

Introdução à Estatística usando o R com Aplicação em Análises Laboratoriais

Profa Carolina & Prof Gilberto

Instituto de Matemática e Estatística
Universidade Federal da Bahia

05 de outubro de 2019

Cronograma do curso

■ Parte 1: Introdução ao R e Estatística Descritiva.

- 05/10/2019 - 7h00 às 12h30: Introdução ao R e Estatística Descritiva.

■ Parte 2: Probabilidade.

- 19/10/2019 - 7h00 às 12h30: Probabilidade.

■ Parte 3: Intervalo de confiança e teste de hipótese.

- 26/10/2019 - 7h00 às 12h30: - 7h00 às 12h30: Intervalos de confiança.
- 09/11/2019 - 7h00 às 12h30: Teste de hipóteses.

■ Parte 4: Regressão linear simples e ANOVA.

- 23/11/2019 - 7h00 às 12h30: Regressão linear simples.
- 30/11/2019 - 7h00 às 12h30: ANOVA.

Antes de começar

Abra o RStudio (editor que usaremos para aprender a usar R)

Instale os seguintes pacotes

- `install.packages('tidyverse')`
- `install.packages('MASS')`
- `install.packages('readxl')`
- `install.packages('xlsx')`

Após instalar, carregue estes pacotes

- `library(MASS)`
- `library(readxl)`
- `library(xlsx)`
- `library(DescTools)`
- `library(tidyverse)`

Origens históricas do R

História do S (precursor do R)

- R é uma linguagem derivada do S
- S foi desenvolvido em Fortran por John Chambers em 1976 no Bell Labs
- S foi desenvolvido para ser um ambiente de análise estatística
- Em 1988, a versão 4 (implementada em C) foi lançada (permitiu portabilidade entre sistemas operacionais)
- Filosofia do S: permitir que usuários possam analisar dados usando estatística com pouco conhecimento de programação

História do R

- Em 1991, Ross Ihaka e Robert Gentleman criaram o R na Nova Zelândia
- Em 1995, Ross e Robert liberam o R sob a licença “GNU General License”, o que tornou o R um software livre
- Em 1997, The Core Group é criado para melhorar e controlar o código fonte do R

Características do R

- Constante melhoramento e atualização
- Portabilidade (roda em praticamente todos os sistemas operacionais)
- Grande comunidade de desenvolvedores que adicionam novas capacidades ao R através de pacotes
- Gráficos de maneira relativamente simples
- Interatividade
- Uma grande comunidade de usuários (especialmente útil para resolução de problemas)

Referência para aprender R

Onde baixar o R/RStudio:

- Baixe o R: <https://cran.r-project.org/>
- RStudio (Editor de R): <https://www.rstudio.com/>

Livros:

- Iniciante no R: Hands-On Programming with R: Write Your Own Functions and Simulations
- Intermediário no R: R for data science
- Avançado no R: Advanced R

Na internet:

- Material em português – Curso-R: <http://material.curso-r.com/>
- Nível intermediário de R – R for data science: <http://r4ds.had.co.nz/>
- Nível avançado de R – R Advanced: <http://adv-r.had.co.nz/>

O que fazer quando estiver em apuros?

■ Documentação do R

```
help(mean) #pedindo ajuda pelo console  
?mean #modo alternativo de pedir ajuda pelo console
```

■ Programador mais experiente mais próximo

■ Stack Overflow: <https://pt.stackoverflow.com/>

■ Google

```
log('G')
```

```
## Error in log("G"): non-numeric argument to mathematical function
```

Pesquisar no Google “Error in log("G") : non-numeric argument to mathematical function”

O RStudio

Componentes do RStudio

- Editor/Scripts: é onde escrevemos nossos códigos.
- Console: é onde rodamos o código e recebemos as saídas.
- Environment: painel com todos os objetos criados na sessão.
- Files: mostra os arquivos no diretório de trabalho. É possível navegar entre diretórios.
- Plots: painel onde os gráficos serão apresentados.
- Help: janela onde a documentação das funções serão apresentadas.
- History: painel com um histórico dos comandos rodados.
- Environment: Objetos criados.
- Packages: Gerenciador de pacotes do Editor RStudio.

Começando a usar o RStudio

- 1 Separe uma pasta para desenvolver a sua análise;
- 2 Crie um novo projeto nesta pasta;
- 3 Como rodar um código no R:
 - Selecione parte do código e clique em Crtl+Enter ou Crtl+R
 - Selecione parte do código e clique no botão "Run"
 - Digite no console o seguinte código: `source("nome do arquivo.R")`
 - Digite Crtl+Shift+R ou Crtl+Shift+S para rodar todo o arquivo ativo
- 4 Instalar pacotes e carregar pacotes:
 - através da interface gráfico do RStudio (Packages)
 - através do comando `install.packages("nome do pacote")`
 - para carregar pacotes: `library("nome do pacote")` ou `require("nome do pacote")`.

Operações aritméticas básicas para números

#Soma

1+1

[1] 2

#subtração

2-1

[1] 1

#divisão

3/2

[1] 1.5

Operações aritméticas básicas para números

```
#potenciação  
2^3
```

```
## [1] 8
```

```
#Resto da divisão de 5 por 3  
5 %% 3
```

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

```
#parte inteira da divisão de 5 por 3  
5 %/% 3
```

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

Figura 1: Divisão inteira %/% e operador resto %%.

$$\begin{array}{r|l} 5 & 3 \\ -3 & 1 = 5 \% \% 3 \\ \hline 2 & 2 = 5 \% / \% 3 \end{array}$$

Tipos básicos de dados no R

R é uma linguagem vetorial e matricial, e os objetos básicos são vetores, listas e matrizes. Números são vetores de comprimento 1.

Vetores são elementos no R caracterizados por todos os valores serem do mesmo tipo. Existem 5 tipos básicos de dados no R:

■ Inteiro (Integer)

```
a <- 1L  
typeof(a)
```

```
## [1] "integer"
```

■ Número complexo (Complex)

```
a <- 2 + 5i  
typeof(a)
```

```
## [1] "complex"
```

Tipos básicos de dados no R (continuação)

■ Lógico (Logic)

```
a <- TRUE  
typeof(a)
```

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

■ Número real (double)

```
a <- 1.3  
typeof(a)
```

```
## [1] "double"
```

■ Carácter (character)

```
a <- "Eu mesmo: Gilberto"  
typeof(a)
```

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

Vetores no R

■ Vetor numérico

```
a <- c(1, 2, 3)
print(a)
```

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

```
class(a)
```

```
## [1] "numeric"
```

■ Vetor caracter

```
a <- c("Gilberto", "Pereira", "Sassi")
print(a)
```

```
## [1] "Gilberto" "Pereira" "Sassi"
```

```
class(a)
```

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

Matrizes no R

■ Matriz numérica

```
(a <- matrix(1:6, nrow = 2, ncol = 3) )
```

```
##      [,1] [,2] [,3]  
## [1,]    1    3    5  
## [2,]    2    4    6
```

```
class(a)
```

```
## [1] "matrix"
```

■ Matriz caracter

```
(a <- matrix(c('a','b','c','d'), nrow = 2, ncol = 2))
```

```
##      [,1] [,2]  
## [1,] "a"  "c"  
## [2,] "b"  "d"
```

```
class(a)
```

```
## [1] "matrix"
```

Operações com matrizes

- soma de matrizes (duas matrizes de mesma dimensão)

```
(A <- matrix(1:6, nrow = 3, ncol = 2))
```

```
##      [,1] [,2]  
## [1,]    1    4  
## [2,]    2    5  
## [3,]    3    6
```

```
B <- matrix(rep(0.1,6), nrow = 3, ncol = 2)  
(C <- A + B)
```

```
##      [,1] [,2]  
## [1,]  1.1  4.1  
## [2,]  2.1  5.1  
## [3,]  3.1  6.1
```


Operações com matrizes

■ Transposição de matriz

```
(D <- t(A))
```

##	[, 1]	[, 2]	[, 3]
## [1,]	1	2	3
## [2,]	4	5	6

■ Multiplicação de matrizes (quando possível)

```
(E <- A %*% t(A))
```

##	[, 1]	[, 2]	[, 3]
## [1,]	17	22	27
## [2,]	22	29	36
## [3,]	27	36	45

Operações com matrizes

■ Matriz identidade

```
(A <- diag(3))
```

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

■ Matriz diagonal

```
(A <- diag(1:3))
```

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

Operações com matrizes

■ Retirar a diagonal principal de um matriz

```
(A <- matrix(1:9, nrow = 3, ncol = 3))
```

```
##      [,1] [,2] [,3]  
## [1,]    1    4    7  
## [2,]    2    5    8  
## [3,]    3    6    9
```

```
(B <- diag(A))
```

```
## [1] 1 5 9
```

Operações com matrizes

■ Retirar uma linha de um matriz

```
A <- matrix(1:9, nrow = 3, ncol = 3)
print(A[1, ]) #selecionar a linha 1
```

```
## [1] 1 4 7
```

```
print(A[,1]) #selecionar a coluna 1
```

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

■ Modificar o valor de um único elemento da matriz

```
A[1,3] <- 0.1 #Atribui 0,1 ao valor A[1,3]
print(A)
```

```
##      [,1] [,2] [,3]
## [1,]    1    4 0.1
## [2,]    2    5 8.0
## [3,]    3    6 9.0
```

Operações com matrizes

■ Determinante da matriz

```
det(A)
```

```
## [1] 20.7
```

■ Matriz inversa de uma matriz quadrada (A %*% ginv(A) %*% A == A) – precisa do pacote MASS

```
ginv(A)
```

```
##           [,1]      [,2]      [,3]  
## [1,] -0.1449275 -1.7101449  1.5217391  
## [2,]  0.2898551  0.4202899 -0.3768116  
## [3,] -0.1449275  0.2898551 -0.1449275
```

Alguns comandos úteis para matrizes

Operador ou função	Descrição
$A * B$	multiplicação ponto-a-ponto
$A \% * \% B$	multiplicação matricial
$A \% o \% B$	multiplicação exterior $A \cdot B^T$
$\text{crossprod}(A, B)$	$A \cdot B^T$
$\text{crossprod}(A)$	$A \cdot A^T$
$t(A)$	transposta da matriz
$\text{diag}(x)$	cria uma matriz diagonal igual a x
$\text{diag}(A)$	retira a diagonal da matriz
$\text{diag}(k)$	cria uma matriz identidade de ordem k
$\text{ginv}(A)$	matriz inversa de uma matriz do pacote MASS
$\text{rowMeans}(A)$	média por linhas
$\text{rowSums}(A)$	soma por linhas
$\text{colMeans}(A)$	médias por colunas
$\text{colSums}(A)$	soma por colunas

Valores especiais

- **NA (Not Available)**: significa que o valor está faltante. Para verificar se um objeto é NA, usamos a função `is.na()`;
- **NaN (Not a Number)**: significa que o resultado da operação envolvendo número não é um número. Para verificar se um objeto é NaN, usamos a função `is.nan()`;
- **Inf (Infinito)**: significa que o valor numérico do objeto é maior que o limite que a máquina suporta. Para verificar se um objeto é Inf, usamos a função `is.infinite()`;
- **NULL (Null)**: ausência de informação. NA está mais associada com ausência de informação em um conjunto de dados e NULL está associado com ausência de informação em programação. `is.null()` verifica se o objeto é NULL.

Listas

Podemos agregar quaisquer tipo de objeto em um único objeto chamado list.

```
#Exemplo
pedido <- list(pedido_id = 8001406,
               nome = "Fulano de Tal",
               cpf = "12345678900",
               itens = list(
                 list(descricao = "Ferrari",
                      frete = 0,
                      valor = 500000),
                 list(descricao = "Dolly",
                      frete = 1.5,
                      valor = 3.90)))
```


Listas

```
#retirar a ID do pedido  
pedido$pedido_id
```

```
## [1] 8001406
```

```
#Quarto elemento da lista  
pedido[3]
```

```
## $cpf
```

```
## [1] "12345678900"
```

```
#valor do quarto elemento da lista  
pedido[[3]]
```

```
## [1] "12345678900"
```

Listas



```
#nome de um atributo da lista  
pedido['nome']
```

```
## $nome  
## [1] "Fulano de Tal"
```

```
#nomes da lista  
names(pedido)
```

```
## [1] "pedido_id" "nome" "cpf" "itens"
```

tibble

Um `data.frame` é o mesmo que uma tabela do SQL ou um spreadsheet do Excel. Seus dados serão armazenados como um objeto `tibble`.

Um `tibble` é uma tabela, em que cada linha é um elemento ou indivíduo da amostra e cada coluna é uma variável.

■ Operação com `data.frame`

- `head()` - Mostra as primeiras 6 observações (linhas).
- `tail()` - Mostra as últimas 6 observações (linhas).
- `dim()` - Número de observações (linhas) e de variáveis (colunas).
- `names()` - Os nomes das variáveis (colunas).
- `add_column()` - adiciona uma variável (coluna) ao `tibble`.
- `add_row()` ou `add_case()` - adiciona novas observações (linhas) ao `tibble`.
- `glimpse()` - sumário sobre o `data.frame`.

tibble

```
(dados <- tibble(temp = c(10,16,22), especie = rep('trigo', 3),  
                 germinacao = c(0,2,0)))
```

```
## # A tibble: 3 x 3  
##   temp especie germinacao  
##   <dbl> <chr>      <dbl>  
## 1    10 trigo         0  
## 2    16 trigo         2  
## 3    22 trigo         0
```

```
head(dados, n= 1) #n primeiras observações da amostra
```

```
## # A tibble: 1 x 3  
##   temp especie germinacao  
##   <dbl> <chr>      <dbl>  
## 1    10 trigo         0
```

tibble

```
tail(dados, n = 1) #n últimas observações da amostra
```

```
## # A tibble: 1 x 3  
##   temp especie germinacao  
##   <dbl> <chr>      <dbl>  
## 1    22 trigo          0
```

```
#adicionando uma variável
```

```
(dados <- add_column(dados, inicio = c(0,0,0), fim = c(10,15,5)))
```

```
## # A tibble: 3 x 5  
##   temp especie germinacao inicio   fim  
##   <dbl> <chr>      <dbl> <dbl> <dbl>  
## 1    10 trigo          0      0    10  
## 2    16 trigo          2      0    15  
## 3    22 trigo          0      0     5
```

```
names(dados) #nome das variáveis
```

```
## [1] "temp"          "especie"       "germinacao"   "inicio"       "fim"
```

tibble

```
(dados <- add_case(dados, temp = 10, especie = 'trigo', germinacao = 7
```

```
## # A tibble: 4 x 5
##   temp especie germinacao inicio    fim
##   <dbl> <chr>      <dbl>   <dbl> <dbl>
## 1    10 trigo         0     0    10
## 2    16 trigo         2     0    15
## 3    22 trigo         0     0     5
## 4    10 trigo         7    NA    NA
```

```
dim(dados) # número de observação e de variáveis
```

```
## [1] 4 5
```

Lendo os dados no R – arquivos excel

Para arquivos excel, usamos as funções `read_xls` e `read_xlsx` do pacote `readxl`.

Informações que precisam ser informadas:

- `path`: caminho completo até o arquivo excel;
- `sheet`: nome da planilha que será lida;
- `range`: delimita as células que serão lidas;
- `col_names`: argumento lógico. Se `TRUE`, a primeira linha de `range` é nome das variáveis.

```
df_iris <- read_xlsx(path = 'dados.xlsx', sheet = 'Iris',  
                     range = "A1:E101",  
                     col_names = TRUE)
```

Lendo os dados no R

```
glimpse(df_iris)
```

```
## Observations: 100
## Variables: 5
## $ sepal_comp <dbl> 4.4, 5.8, 5.4, 6.4, 6.7, 5.5, 5.6, 6.8, 6.1, 6.
## $ sepal_larg <dbl> 3.2, 2.7, 3.4, 2.7, 3.0, 2.5, 2.7, 2.8, 2.8, 2.
## $ petal_comp <dbl> 1.3, 4.1, 1.5, 5.3, 5.0, 4.0, 4.2, 4.8, 4.7, 5.
## $ petal_larg <dbl> 0.2, 1.0, 0.4, 1.9, 1.7, 1.3, 1.3, 1.4, 1.2, 1.
## $ especie <chr> "setosa", "versicolor", "setosa", "virginica",
```


Lendo os dados no R – arquivos de texto (csv ou txt)

Para arquivos de texto, usamos as funções `read_delim` do pacote `readr` (incluso no `tidyverse`).

Informações que precisam ser informadas:

- `file`: caminho completo até o arquivo `.txt` ou `.csv`;
- `col_names`: argumento lógico. Se `TRUE`, a primeira linha do arquivo `.txt` é o nome das variáveis.
- `delim`: caracter delimitador ou divisor das variáveis ou colunas.
- `locale`: opções para ler arquivos. Aqui no curso, vamos usar principalmente para especificar o sinal de decimal.

```
# lendo arquivos txt
df_iris <- read_delim(file = "iris.txt", col_names = TRUE,
                      delim = "|", locale = locale(decimal_mark = ","))

## Parsed with column specification:
## cols(
##   sepala_comp = col_double(),
##   sepala_larg = col_double(),
##   petala_comp = col_double(),
##   petala_larg = col_double(),
##   especie = col_character()
## )
```

Lendo os dados no R – arquivos de texto (continuação)

```
# lendo arquivos csv (formato europeu -- usado no Brasil)  
df_iris <- read_csv2(file = "iris.csv", col_names = TRUE)
```

```
## Using ',' as decimal and '.' as grouping mark. Use read_delim() for  
## Parsed with column specification:  
## cols(  
##   sepala_comp = col_double(),  
##   sepala_larg = col_double(),  
##   petala_comp = col_double(),  
##   petala_larg = col_double(),  
##   especie = col_character()  
## )
```

Conceitos básicos

Começamos com alguns conceitos básicos, que usaremos durante todo esse curso.

- **População:** Todos os elementos ou indivíduos alvo do estudo;
- **Amostra:** Parte da população;
- **Parâmetro:** característica da população (grandeza);
- **Estimativa:** característica da amostra. Usamos a estimativa para aproximar o parâmetro;
- **Variável:** característica de um elemento da população (mensurando ou analito). Geralmente usamos uma letra maiúscula do alfabeto latino para representar uma variável (mensurando ou analito), e uma letra minúscula do alfabeto latino para representar o valor de uma variável para um elemento (indicação) da população. Por exemplo, podemos representar a variável "Teor de hidrócloro" por X e um indicação da amostra por $x = 25, 1 \text{ mg/comprimido}$.

Classificação de variáveis

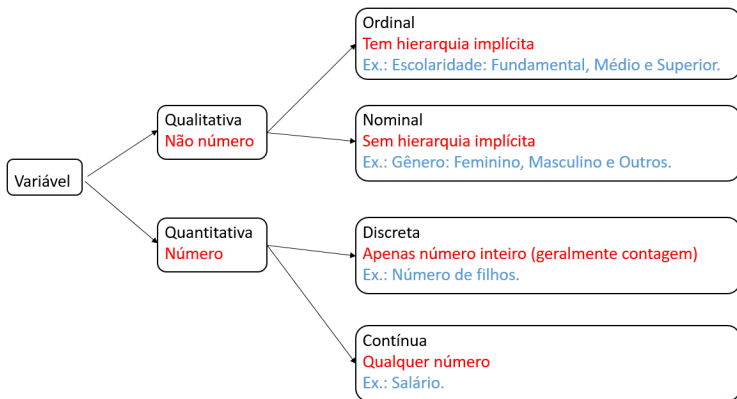


Figura 2: Classificação de variáveis.

Tabela de distribuição de frequência – Variável qualitativa

A primeira coisa que fazemos é contar!

Seja X uma variável qualitativa com valores possíveis B_1, \dots, B_k , então construímos a tabela de distribuição de frequências como ilustrado na Tabela 1.

Tabela 1: Tabela de distribuição de frequências – variável qualitativa.

X	Frequência	Frequência Relativa	Porcentagem
B_1	n_1	$f_1 = \frac{n_1}{n}$	$100 \cdot f_1$
B_2	n_2	$f_2 = \frac{n_2}{n}$	$100 \cdot f_2$
\vdots	\vdots	\vdots	\vdots
B_k	n_k	$f_k = \frac{n_k}{n}$	$100 \cdot f_k$
Total	n	1	100%

Em que n_i , $i = 1, \dots, k$ é o número de indivíduos com valor de X igual a B_i .

Tabela de distribuição de frequência – Variável qualitativa

A primeira coisa que podemos fazer é construir a tabela de distribuição de frequência.

```
tab_freq <- df_iris %>%  
  group_by(especie) %>%  
  summarise(frequencia = n()) %>%  
  mutate(frequencia_relativa = frequencia / sum(frequencia),  
         porcentagem = 100 * frequencia_relativa)  
tab_freq %>%  
  add_case(especie = 'Total',  
           frequencia=sum(tab_freq$frequencia),  
           frequencia_relativa = sum(tab_freq$frequencia_relativa),  
           porcentagem = sum(tab_freq$porcentagem))
```

```
## # A tibble: 4 x 4  
##   especie   frequencia frequencia_relativa porcentagem  
##   <chr>      <int>          <dbl>          <dbl>  
## 1 setosa      32            0.32            32  
## 2 versicolor 37            0.37            37  
## 3 virginica   31            0.31            31  
## 4 Total     100            1              100
```

Gráfico no R

Vamos construir o gráfico de barras para a variável `especie`.

Vamos usar o pacote `ggplot2` já incluso no pacote `tidyverse`.

O gráficos usando `ggplot` tem o seguinte formato:

```
ggplot(data = <data possible tibble>) +  
  <Geom functions>(mapping = aes(<MAPPINGS>))
```



Universidade
Federal da Bahia

Gráfico de barras – variável qualitativa

Para a variável `especie`, temos que

```
ggplot(data = df_iris) +  
  geom_bar(mapping = aes(x = especie, y = ..prop..,  
                          group = 1),  
            fill = 'blue') +  
  labs(x = 'Espécie', y = 'Frequência Relativa',  
        title = 'Gráfico de Barras')
```

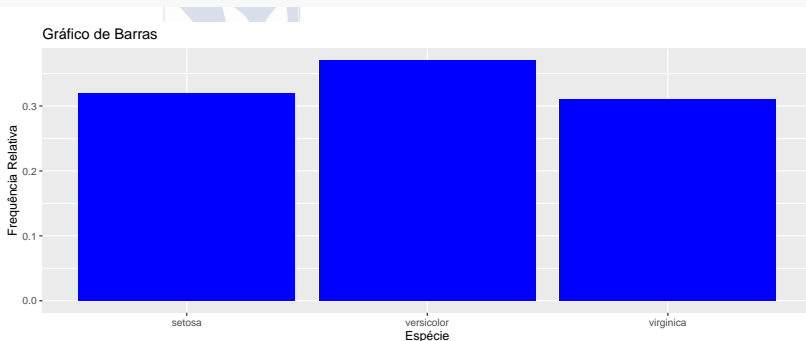


Tabela de distribuição de frequências – variável quantitativa discreta

```
df_arroz <- read_xlsx("dados.xlsx", sheet="Arroz")
tab_arroz <- df_arroz %>% group_by(germinado) %>%
  summarise(frequencia = n()) %>%
  mutate(frequencia_relativa = frequencia / sum(frequencia),
         porcentagem = frequencia_relativa * 100)
tab_arroz %>% add_row(germinado = 'Total',
                    frequencia =
                      sum(tab_arroz$frequencia),
                    frequencia_relativa =
                      sum(tab_arroz$frequencia_relativa),
                    porcentagem =
                      sum(tab_arroz$porcentagem))
```

Tabela de distribuição de frequências – variável quantitativa discreta

```
## # A tibble: 13 x 4
##   germinado frequencia frequencia_relativa porcentagem
##   <chr>      <int>          <dbl>          <dbl>
## 1 0          57          0.722          72.2
## 2 1           7          0.0886         8.86
## 3 2           5          0.0633         6.33
## 4 3           1          0.0127         1.27
## 5 4           2          0.0253         2.53
## 6 5           1          0.0127         1.27
## 7 6           1          0.0127         1.27
## 8 9           1          0.0127         1.27
## 9 10          1          0.0127         1.27
## 10 12          1          0.0127         1.27
## 11 13          1          0.0127         1.27
## 12 17          1          0.0127         1.27
## 13 Total       79          1            100
```

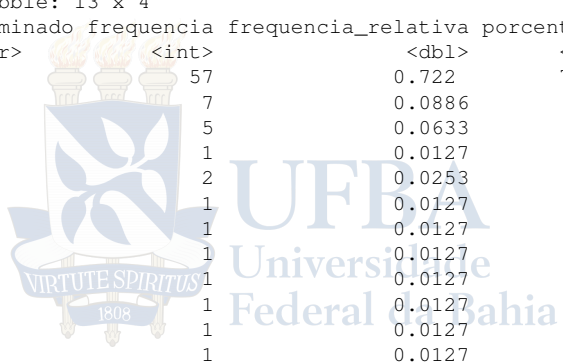
The image contains a large, semi-transparent watermark of the UFBA (Universidade Federal da Bahia) logo. The logo features a shield with a stylized plant, three torches above it, and a banner below with the text 'VIRTUTE SPIRITUS' and the year '1808'. The text 'UFBA' and 'Universidade Federal da Bahia' are also visible in the background.

Gráfico de barras – variável quantitativa discreta

Gráfico de barras.

```
ggplot(data = df_arroz) +  
  geom_bar(mapping = aes(x = germinado),  
            fill = 'blue') +  
  labs(x = 'Número de sementes germinadas', y = "Frequência",  
        title = 'Gráfico de barras')
```

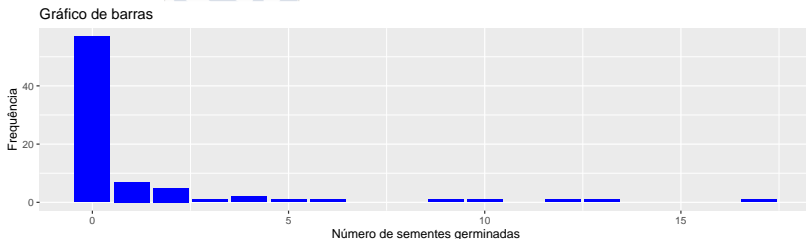


Tabela de distribuição de frequências – variável quantitativa contínua

Vamos construir um histograma para a comprimento de pétala para a espécie versicolor.

```
df_versicolor <- df_iris %>% filter(especie %in% 'versicolor')
k <- (1 + nrow(df_versicolor) %>% log2()) %>% ceiling()
tabela <- df_versicolor %>%
  group_by(petala_qual = cut(petala_comp, breaks = k,
                             include.lowest = T, right = F)) %>%
  summarise(frequencia = n()) %>%
  mutate(frequencia_relativa = frequencia / sum(frequencia),
         porcentagem = frequencia_relativa * 100)
tabela %>% add_row(petala_qual = 'Total',
                  frequencia = sum(tabela$frequencia),
                  frequencia_relativa =
                    sum(tabela$frequencia_relativa),
                  porcentagem = sum(tabela$porcentagem))
```

Tabela de distribuição de frequências – variável quantitativa contínua

```
## # A tibble: 8 x 4
##   petala_qual frequencia frequencia_relativa porcentagem
##   <fct>         <int>          <dbl>          <dbl>
## 1 [3,3.3)         1          0.0270          2.70
## 2 [3.3,3.6)       1          0.0270          2.70
## 3 [3.6,3.9)       2          0.0541          5.41
## 4 [3.9,4.2)       8          0.216           21.6
## 5 [4.2,4.5)       5          0.135           13.5
## 6 [4.5,4.8)      14          0.378           37.8
## 7 [4.8,5.1]       6          0.162           16.2
## 8 Total          37          1             100
```

Histograma – variável quantitativa contínua

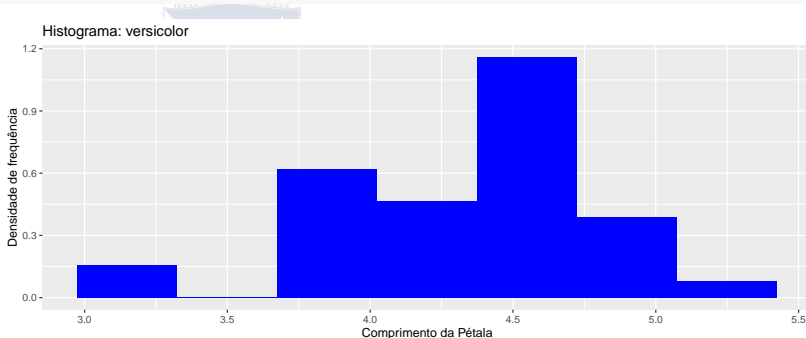
Nos gráficos de barras, a frequência (ou frequência relativa ou porcentagem) está no eixo y, ou seja, na altura da barra.

O histograma tem uma interpretação ligeiramente diferente: a área da barra é a frequência relativa.

- Para variável quantitativa contínua, dividimos os valores em faixas de valores e calculamos a frequência relativa para cada faixa.
- Para a barra correspondente à faixa $[a, b)$ a altura da barra precisa ser $\frac{f}{b - a}$, em que f é a frequência relativa da faixa $[a, b)$.
- Chamamos a razão $\frac{f}{b - a}$ de densidade de frequência.
- Número de faixas, podemos usar a regra de Sturge: $[1 + \log_2(n)]$.

Histograma – variável quantitativa contínua

```
ggplot(data = df_versicolor)+  
  geom_histogram(mapping = aes(x = petala_comp, y = ..density..),  
                 bins = k, fill = 'blue')+  
  labs(x = 'Comprimento da Pétala',  
       y = 'Densidade de frequência',  
       title = 'Histograma: versicolor')
```



Medidas de Resumo (variável quantitativa)

A ideia é encontrar um ou alguns valores que sintetizem todas as indicações.

Medidas de posição (tendência central)

A ideia é encontrar um valor que representa “bem” todas as indicações.

- **Média:** $\bar{X} = \frac{x_1 + \dots + x_n}{n}$
- **Mediana:** valor que divide a sequência ordenada de valores em duas partes iguais.

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

em que $x_{(j)}$ é o j-ésimo menor valor da variável quantitativa X .

Medidas de dispersão

A ideia é medir a homogeneidade das indicações.

- **Variância:** $s^2 = \frac{(x_1 - \bar{X})^2 + \dots + (x_n - \bar{X})^2}{n - 1}$;
- **Desvio padrão:** $s = \sqrt{s^2}$ (mesma unidade dos dados);
- **coeficiente de variação** $cv = \frac{s}{\bar{X}} \cdot 100\%$ (adimensional, ou seja, “sem unidade”)

Medidas de Resumo: exemplo

Podemos usar a função `summarise` do pacote `dplyr` (inclusive no pacote `tidyverse`).

```
# Média para o comprimento de pétala para a espécie versicolor
df_versicolor %>% summarise(media = mean(petala_comp),
                             s2 = var(petala_comp),
                             s = sd(petala_comp),
                             mediana = median(petala_comp),
                             cv = s * 100 / media)
```

```
## # A tibble: 1 x 5
##   media    s2      s mediana    cv
##   <dbl> <dbl> <dbl>   <dbl> <dbl>
## 1   4.34  0.210  0.459     4.5  10.6
```

Medidas de Resumo: exemplo

Podemos usar a função `summarise` do pacote `dplyr` (inclusive no pacote `tidyverse`).

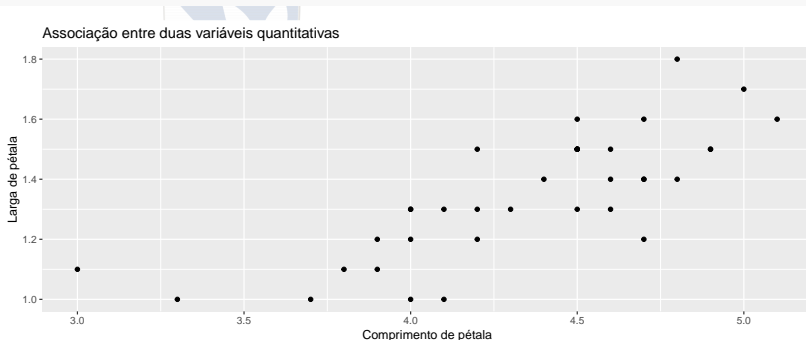
```
# Média para o comprimento de pétala para cada espécie
df_iris %>% group_by(especie) %>%
  summarise(media = mean(petala_comp),
             s2 = var(petala_comp),
             s = sd(petala_comp),
             mediana = median(petala_comp),
             cv = s * 100 / media)
```

```
## # A tibble: 3 x 6
##   especie      media      s2      s mediana    cv
##   <chr>      <dbl>    <dbl> <dbl>    <dbl> <dbl>
## 1 setosa      1.45  0.0252  0.159    1.45  10.9
## 2 versicolor  4.34  0.210   0.459    4.5   10.6
## 3 virginica   5.54  0.288   0.537    5.5    9.70
```

Associação entre duas variáveis quantitativas

Para duas variáveis quantitativas, estudamos a associação entre as duas variáveis usando o gráfico de dispersão. Além disso, podemos calcular o coeficiente de correlação linear de Pearson.

```
ggplot(data = df_versicolor) +  
  geom_point(aes(x=petala_comp, y = petala_larg)) +  
  labs(x = 'Comprimento de pétala', y = 'Larga de pétala',  
       title = 'Associação entre duas variáveis quantitativas')
```



Associação entre duas variáveis quantitativas

Também podemos calcular o coeficiente de correlação linear de Pearson. Lembre que se X e Y são duas variáveis quantitativas com valores

X	x_1	x_2	\cdots	x_n
Y	y_1	y_2	\cdots	y_n

Então, o coeficiente de correlação linear é dado por

$$r = \left(\frac{(x_1 - \bar{x}) \cdot (y_1 - \bar{y})}{s_x s_y} \right) + \cdots + \left(\frac{(x_n - \bar{x}) \cdot (y_n - \bar{y})}{s_x s_y} \right)$$

#No R, o cálculo é bem simples

```
with(df_versicolor, cor(petala_comp, petala_larg))
```

```
## [1] 0.7765857
```