 24.42447
```

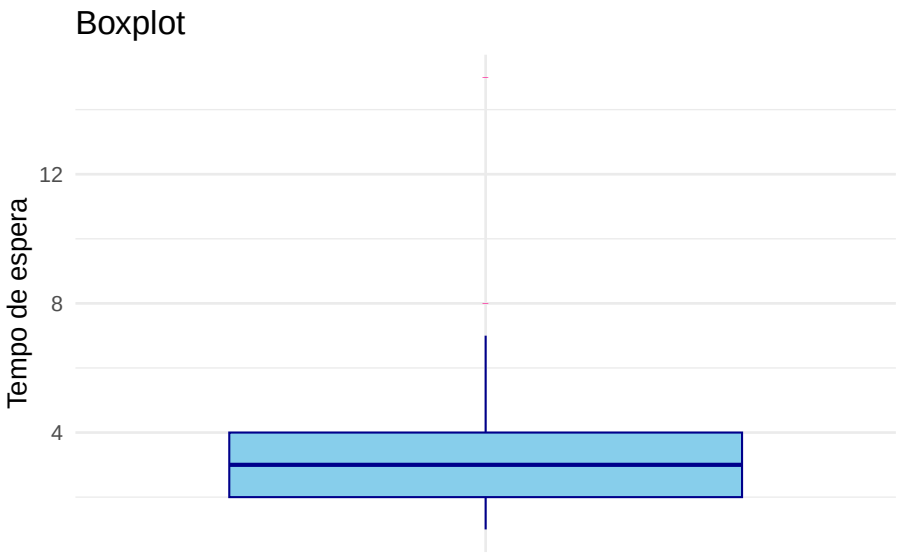
```
hist(
  mtcars$hp[mtcars$vs == 0],
  main = "Histograma de HP (vs == 0)",
  xlab = "HP",
  col = "pink",
  border = "white"
)
```




```
hist(
  mtcars$hp[mtcars$vs == 1],
  main = "Histograma de HP (vs == 1)",
  xlab = "HP",
  col = "pink",
  border = "white"
)
```



- **Exercício 3**



- **Exercício 4**

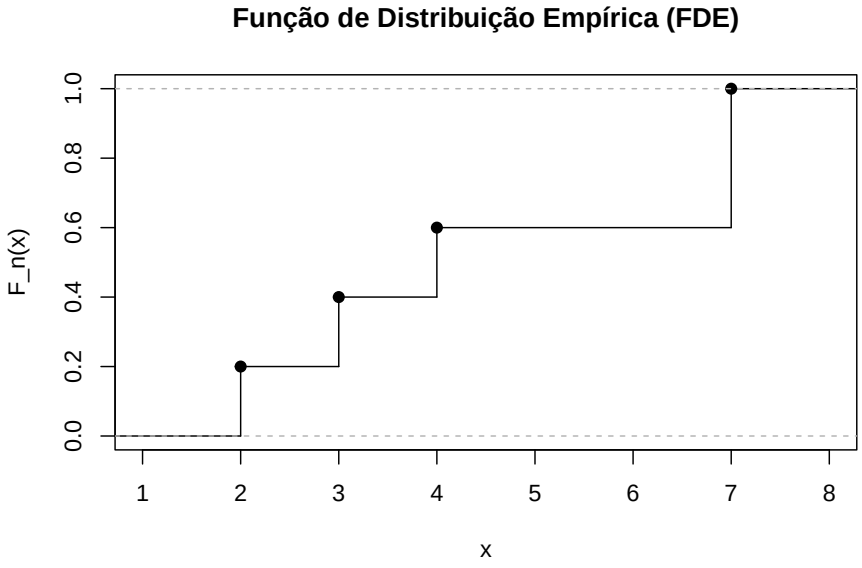
item (a)
$\{2, 3, 4, 7, 7\}$

item (b)

x	FDE
2	0.2
3	0.4
4	0.6
7	1.0

item (c)

```
dados <- c(3, 7, 4, 2, 7)

fde <- ecdf(dados)

plot(fde, main = "Função de Distribuição Empírica (FDE)",
     xlab = "x", ylab = "F_n(x)", verticals = TRUE, do.points = TRUE, pch = 19)
```



	Doenca		
	Doente	Não Doente	**Total**
Teste			
Negativo	50 (25%)	75 (38%)	125 (63%)
Positivo	50 (25%)	25 (13%)	75 (38%)
Total	100 (50%)	100 (50%)	200 (100%)

11.5 Capítulo 7

- **Exercício 1**

Primeiro, interpretando o exercício:

Das 200 pessoas estudadas, 100 pacientes são doentes e 100 não são doentes. No teste, dos 75 resultados positivos, 25 são falsos-positivos, ou seja, temos 50 pacientes *positivados* e *com HIV*. Já dos 125 resultados negativos, 50 são falsos-negativos, ou seja, temos 75 pacientes *negativados* e *sem HIV*.

item (a)

```
library(dplyr)
library(gtsummary)

# Criar os dados com data.frame()
dados_hiv <- data.frame(
  Teste = c(rep("Positivo", 75), rep("Negativo", 125)),
  Doenca = c(
    rep("Doente", 50),         # verdadeiros positivos
    rep("Não Doente", 25),     # falsos positivos
    rep("Doente", 50),         # falsos negativos
    rep("Não Doente", 75)      # verdadeiros negativos
))

dados_hiv %>%
  tbl_cross(
    row = Teste,
    col = Doenca,
    percent = "cell"
  ) %>%
  bold_labels()
```

item (b)

Seja: *vp*: verdadeiro positivo (tem HIV e teste foi positivo) *fp*: falso positivo (não tem HIV, mas teste deu positivo) *fn*: falso negativo (tem HIV, mas teste deu negativo) *vn*: verdadeiro negativo (não tem HIV e teste deu negativo)

Sensibilidade: a probabilidade do teste dar positivo, dado que a pessoa está doente.

$$S = \frac{\text{vp}}{\text{vp + fn}} = \frac{50}{50+50} = 0,5$$

Especificidade: a probabilidade do teste dar negativo, dado que a pessoa não está doente.

$$E = \frac{\text{vn}}{\text{vn + fp}} = \frac{75}{75+25} = 0,75$$

Item (c)

Valor Preditivo Positivo: a probabilidade da pessoa estar doente, dado que o teste deu positivo.

$$VPP = \frac{\text{vp}}{\text{vp + fp}} = \frac{50}{50+25} \approx 0,67$$

Valor Preditivo Negativo: a probabilidade da ausência de doença quando o teste deu negativo.

$$VPN = \frac{\text{vn}}{\text{vn + fn}} = \frac{75}{75+50} = 0,6$$

item (d)

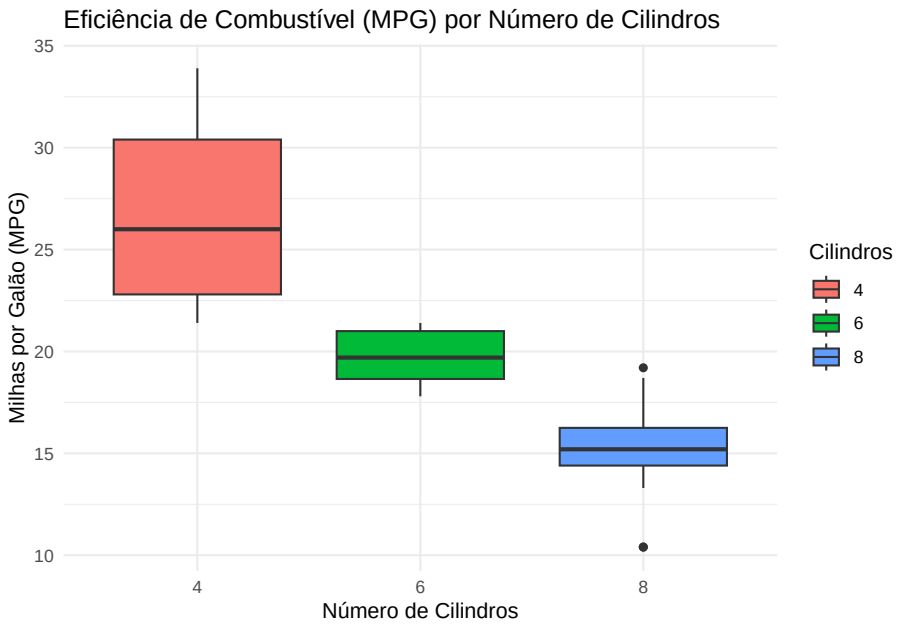
A *acúracia* é a probabilidade do teste fornecer resultados corretos, ou seja, ser positivo nos doentes e negativo nos não doentes.

$$AC = \frac{\text{vp + vn}}{\text{total}} = \frac{50+75}{200} = 0,625$$

11.6 Capítulo 8

- **Exercício 1**

```r
library(ggplot2)

# Carregar o dataset mtcars (já está disponível no R)
data("mtcars")

ggplot(mtcars, aes(x = factor(cyl), y = mpg, fill = factor(cyl))) +
  geom_boxplot() +
  labs(
    title = "Eficiência de Combustível (MPG) por Número de Cilindros",
    x = "Número de Cilindros",
    y = "Milhas por Galão (MPG)",
    fill = "Cilindros"
  ) +
  theme_minimal()
```



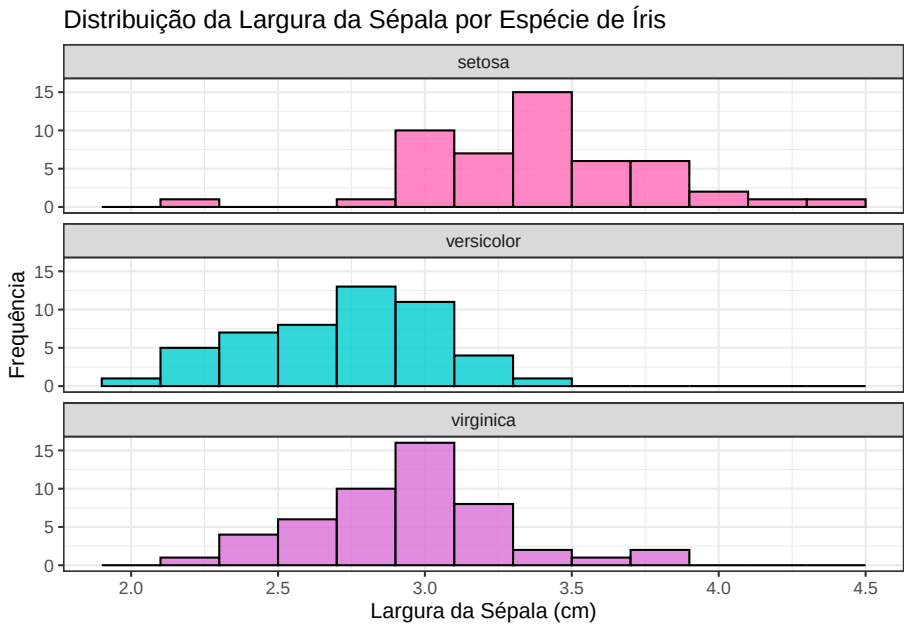
O boxplot mostra uma associação negativa entre o número de cilindros e a eficiência de combustível (MPG).

Podemos observar que para carros com 4 cilindros (4-cyl) mediana de MPG mais alta e uma dispersão relativamente grande, indicando maior eficiência. Já em carros com 6 cilindros (6-cyl) a mediana de MPG é menor do que os de 4 cilindros e com menor dispersão. Por fim, carros com 8 cilindros (8-cyl) possuem a mediana de MPG mais baixa, sugerindo que, em média, são os menos eficientes em termos de combustível.

Ou seja, o gráfico sugere que, à medida que o número de cilindros aumenta, a eficiência de combustível tende a diminuir.

- **Exercício 2**

```
# Carregar o pacote ggplot2
library(ggplot2)

# Carregar o dataset iris (já está disponível no R)
data("iris")

# Opcional, definir as cores para cada espécie, similar ao seu modelo
cores_especies <- c("setosa" = "hotpink",
                    "versicolor" = "darkturquoise",
                    "virginica" = "orchid")


ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
  geom_histogram(alpha = 0.8, color = "black", binwidth = 0.2) +
  scale_fill_manual(values = cores_especies) +
  facet_wrap(~Species, ncol = 1) +
  labs(
    title = "Distribuição da Largura da Sépala por Espécie de Íris",
    x = "Largura da Sépala (cm)",
    y = "Frequência"
  ) +
  theme_bw() +
  theme(legend.position = "none")
```



11.7 Capítulo 9

- **Exercício 1**

item (a)

Simular os lançamentos de um dado honesto:

```
set.seed(123)

n <- 10000

lancamentos <- sample(1:6, size = n, replace = TRUE)
```

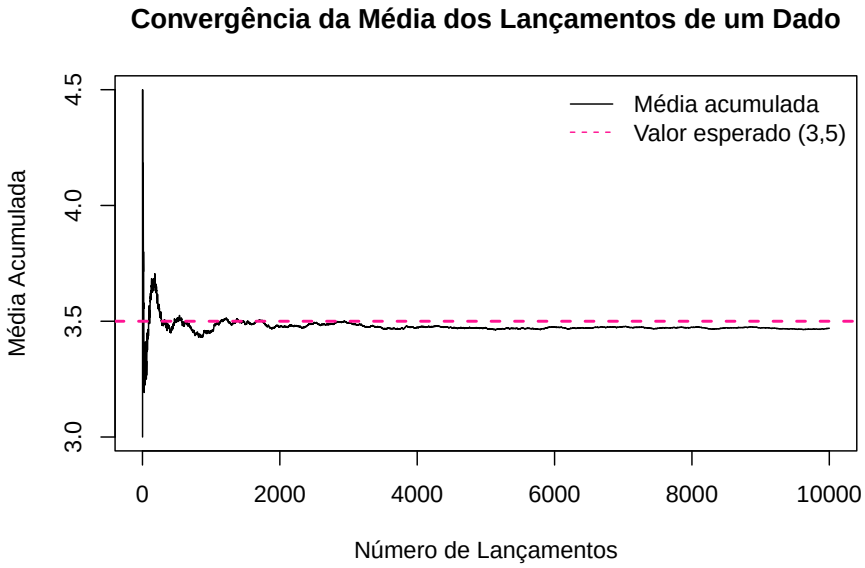
item (b)

Calcular a média acumulada depois de cada lançamento:

```
somas <- cumsum(lancamentos)

# Divide cada soma pelo número de lançamentos até aquele ponto
medias <- somas / (1:n)
```

item (c)

```
plot(medias, type = "l",
     main = "Convergência da Média dos Lançamentos de um Dado",
     xlab = "Número de Lançamentos",
     ylab = "Média Acumulada",
     col = "black")

# Adicionar linha horizontal com o valor esperado (3,5)
abline(h = 3.5, col = "deeppink", lty = 2, lwd = 2)

# Legenda
legend("topright", legend = c("Média acumulada", "Valor esperado (3,5)"),
       col = c("black", "deeppink"), lty = c(1, 2), bty = "n")
```



- **Exercício 2**

item (a)

```
set.seed(123)

n_simulacoes <- 10000
```

```r
n_questoes <- 10

# A probabilidade de acerto ao chutar uma questão é 1 em 4 (0,25)
p_acerto <- 1 / 4

acertos <- rbinom(n_simulacoes, size = n_questoes, prob = p_acerto)
```

item (b)

```r
sucessos <- sum(acertos >= 4)

prob_estimada <- sucessos / n_simulacoes

cat("Estimativa de P(acertar pelo menos 4 questões):", prob_estimada, "\n")
```

```
## Estimativa de P(acertar pelo menos 4 questões): 0.2183
```

item (c)

```r
acertos <- rbinom(n_simulacoes, size = n_questoes, prob = p_acerto)
```

- **Exercício 3**

item (a)

```r
set.seed(123)

n_simulacoes <- 10000

dado1 <- sample(1:6, size = n_simulacoes, replace = TRUE)
dado2 <- sample(1:6, size = n_simulacoes, replace = TRUE)
```

item (b)

```r
somas <- dado1 + dado2

# Verificar se soma > 8 OU soma == 5
condicao_satisfeita <- (somas > 8) | (somas == 5)

# Contar quantas vezes a condição foi satisfeita
numero_condicao_satisfeita <- sum(condicao_satisfeita)

cat("Número de vezes que a condição foi satisfeita:", numero_condicao_satisfeita, "\n")
```

```
## Número de vezes que a condição foi satisfeita: 3812
```

item (c)

```
# Estimar a Probabilidade
probabilidade_estimada <- numero_condicao_satisfeita / n_simulacoes

cat("Probabilidade estimada:", probabilidade_estimada, "\n")
```

```
## Probabilidade estimada: 0.3812
```