

# INTRODUÇÃO À PROGRAMAÇÃO

## UTILIZANDO O R

Aula 2 – Conceitos (não tão) básicos

**Atenção:** estes slides foram pensados para serem usados na aula presencial, então algumas vezes pode ser ruim de usar eles caso você não tenha assistido a aula antes. Por exemplo, alguns dos exercícios foram feitos para dar errado para que se discuta a mensagem de erro na própria aula.



1º Meetup

# R-Ladies Floripa

Sábado, 25 de maio e 01 de junho, das 13h às 18h.

Minicurso:  
Introdução à  
programação  
usando o R e  
o R Studio

Quer aprender a  
programar mas não sabe  
por onde começar?  
Este curso é para você!



Organizadoras locais:



Aishameriane Schmidt  
@AishamerianeS



Camila Weber  
@camilatweber

Apoio: Grupo DOT e Aquarela Analytics



Mais informações e inscrições: <https://www.meetup.com/pt-BR/r-ladies-florianopolis/>

### 3. Recreio



Das 15:00 às 15:25



## *4. Material do curso*





Ou acesse

<https://tinyurl.com/aisha-rladies>

Acesse o site do  
R-Ladies Floripa no  
Meetup!

5. “Sujando” as mãos no código



A close-up photograph showing a person's hands working on a pottery wheel. The hands are covered in wet clay, and they are shaping a small, rounded object. The pottery wheel is made of wood and is spinning rapidly, creating a blurred effect. In the background, there are some green plants and a person's hair. The overall atmosphere is focused and traditional.

*Exercícios práticos*

*Da aula  
anterior...*



# Nós falamos sobre:

---

- A importância de saber programação;
- Alguns recursos para aprender a programar;
- Um panorama do R;
- Recursos para estudar mais sobre R;



# Nós falamos sobre:

- Instalando o R e o R Studio;
- Salvando variáveis (biscoito "<- " e bolacha "=")
- Operações matemáticas (+, -, ÷, ×, etc)
- Tipos de variáveis (numeric, character, integer, logical, complex)
- Vetores e operações com vetores



# *Nós falamos sobre:*

Conceitos e  
Motivação

Familiaridade com as  
ferramentas  
separadamente



*E o que  
vamos fazer  
hoje?*



TO-DO LIST

NOTHING





# AULA 2

- ✓ *Avisos gerais*
- ✓ *Salvando código, limpando workspace*
- ✓ *Vetores (cont.)*
- ✓ *Matrizes*
- ✓ *Condicionais e Loops*
- ✓ *Instalação de pacotes*
- ✓ *R Markdown*
- ✓ *Leitura de dados externos*
- ✓ *Manipulação de data frames*
- ✓ *Gráficos*
- ✓ *Encerramento*



# AULA 2

- ✓ *Avisos gerais*
- ✓ *Salvando código, limpando workspace*
- ✓ *Vetores (cont.)*
- ✓ *Matrizes*
- ✓ *Condicionais e Loops*
- ✓ *Instalação de pacotes*
- ✓ *R Markdown*
- ✓ *Leitura de dados externos*
- ✓ *Manipulação de data frames*
- ✓ *Gráficos*
- ✓ *Encerramento*



Quando seu código aparece coloridinho e com asterisco, ele não está salvo (seja da primeira vez ou das próximas) e você precisa salvar ele na sua máquina.

The screenshot shows the RStudio interface. A yellow arrow points from the text above to the title bar of the code editor window, which displays "Código.R\*" with a red asterisk. Below the title bar, there is a message: "Source on Save" followed by a small icon. The main code editor area contains R script code. At the bottom of the screen, the Windows taskbar is visible with icons for File Explorer, Google Chrome, Microsoft Edge, Mail, and the Start button.

```
1 # Trabalho economia internacional
2 # Aishameriane Schmidt
3 # Maio 2019
4
5 # Lendo os dados
6 dados <- read.csv("D:\\\\Onedrive - Aisha\\\\OneDrive\\\\Documentos\\\\Graduação Economia UDESC\\\\2019-1\\\\Economia
7
8 head(dados)
9 names(dados)[1:2] <- c("Country", "Country_code")
10
11 library(ggplot2)
12 library(Synth)
13 #library(devtools) # Se o SCTools não estiver instalado
14 #install_github("bcastanho/SCTools")
15 library(SCTools)
16
17
18 set.seed(6969)
19
20 summary(dados)
21 summary(dados$ECI_extended[which(dados$Treatment ==1)])
22 summary(dados$ECI_extended[which(dados$Treatment ==0)])
23
24 countries <- data.frame(unique(dados$Country_code), seq(1, to = length(unique(dados$Country_code))))
25 head(countries)
26 colnames(countries) <- c("Country_code", "Country_index")
27 dados2 <- merge(dados, countries, by = "Country_code")
28
```

3:13 (Top Level) R Script

Console Terminal

```
R IS A COLLABORATIVE PROJECT WITH MANY CONTRIBUTORS.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> Y <- TRUE
> class(Y)
[1] "logical"
> |
```

Environment

Global Envir

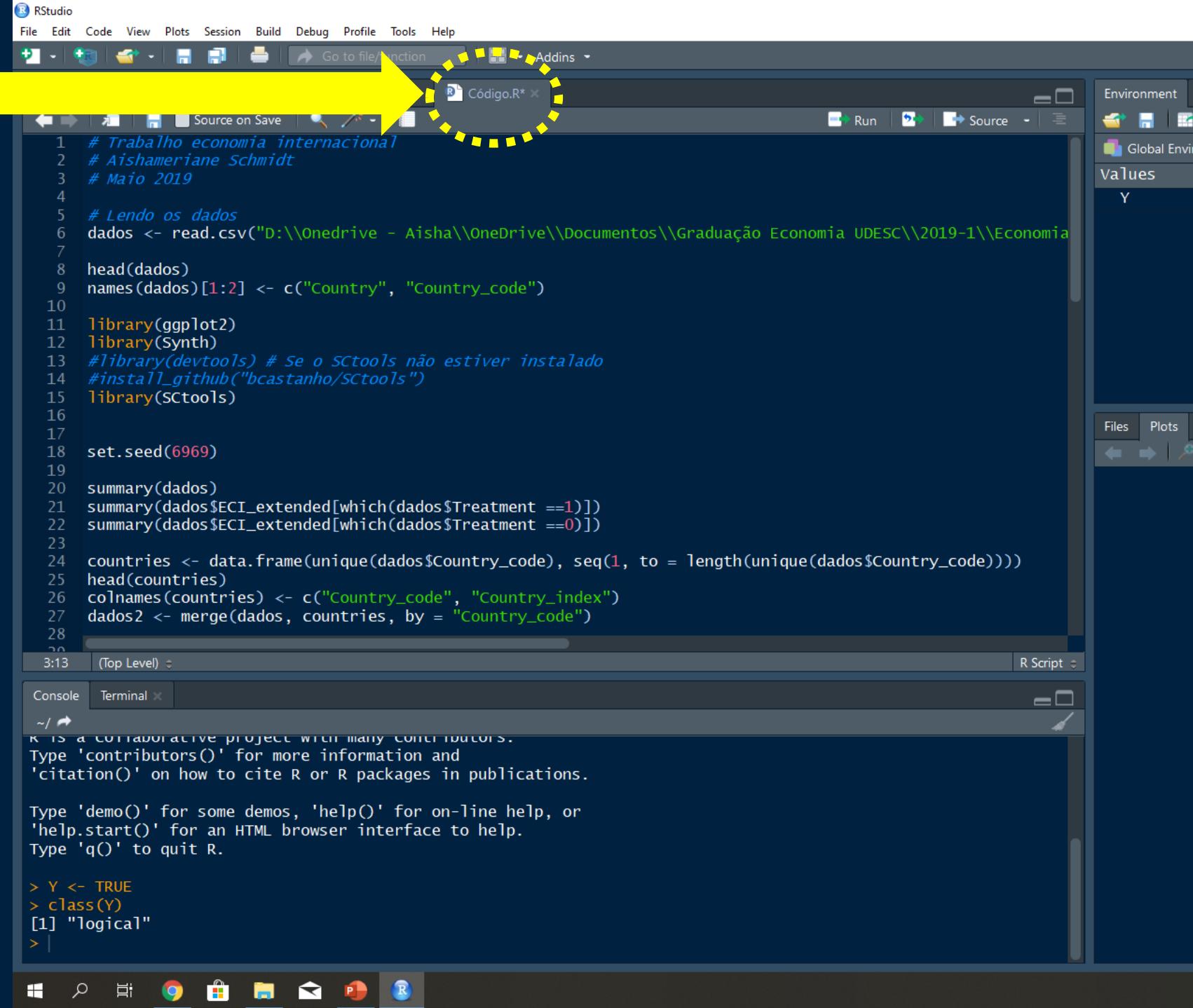
Values

Y

Files Plots

Quando seu código aparece coloridinho e com asterisco, ele não está salvo (seja da primeira vez ou das próximas) e você precisa salvar ele na sua máquina.

→ Caso seja um código já salvo, basta apertar no disquetinho



The screenshot shows the RStudio interface. A large yellow arrow points from the top left towards the save icon in the toolbar. The save icon is a blue disk with a white asterisk (\*). The main code editor window displays an R script titled "Código.R\*". The code itself is written in Portuguese and involves reading a CSV file, loading libraries (ggplot2, Synth, SCTools), setting a seed, summarizing data, creating a countries data frame, and merging it with the main dataset. Below the code editor is the R console, which shows the standard R startup message and a few commands entered by the user, such as `Y <- TRUE` and `class(Y)`.

```
# Trabalho economia internacional
# Aishameriane Schmidt
# Maio 2019

# Lendo os dados
dados <- read.csv("D:\\\\Onedrive - Aisha\\\\OneDrive\\\\Documentos\\\\Graduação Economia UDESC\\\\2019-1\\\\Economia\\dados.csv")

head(dados)
names(dados)[1:2] <- c("Country", "Country_code")

library(ggplot2)
library(Synth)
#library(devtools) # Se o SCTools não estiver instalado
#install_github("bcastanho/SCTools")
library(SCTools)

set.seed(6969)

summary(dados)
summary(dados$ECI_extended[which(dados$Treatment ==1)])
summary(dados$ECI_extended[which(dados$Treatment ==0)])

countries <- data.frame(unique(dados$Country_code), seq(1, to = length(unique(dados$Country_code))))
head(countries)
colnames(countries) <- c("Country_code", "Country_index")
dados2 <- merge(dados, countries, by = "Country_code")
```

```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

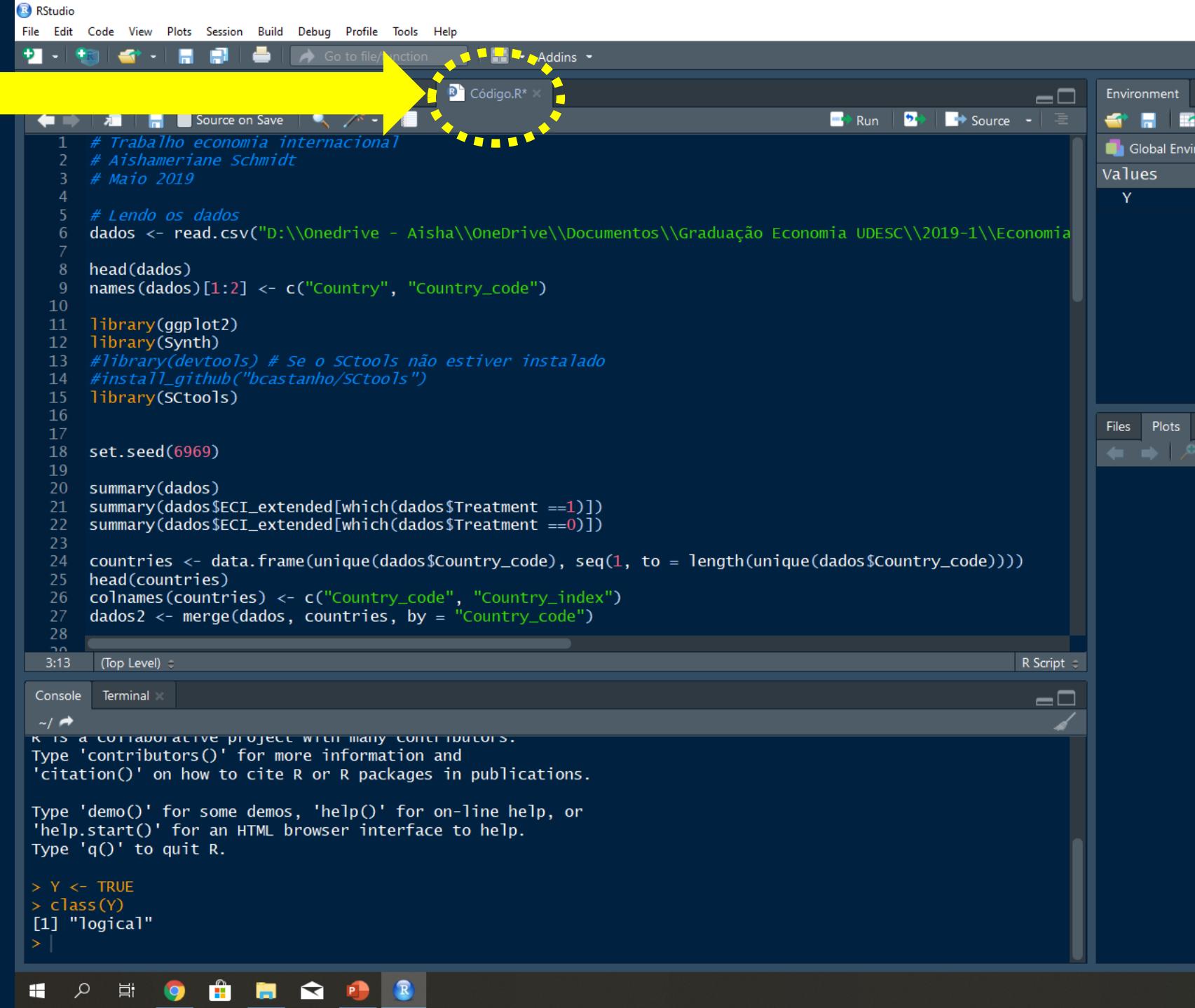
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> Y <- TRUE
> class(Y)
[1] "logical"
>
```

Quando seu código aparece coloridinho e com asterisco, ele não está salvo (seja da primeira vez ou das próximas) e você precisa salvar ele na sua máquina.

→ Caso seja um código já salvo, basta apertar no disquetinho

→ Se for um código novo, escolha a pasta e nome pro arquivo e então escolha a codificação (eu uso geralmente UTF-8, a primeira da lista)



The screenshot shows the RStudio interface. A large yellow arrow points from the top left towards the save icon in the toolbar. The toolbar also includes icons for file operations like Open, Save, Print, and Run. The main workspace shows an R script named "Código.R\*" with the following code:

```
1 # Trabalho economia internacional
2 # Aishameriane Schmidt
3 # Maio 2019
4
5 # Lendo os dados
6 dados <- read.csv("D:\\\\Onedrive - Aisha\\\\OneDrive\\\\Documentos\\\\Graduação Economia UDESC\\\\2019-1\\\\Economia")
7
8 head(dados)
9 names(dados)[1:2] <- c("Country", "Country_code")
10
11 library(ggplot2)
12 library(Synth)
13 #library(devtools) # Se o SCTools não estiver instalado
14 #install_github("bcastanho/SCTools")
15 library(SCTools)
16
17
18 set.seed(6969)
19
20 summary(dados)
21 summary(dados$ECI_extended[which(dados$Treatment ==1)])
22 summary(dados$ECI_extended[which(dados$Treatment ==0)])
23
24 countries <- data.frame(unique(dados$Country_code), seq(1, to = length(unique(dados$Country_code))))
25 head(countries)
26 colnames(countries) <- c("Country_code", "Country_index")
27 dados2 <- merge(dados, countries, by = "Country_code")
28
```

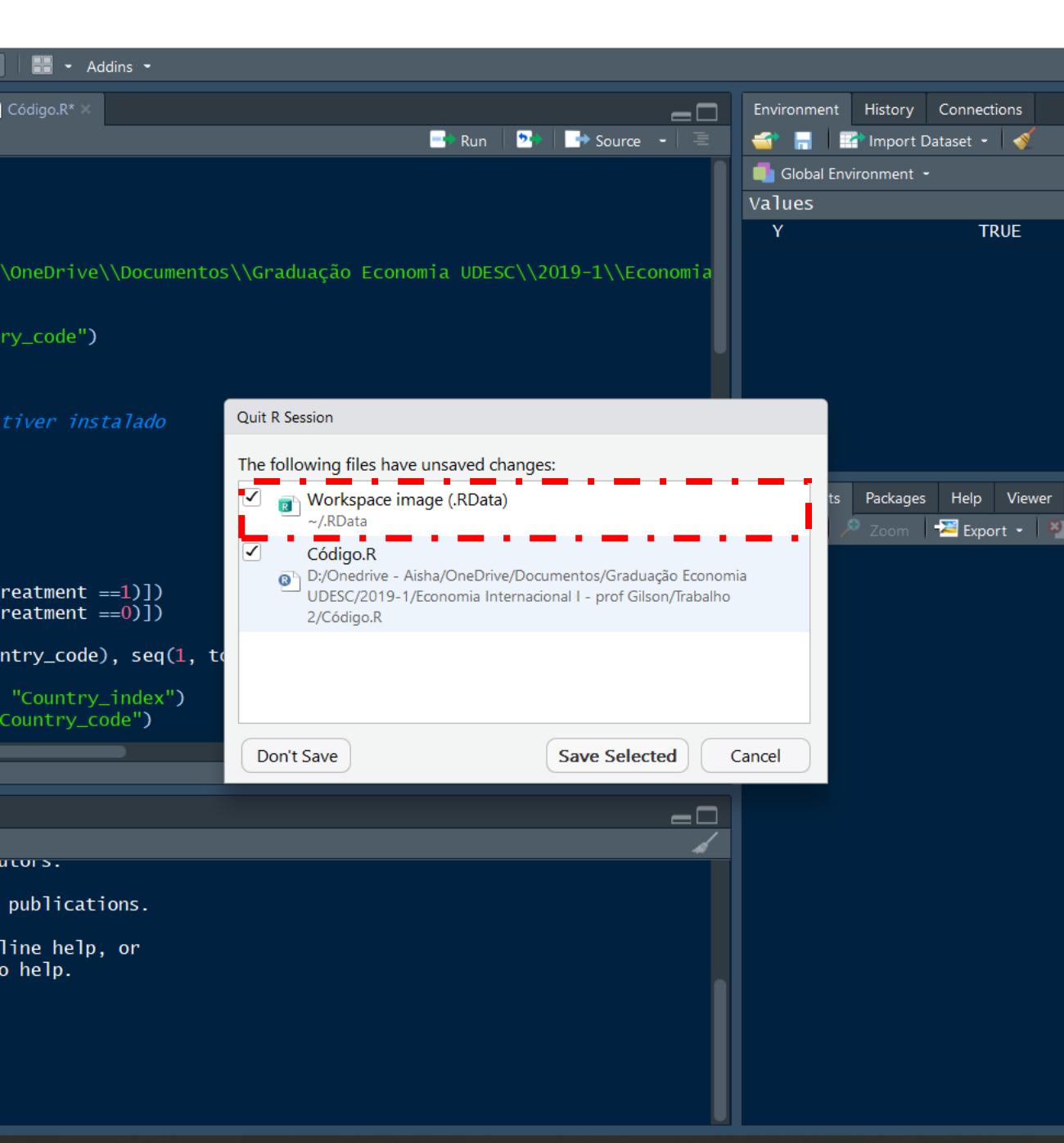
The status bar at the bottom indicates "3:13 (Top Level)". The bottom pane shows the R console output:

```
R IS A COLLABORATIVE PROJECT WITH MANY CONTRIBUTORS.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

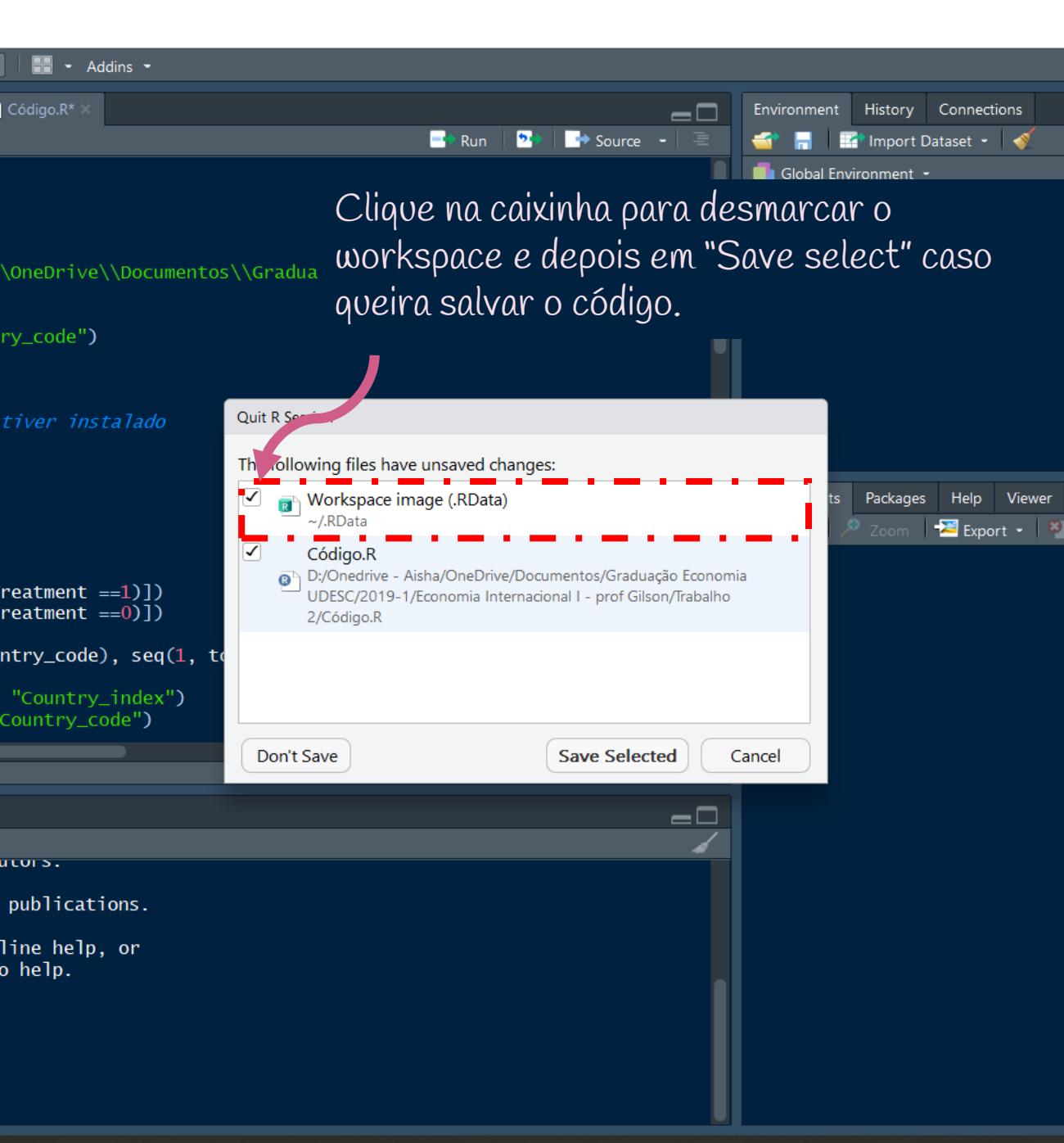
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> Y <- TRUE
> class(Y)
[1] "logical"
>
```

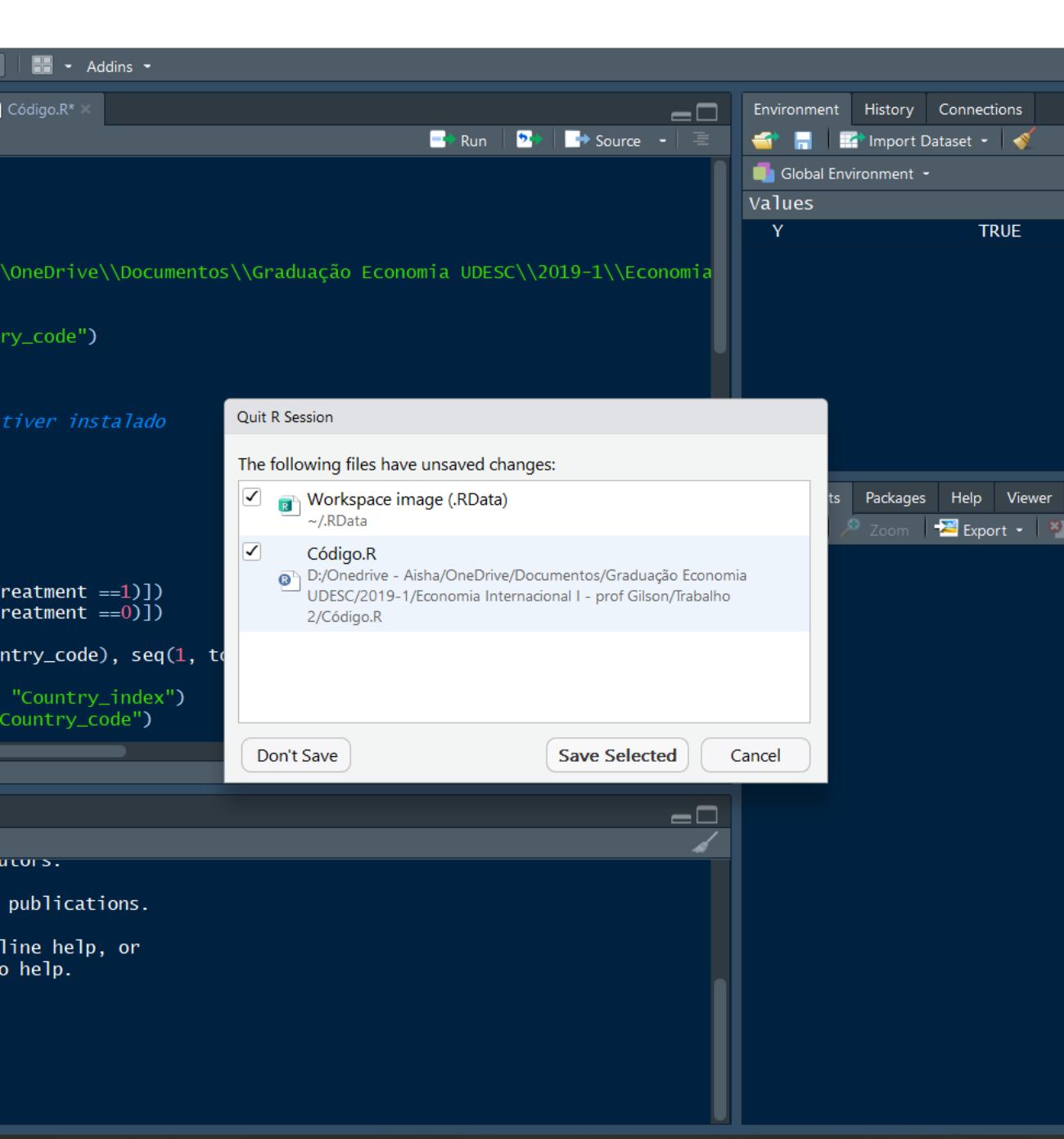
The right side of the interface shows the Environment and Global Environmen tabs, along with a Values section and a Y variable entry field.



Eu não recomendo salvar o workspace (salvo em casos bem específicos), mas se acontecer, toda vez que você abrir seu RStudio, ele vai carregar as variáveis e data frames da sessão anterior que estavam salvos na memória.



Eu não recomendo salvar o workspace (salvo em casos bem específicos), mas se acontecer, toda vez que você abrir seu RStudio, ele vai carregar as variáveis e data frames da sessão anterior que estavam salvos na memória.



Para limpar o workspace que ficou salvo, execute o seguinte comando:

```
unlink(".RData")
```

# AULA 2

- ✓ *Avisos gerais*
- ✓ *Salvando código, limpando workspace*
- ✓ *Vetores (cont.)*
- ✓ *Matrizes*
- ✓ *Loops*
- ✓ *Instalação de pacotes*
- ✓ *R Markdown*
- ✓ *Leitura de dados externos*
- ✓ *Manipulação de data frames*
- ✓ *Gráficos*
- ✓ *Encerramento*

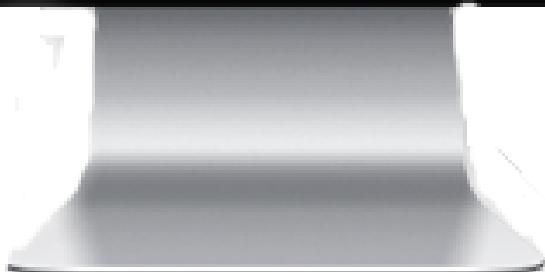


```
# Calculando os gastos na festinha junina  
# Ganhos no jogo da pescaria, por dia da semana (seg a sex)  
pescaria <- c(14, -5, 20, -12, 24)
```

```
# Ganhos no jogo de argola, também por dia  
argola <- c(-2.4, -5, 10, -3.5, 1)
```

```
# Vamos dar nomes para identificar os dias da semana  
dias_da_semana <- c("Seg", "Ter", "Qua", "Qui", "Sex")  
names(pescaria) <- dias_da_semana  
names(argola) <- dias_da_semana
```

```
# Você pode usar a função which() para encontrar valores da seguinte forma  
x <- c("a", "b", "c")  
names(x) <- c("Primeiro", "Segundo", "Terceiro")  
which(names(x) %in% c("Primeiro", "Terceiro"))  
x[which(names(x) %in% c("Primeiro", "Terceiro"))]  
  
# Utilizando isso,  
# Imprima na tela dos ganhos da argola na segunda e quarta feira  
  
# Imprima na tela os ganhos da pescaria de segunda a quinta  
  
# Some os ganhos da pescaria de segunda a quinta (Dica: função sum())  
  
# Calcule o valor total arrecadado por dia (na argola e pescaria) e na semana
```



```
# Imprima na tela dos ganhos da argola na segunda e quarta feira (dias 1 e 3)
argola[which(names(argola) %in% c("Seg", "Qua"))]
```

```
# Imprima na tela os ganhos da pescaria de segunda a quinta (dias 1 a 4)
argola[-which(names(argola) %in% c("Sex"))]
```

```
# Imprima na tela dos ganhos da argola na segunda e quarta feira (dias 1 e 3)
argola[which(names(argola) %in% c("Seg", "Qua"))]

# Imprima na tela os ganhos da pescaria de segunda a quinta (dias 1 a 4)
argola[-which(names(argola) %in% c("Sex"))]
OU

argola[which(names(argola) %in% "Seg")):which(names(argola) %in% "Qui")]
OU

argola[which(names(argola) %in% c("Seg","Qui"))[1]:which(names(argola) %in%
c("Seg","Qui"))[2]]
OU

posicao <- which(names(argola) %in% c("Seg","Qui"))
argola[posicao[1]:posicao[2]]
```

```
# Imprima na tela os ganhos da pescaria de segunda a quinta (dias 1 a 4)
argola[-which(names(argola) %in% c("Sex"))]

# Some os ganhos da pescaria de segunda a quinta
sum(argola[-which(names(argola) %in% c("Sex"))])

# Calcule o valor total arrecadado por dia (na argola e pescaria) e na semana
argola + pescaria

sum(argola + pescaria)
```

# AULA 2

- ✓ *Avisos gerais*
- ✓ *Salvando código, limpando workspace*
- ✓ *Vetores (cont.)*
- ✓ ***Matrizes***
- ✓ *Condicionais e Loops*
- ✓ *Instalação de pacotes*
- ✓ *R Markdown*
- ✓ *Leitura de dados externos*
- ✓ *Manipulação de data frames*
- ✓ *Gráficos*
- ✓ *Encerramento*



## TRABALHANDO COM MATRIZES

- Matrizes são objetos do R e tem duas dimensões, indicadas pelas linhas e pelas colunas

# FUNÇÃO MATRIX

`matrix {base}`

- `matrix` creates a matrix from the given set of values.
- `as.matrix` attempts to turn its argument into a matrix
- `is.matrix` tests if its argument is a (strict) matrix

Fonte: Texto de ajuda da função `rep()`.

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/matrix.html>

# FUNÇÃO MATRIX

```
matrix {base}
```

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/matrix.html>

# FUNÇÃO MATRIX

```
matrix {base}
```

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```

Dados para serem colocados na matriz

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/matrix.html>

# FUNÇÃO MATRIX

```
matrix {base}
```

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```

Número de linhas. Se não informado, será usado 1.

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/matrix.html>

# FUNÇÃO MATRIX

```
matrix {base}
```

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```



Número de colunas. Se não informado, será usado 1.

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/matrix.html>

# FUNÇÃO MATRIX

```
matrix {base}
```

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```



Determina se os números serão dispostos primeiro na coluna ou primeiro na linha.

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/matrix.html>

# FUNÇÃO MATRIX

```
matrix {base}
```

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```

Dá nomes para as linhas e colunas da matrix

Fonte: Texto de ajuda da função rep().

Disponível em: <https://stat.ethz.ch/R-manual/R-devel/library/base/html/matrix.html>

SUJANDO AS MÃOS



# Armazene a seguinte matriz na variável x (use a função `matrix()`)

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

# Agora armazene a seguinte matriz na variável y

$$\begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

# Armazene o vetor (1,2,3) na variável v

```
# Armazene a seguinte matriz na variável x  
x<- matrix(1:9, nrow = 3, ncol=3, byrow=T)  
  
# Agora armazene a seguinte matriz na variável y  
y<- matrix(1:9, nrow = 3, ncol=3)  
  
# Armazene o vetor (1,2,3) na variável v  
v<- c(1,2,3)  
  
# (Bônus) Some os elementos da diagonal principal de x com os da matriz y  
usando o comando diag()  
  
diag(x) + diag(y)
```

Sugestão: imprima x na tela e depois digite apenas diag(x) para entender o que o comando está fazendo.

```
# Some x e y
```

```
# Some x e v
```

```
# Mutiplique x e y usando * e depois usando %*%
```

```
# Crie uma matriz com as letras a, b, c, d, e, f, g, h dispostas em 4 linhas
```

```
# Crie uma matriz com os números de 1 a 4 dispostos em colunas e utilize o comando solve() para invertê-la
```

$$\mathbf{A}^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{\det(\mathbf{A})} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

```
# Some x e y  
x+y  
  
# Some x e v  
x+v  
  
# Mutiplique x e y usando * e depois usando %*%  
  
x*y  
x %*% y  
  
# Crie uma matriz com as letras a, b, c, d, e, f, g, h dispostas em 4 linhas  
matrix(letters[1:8], nrow=4, ncol = 2, byrow=T)  
  
# Crie uma matriz com os números de 1 a 4 dispostos em colunas e utilize o  
comando solve() para invertê-la  
solve(matrix(1:4, nrow=2, ncol=2))
```

Sugestão: calcule a inversa fazendo  
“manualmente” no R e depois compare com a  
inversa da função solve()

```
# Receita da vendas da barraca de pipoca (doce x salgada)
sexta    <- c(27.50, 53.30)
sabado   <- c(34.60, 75.20)
domingo <- c(45.80, 32.40)

# Armazene os vetores na variável receitas

# Declare receitas como matriz

# Utilize o comando rownames() e informe o dia correspondente a cada linha

# Utilize o comando colnames() e informe o tipo de pipoca em cada coluna

# Imprima na tela a matriz receitas
```

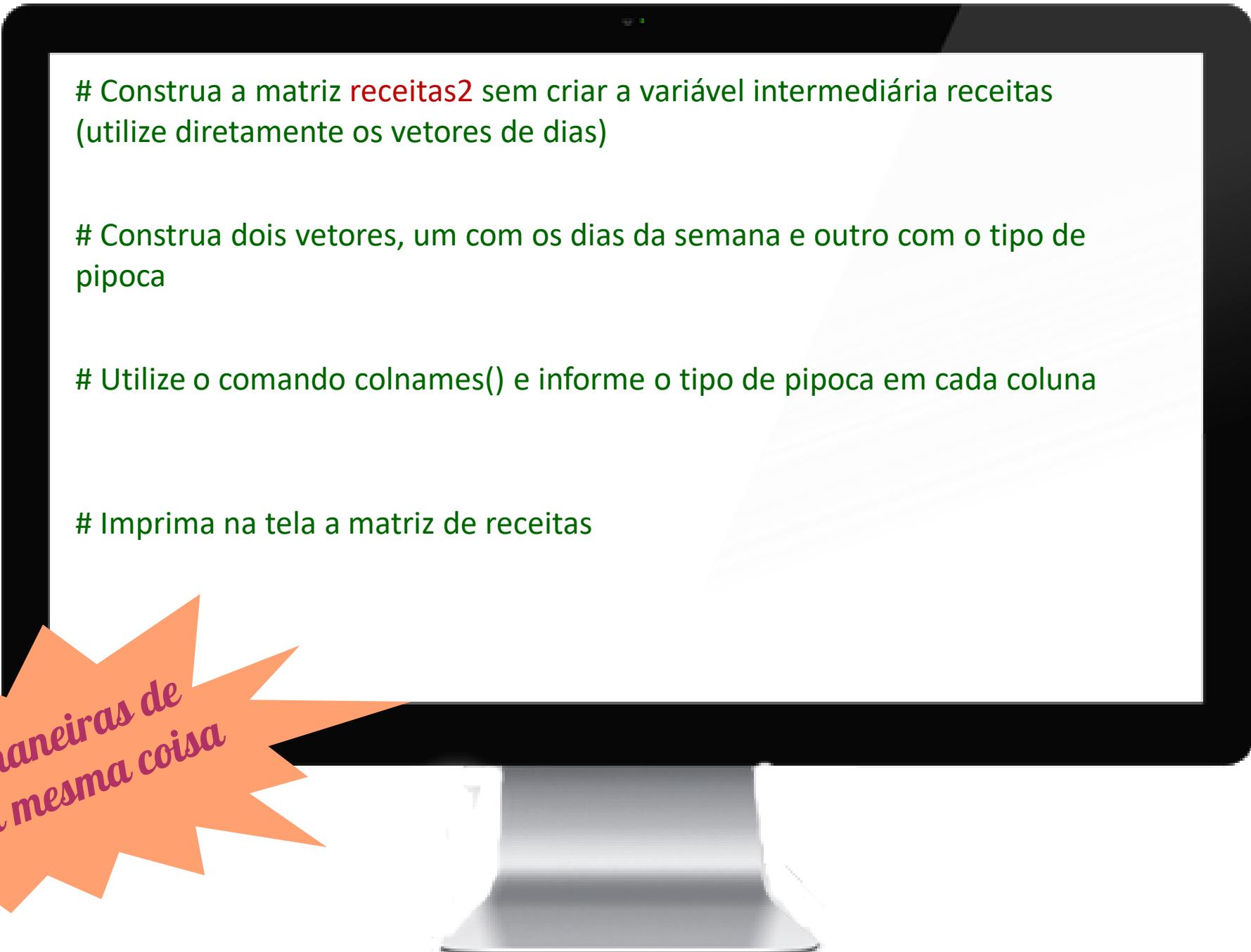
```
# Receita da vendas da barraca de pipoca (doce x salgada)
sexta    <- c(27.50, 53.30)
sabado   <- c(34.60, 75.20)
domingo <- c(45.80, 32.40)

# Armazene os vetores na variavel receitas
receitas <- c(sexta, sabado, domingo)

# Declare receitas como matriz
matrix_receitas <- matrix(receitas, nrow = 3, byrow = T)

# Utilize o comando rownames() e informe o dia correspondente a cada linha
rownames(matrix_receitas) <- c("Sexta", "Sábado", "Domingo")

# Utilize o comando colnames() e informe o tipo de pipoca em cada coluna
colnames(matrix_receitas) <- c("Doce", "Salgada")
# Imprima na tela a matriz de receitas
```



```
# Construa a matriz receitas2 sem criar a variável intermediária receitas  
(utilize diretamente os vetores de dias)
```

```
# Construa dois vetores, um com os dias da semana e outro com o tipo de  
pipoca
```

```
# Utilize o comando colnames() e informe o tipo de pipoca em cada coluna
```

```
# Imprima na tela a matriz de receitas
```

1001 maneiras de  
fazer a mesma coisa

```
# Construa a matriz_receitas2 sem criar a variável intermediária receitas  
# (utilize diretamente os vetores de dias)  
receitas2 <- matrix(c(sexta, sabado, domingo), nrow = 3, byrow = T)  
  
# Construa dois vetores, um com os dias da semana e outro com o tipo de  
# pipoca  
dias_da_semana <- c("Sexta", "Sábado", "Domingo")  
  
# Utilize o comando colnames() e informe o tipo de pipoca em cada coluna  
tipo_pipoca <- c("Doce", "Salgada")  
rownames(receitas2) <- dias_da_semana  
colnames(receitas2) <- tipo_pipoca  
  
# Imprima na tela a matriz receitas2 e compare com a matriz receitas  
receitas2  
receitas == receitas2
```

```
# Salve os dias da semana na variável outra_matriz utilizando o comando  
cbind() e imprima na tela
```

```
# Utilizando o comando matriz_receitas <- matrix(receitas, nrow = 3, byrow =  
T) declare um novo argumento na função chamado dimnames que é igual a  
list(dias_da_semana, tipo_pipoca)
```

1001 maneiras de  
fazer a mesma coisa

```
# Salve os vetores de dias da semana na variável outra_matriz utilizando o comando cbind() e imprima na tela
outra_matriz <- cbind(sexta, sabado, domingo)
outra_matriz
```

# Utilizando o comando `matriz_receitas <- matrix(receitas, nrow = 3, byrow = T)` declare um novo argumento na função chamado `dimnames` que é igual a `list(dias_da_semana, tipo_pipoca)`

```
matriz_receitas <- matrix(receitas, nrow = 3, byrow = T,  
                           dimnames = list(dias_da_semana, tipo_pipoca))
```

ou

```
# Crie a variável receita_por_dia que irá receber o resultado da função  
rowSums() da matriz_receitas e imprima na tela
```

```
# Faça o mesmo para as receitas por tipo de pipoca vendido (procure por  
uma função parecida com rowSums() mas que se aplique às colunas)
```

```
# Junte o vetor de receita_por_dia com a matriz_receitas na nova matriz  
barraquinha.
```

```
# Crie a variável receita_por_dia que irá receber o resultado da função  
rowSums() da matriz_receitas e imprima na tela  
receita_por_dia <- rowSums(matriz_receitas)  
receita_por_dia
```

```
# Faça o mesmo para as receitas por tipo de pipoca vendido (procure por  
uma função parecida com rowSums() mas que se aplique às colunas)
```

```
apropos("Sums")  
receita_por_tipo <- colSums(matriz_receitas)  
receita_por_tipo
```

```
# Junte o vetor de receita_por_dia com a matriz_receitas na nova matriz  
barraquinha
```

```
barraquinha <- cbind(matriz_receitas, receita_por_dia)
```

```
# Você recebeu o faturamento da segunda e terça feira, que foi de (45.7,  
53.8) e (23.4, 45.6) . Salve em dois vetores.
```

```
# Calcule o faturamento de cada dia e inclua em cada vetor
```

```
# Utilize o comando apropriado para juntar essas duas linhas com a matriz  
barraquinha (dica: é parecido com cbind() )
```

```
# Você recebeu o faturamento da segunda e terça feira, que foi de (45.7,  
53.8) e (23.4, 45.6) . Salve em dois vetores.
```

```
segunda <- c(45.7, 53.8)
```

```
terca <- c(23.4, 45.6)
```

```
# Calcule o faturamento de cada dia e inclua em cada vetor
```

```
segunda <- c(segunda, sum(segunda))
```

```
terca <- c(terca, sum(terca))
```

```
# Utilize o comando apropriado para juntar essas duas linhas com a matriz
```

```
barraquinha
```

```
barraquinha <- rbind(barraquinha, segunda, terca)
```

## TRABALHANDO COM MATRIZES

- Podemos “chamar” elementos de matrizes utilizando seu endereço em termos de linhas e/ou colunas.

# TRABALHANDO COM MATRIZES

- Uma matriz  $B_{n \times m}$  pode ser manipulada utilizando os seguintes comandos:
  - $B[i,j]$  - Chama o elemento  $i$  da coluna  $j$

$B[,$

•  $B[i,:]$  - Chama a coluna  $i$

•  $B[:,j]$  - Chama a coluna de  $i$  a  $j$

•  $B[c]$  - Chama a coluna  $c$

# TRABALHANDO COM MATRIZES

- Uma matriz  $B_{n \times m}$  pode ser manipulada utilizando os seguintes comandos:
  - $B[i,j]$  - Chama o elemento  $i$  da coluna  $j$
  - $B[,j]$  - Chama a coluna  $j$
  - $B[i,:]$  - Chama a coluna  $i$
  - $B[:,i]$  - Chama a coluna de  $i$  a  $n$
  - $B[cols]$  - Chama as colunas de  $i$  a  $n$

# TRABALHANDO COM MATRIZES

- Uma matriz  $B_{n \times m}$  pode ser manipulada utilizando os seguintes comandos:
  - $B[i,j]$  - Chama o elemento  $i$  da coluna  $j$
  - $B[,j]$  - Chama a coluna  $j$
  - $B[i, ]$  - Chama a linha  $i$
  - $B[i:n, j:m]$  - Chama a sub-matriz de linhas  $i$  a  $n$  e colunas  $j$  a  $m$
  - $B[cols]$  - Chama a coluna de índice  $cols$

# TRABALHANDO COM MATRIZES

- Uma matriz  $B_{n \times m}$  pode ser manipulada utilizando os seguintes comandos:
  - $B[i,j]$  - Chama o elemento  $i$  da coluna  $j$
  - $B[,j]$  - Chama a coluna  $j$
  - $B[i,]$  - Chama a linha  $i$
  - $B[i:n,]$  - Chama as linhas de  $i$  a  $n$
  - $B[::j]$  - Chama as linhas de 0 a  $n-1$  e a coluna  $j$

## TRABALHANDO COM MATRIZES

- Uma matriz  $B_{n \times m}$  pode ser manipulada utilizando os seguintes comandos:
  - $B[i,j]$  – Chama o elemento  $i$  da coluna  $j$
  - $B[,j]$  – Chama a coluna  $j$
  - $B[i,]$  – Chama a linha  $i$
  - $B[i:n,]$  – Chama as linhas de  $i$  a  $n$
  - $B[c(i,j,k),]$  – Chama as linhas  $i, j$  e  $k$ .

SUJANDO AS MÃOS



```
# Recupere o elemento da coluna 3 e linha 2 da matriz barraquinha
```

```
# Recupere os dois primeiros elementos da linha 1 da matriz barraquinha
```

```
# Recupere os elementos 1 e 3 da linha 2 da matriz barraquinha
```

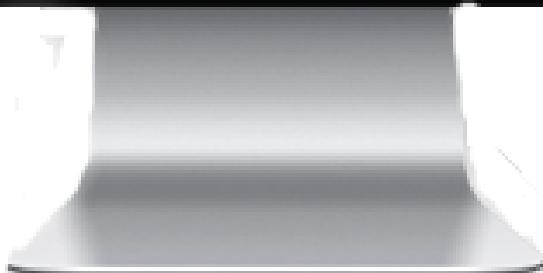
```
# Recupere o elemento da coluna 3 e linha 2 da matriz barraquinha  
barraquinha[2,3]  
  
# Recupere os dois primeiros elementos da linha 1 da matriz barraquinha  
barraquinha[1,1:2]  
  
# Recupere os elementos 1 e 3 da linha 2 da matriz barraquinha  
barraquinha[2,c(1,3)]
```

```
# Salve na variável doce os valores da pipoca doce da barraquinha
```

```
# Calcule o valor total recebido em pipoca doce e compare com o comando  
colSums()
```

```
# Calcule a média do gasto por dia em pipoca doce
```

```
# salve na variável doce os valores da pipoca doce da barraquinha  
doce <- matriz_receitas[,1 ]  
  
# Calcule o valor total recebido em pipoca doce e compare com o comando  
colSums()  
  
sum(doce)  
colSums(matriz_receitas)  
  
# Calcule a média do gasto por dia em pipoca doce usando a função mean()  
mean(doce)
```



# AULA 2

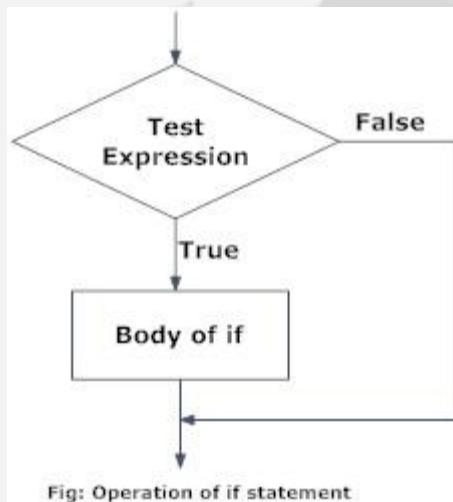
- ✓ *Avisos gerais*
- ✓ *Salvando código, limpando workspace*
- ✓ *Vetores (cont.)*
- ✓ *Matrizes*
- ✓ *Condicionais e Loops*
- ✓ *Instalação de pacotes*
- ✓ *R Markdown*
- ✓ *Leitura de dados externos*
- ✓ *Manipulação de data frames*
- ✓ *Gráficos*
- ✓ *Encerramento*



# ESTRUTURA IF()

```
if (condição) expressão
```

```
if (condição) expressão1 else expressão2
```



```
x<-5  
if(x>0){  
    print("Número Positivo")  
} else {  
    print("Número negativo")}
```

Fonte: <http://www.programiz.com/r-programming/if-else-statement>

# COMPARANDO COISAS

Nome	Descrição
<code>==</code>	Igualdade
<code>&gt;</code>	Maior que
<code>&lt;</code>	Menor que
<code>&gt;=</code>	Maior ou igual
<code>&lt;=</code>	Menor ou igual
<code>!=</code>	Diferente

SUJANDO AS MÃOS



```
# Número de clientes que frequentaram a barraca de pescaria  
barraca <- "Pescaria"  
clientes <- 14
```

```
# Use a função if() para verificar se o valor armazenado na variável barraca é  
Pescaria
```

```
# Faça um teste que verifica se a Pescaria é popular (popularidade é atingida  
quando se tem pelo menos 15 clientes)
```

```
# Número de clientes que frequentaram a barraca de pescaria  
barraca <- "Pescaria"  
clientes <- 14
```

```
# Use a função if() para verificar se o valor armazenado na variável barraca é  
Pescaria
```

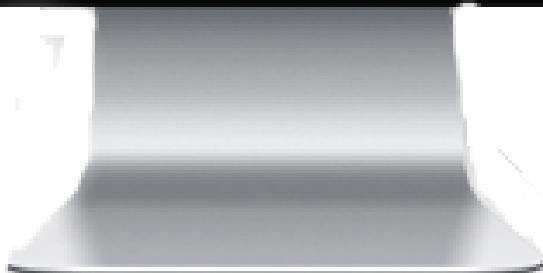
```
if (barraca == "Pescaria") {  
    print("Exibindo informações da Pescaria")  
    clientes  
}
```

```
# Faça um teste que verifica se a Pescaria é popular (popularidade é atingida  
quando se tem pelo menos 15 clientes)
```

```
# Número de clientes que frequentaram a barraca de pescaria
barraca <- "Pescaria"
clientes <- 14

# Use a função if() para verificar se o valor armazenado na variável barraca é
# Pescaria
if (barraca == "Pescaria") {
  print("Exibindo informações da Pescaria")
  clientes
}
# Faça um teste que verifica se a Pescaria é popular (popularidade é atingida
# quando se tem pelo menos 15 clientes)
if (clientes >= 15) {

  print("A pescaria é popular!")
}
```



# Faça um teste para verificar se  $x = 2$  é menor ou igual que  $z = 4$ , em caso positivo, escreva “x é menor que z” na tela.

# Faça um teste para verificar se  $x = 3.14$  é menor que o número pi, em caso positivo, imprima x na tela, em caso negativo, imprima pi na tela. Modifique o valor de x para 5 e refaça o teste.

```
# Faça um teste para verificar se x = 2 é menor ou igual que z = 4, em caso positivo, escreva “x é menor que z” na tela.
```

```
x <- 2
```

```
z <- 4
```

```
if (x <= z) {
```

```
    sprintf("x (%s) é menor que z (%s)", x, z) }
```

```
# Faça um teste para verificar se x = 3.14 é menor que o número pi, em caso positivo, imprima x na tela, em caso negativo, imprima pi na tela. Modifique o valor de x para 5 e refaça o teste.
```

```
x <- 3.14
```

```
pi
```

```
if (x < pi) {
```

```
    x
```

```
} else {
```

```
    pi
```

```
}
```

# Agora incremente o teste de popularidade anterior para comportar as seguintes faixas:

# Mais que 15 clientes é super popular

# Entre 10 e 15 clientes é popularidade média

# Os outros valores são popularidade baixa

Dica: utilize if - else if - else

# Agora incremente o teste de popularidade anterior para comportar as seguintes faixas:

# Mais que 15 likes é super popular

# Entre 10 e 15 likes é popularidade média

# Os outros valores são popularidade baixa

```
if (clientes > 15) {
```

```
    print("Pescaria é popular!")
```

```
} else if (clientes <= 15 & clientes > 10) {
```

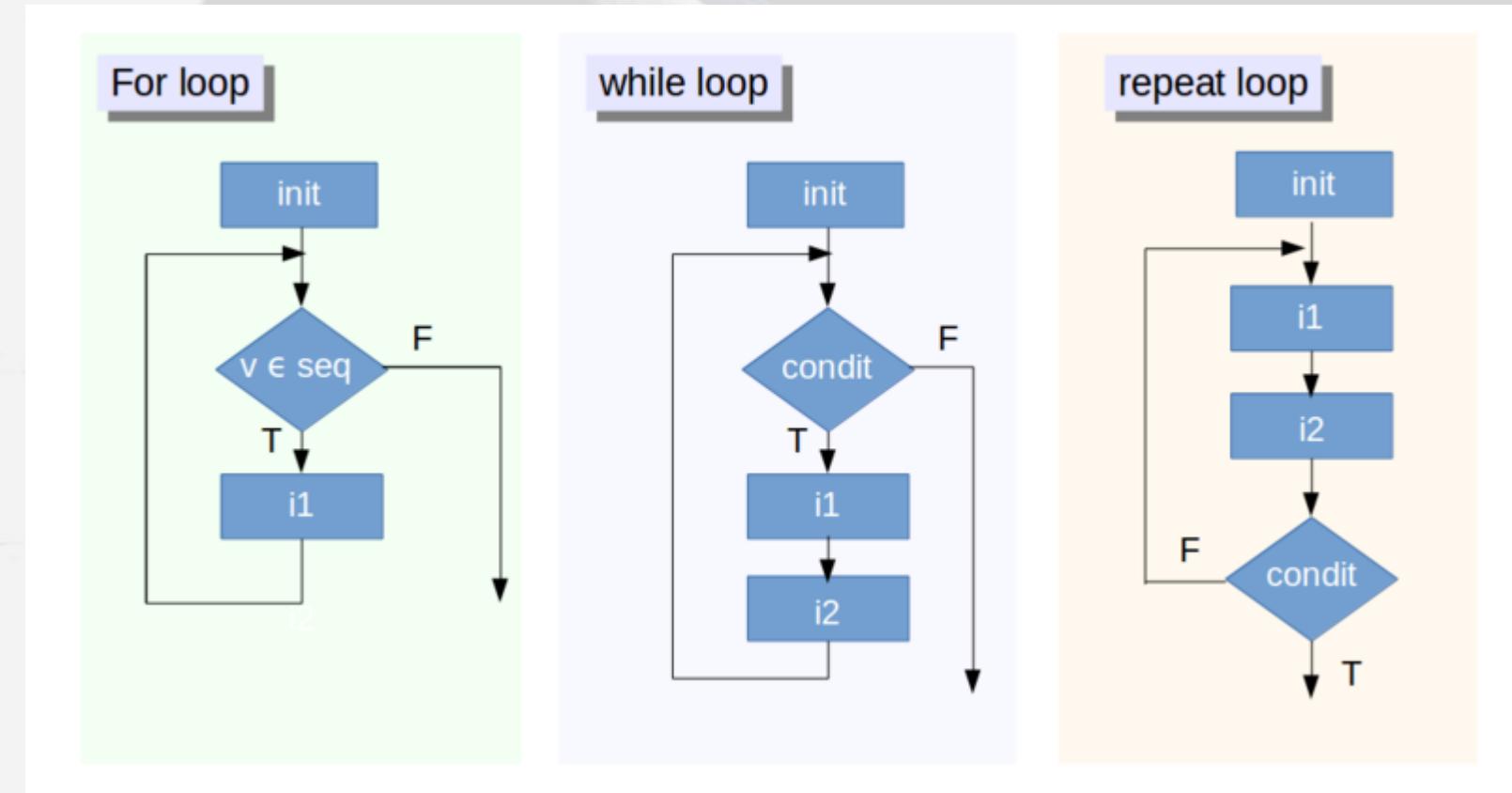
```
    print("O número de clientes está na média")
```

```
} else {
```

```
    print("Tente ser mais popular!")
```

```
}
```

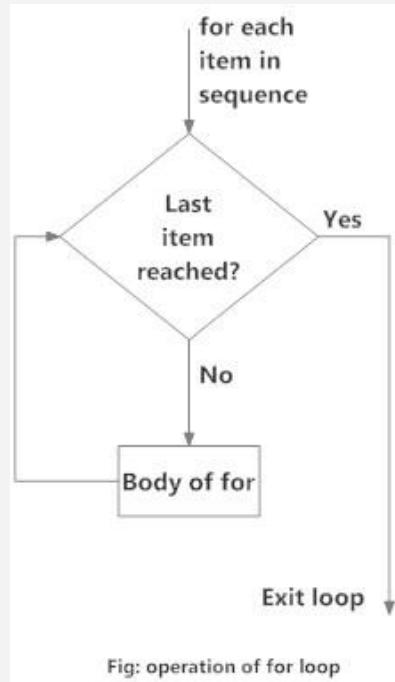
# ESTRUTURAS DE CONTROLE



Fonte: <https://www.datacamp.com/community/tutorials/tutorial-on-loops-in-r>

# ESTRUTURA FOR()

for (variável em um intervalo) expressão



Faça este  
exemplo!

```
For (i in 1:5) {  
  print(i)  
}
```

Fonte: <http://www.programiz.com/r-programming/for-loop>

SUJANDO AS MÃOS



# Escreva um laço que imprima na tela os valores na sequencia que vai de 0 a 1 em pulos de 0.3

# Refaça o exercício anterior porém a cada etapa faça aparecer “Valor do loop” antes de exibir o valor.

# Escreva um laço que imprima na tela os valores na sequencia que vai de 0 a 1 em pulos de 0.3

```
for (ola in seq(0,1,by=0.3)) {  
  print(ola)  
}
```

# Refaça o exercício anterior porém a cada etapa faça aparecer “Valor do loop” antes de exibir o valor. DICA: Procure por cat(), paste() ou sprintf()

# Escreva um laço que imprima na tela os valores na sequencia que vai de 0 a 1 em pulos de 0.3

```
for (ola in seq(0,1,by=0.3)) {  
  print(ola)  
}
```

# Refaça o exercício anterior porém a cada etapa faça aparecer “Valor do loop” antes de exibir o valor. DICA: Procure por cat(), paste() ou sprintf()

```
for (ola in seq(0,1,by=0.3)) {  
  x <- cat("Valor do loop:", ola, "\n")  
  x  
}
```

# Escreva um laço for que escreva na tela as 15 primeiras letras do alfabeto  
DICA: verifique o objeto letters()

# Crie uma matriz x 3x3 toda nula. Armazene as 15 primeiras letras do alfabeto dispostas em linhas na matriz x. Obs: primeiro crie uma matriz com zeros.

# Escreva um laço for que escreva na tela as 15 primeiras letras do alfabeto

DICA: verifique o objeto letters()

```
for (i in 1:15) {  
  print(letters[i])  
}
```

# Crie uma matriz x 3x3 toda nula. Armazene as 15 primeiras letras do alfabeto dispostas em linhas na matriz x. Obs: primeiro crie uma matriz com zeros.

# Escreva um laço for que escreva na tela as 15 primeiras letras do alfabeto

DICA: verifique o objeto letters()

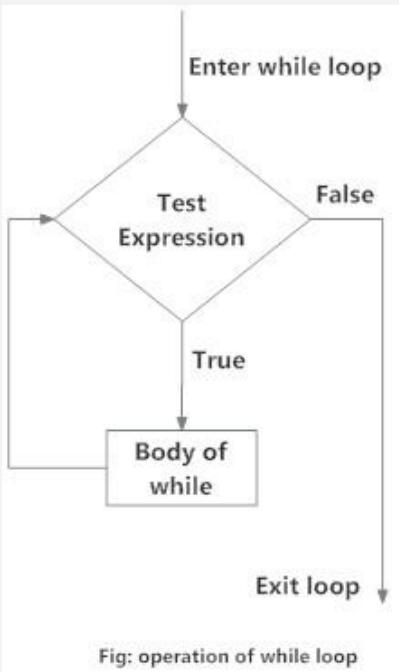
```
for (i in 1:15) {  
  print(letters[i])  
}
```

# Crie uma matriz x 3x3 toda nula. Armazene as 15 primeiras letras do alfabeto dispostas em linhas na matriz x. Obs: primeiro crie uma matriz com zeros.

```
x<- matrix(rep(0,9), nrow=3, ncol=3)  
for (i in 1:nrow(x)) {  
  for (j in 1:ncol(x)) {  
    x[i,j]<-letters[ (i-1) * ncol(x) + j ]  
  }  
}
```

# ESTRUTURA WHILE()

while (condição) expressão



```
x <- 1  
while (x < 5) {  
    print(x)  
    x <- x+1  
}
```

Fonte: <http://www.programiz.com/r-programming/for-loop>

SUJANDO AS MÃOS



```
# Salve na variável x o número 5 e faça um laço while que enquanto x for menor que 10, o R imprima na tela “Olá”.
```

```
# Agora, faça um laço for() indo de 1 a 15 e dentro dele um laço while que verifica se x é maior que a variável de controle do laço for() (por exemplo, i).
```

Se isso ocorrer, é impresso o valor de i na tela. Após, utilize break (para que o while() seja encerrado)

# Salve na variável x o número 5 e faça um laço while que enquanto x for menor que 10, o R imprima na tela “Olá”.

```
x<-5  
while (x < 10) {  
  print("Olá")  
}
```

# Agora, faça um laço for() indo de 1 a 15 e dentro dele um laço while que verifica se x é maior que a variável de controle do laço for(). Se isso ocorrer, é impresso o valor de i na tela. Após, utilize break (para que o while() seja encerrado)

# Salve na variável x o número 5 e faça um laço while que enquanto x for menor que 10, o R imprima na tela “Olá”.

# Agora, faça um laço for() indo de 1 a 15 e dentro dele um laço while que verifica se x é maior que a variável de controle do laço for(). Se isso ocorrer, é impresso o valor de i na tela. Após, utilize break (para que o while() seja encerrado)

```
for (i in 1:15) {  
  while (x > i) {  
    print(i)  
    break  
  }  
}
```

# AULA 2

- ✓ *Avisos gerais*
- ✓ *Salvando código, limpando workspace*
- ✓ *Vetores (cont.)*
- ✓ *Matrizes*
- ✓ *Condicionais e Loops*
- ✓ *Leitura de dados externos*
- ✓ *Instalação de pacotes*
- ✓ *R Markdown*
- ✓ *Manipulação de data frames*
- ✓ *Gráficos*
- ✓ *Encerramento*



## LENDO DADOS EXTERNOS

- O R pode ler dados nos mais variados formatos (csv, txt, xls, ...)
- Existem pacotes para leitura de formatos específicos (arquivos de Matlab, Stata, SPSS, ...)

## LENDO DADOS EXTERNOS

- O primeiro passo é saber onde no computador está o arquivo que será lido, por exemplo:

D:\Onedrive - Aisha\OneDrive\Documentos\R-Ladies\Floripa Chapter\Encontro  
1 - Mini curso - Maio 2019\Material mini curso\Material em PDF\

## LENDO DADOS EXTERNOS

- O primeiro passo é saber onde no computador está o arquivo que será lido, por exemplo:  
D:\Onedrive - Aisha\OneDrive\Documentos\R-Ladies\Floripa Chapter\Encontro  
1 - Mini curso - Maio 2019\Material mini curso\Material em PDF\
- Outra possibilidade é dizer ao R em qual diretório você está trabalhando e com isso apenas informar os nomes dos arquivos, quando necessário  
`getwd()` e `setwd("diretorio")`

# LENDO DADOS EXTERNOS

var1	var2	var3	var4
obs11	obs12	obs13	obs14
obs21	obs22	obs23	obs24
obs31	obs32	obs33	obs34
obs41	obs42	obs43	obs44

Linhas: Observações (casos)

Colunas: Variáveis

# LENDO DADOS EXTERNOS

```
read.csv {utils}  
  
read.csv(file, header = TRUE, sep = ",", quote = "\"\"", dec = ".", fill = TRUE,  
comment.char = "", ...)
```

Se o arquivo está no working directory, basta o nome com o ponto e a extensão, entre aspas. Senão, deve-se colocar o caminho inteiro para o arquivo, utilizando \ ou // na separação de pastas.

read.csv() é um “caso particular” de read.table(). Um bom texto explicativo está aqui:  
<https://www.datacamp.com/community/tutorials/r-tutorial-read-excel-into-r>

# LENDO DADOS EXTERNOS

```
read.csv {utils}  
  
read.csv(file, header = TRUE, sep = ",", quote = "\"\"", dec = ".", fill = TRUE,  
comment.char = "", ...)
```

Por default, a função assume que a primeira linha dos dados é de cabeçalho

# LENDO DADOS EXTERNOS

```
read.csv {utils}  
  
read.csv(file, header = TRUE, sep = “,”, quote = “\””, dec = “.”, fill = TRUE,  
comment.char = “”, ...)
```



A diferença entre o `read.csv` e o `read.csv2` é que o primeiro utiliza a vírgula como separador e o segundo utiliza o ponto e vírgula. Porém isso pode ser alterado neste argumento.

# LENDO DADOS EXTERNOS

```
read.csv {utils}  
  
read.csv(file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill = TRUE,  
comment.char = "", ...)
```



Se o seu arquivo contém aspas ou asteriscos, é necessário informar aqui para que ele não considere como colunas distintas

# LENDO DADOS EXTERNOS

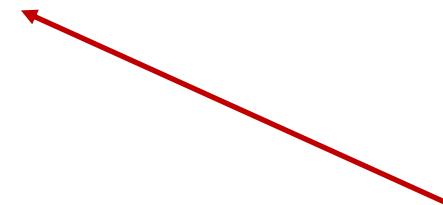
```
read.csv {utils}  
  
read.csv(file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill = TRUE,  
comment.char = "", ...)
```



Este argumento é para informar como é o separador decimal.  
Note que se o seu arquivo é separado por vírgulas, o  
separador decimal deveria ser outro símbolo...

# LENDO DADOS EXTERNOS

```
read.csv {utils}  
  
read.csv(file, header = TRUE, sep = ",", quote = "\"\"", dec = ".",  
fill = TRUE, comment.char = "", ...)
```



Caso existam observações com tamanhos distintos de informação, o R preenche os espaços vazios até ter um tamanho padrão.

# LENDO DADOS EXTERNOS

```
read.csv {utils}  
  
read.csv(file, header = TRUE, sep = ",", quote = "\"", dec = ".",  
fill = TRUE, comment.char = "", ...)
```



Se houverem informações em formato de comentário, usa-se  
essa opção para que o R ignore o que vem depois do caractere

SUJANDO AS MÃOS



```
# Encontre o working directory usando getwd() e em seguida mude-o para a  
pasta onde estão os arquivos que você fez download usando setwd()
```

```
# Leia o arquivo dados1.csv
```

```
# Armazene o conteúdo de dados1.csv na variável dados
```

```
# Encontre o working directory usando getwd() e em seguida mude-o para a  
pasta onde estão os arquivos que você fez download usando setwd()  
  
getwd()  
setwd("C://Downloads")  
  
# Leia o arquivo dados1.csv  
  
read.csv("dados1.csv")  
read.csv("C://Downloads//dados1.csv")  
  
# Armazene o conteúdo de dados1.csv na variável dados  
  
dados <- Read.csv("C://Downloads//dados1.csv")
```

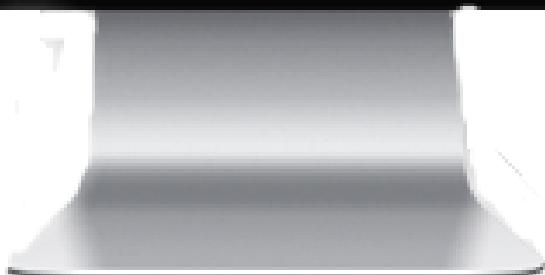
```
# Utilize a função summary para ver o que há na variável dados
```

```
# Verifique a classe do objeto dados
```

```
# Utilize dim() para ver a dimensão do objeto dados
```

```
# Utilize head() para ver as primeiras linhas e tail() para ver as últimas, utilize  
o argumento n=3 para ter apenas 3 observações.
```

```
# Utilize a função summary para ver o que há na variável dados  
summary(dados)  
  
# Verifique a classe do objeto dados  
class(dados)  
  
# Utilize dim() para ver a dimensão do objeto dados  
dim(dados)  
  
# Utilize head() para ver as primeiras linhas e tail() para ver as últimas, utilize  
o argumento n=3 para ter apenas 3 observações.  
head(dados, n=3)  
tail(dados , n=3)
```





```
# Importe novamente os dados, porém utilizando header = FALSE e repita o  
procedimento anterior
```

```
# Utilize a função summary para ver o que há na variável dados
```

```
# Verifique a classe do objeto dados
```

```
# Utilize dim() para ver a dimensão do objeto dados
```

```
# Utilize head() para ver as primeiras linhas e tail() para ver as últimas
```

```
# Importe novamente os dados, porém utilizando head = FALSE e repita o  
procedimento anterior  
dados <- read.csv("C://Downloads//dados1.csv", head = FALSE)  
  
# Utilize a função summary para ver o que há na variável dados  
summary(dados)  
  
# Verifique a classe do objeto dados  
class(dados)  
  
# Utilize dim() para ver a dimensão do objeto dados  
dim(dados)  
  
# Utilize head() para ver as primeiras linhas e tail() para ver as últimas  
head(dados)  
tail(dados)
```

## LENDO DADOS DO EXCEL

- Para leitura de dados em formato xls ou xlsx, precisaremos de funções que não estão nos pacotes básicos do R.
- Ao trabalhar com pacotes, o primeiro passo é fazer a sua instalação

## PACOTES EXTERNOS

- Podem ser instalados diretamente do cran usando o menu ou o comando `install.packages(nomedopacote)` ou a partir de arquivos locais
- Usamos o comando `library(nomedopacote)` para carregá-lo depois de instalado – isso deve ser feito a cada nova sessão do R

# The R Project for Statistical Computing

## Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and Mac OS. To [download R](#), please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

## News

- [The R Journal Volume 9/1](#) is available.
- [R version 3.4.1 \(Single Candle\)](#) has been released on Friday 2017-06-30.
- [R version 3.3.3 \(Another Canoe\)](#) has been released on Monday 2017-03-06.
- [The R Journal Volume 8/2](#) is available.
- [useR! 2017](#) (July 4 - 7 in Brussels) has opened registration and more at <http://user2017.brussels/>
- Tomas Kalibera has joined the R core team.
- The R Foundation welcomes five new ordinary members: Jennifer Bryan, Dianne Cook, Julie Josse, Tomas Kalibera, and Balasubramanian Narasimhan.
- [The R Journal Volume 8/1](#) is available.
- The [useR! 2017](#) conference will take place in Brussels, July 4 - 7, 2017.
- [R version 3.2.5 \(Very, Very Secure Dishes\)](#) has been released on 2016-04-14. This is a rebadging of the quick-fix release 3.2.4-revised.

SUJANDO AS MÃOS



```
# Com o arquivo dados1.xlsx salvo no computador, use a função read_excel()  
para ler o arquivo  
  
read_excel("C://Downloads//dados1.xlsx")
```

```
# Com o arquivo dados1.xlsx salvo no computador, use a função read_excel()  
para ler o arquivo  
  
read_excel("C://Downloads//dados1.xlsx")
```

```
# Podemos instalar pacotes utilizando o comando install.packages("nome")
```

```
# Mesmo que o pacote esteja instalado, é necessário fazer seu carregamento  
cada vez que o R é inicializado
```

```
# Com o arquivo dados1.xlsx salvo no computador, use a função read_excel()  
para ler o arquivo
```

```
read_excel("C://Downloads//dados1.xlsx")
```

```
# Podemos instalar pacotes utilizando o comando install.packages("nome")
```

```
install.packages ("readxl")
```

```
# Mesmo que o pacote esteja instalado, é necessário fazer seu carregamento  
cada vez que o R é inicializado
```

```
library(readxl)
```

# Com o arquivo dados1.xlsx salvo no computador, use a função read\_excel() para ler o arquivo

# Utilize o argumento skip = 1 na função e veja o que acontece

# Utilize o comando dir() (sem argumentos) e depois excel\_sheets("dados1.xlsx")

```
# Com o arquivo dados1.xlsx salvo no computador, use a função read_excel() para  
ler o arquivo
```

```
dados <- read_excel("C://Downloads//dados1.xlsx")
```

```
# Utilize o argumento skip = 1 na função e veja o que acontece
```

```
dados <- read_excel("C://Downloads//dados1.csv", skip = 1)
```

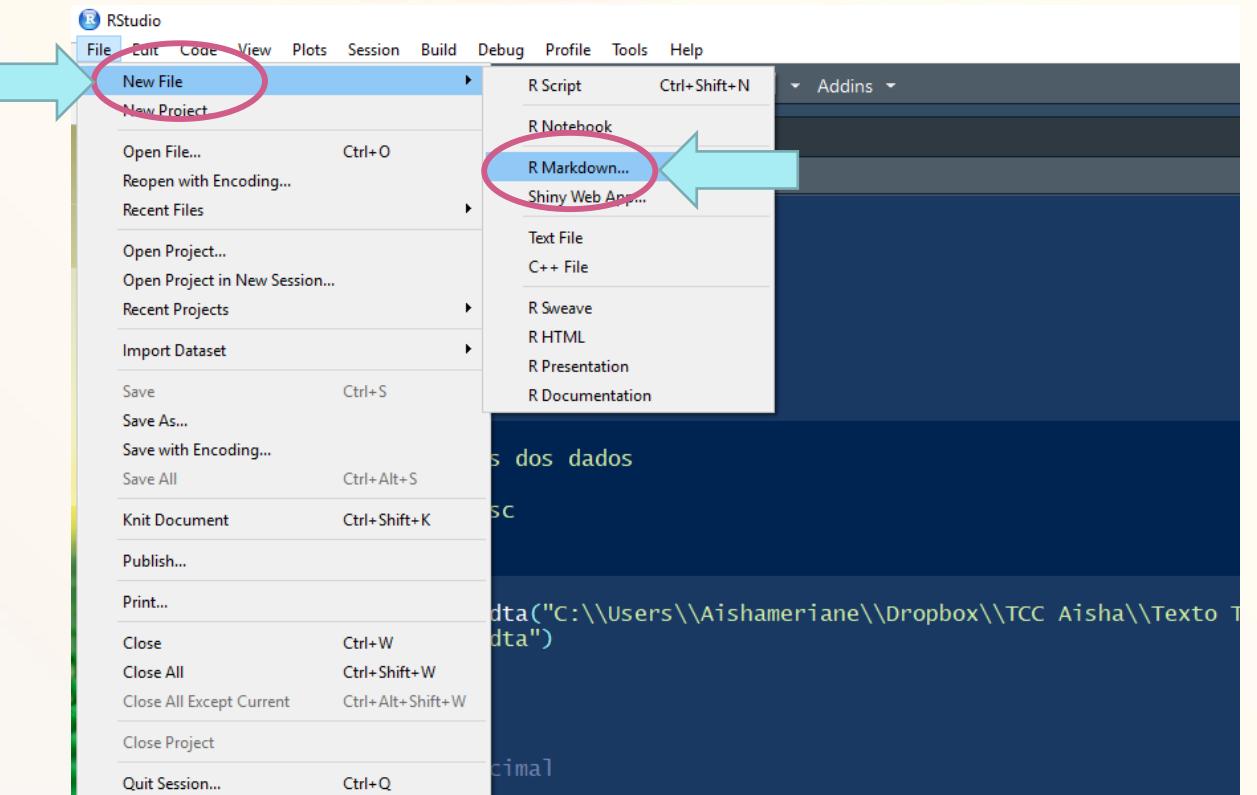
```
# Utilize o comando dir() (sem argumentos) e depois excel_sheets("dados1.xlsx")
```

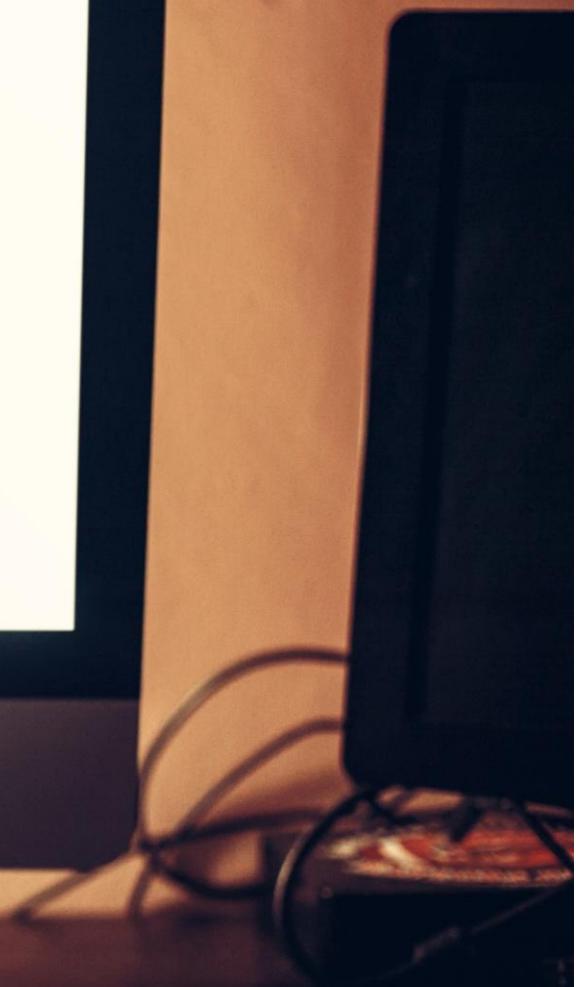
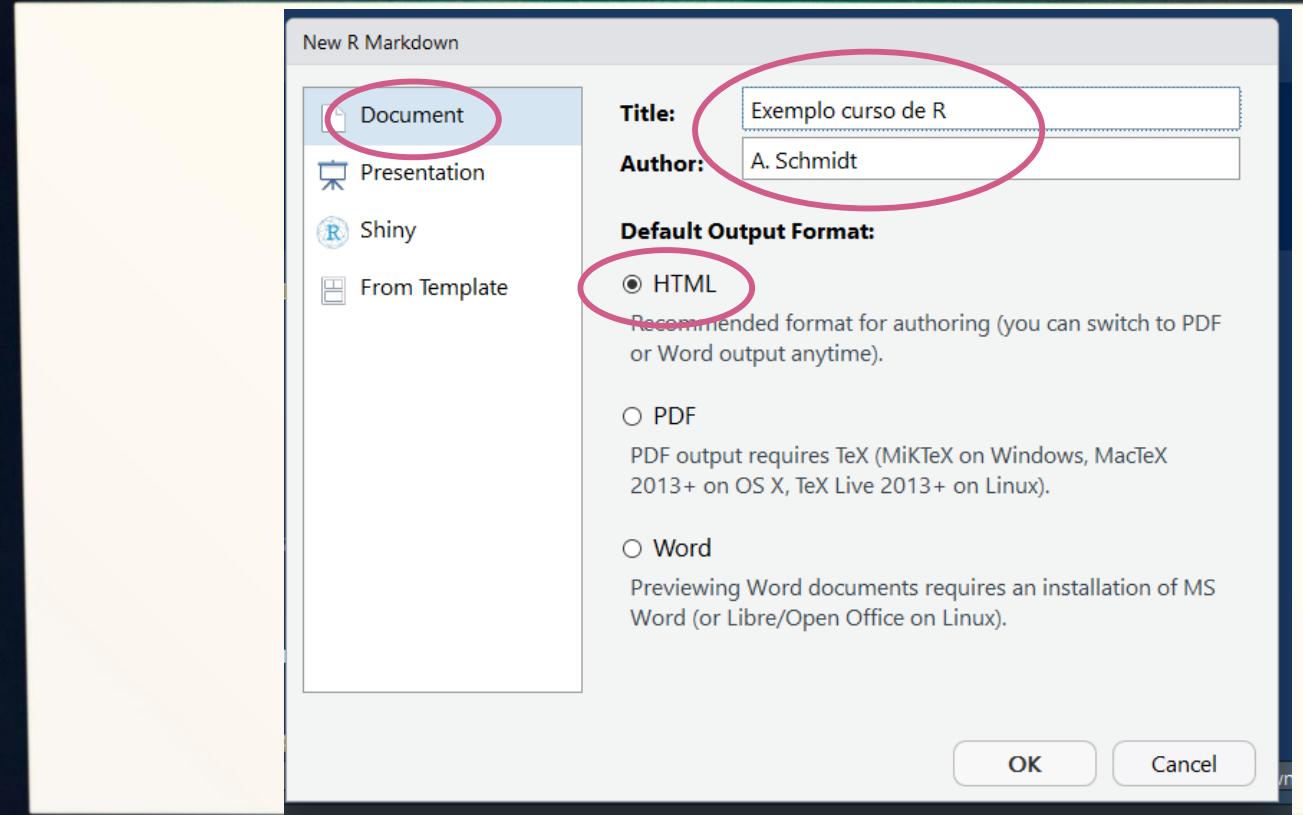
```
# Para importar outras abas de uma mesma planilha, pode-se utilizar o argumento  
sheet = 2 ou sheet = "plan2"
```

# AULA 2

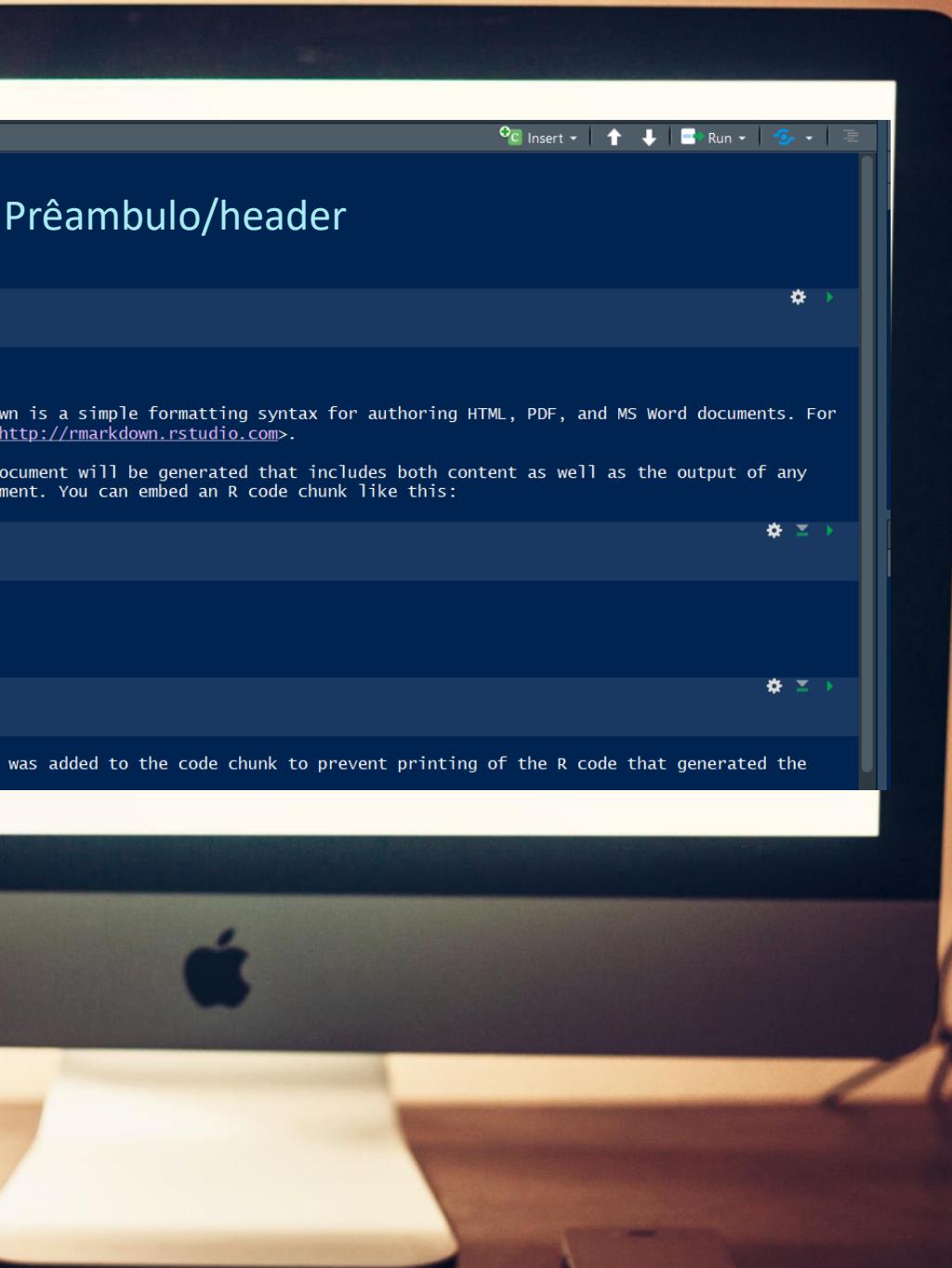
- ✓ *Avisos gerais*
- ✓ *Salvando código, limpando workspace*
- ✓ *Vetores (cont.)*
- ✓ *Matrizes*
- ✓ *Condicionais e Loops*
- ✓ *Leitura de dados externos*
- ✓ *Instalação de pacotes*
- ✓ *R Markdown*
- ✓ *Manipulação de data frames*
- ✓ *Tabelas e Gráficos*
- ✓ *Encerramento*







```
1 ---  
2 title: "Exemplo curso de R"  
3 author: "A. Schmidt"  
4 date: "23 de maio de 2018"  
5 output: html_document  
6 ---  
7  
8 ``{r setup, include=FALSE}  
9 knitr::opts_chunk$set(echo = TRUE)  
10  
11 ## R Markdown  
12  
13 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For  
more details on using R Markdown see <http://rmarkdown.rstudio.com>.  
15  
16 When you click the **Knit** button a document will be generated that includes both content as well as the output of any  
embedded R code chunks within the document. You can embed an R code chunk like this:  
17  
18 ``{r cars}  
19 summary(cars)  
20  
21 ## Including Plots  
22  
23 You can also embed plots, for example:  
24  
25 ``{r pressure, echo=FALSE}  
26 plot(pressure)  
27  
28  
29  
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the  
plot.
```



```
1 *--  
2 |title: "Exemplo curso de R"  
3 |author: "A. Schmidt"  
4 |date: "23 de maio de 2018"  
5 |output: html_document  
6 ---  
7  
8 ``{r setup, include=FALSE}  
9 knitr::opts_chunk$set(echo = TRUE)  
10  
11 ## R Markdown  
12  
13 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For  
more details on using R Markdown see <http://rmarkdown.rstudio.com>.  
15  
16 When you click the **Knit** button a document will be generated that includes both content as well as the output of any  
embedded R code chunks within the document. You can embed an R code chunk like this:  
17  
18 ``{r cars}  
19 summary(cars)  
20  
21 ## Including Plots  
22  
23 You can also embed plots, for example:  
24  
25 ``{r pressure, echo=FALSE}  
26 plot(pressure)  
27  
28  
29  
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the  
plot.
```

## Chunks (blocos de código)

## Chunks (blocos de código)

## Chunks (blocos de código)

## Os blocos de texto são escritos em latex e/ou markdown

```
1 *  
2 title: "Exemplo curso de R"  
3 author: "A. Schmidt"  
4 date: "23 de maio de 2018"  
5 output: html_document  
6 ---  
7  
8 ```{r setup, include=FALSE}  
9 knitr::opts_chunk$set(echo = TRUE)  
10  
11 ## R Markdown  
12  
13 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For  
more details on using R Markdown see <http://rmarkdown.rstudio.com>.  
15  
16 When you click the Knit button a document will be generated that includes both content as well as the output of any  
embedded R code chunks within the document. You can embed an R code chunk like this:  
17  
18 ```{r cars}  
19 summary(cars)  
20  
21 ## Including Plots  
22  
23 You can also embed plots, for example:  
24  
25 ```{r pressure, echo=FALSE}  
26 plot(pressure)  
27  
28  
29  
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the  
plot
```

## Recursos para aprender mais de R Markdown:

<https://rmarkdown.rstudio.com/lesson-1.html>

[https://rmarkdown.rstudio.com/authoring quick tour.html](https://rmarkdown.rstudio.com/authoring_quick_tour.html)

<http://www.botanicaamazonica.wiki.br/labotam/doku.php?id=bot89:precurso:rmarkdown:inicio>

<http://www.jacolienvanrij.com/Tutorials/tutorialMarkdown.html>

<https://bookdown.org/yihui/rmarkdown/>

**Recursos a mais para analisar os Pokémons:**

<https://rpubs.com/mattdray/beginner-r-feat-pokemon>

<https://www.kaggle.com/jonathanbouchet/pokemon-data-clusters>

[https://github.com/ayoubimaya/pokemon/blob/master/Old%20Pokemon%20Dataset/pokemanz\\_01.R](https://github.com/ayoubimaya/pokemon/blob/master/Old%20Pokemon%20Dataset/pokemanz_01.R)

SUJANDO AS MÃOS



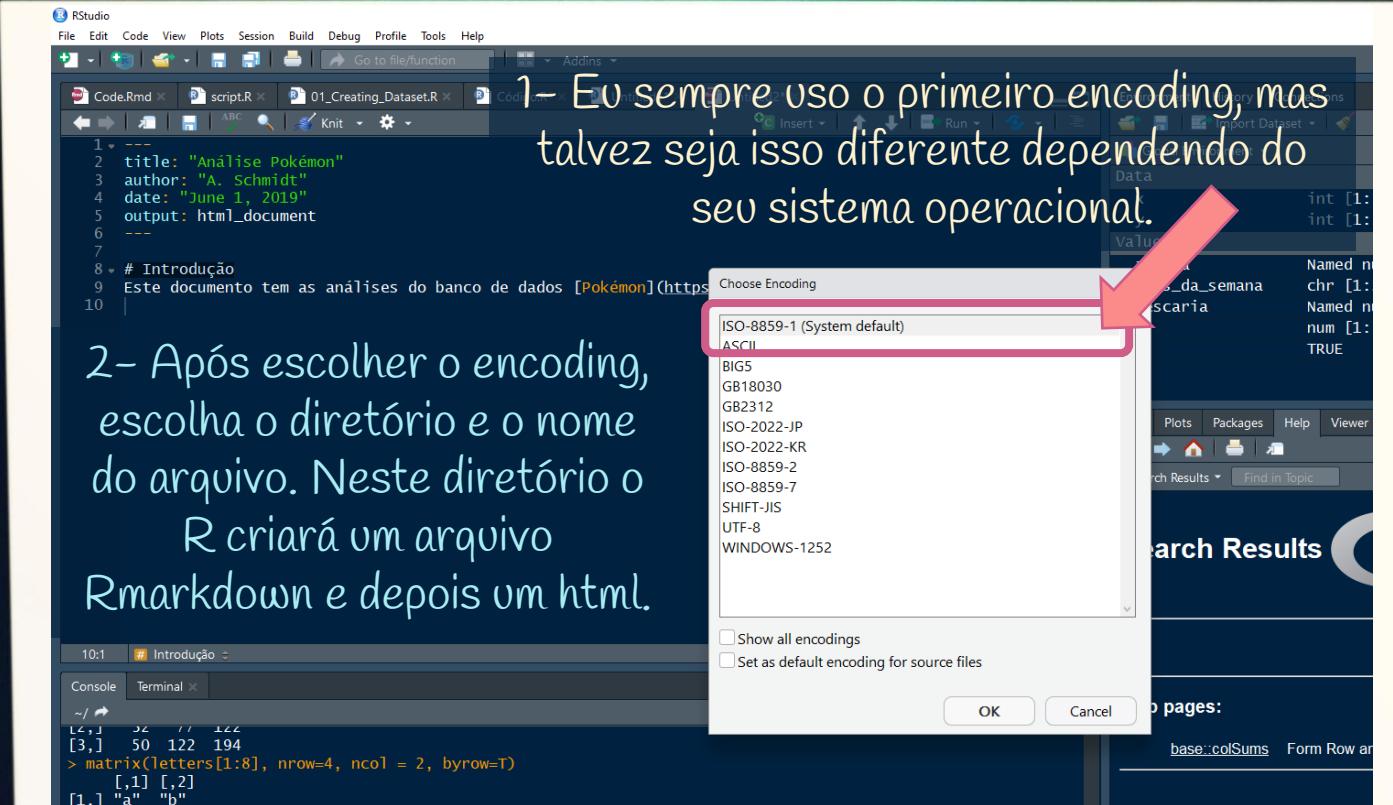
```
# Abra um novo documento R Markown no seu R Studio cuja saída seja html  
# Apague tudo que está abaixo do preâmbulo
```

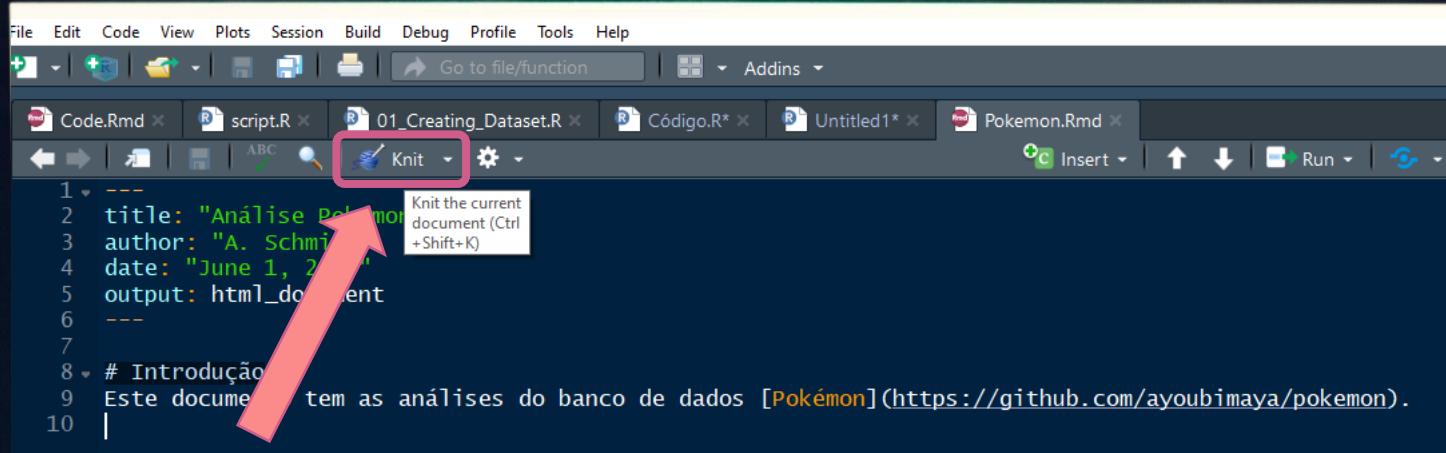
```
---
```

```
title: "Análise Pokémon"  
author: "A. Schmidt"  
date: "June 1, 2019"  
output: html_document
```

```
---
```

```
# Abra um novo documento R Markdown no seu R Studio cuja saída seja html  
# Apague tudo que está abaixo do preâmbulo  
# Você pode escrever como desejar nas partes de fora do código, para isso  
use códigos Markdown: https://www.markdowntutorial.com/.  
# Salve o arquivo  
---  
title: "Análise Pokémon"  
author: "A. Schmidt"  
date: "June 1, 2019"  
output: html_document  
---  
  
# Introdução  
  
Este documento tem as análises do banco de dados  
[Pokémon](https://github.com/ayoubimaya/pokemon)
```





A screenshot of the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with various icons. A red arrow points to the 'Knit' button, which has a blue needle icon and a dropdown menu. A tooltip for the 'Knit' button says 'Knit the current document (Ctrl + Shift + K)'. The main workspace shows an Rmd file named 'Pokemon.Rmd' with the following code:

```
1 ---  
2 title: "Análise Pokémon"  
3 author: "A. Schmid"  
4 date: "June 1, 2019"  
5 output: html_document  
6 ---  
7  
8 # Introdução  
9 Este documento tem as análises do banco de dados [Pokémon] (https://github.com/ayoubimaya/pokemon).  
10 |
```

3 – Aperte no botão "Knit" sempre que quiser gerar o documento (no nosso caso será um html, mas poderia ser pdf ou word). Você poderia fazer isso sem ter salvo o documento primeiro e aí o Rstudio primeiro pede para salvar e depois 'tricota'. Sempre que você apertar Knit, ele salva a versão atual do código.

RStudio

Edit Code View Plots Session Build Debug Profile T

Code.Rmd x script.R x 01\_Creating\_Datasets

```
1 title: "Análise Pokémon"  
2 author: "A. Schmidt"  
3 date: "June 1, 2019"  
4 output: html_document  
5 ---  
6  
7  
8 # Introdução  
9 Este documento tem as análises do  
10
```

D:/Onedrive - Aisha/OneDrive/Documents/R-Ladies/Floripa Chapter/Encontro 1 - Mini curso - Maio 2019/Material mi

Pokémon.html | Open in Browser | Find

# Análise Pokémon

A. Schmidt

June 1, 2019

## Introdução

Este documento tem as análises do banco de dados Pokémon.

4 – O R Studio exibe uma pré-visualização do seu documento html em uma janela própria, mas ele estará salvo no seu computador e pode ser acessado usando qualquer programa compatível também.

The image shows a computer monitor displaying RStudio on the left and a browser window on the right. The RStudio interface shows an Rmd file named 'Code.Rmd' with the following content:

```
1 title: "Análise Pokémon"  
2 author: "A. Schmidt"  
3 date: "June 1, 2019"  
4 output: html_document  
5 ---  
6  
8 # Introdução  
9 Este documento tem as análises do  
10
```

The browser window shows the generated HTML document titled 'Análise Pokémon' by 'A. Schmidt' on 'June 1, 2019'. A callout box points to the title and author information with the text: 'Título, autor e data (estão no preâmbulo)'. Below the title is a section titled 'Introdução' with the text: 'Este documento tem as análises do banco de dados Pokémon.'

RStudio

Edit Code View Plots Session Build Debug Profile Tools

Code.Rmd x script.R x 01\_Creating\_Datasets.R

```
1 ---  
2 title: "Análise Pokémon"  
3 author: "A. Schmidt"  
4 date: "June 1, 2019"  
5 output: html_document  
6 ---  
7  
8 # Introdução ← →  
9 Este documento tem as análises do banco de dados.  
10
```

D:/Onedrive - Aisha/OneDrive/Documents/R-Ladies/Floripa Chapter/Encontro 1 - Mini curso - Maio 2019/Material mi

Pokemon.html | Open in Browser | Find

# Análise Pokémon

A. Schmidt

June 1, 2019

## Introdução

Este documento tem as análises do banco de dados.

Use #, ## e ### para títulos, subtítulos e subsubtítulos

RStudio

Edit Code View Plots Session Build Debug Profile Tools

Code.Rmd script.R 01\_Creating\_Datasets

```
1 ---  
2 title: "Análise Pokémon"  
3 author: "A. Schmidt"  
4 date: "June 1, 2019"  
5 output: html_document  
6 ---  
7  
8 # Introdução  
9 Este documento tem as análises do Pokémon.  
10
```

D:/Onedrive - Aisha/OneDrive/Documents/R-Ladies/Floripa Chapter/Encontro 1 - Mini curso - Maio 2019/Material mi

Pokemon.html Open in Browser Find

# Análise Pokémon

A. Schmidt

June 1, 2019

## Introdução

Este documento tem as análises do banco de dado [Pokémon](#).



Nossos links já  
aparecem bonitinhos :D

# Agora vamos criar um chunk de código R dentro do nosso arquivo Markdown.

# Você precisa abrir ele com três acentos e fechar com três acentos agudos e incluir {r} na abertura, da seguinte forma:

```{r}

```

# Agora vamos criar um chunk de código R dentro do nosso arquivo Markdown.

# Você precisa abrir ele com três acentos e fechar com três acentos agudos e incluir {r} na abertura, da seguinte forma:

```{r}

Tudo que vier aqui deve ser código em R e valem as regras que já aprendemos

```

```
# Nosso primeiro código será para buscar os bancos de dados.  
# Existem dois bancos em formato csv, Pokémon 1 e Pokémon 2, que estão  
disponíveis em https://tinyurl.com/pokemon-rladies-01 e  
https://tinyurl.com/pokemon-rladies-02
```

## # Introdução

Este documento tem as análises do banco de dados  
[Pokémon](<https://github.com/ayoubimaya/pokemon>)

```
```{r}  
Pokemon_01 <- read.csv('https://tinyurl.com/pokemon-rladies-01')  
```
```

```
# Nosso primeiro código será para buscar os bancos de dados.  
# Existem dois bancos em formato csv, Pokémon 1 e Pokémon 2, que estão  
disponíveis em https://tinyurl.com/pokemon-rladies-01 e  
https://tinyurl.com/pokemon-rladies-02
```

### # Introdução

Este documento tem as análises do banco de dados  
[Pokémon](<https://github.com/ayoubimaya/pokemon>)

```
```{r}  
Pokemon_01 <- read.csv('https://tinyurl.com/pokemon-rladies-01')  
```
```

*Faça o mesmo para o segundo arquivo*

```
# Explore os objetos Pokemon_01 e Pokemon_02 usando as funções  
head(), tail(), class() e summary()
```

```
# Explore os objetos Pokemon_01 e Pokemon_02 usando as funções head(),  
tail(), class() e summary()
```

```
```{r}
```

```
Pokemon_01 <- read.csv('https://tinyurl.com/pokemon-rladies-01')  
Pokemon_02 <- read.csv('https://tinyurl.com/pokemon-rladies-02')
```

```
head(Pokemon_01, n = 10)  
head(Pokemon_02, n = 3)
```

```
class(Pokemon_01)  
class(Pokemon_02)
```

```
summary(Pokemon_01)  
summary(Pokemon_02)
```

```
...
```

*Você pode usar o argumento `n =` para ver mais ou menos linhas*

```
# Explore os objetos Pokemon_01 e Pokemon_02 usando as funções head(),  
tail(), class() e summary()  
  
```{r}  
Pokemon_01 <- read.csv('https://tinyurl.com/pokemon-rladies-01')  
Pokemon_02 <- read.csv('https://tinyurl.com/pokemon-rladies-02')  
  
head(Pokemon_01, n = 10)  
head(Pokemon_02, n = 3)  
  
class(Pokemon_01)  
class(Pokemon_02)  
  
summary(Pokemon_01)  
summary(Pokemon_02)  
```
```

*Automaticamente ele salvou  
como data frame, então  
diferente de uma matriz, as  
colunas podem ser de tipos  
diferentes*

```
# Explore os objetos Pokemon_01 e Pokemon_02 usando as funções head(),  
tail(), class() e summary()  
```{r}  
Pokemon_01 <- read.csv('https://tinyurl.com/pokemon-rladies-01')  
Pokemon_02 <- read.csv('https://tinyurl.com/pokemon-rladies-02')  
  
head(Pokemon_01, n = 10)  
head(Pokemon_02, n = 3)  
  
class(Pokemon_01)  
class(Pokemon_02)  
  
summary(Pokemon_01)  
summary(Pokemon_02)  
...``
```

*Depende se a variável é  
numérica, string ou fator.*

## JUNTANDO BANCOS DE DADOS

- Agora queremos ter todos os dados em um mesmo objeto.



## JUNTANDO BANCOS DE DADOS

- Agora queremos ter todos os dados em um mesmo objeto.
- Para isso, precisamos “juntá-los”, mas de forma que as características do Bulbassauro não se misturem com os atributos do Togepi.

## JUNTANDO BANCOS DE DADOS

- Agora queremos ter todos os dados em um mesmo objeto.
- Para isso, precisamos “juntá-los”, mas de forma que as características do Bulbassauro não se misturem com os atributos do Togepi.
- A função `merge()` permite juntar data frames que têm uma chave em comum identificando os casos unicamente.

# JUNTANDO BANCOS DE DADOS

```
base {base}
```

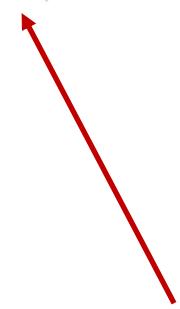
```
merge(x, y, ...)
```

*Os dois data frames ou objetos que queremos juntar*

# JUNTANDO BANCOS DE DADOS

```
base {base}
```

```
merge(x, y, ...)
```



Veja o `help` para as demais opções. Hoje iremos usar o argumento “`by`” para indicar a variável que contém a chave

SUJANDO AS MÃOS



```
# A função merge() permite juntar data frames  
# Utilize o argumento by para selecionar a coluna que contém a chave  
identificadora (utilize aspas no nome desta coluna)  
# Explore o novo data frame da mesma forma como você fez nos outros.  
# Tem algo que te incomoda?
```

```{r}

```

```
# A função merge() permite juntar data frames  
# Utilize o argumento by para selecionar a coluna que contém a chave  
identificadora (utilize aspas no nome desta coluna)
```

```
```{r}
```

```
Pokemon <- merge(Pokemon_01, Pokemon_02, by = "ID")  
head(Pokemon)  
class(Pokemon)  
summary(Pokemon)
```

```
```
```



# Vamos limpar nosso banco de dados removendo a coluna que está duplicada e renomeando a coluna de nome

```{r}

Pokemon <- merge(Pokemon\_01, Pokemon\_02, by = "ID")

head(Pokemon[, -7])

Pokemon <- Pokemon[, -7]  
head(Pokemon)

```

*Primeiro teste se o que você quer fazer irá acontecer mesmo!*

```
# Vamos limpar nosso banco de dados removendo a coluna que está  
duplicada e renomeando a coluna de nome
```

```
```{r}
```

```
Pokemon <- merge(Pokemon_01, Pokemon_02, by = "ID")
```

```
head(Pokemon[, -7])
```

```
Pokemon <- Pokemon[, -7]
```

```
head(Pokemon)
```

```
```
```



*Só então sobrescreva o objeto!*

```
# Vamos limpar nosso banco de dados removendo a coluna que está  
duplicada e renomeando a coluna de nome
```

```
```{r}
```

```
Pokemon <- merge(Pokemon_01, Pokemon_02, by = "ID")
```

```
head(Pokemon[, -7])
```

```
Pokemon <- Pokemon[, -7]
```

```
head(Pokemon)
```

```
```
```

*E dê uma conferida para ver se  
deu certo ;)*

```
# Vamos limpar nosso banco de dados removendo a coluna que está  
duplicada e renomeando a coluna de nome
```

```
```{r}  
Pokemon <- merge(Pokemon_01, Pokemon_02, by = "ID")  
  
head(Pokemon[, -7])  
Pokemon <- Pokemon[, -7]  
head(Pokemon)  
  
names(Pokemon)[2] <- "Nome"  
head(Pokemon)  
```
```

*Não precisamos renomear todas as colunas, somente a segunda. Isto é, queremos que o segundo valor de names (Pokemon) seja Nome*

## TABELAS EM R E RMARKDOWN

- Muitas vezes queremos olhar nossos dados em tabelas, para fazer cruzamentos dos nossos dados.
- Existem muitos pacotes para fazer tabelas “bonitas”, como o Kaggle e Stargazer (que é compatível com Latex).
- Também podemos usar nosso bom e velho pacote base para começar a manipular os dados.

## Recursos adicionais para montar tabelas:

<https://www.r-statistics.com/2013/07/tailor-your-tables-with-stargazer-new-features-for-latex-and-text-output/>

<https://rmarkdown.rstudio.com/lesson-7.html>

[https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome\\_table\\_in\\_html.html](https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_html.html)

<https://bookdown.org/yihui/rmarkdown/tufte-tables.html>

```
# A função table() consegue cruzar duas variáveis categóricas  
# Vamos fazer o cruzamento entre as variáveis Tipo1 e Lendario, para ver  
quais os Tipos mais frequentes entre os Pokemons lendários
```

```
```{r}
```

```
```
```

```
# A função table() consegue cruzar duas variáveis categóricas  
# Vamos fazer o cruzamento entre as variáveis Tipo1 e Lendario, para ver  
quais os Tipos mais frequentes entre os Pokemons lendários
```

```
```{r}
```

```
table(Pokemon$Tipo1, Pokemon$Lendario)
```

```
```
```

# Podemos salvar esta tabela em uma variável e então usar a função prop.table() para montar a tabela de frequências e descobrir que 21% dos Pokémons lendários são do tipo Dragão!

```
```{r}
```

```
Tabela <- table(Pokemon$Tipo1, Pokemon$Lendario)  
round(prop.table(Tabela, 2),2)
```

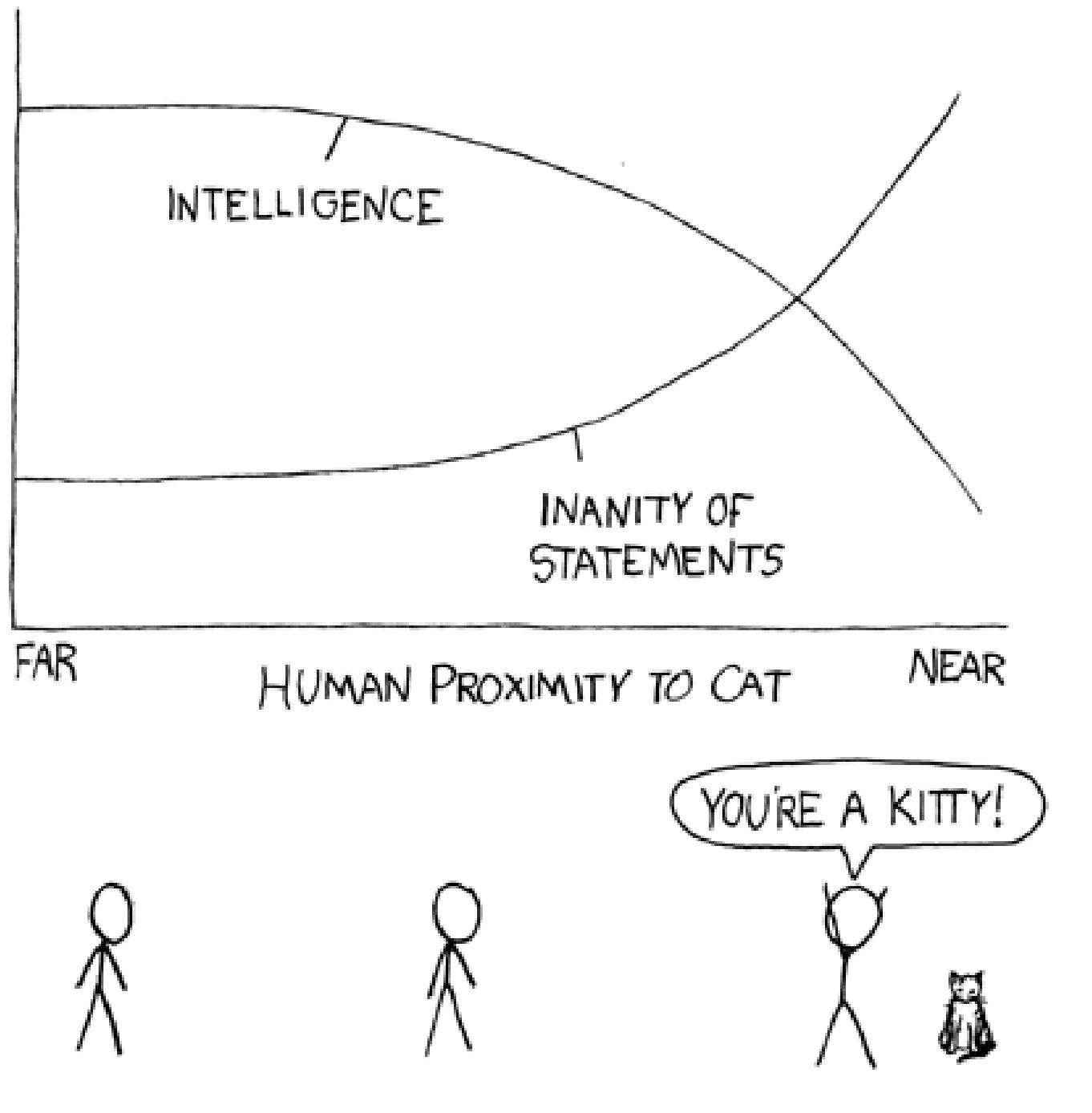
```
```
```

# Podemos salvar esta tabela em uma variável e então usar a função prop.table() para montar a tabela de frequências e descobrir que 21% dos Pokémons lendários são do tipo Dragão!

```
```{r}
```

```
Tabela <- table(Pokemon$Tipo1, Pokemon$Lendario)  
round(prop.table(Tabela, 2),2)  
```
```

*round () arredonda para o número de casas informado no segundo argumento*

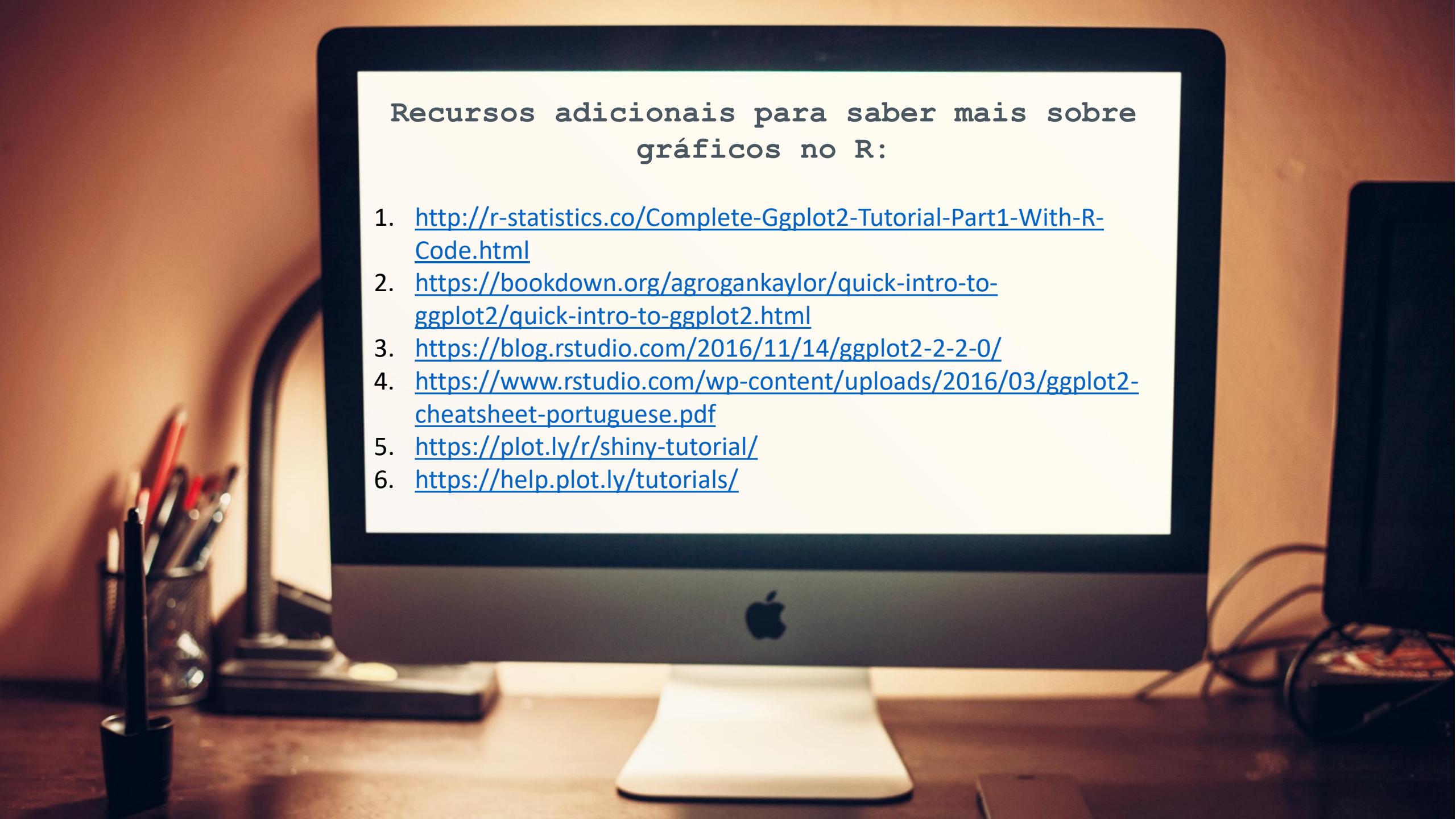


# Gráficos!

Tirinha do xkcd.  
Disponível em: <https://xkcd.com/231/>

## GRÁFICOS EM R

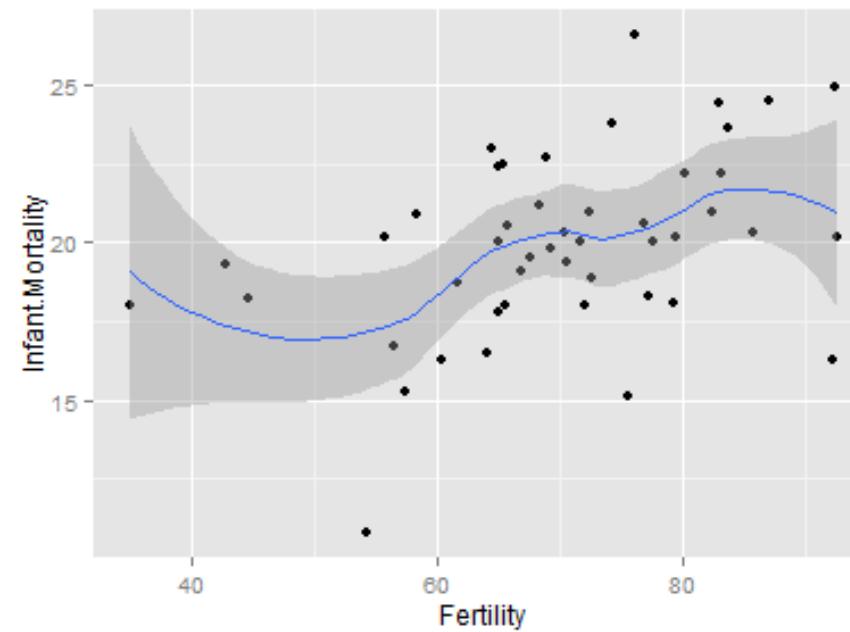
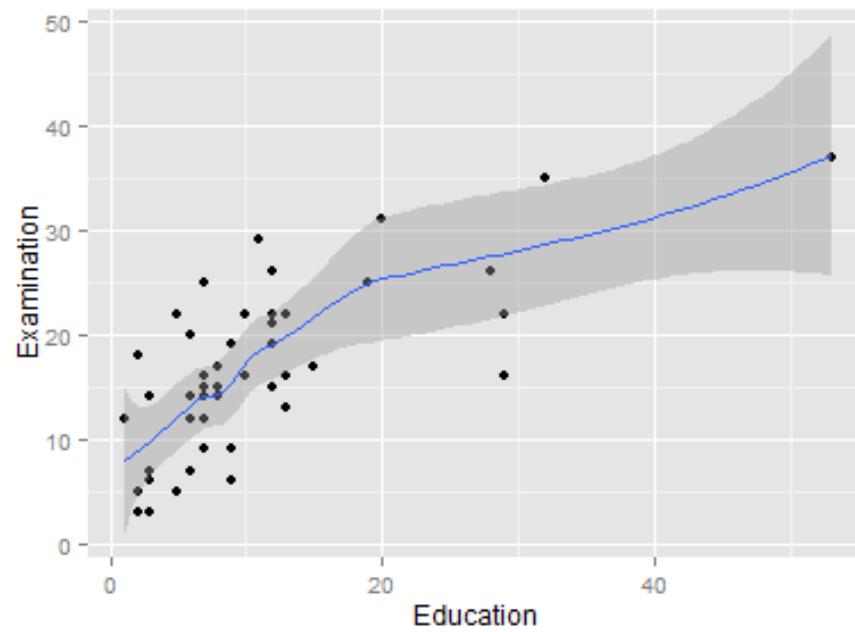
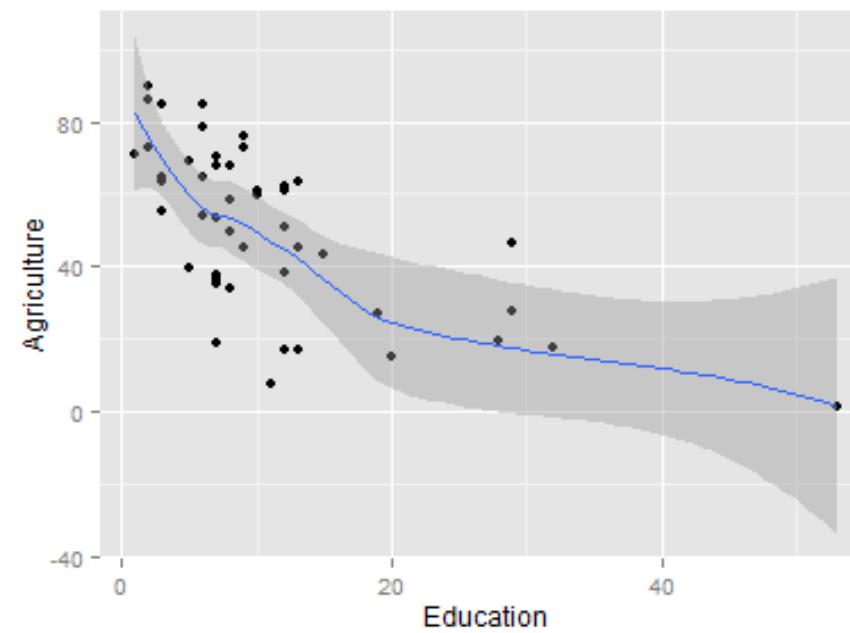
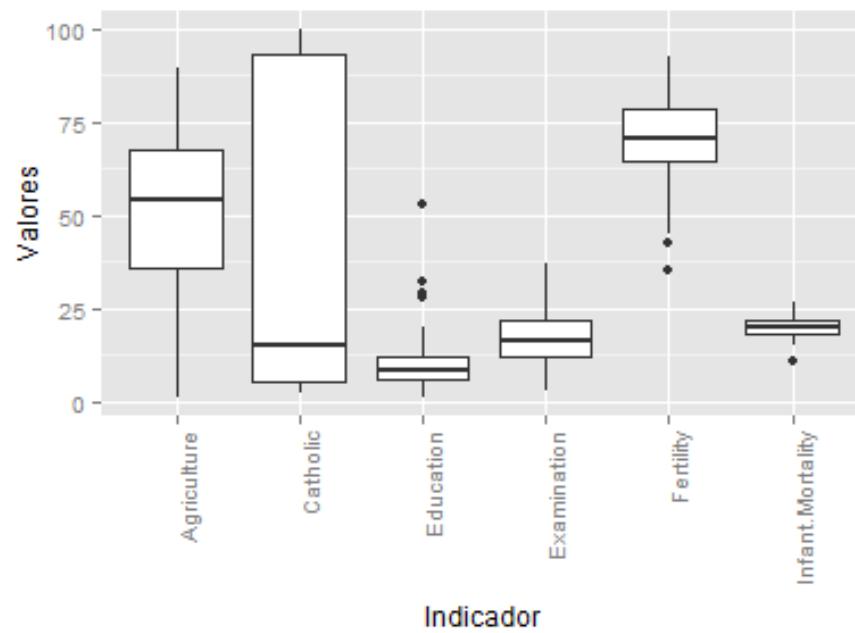
- O pacote base do R tem a função `plot()` que faz um gráfico simples. Também temos a função `hist()` para histogramas e `boxplot()` para boxplots.
- Para além disso, o pacote `ggplot2` abre muitas possibilidades de gráficos.
- Gráficos interativos podem ser feitos em `shiny` ou `plotly`.



## Recursos adicionais para saber mais sobre gráficos no R:

1. <http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html>
2. <https://bookdown.org/agrogankaylor/quick-intro-to-ggplot2/quick-intro-to-ggplot2.html>
3. <https://blog.rstudio.com/2016/11/14/ggplot2-2-2-0/>
4. <https://www.rstudio.com/wp-content/uploads/2016/03/ggplot2-cheatsheet-portuguese.pdf>
5. <https://plot.ly/r/shiny-tutorial/>
6. <https://help.plot.ly/tutorials/>

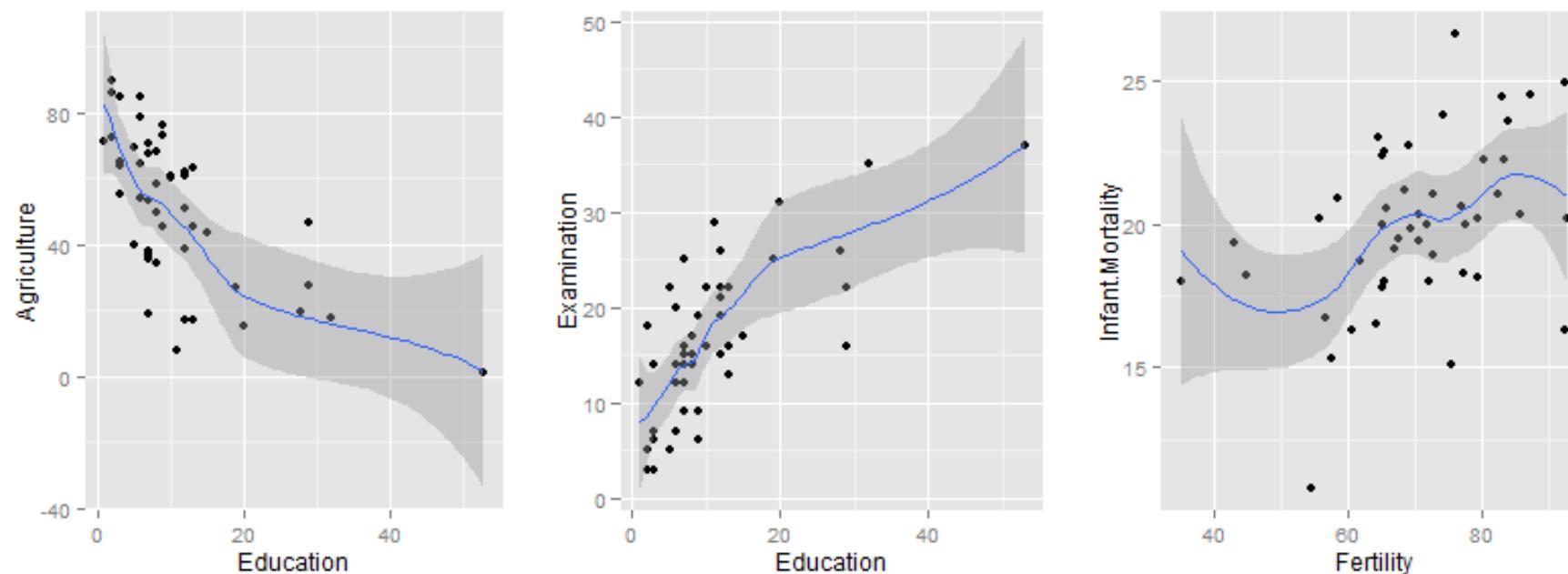
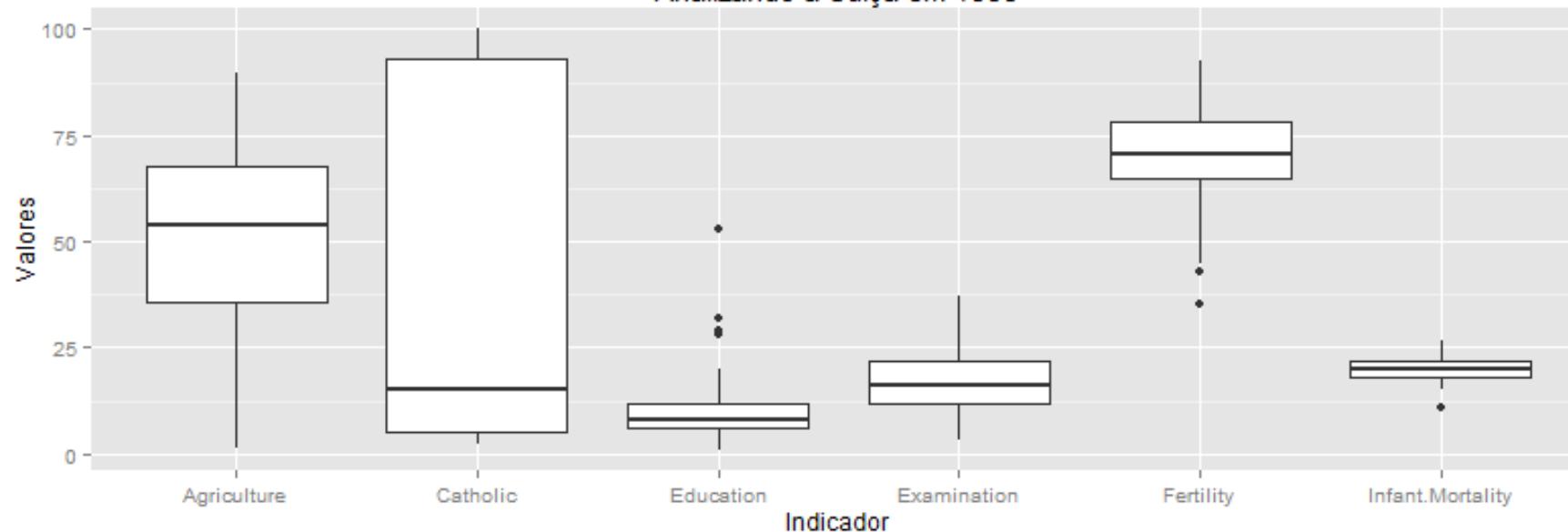
### Analizando a Suíça em 1888



**Exemplo de  
gráfico no R  
(usando ggplot2)**

Fonte:  
<http://www.dadosaleatorios.com.br/2014/10/fazendo-multiplos-graficos-com-o-ggplot2.html>

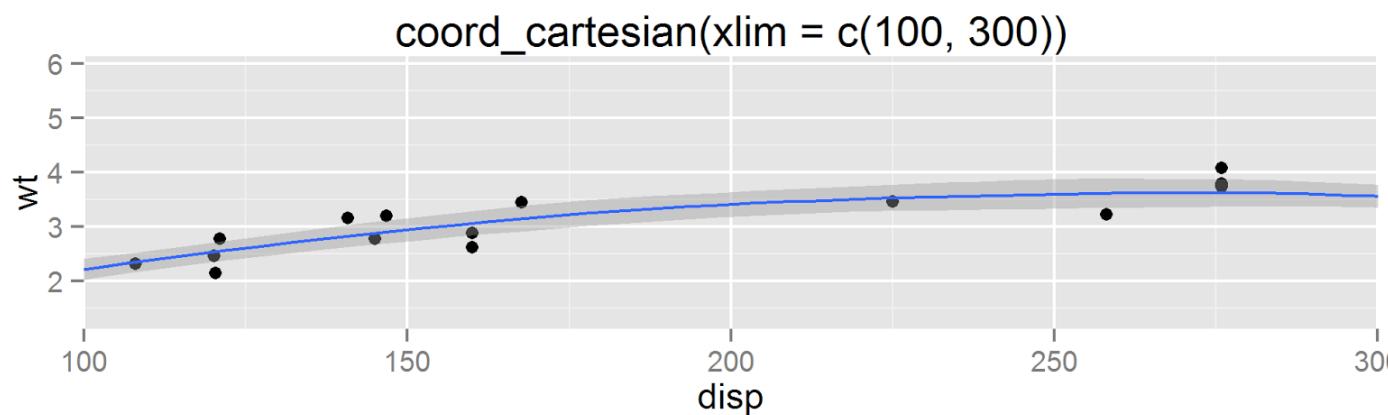
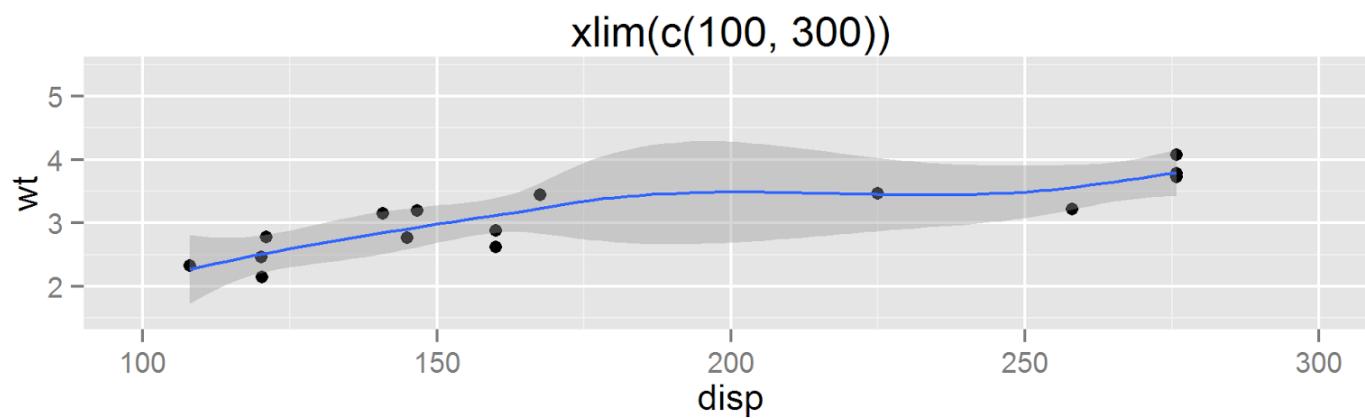
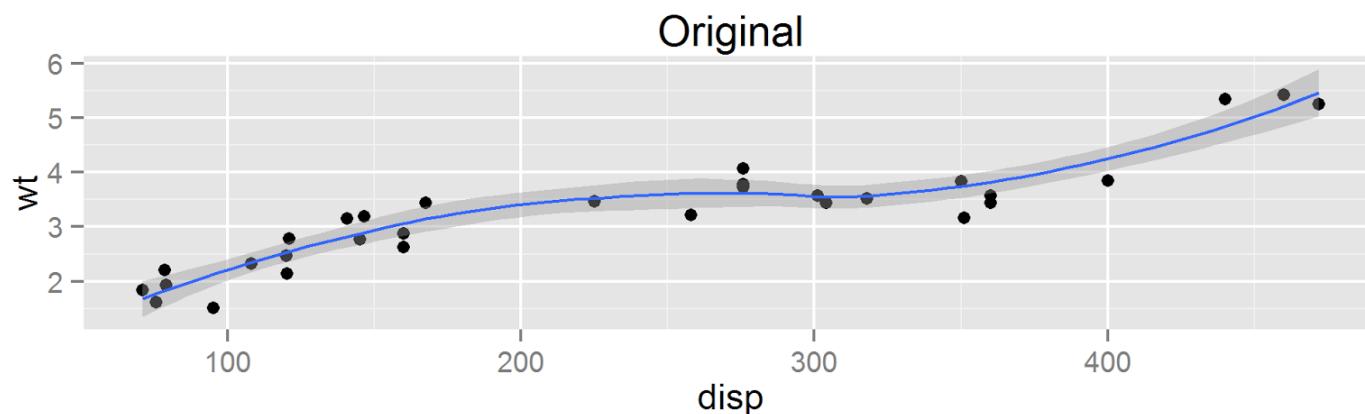
## Analizando a Suiça em 1888



**Exemplo de  
gráfico no R  
(usando ggplot2)**

Fonte:  
<http://www.dadosaleatorios.com.br/2014/10/fazendo-multiplos-graficos-com-o-ggplot2.html>

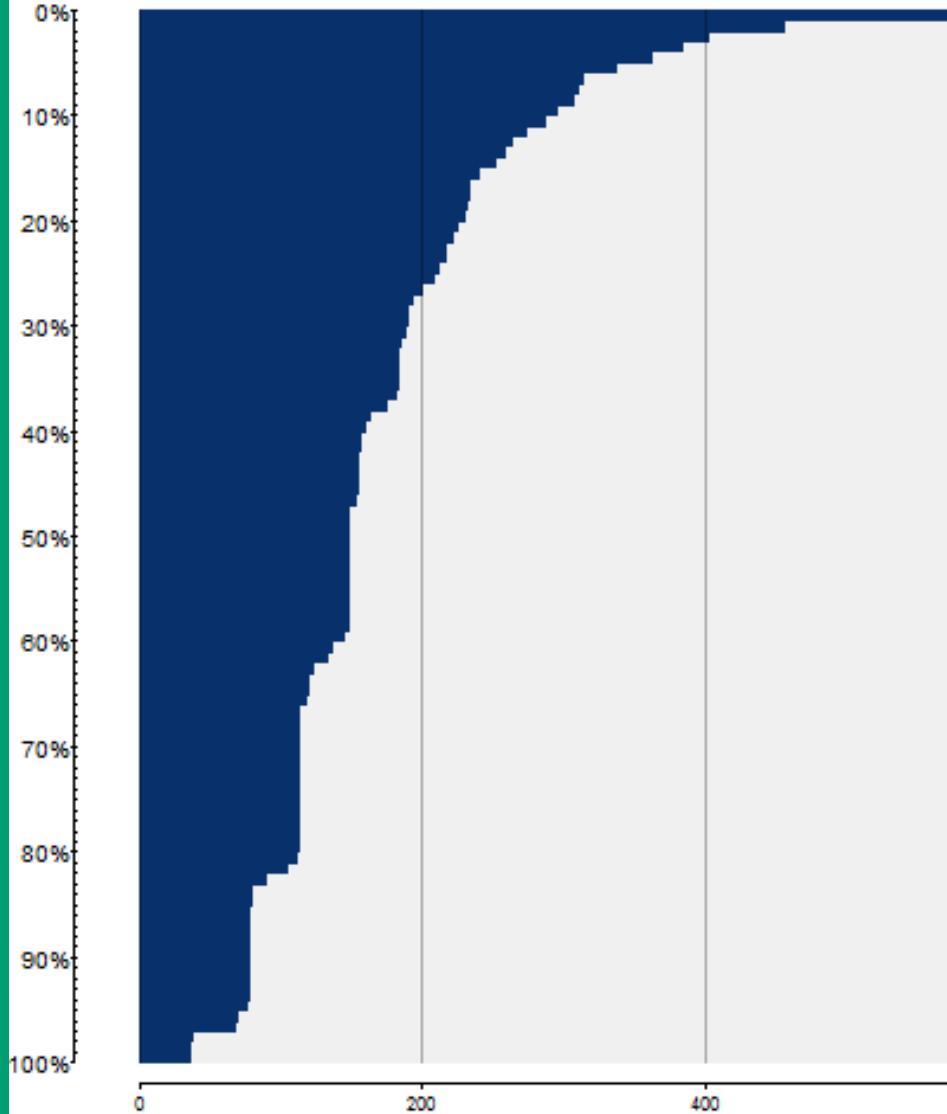
# Exemplo de gráfico no R (usando ggplot2)



Fonte:  
<http://www.dadosaleatorios.com.br/2014/10/alterando-escala-dos-eixos-no-ggplot2.html>

## Percentis da parcela paga e a distribuição pelo Brasil

Valor.Parcela



row bins: 100

objects:

14,047,474

140,475 (per bin)

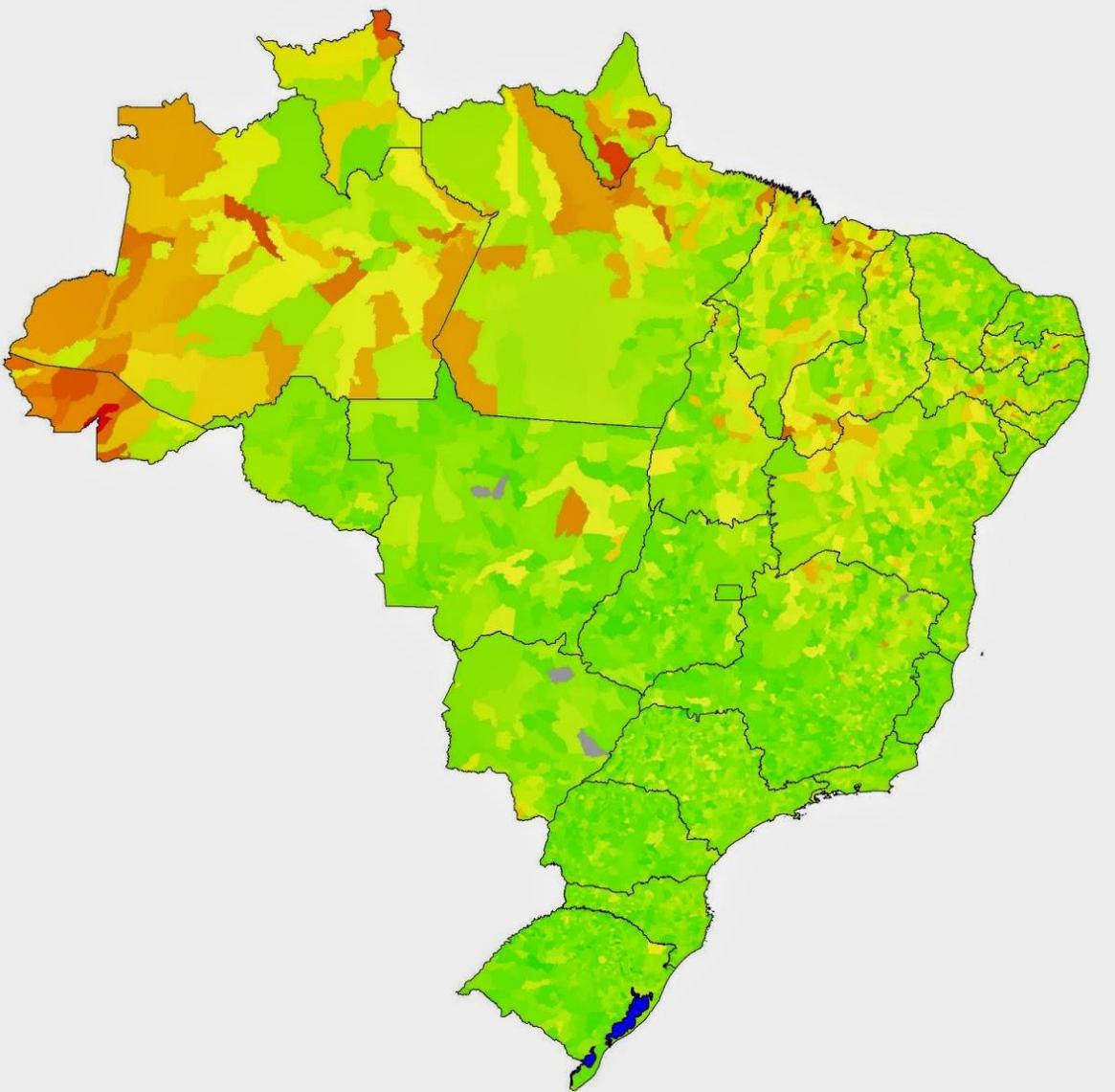
Região



Analisando  
microdados do  
governo  
(Bolsa Família)

Fonte:  
<http://www.dadosaleatorios.com.br/2014/12/dados-do-bolsa-familia.html>

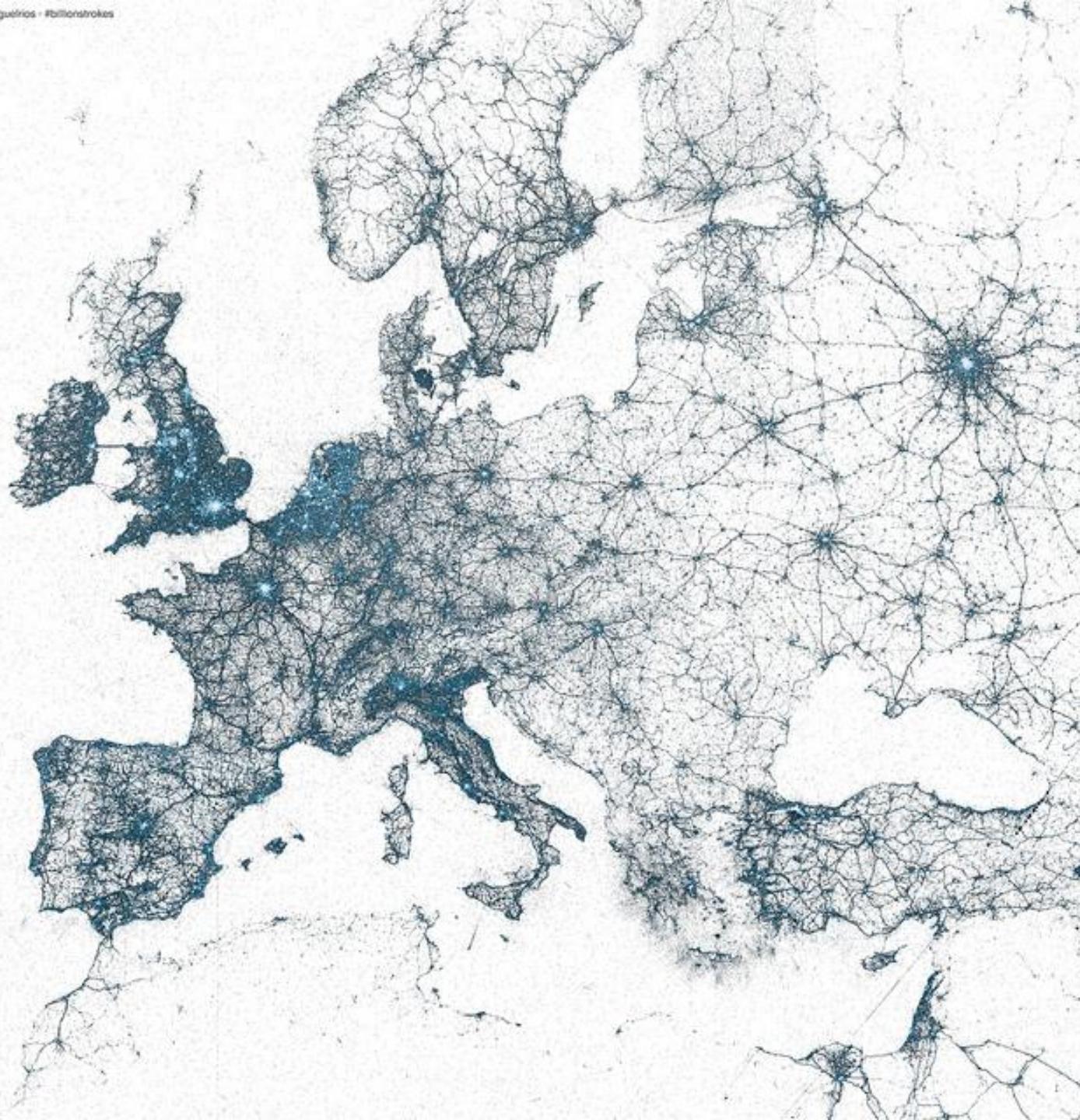
Benefício médio do Bolsa Família  
Novembro/2014



# Analisando microdados do governo (Bolsa Família)

Fonte:

<http://www.dadosaleatorios.com.br/2014/12/dados-do-bolsa-familia.html>



# The arteries of the World, in tweets

(as artérias do mundo em tweets)

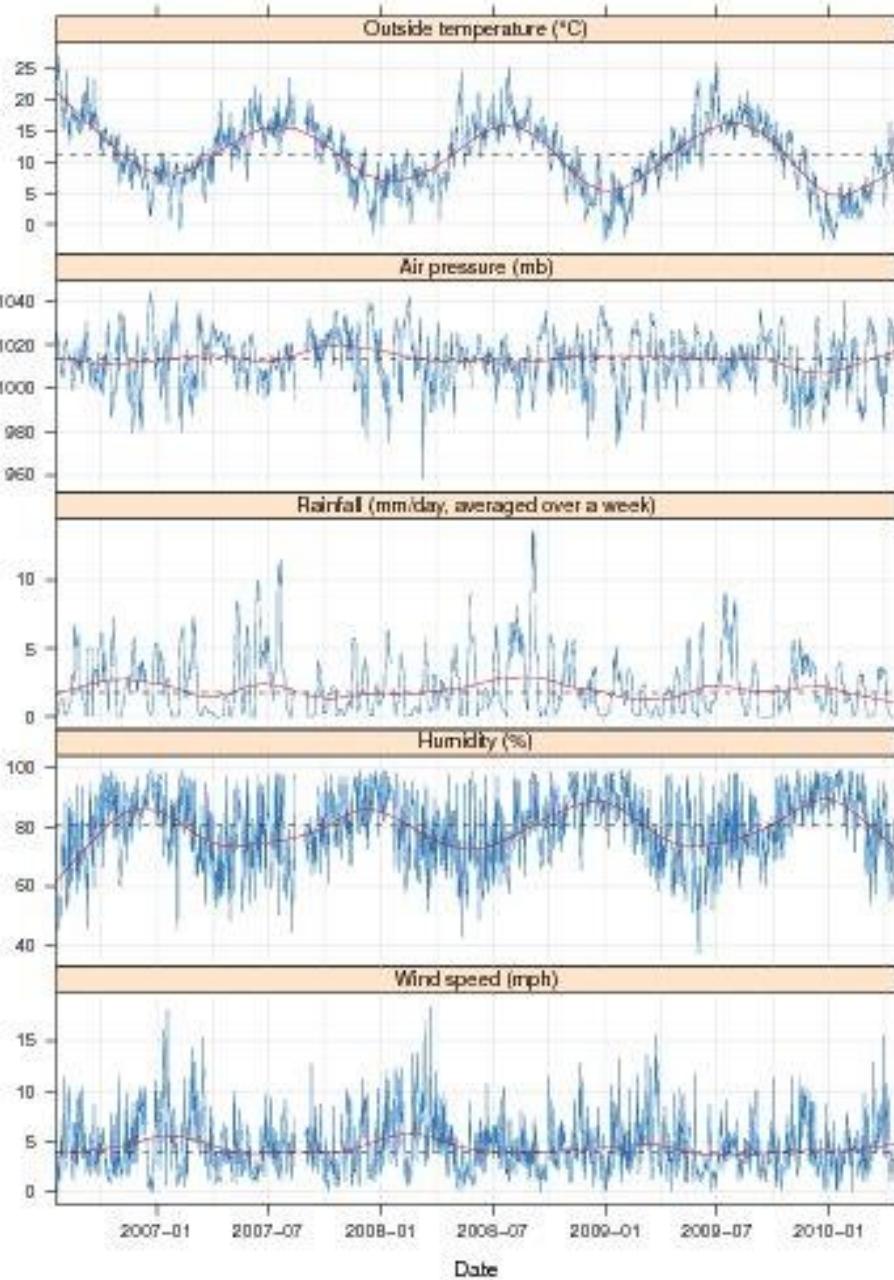
O gráfico mostra a localização dos tweets desde 2009, geolocalizados na Europa.

<http://blog.revolutionanalytics.com/2013/05/the-arteries-of-the-world-in-tweets.html>

### Birmingham Wast Hills Observatory average midday weather

raw data  
smoothed curve

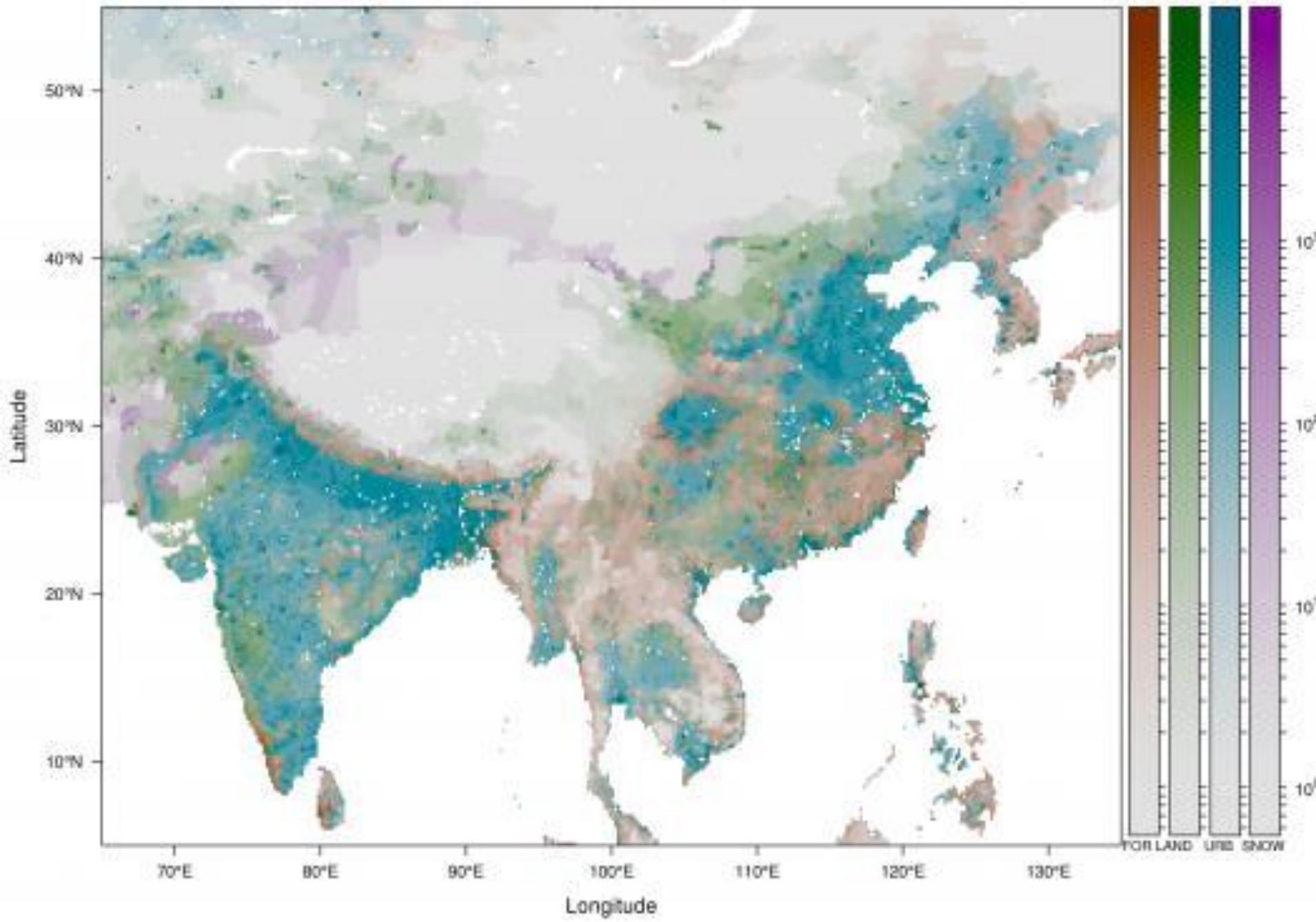
median value



# Gráficos de séries temporais

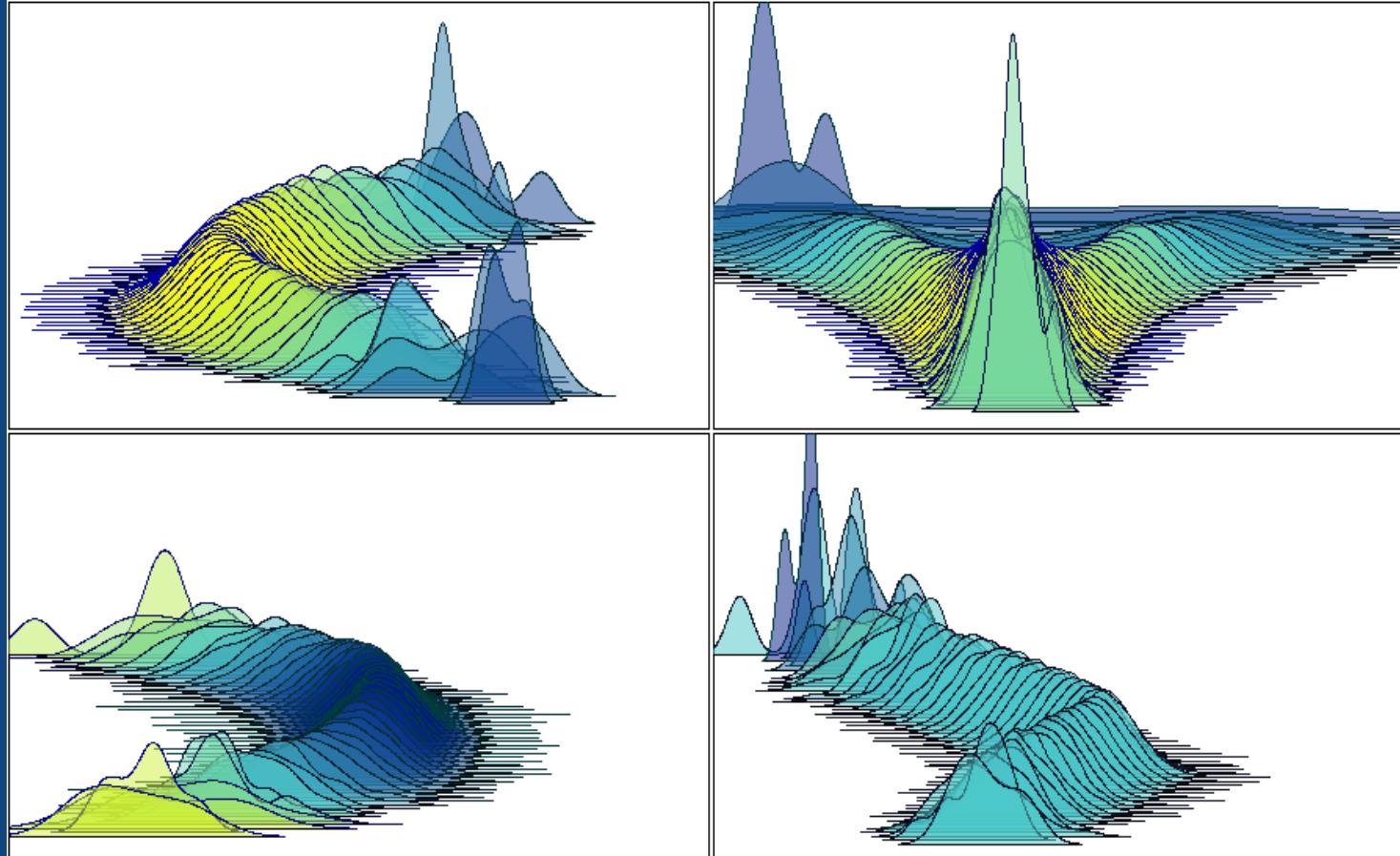
<http://www.sr.bham.ac.uk/~ajrs/R/r-gallery.html>

# Recriando mapas como os que aparecem em jornais

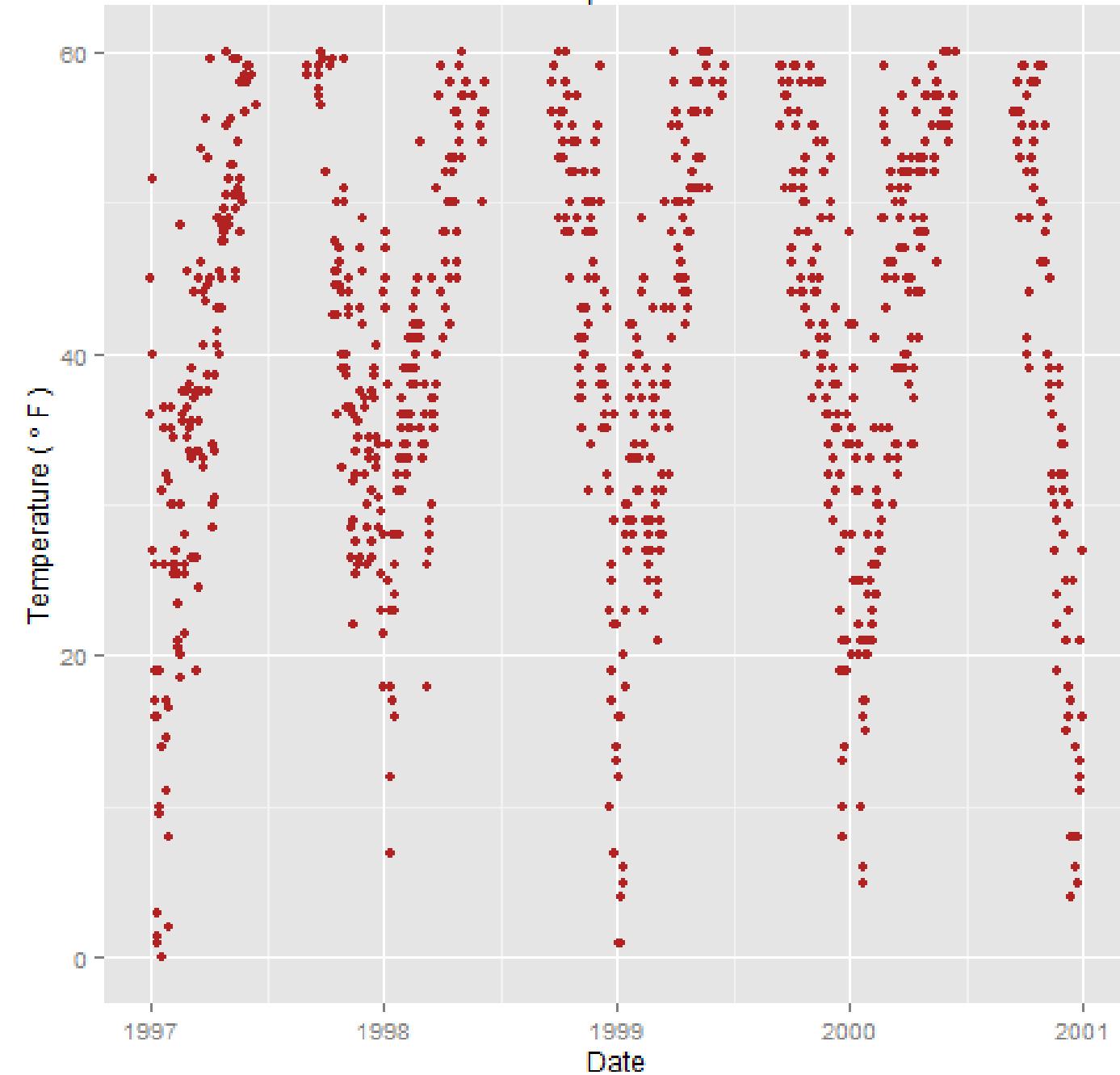


<http://www.r-bloggers.com/creating-beautiful-maps-with-r/>

# Gráficos de superficies

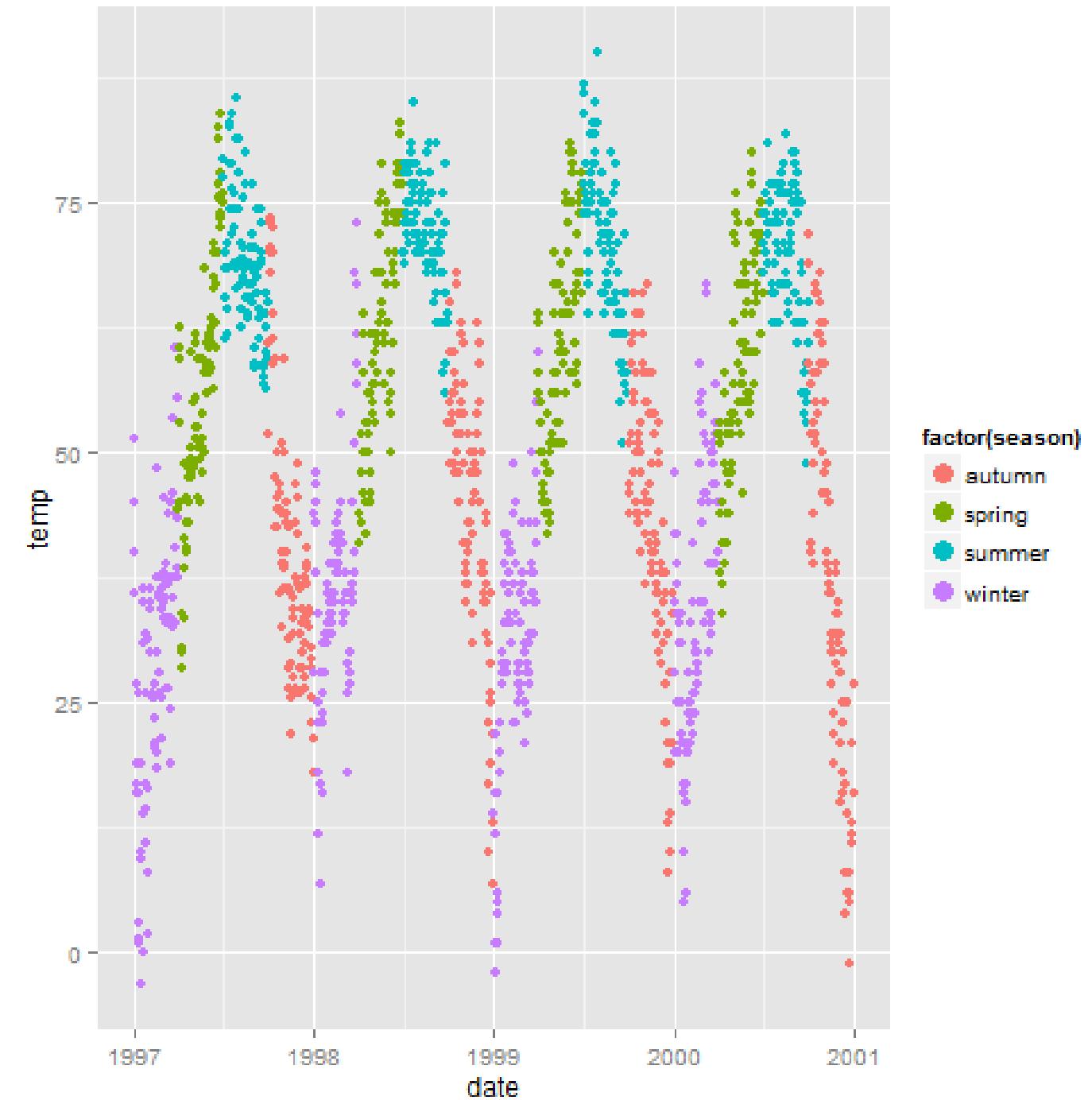


Temperature



Gráficos com elementos  
destacados

# Gráficos com elementos destacados



# EXPECTATION



Belos gráficos

# EXPECTATION VS REALITY



Belos gráficos



O gráfico que você aprende a fazer na  
aula da Aisha.

SUJANDO AS MÃOS



```
# A função plot() permite fazer alguns gráficos simples usando os pacotes  
pré-instalados do R.  
# Execute help(plot) para ver suas opções  
# Faça um gráfico de dispersão (pontos) dos atributos de ataque de todos os  
Pokémons  
# Repita o gráfico apenas para os Pokémon Lendários
```

```
```{r}
```

```
```
```

```
# A função plot() permite fazer alguns gráficos simples usando os pacotes  
pré-instalados do R.  
# Execute help(plot) para ver suas opções  
# Faça um gráfico de dispersão (pontos) dos atributos de ataque de todos os  
Pokémons  
# Repita o gráfico apenas para os Pokémon Lendários  
  
```{r}  
plot(Pokemon$Ataque, type = 'p', ylab = "Ataque", xlab = "ID")  
plot(Pokemon$Ataque[which(Pokemon$Lendario == T)], type = 'p', ylab =  
"Ataque", xlab = "ID")  
```
```

```
# O gráfico anterior não é muito informativo para nós.  
# Vamos fazer um box plot que tenha no eixo x o valor do ataque e no eixo y  
a categoria (se é lendário ou não).  
# Procure no help ou na internet sobre a função boxplot() e tente fazer o  
gráfico.
```

```
```{r}
```

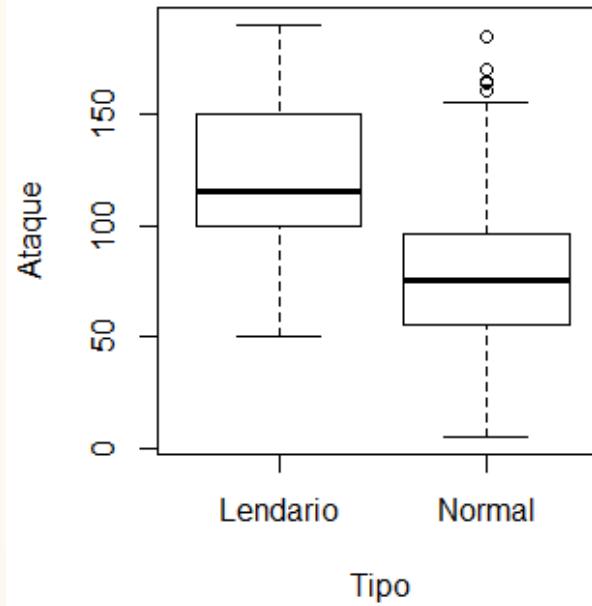
```
```
```

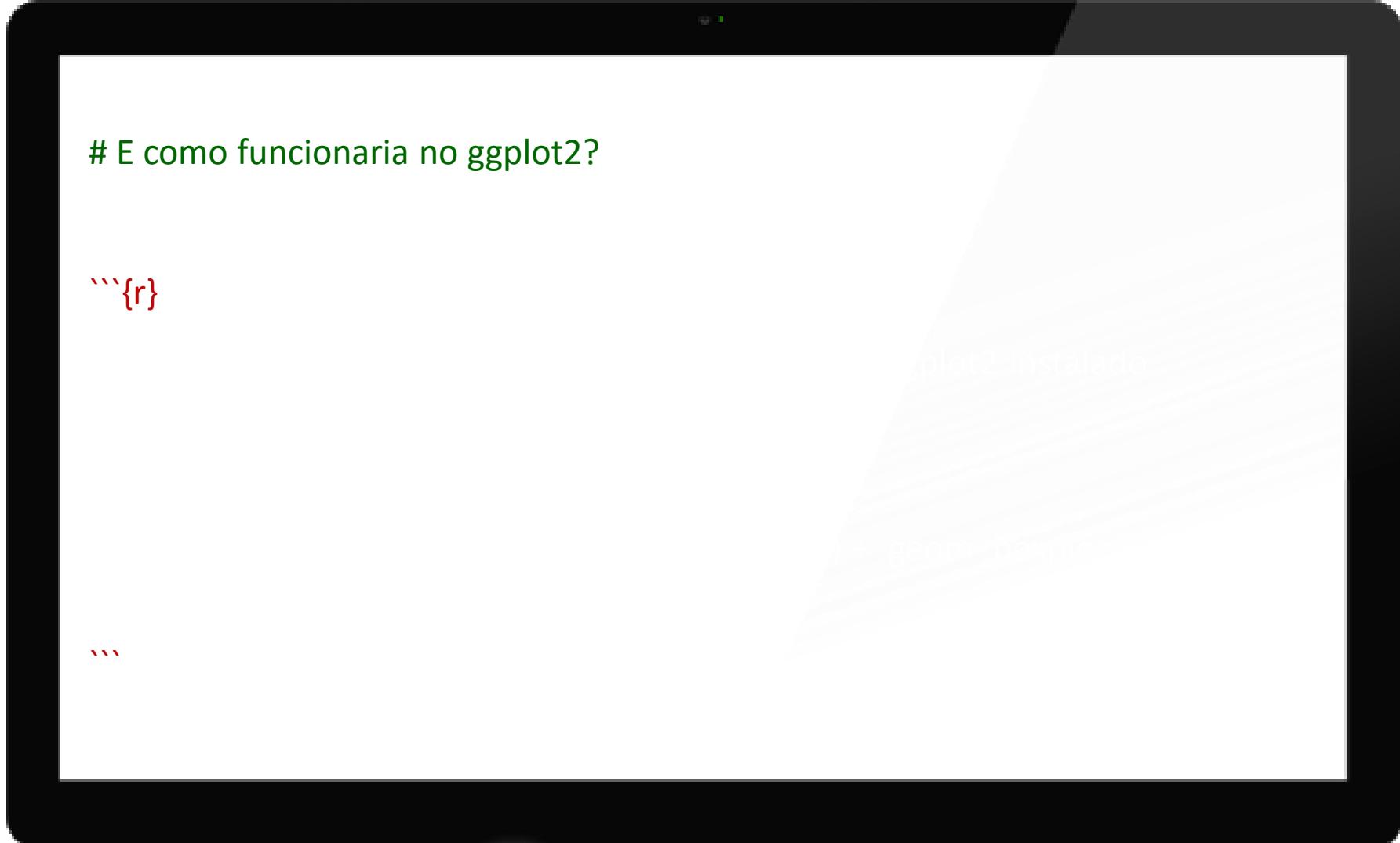
```
# O gráfico anterior não é muito informativo para nós.  
# Vamos fazer um box plot que tenha no eixo x o valor do ataque e no eixo y  
a categoria (se é lendário ou não).  
# Procure no help ou na internet sobre a função boxplot() e tente fazer o  
gráfico.
```

```
```{r}  
table(Pokemon$Lendario)  
Pokemon$Lendario[Pokemon$Lendario == TRUE] <- "Lendario"  
Pokemon$Lendario[Pokemon$Lendario == FALSE] <- "Normal"  
  
```
```

*Vamos renomear os valores da variável  
lendário para ficar mais bonito no gráfico*

```
# O gráfico anterior não é muito informativo para nós.  
# Vamos fazer um box plot que tenha no eixo x o valor do ataque e no eixo y  
# a categoria (se é lendário ou não).  
# Procure no help ou na internet sobre a função boxplot() e tente fazer o  
# gráfico.  
  
```{r}  
table(Pokemon$Lendario)  
Pokemon$Lendario[Pokemon$Lendario == TRUE] <- "Lendario"  
Pokemon$Lendario[Pokemon$Lendario == FALSE] <- "Normal"  
  
boxplot(Pokemon$Ataque ~ Pokemon$Lendario, ylab = "Ataque", xlab =  
"Tipo")  
```
```



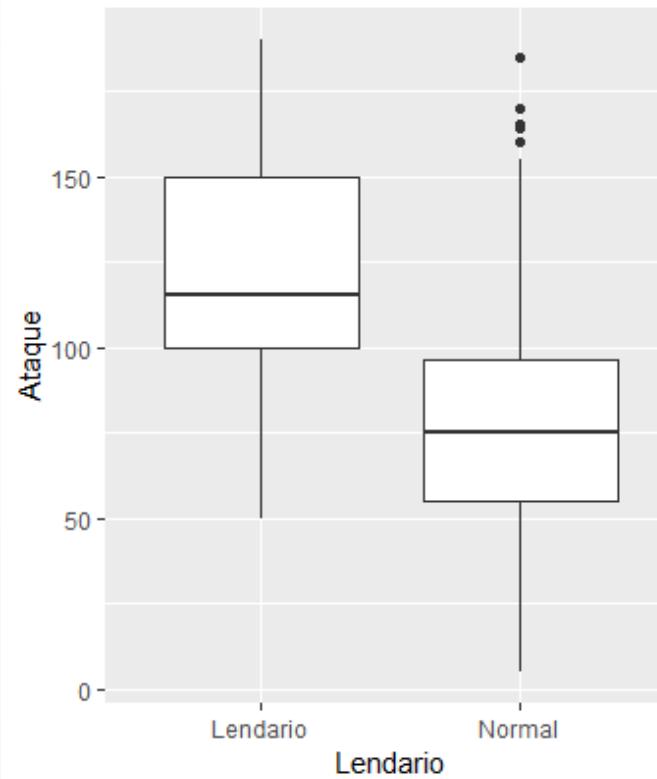


```
# E como funcionaria no ggplot2?
```

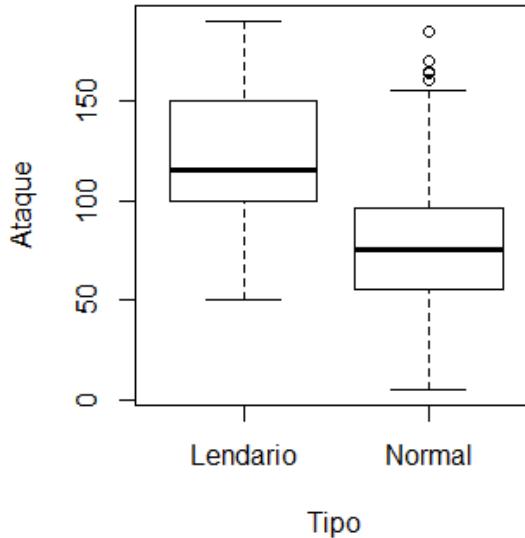
```
```{r}
```

```
```
```

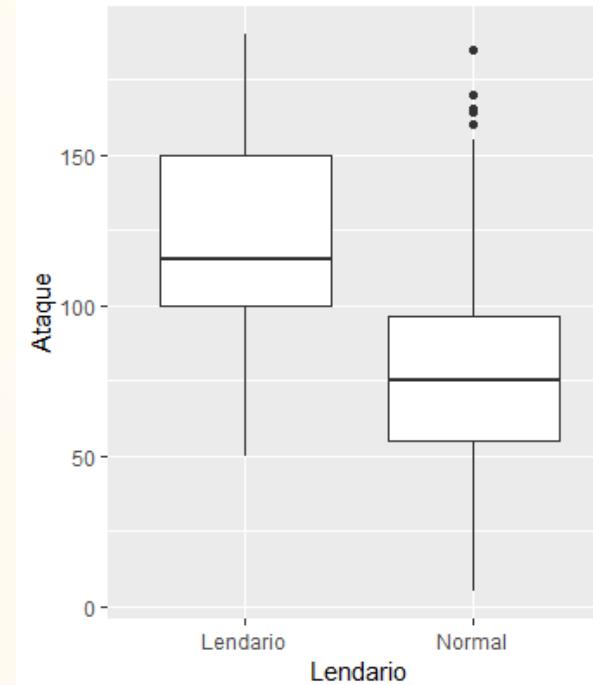
```
# E como funcionaria no ggplot2?  
  
```{r}  
# Descomente a linha abaixo caso você não tenha o ggplot2 instalado  
# install.packages("ggplot2")  
  
library(ggplot2)  
  
p <- ggplot(Pokemon, aes(x=Lendario, y=Ataque)) + geom_boxplot()  
p  
...  
```
```

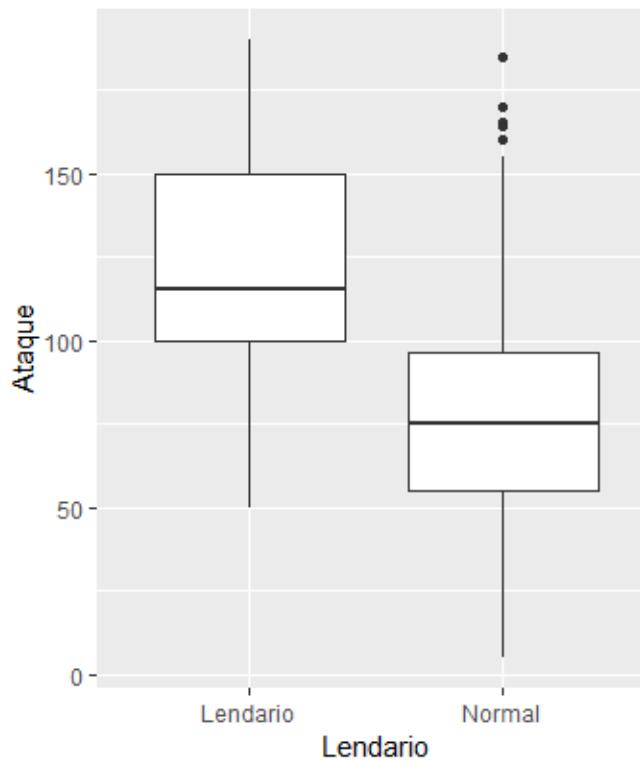


boxplot()



ggplot2





A partir daqui dá para  
customizar seu gráfico de  
diversas formas!

<http://www.sthda.com/english/wiki/ggplot2-box-plot-quick-start-guide-r-software-and-data-visualization>

A yellow cartoon character with a large head, small body, and a tuft of hair is sitting on a teal sofa, knitting a pink garment. The character has a joyful expression with a wide smile and closed eyes. A pink ball of yarn lies on the sofa next to the character. The background features a pink floral pattern.

Tricote (knit) seu código!

D:/Onedrive - Aisha/OneDrive/Documentos/R-Ladies/Floripa Chapter/Encontro 1 - Mini curso - Maio 2019/Material mini curso/Material em PDF/Pokemon.html

Pokemon.html | Open in Browser | Find

Análise Pokémon

A. Schmidt

June 1, 2019

## Introdução

Este documento tem as análises do banco de dados Pokémon.

## Leitura dos dados

Vamos buscar os dados de duas fontes: [Fonte 1](#) e [Fonte 2](#).

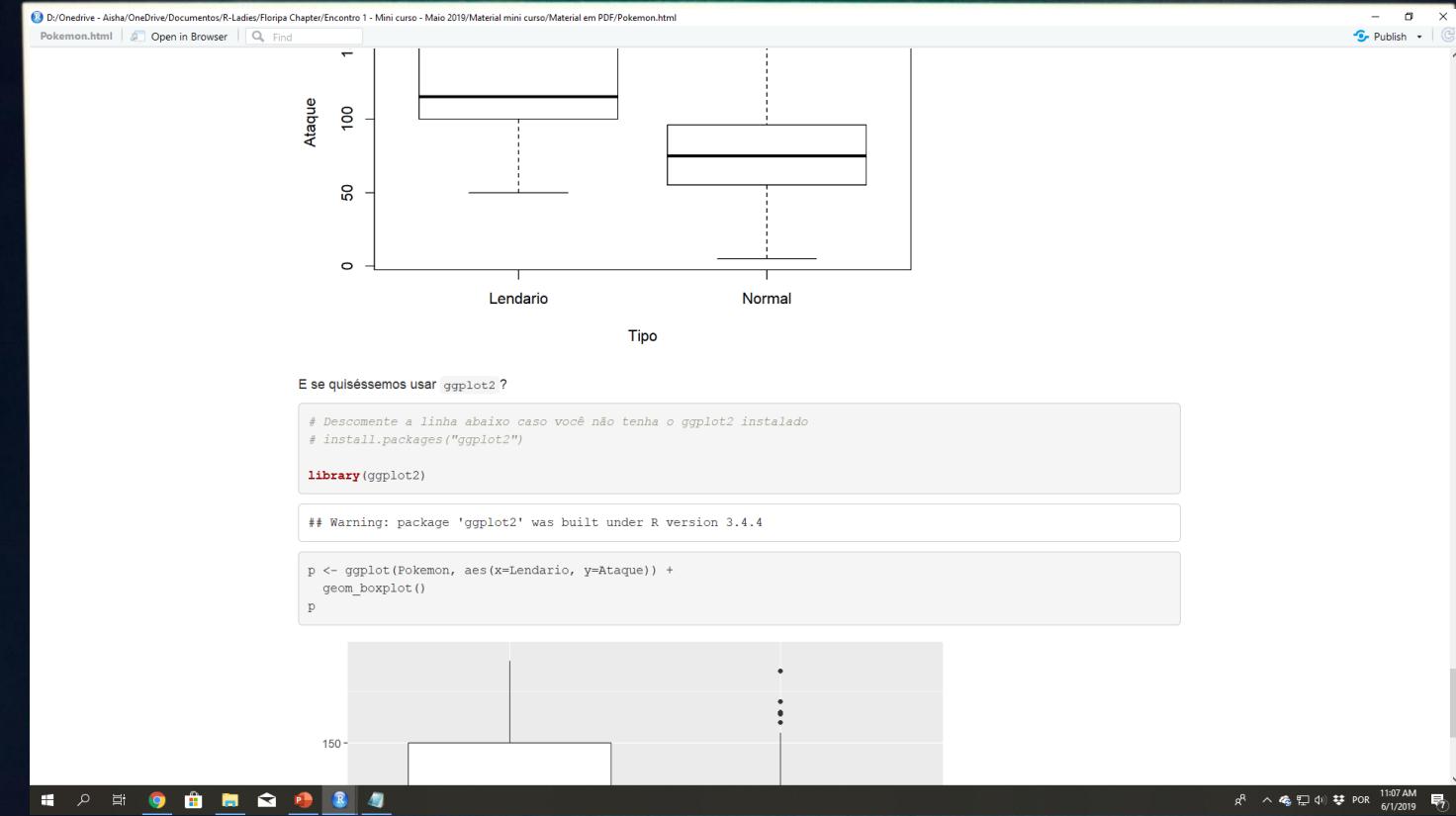
```
Pokemon_01 <- read.csv('https://tinyurl.com/pokemon-rladies-01')
Pokemon_02 <- read.csv('https://tinyurl.com/pokemon-rladies-02')

head(Pokemon_01, n=10)
```

| ##    | ID | Nome          | Tipol       | Typo2  | Geracao | Lendario |       |
|-------|----|---------------|-------------|--------|---------|----------|-------|
| ## 1  | 1  | Bulbasaur     | Grass       | Poison | 1       | FALSE    |       |
| ## 2  | 2  | Ivysaur       | Grass       | Poison | 1       | FALSE    |       |
| ## 3  | 3  | Venusaur      | Grass       | Poison | 1       | FALSE    |       |
| ## 4  | 3  | VenusaurMega  | Venusaur    | Grass  | Poison  | 1        | FALSE |
| ## 5  | 4  | Charmander    | Fire        |        |         | 1        | FALSE |
| ## 6  | 5  | Charmeleon    | Fire        |        |         | 1        | FALSE |
| ## 7  | 6  | Charizard     | Fire        | Flying | 1       | FALSE    |       |
| ## 8  | 6  | CharizardMega | Charizard X | Fire   | Dragon  | 1        | FALSE |
| ## 9  | 6  | CharizardMega | Charizard Y | Fire   | Flying  | 1        | FALSE |
| ## 10 | 7  | Squirtle      | Water       |        |         | 1        | FALSE |

```
head(Pokemon_02, n = 3)
```

| ##   | ID | Nome      | Total | Vida | Ataque | Defesa | AtaqueEsp | DefesaEsp | Velocidade |
|------|----|-----------|-------|------|--------|--------|-----------|-----------|------------|
| ## 1 | 1  | Bulbasaur | 318   | 45   | 49     | 49     | 65        | 65        | 45         |
| ## 2 | 2  | Ivysaur   | 405   | 60   | 62     | 63     | 80        | 80        | 60         |
| ## 3 | 2  | Venusaur  | 505   | 80   | 92     | 92     | 100       | 100       | 90         |





**END**

A close-up shot of a man with long, dark hair and a full beard. He is looking upwards and slightly to his right with a contemplative expression. The background is blurred, showing what appears to be a rustic wooden structure and other people in the distance.

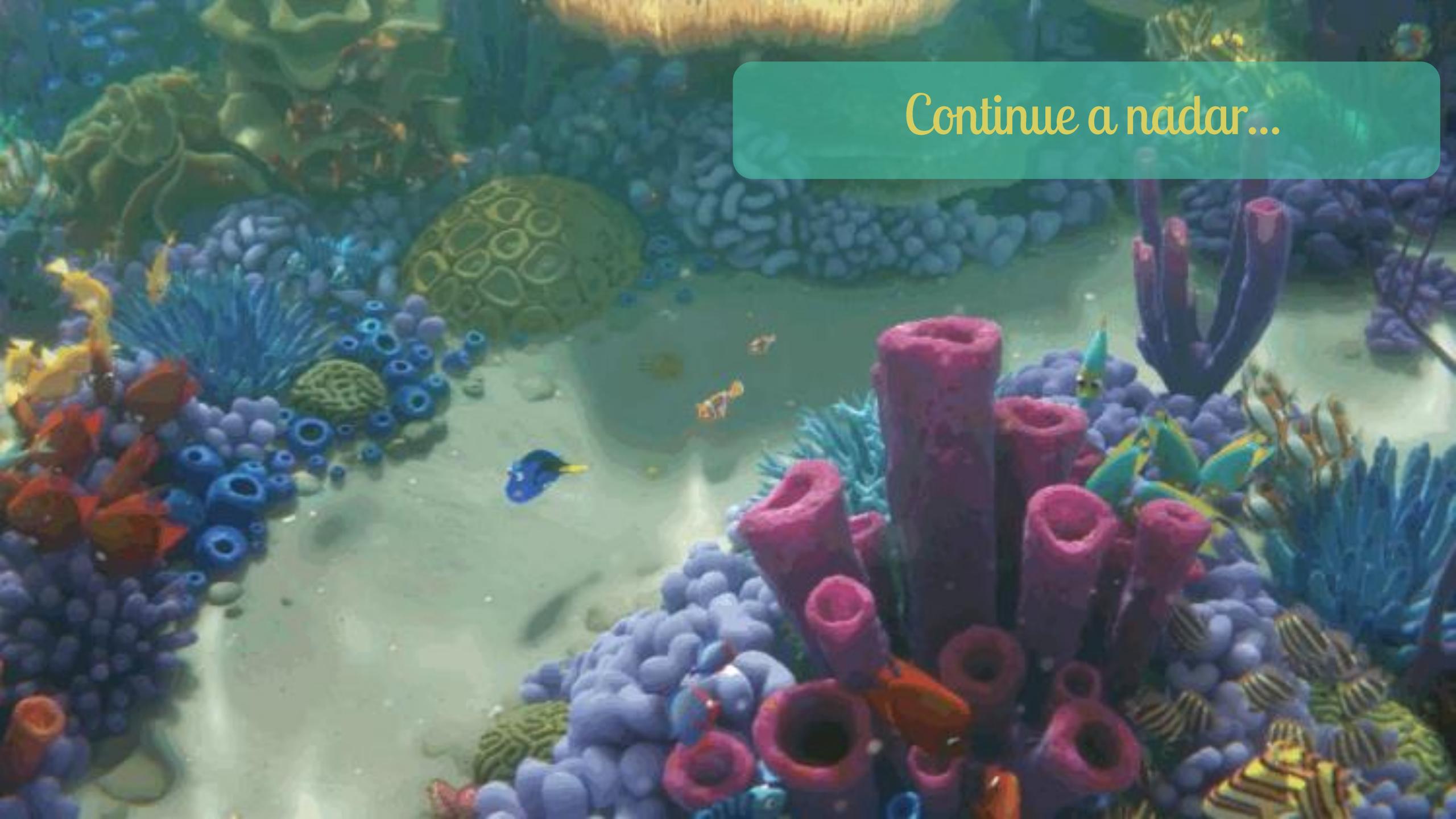
O que acontece agora?

*So, what happens now?*

stirr

O fim é um novo começo



The background is a detailed 3D rendering of an underwater environment. A sandy ocean floor stretches across the center, dotted with small, colorful fish. To the left, a large, yellow and orange striped fish swims near a cluster of purple and blue coral. In the upper left, a large green sea turtle with a yellow patterned shell swims gracefully. The right side of the image features a dense coral reef with various types of coral, including red, pink, and purple tube corals and blue brain corals. Sunlight filters down from the surface in bright rays, illuminating the water and creating a peaceful atmosphere.

Continue a nadar...

Programação requer prática...



E determinação!



# SUPPORT



Created by Valeria Chiquin  
from Noun Project

# YOUR

# LOCAL

# GIRL



Created by Tatyana  
from Noun Project

# GANG

# SUPPORT



Created by Valeria Chiquin  
from Noun Project

# LOCAL

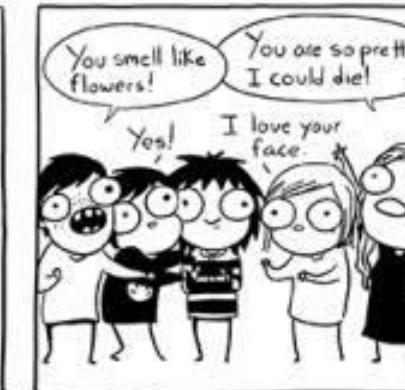
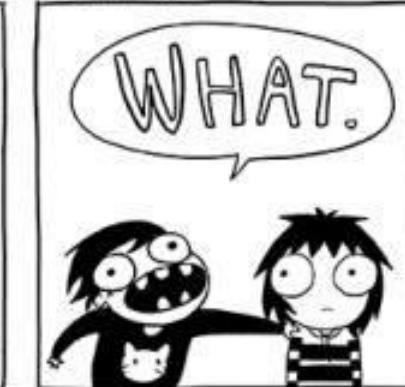
# GIRL



Created by Tatyana  
from Noun Project

# GANG

## WHY FEMALE FRIENDSHIPS ARE IMPORTANT



© Sarah Andersen

# R-Ladies Floripa

## Precisamos de você!

- Organização dos próximos encontros
- Palestrantes
- Locais para os encontros
- Patrocínios
- Ideias!



# R-Ladies Floripa

## Precisamos de você!

- Organização dos próximos encontros
- Palestrantes
- Locais para os encontros
- Patrocínios
- Ideias!

Procure a Camila Weber  
mais próxima de você!



goodbye





**Thank You**

