

Introdução ao uso de dados geoespaciais no R

2 Funcionamento da linguagem R

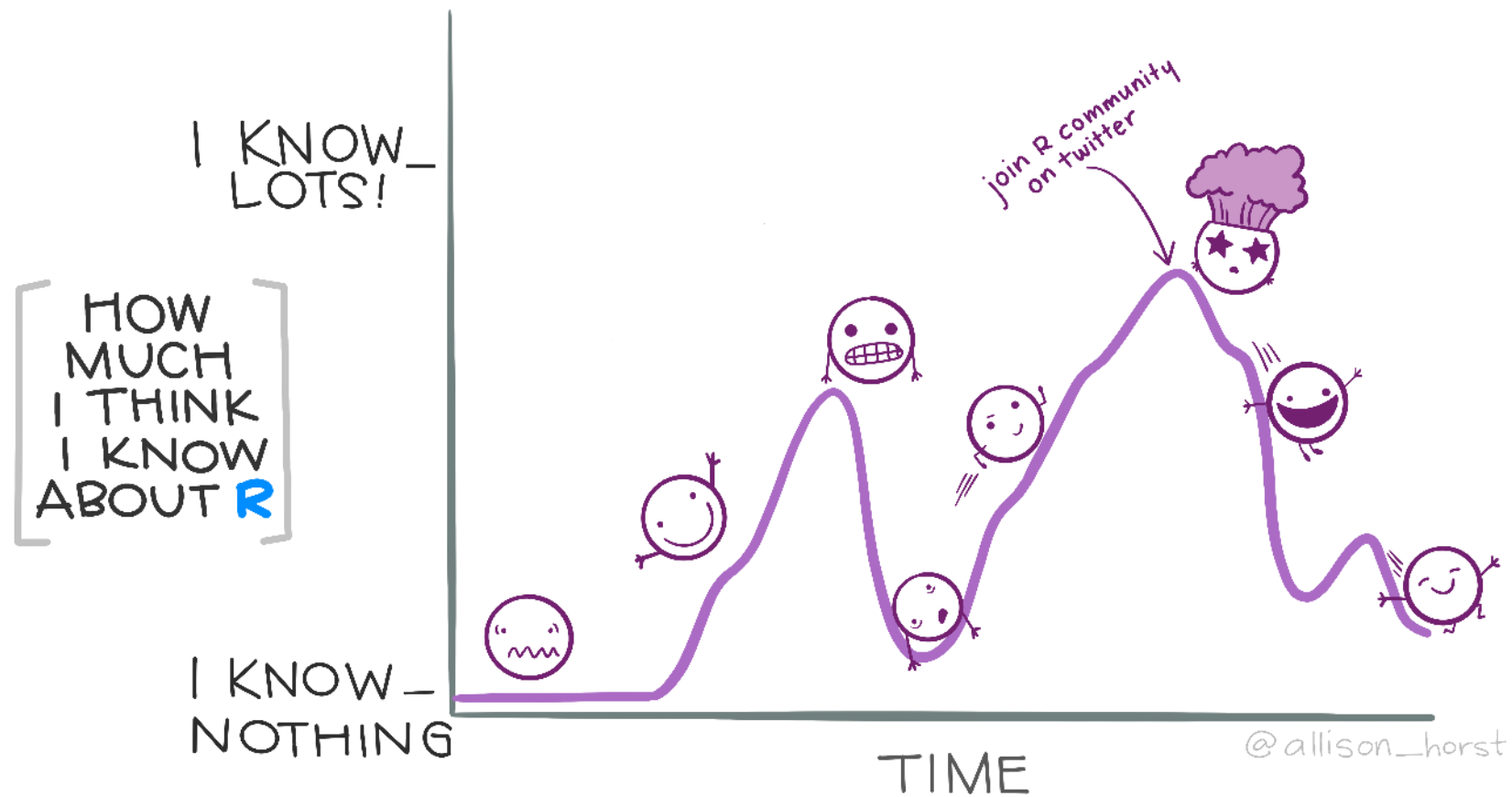
Maurício H. Vancine

Milton C. Ribeiro

UNESP - Rio Claro

Laboratório de Ecologia Espacial e Conservação (LEEC)

25/10/2021-05/11/2021



[@allison_horst](#)

2 Funcionamento da linguagem R

Conteúdo

1. Linguagem R
2. RStudio
3. Console
4. Scripts
5. Operadores
6. Objetos
7. Funções
8. Pacotes
9. Ajuda
10. Ambiente
11. Citações
12. Principais erros
13. Principal material de estudo



1. Linguagem R

Definição

O R é uma **linguagem de programação livre** (*open source*), direcionada à **manipulação, análise e visualização de dados**, com diversas **expansões** (*pacotes*) para o uso de **dados com formatos específicos**



1. Linguagem R

Cinco motivos para usar R

1. R é completamente **gratuito**
2. Pessoas da comunidade **disponibilizam** seu trabalho em R
3. R possui um ecossistema que incentiva a **reprodutibilidade**
4. R tem uma **comunidade** vibrante e crescente
5. Os focos da linguagem são **modelagem, visualização e análise de dados**

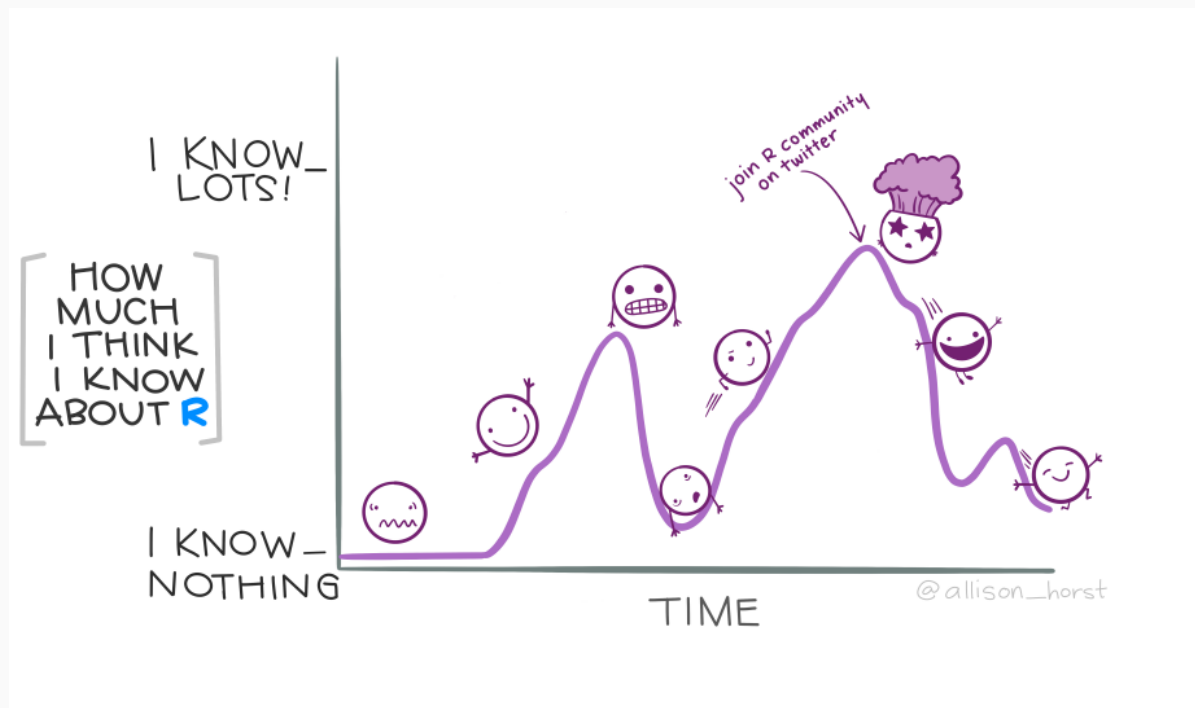


[Curso-R](#)

5 MOTIVOS
PARA
USAR R

1. Linguagem R

É legal, mas nem sempre é fácil...



[@allison_horst](#)



E de onde surgiu o R?

1. Linguagem R

Histórico - Linguagem S

John M. Chambers (Stanford University, CA, EUA)

- Old S (1976-1987)
- New S (1988-1997)
- S4 (1998)

- Interface: S-PLUS (1988-2008)



[Wickham \(2013\)](#).

1. Linguagem R

Histórico - Linguagem R

Robert Gentleman e **Ross Ihaka** (Auckland University, NZ)

Versões

- Desenvolvimento (1997-2000)
- Versão 1 (2000-2004)
- Versão 2 (2004-2013)
- Versão 3 (2013-2020)
- Versão 4 (2020-atual)

IDE (*Integrated Development Environment*)

- Interface: RStudio (2011-atual)
- Atualmente: **R Core Team**

[Wickham \(2013\)](#).



1. Linguagem R

Base R

Base R Cheat Sheet

Getting Help

Accessing the help files

?mean
Get help of a particular function.
help.search('weighted mean')
Search the help files for a word or phrase.
help(package = 'dplyr')
Find help for a package.

More about an object

str(iris)
Get a summary of an object's structure.
class(iris)
Find the class an object belongs to.

Using Libraries

install.packages('dplyr')
Download and install a package from CRAN.

library(dplyr)
Load the package into the session, making all its functions available to use.

dplyr::select
Use a particular function from a package.

data(iris)
Load a built-in dataset into the environment.

Working Directory

getwd()
Find the current working directory (where inputs are found and outputs are sent).

setwd('C://file/path')
Change the current working directory.

Use projects in RStudio to set the working directory to the folder you are working in.

Vectors

Creating Vectors

<code>c(2, 4, 6)</code>	<code>2 4 6</code>	An ordered list of values
<code>2:6</code>	<code>2 3 4 5 6</code>	An integer sequence
<code>seq(2, 3, by=0.5)</code>	<code>2.0 2.5 3.0</code>	A numeric sequence
<code>rep(1:2, times=3)</code>	<code>1 2 1 2 1 2</code>	Repeated values
<code>rep(1:2, each=3)</code>	<code>1 1 1 2 2 2</code>	Repeated elements of a vector

Vector Functions

sort(x) Return a sorted	rev(x) Return a reversed
table(x) See counts of values	unique(x) See unique values

Selecting Vector Elements

By Position

<code>x[4]</code>	The fourth element
<code>x[-4]</code>	All but the fourth
<code>x[2:4]</code>	Elements two to four
<code>x[-(2:4)]</code>	All elements except two to four
<code>x[c(1, 5)]</code>	Elements one and five

By Value

<code>x[x == 10]</code>	Elements which are equal to 10
<code>x[x < 0]</code>	All elements less than zero
<code>x[x %in% c(1, 2, 5)]</code>	Elements in the set 1, 2, 5

Named Vectors

<code>x['apple']</code>	Element with name 'apple'
-------------------------	---------------------------

Programming

For Loop

for (variable in sequence){
 do something
}

Example

```
for (i in 1:4){  
  i <- i + 10  
  print(i)  
}
```

While Loop

```
while (condition){  
  do something  
}
```

Example

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

If Statements

```
if (condition){  
  do something  
} else {  
  do something different  
}
```

Example

```
if (i > 3){  
  print('Yes')  
} else {  
  print('No')  
}
```

Functions

```
function_name <- function(var){  
  do something  
  return(new_variable)  
}
```

Example

```
square <- function(x){  
  squared <- x*x  
  return(squared)  
}
```

Reading and Writing Data

Input	Output	Description
<code>df <- read.table('file.txt')</code>	<code>write.table(df, 'file.txt')</code>	Read and write a delimited text file
<code>df <- read.csv('file.csv')</code>	<code>write.csv(df, 'file.csv')</code>	Read and write a comma separated value file. This is a special case of read/write table.
<code>load('file.RData')</code>	<code>save(df, file = 'file.RData')</code>	Read and write an R data file. R has a special file type.

Condition	A == B	Are equal	A > B	Bigger than	A < B	Smaller than	A >= B	Greater than or equal to	A <= B	Less than or equal to	is.na(x)	is missing
	A != B	Not equal	A < B	Less than	A <= B	Less than or equal to	is.null(x)	is null				

RStudio is a trademark of RStudio, Inc. © 2020 RStudio, Inc. | rstudio.com | info@rstudio.com

Each row of this page is generated by a package version. | Updated: 2/20

E o que o R pode fazer?

1. Linguagem R

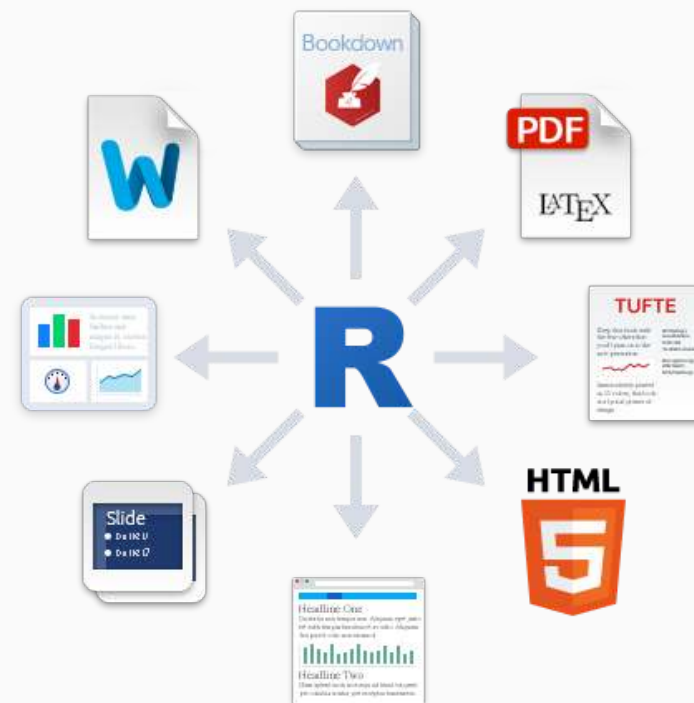
Aplicações

Análises e visualização de dados

- Estatísticas univariadas e multivariadas
- Análises de dados ecológicos (população, comunidades e ecossistemas)
- Análise de dados espaciais, temporais e sonoros
- Análise de dados funcionais, genéticos e filogenéticos
- Análise de dados geoespaciais e sensoriamento remoto
- Visualização de todos os dados anteriores

R Markdown

- Textos em HTML, PDF, Word, ODT, Markdown
- Apresentação de slides
- Websites e Blogs
- Livros e artigos
- Shiny



Há uns 15 anos, um nome tem se destacado no avanço da linguagem R, na parte de *manipulação, visualização e análise de dados* (tidyverse e tidymodels)

1. Linguagem R

Hadley Wickham

Cientista Chefe no RStudio e Professor Adjunto de Estatística na Universidade de Auckland, Stanford e Rice



[Hadley Wickham](#)

Há uns 10 anos, outro nome tem se destacado no avanço da linguagem R, na parte de *textos, sites e apresentações (R Markdown)*

1. Linguagem R

Yihui Xie

Engenheiro de software no RStudio



[Yihui Xie](#)

2. RStudio





Todos conhecem o RStudio?



2. RStudio

IDE

Integrated Development Environment ou Ambiente de Desenvolvimento Integrado

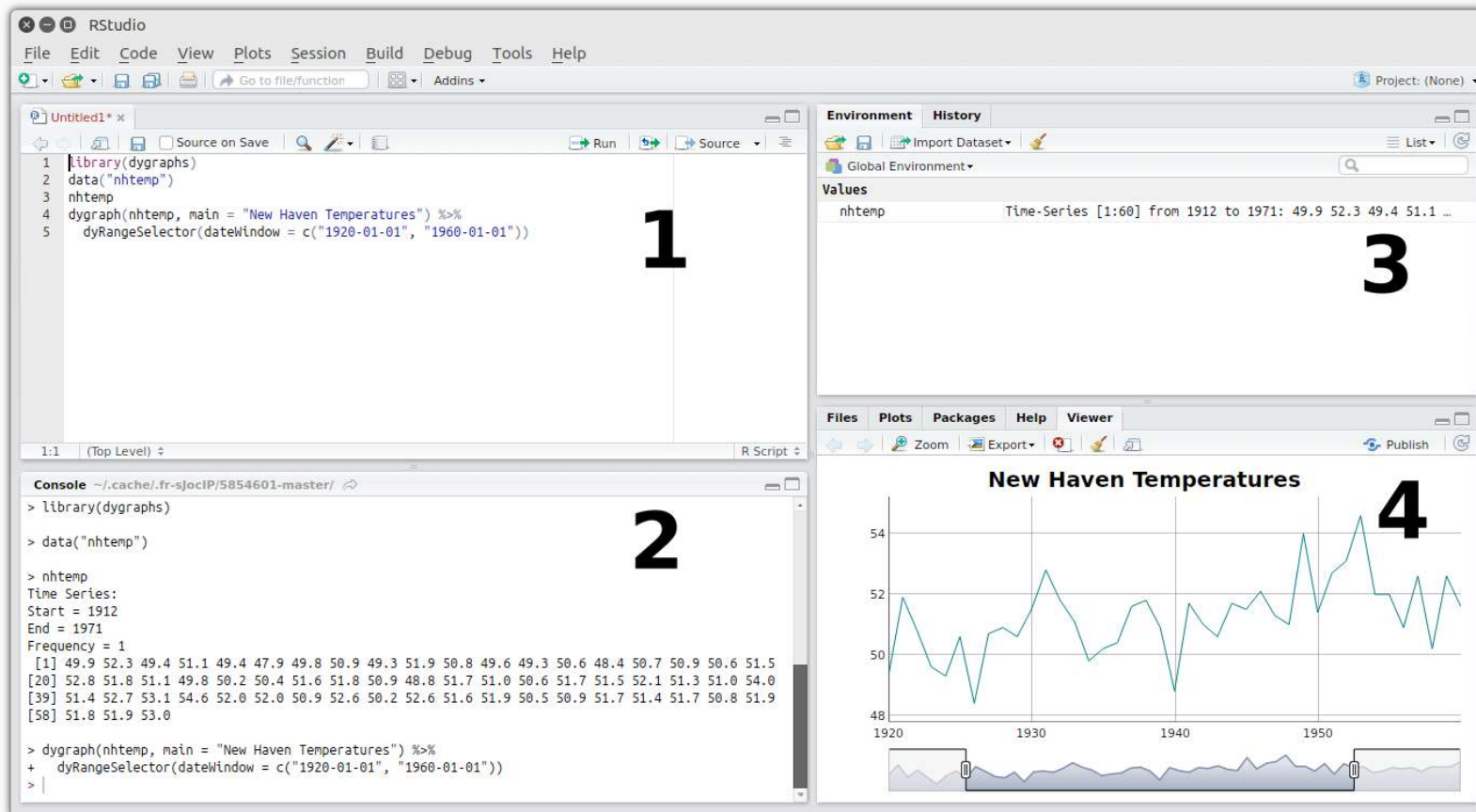
R: Do not open this	RStudio: Open this
	
R: Engine	RStudio: Dashboard
	

RStudio IDE Cheatsheet



2. RStudio

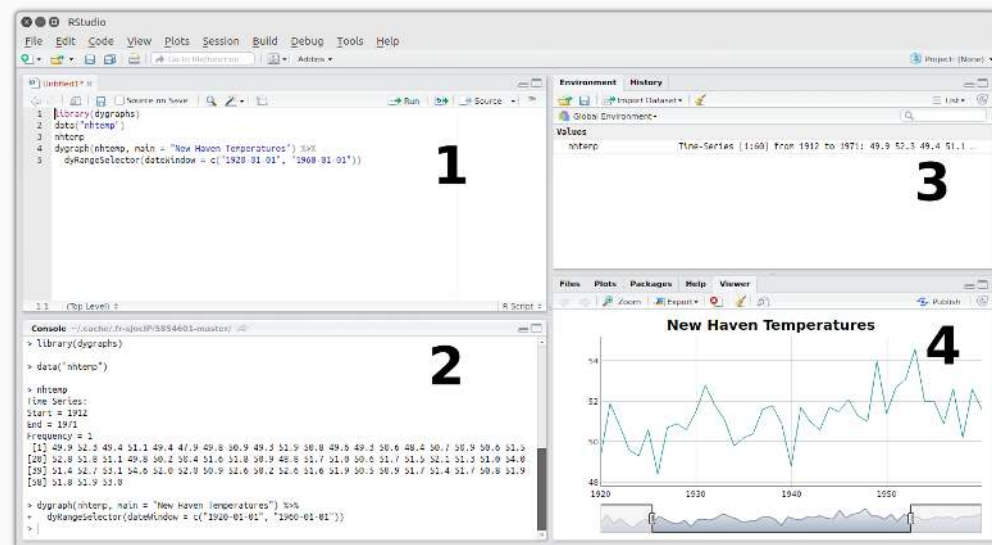
Interface



2. RStudio

Janelas e abas

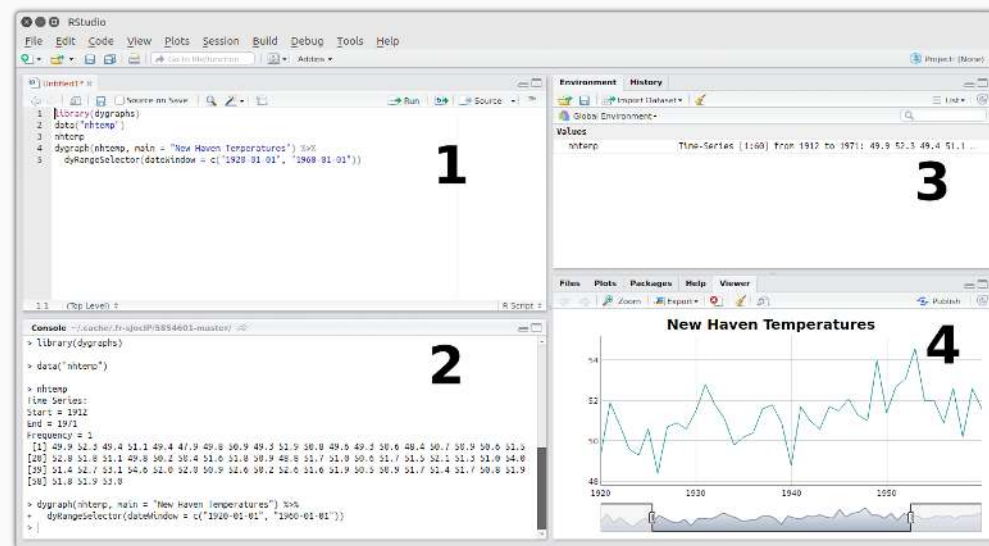
- 1. Editor/Script:** é onde escrevemos nossos códigos em R ou R Markdown
- 2. Console:** é onde os códigos são rodados e vemos as saídas
- 3. Environment:** painel com todos os objetos criados na sessão
- 3. History:** painel com o histórico dos códigos rodados
- 3. Connection:** painel para conectar banco de dados
- 3. Git:** painel do controle de versão
- 3. Tutorial:** painel de tutoriais
- 4. Files:** painel que mostra os arquivos no diretório de trabalho
- 4. Plots:** painel onde os gráficos são apresentados
- 4. Packages:** painel que lista os pacotes
- 4. Help:** painel onde a documentação das funções é exibida
- 4. Viewer:** painel de visualização



2. RStudio

Atalhos

- **f1**: abre o painel de *Help*
- **ctrl + Enter**: roda a linha selecionada no script
- **ctrl + Shift + N**: abre um novo script
- **ctrl + S**: salva um script
- **ctrl + Z**: desfaz uma operação
- **ctrl + shift + Z**: refaz uma operação
- **alt + -**: insere um sinal de atribuição (<-)
- **ctrl + Shift + M**: insere um operador pipe (%>%)
- **ctrl + Shift + C**: comenta uma linha no script - insere um (#)
- **ctrl + Shift + R**: insere uma sessão (# -----)
- **ctrl + Shift + H**: abre uma janela para selecionar o diretório de trabalho
- **ctrl + Shift + f10**: reinicia o console
- **ctrl + L**: limpa os códigos do console
- **alt + Shift + K**: abre uma janela com todos os atalhos disponíveis



2. RStudio

Projeto R (.Rproj) (Tesseract)

- Facilita o trabalho em múltiplos ambientes
- Cada projeto possui seu diretório, documentos e workspace
- Permite versionamento

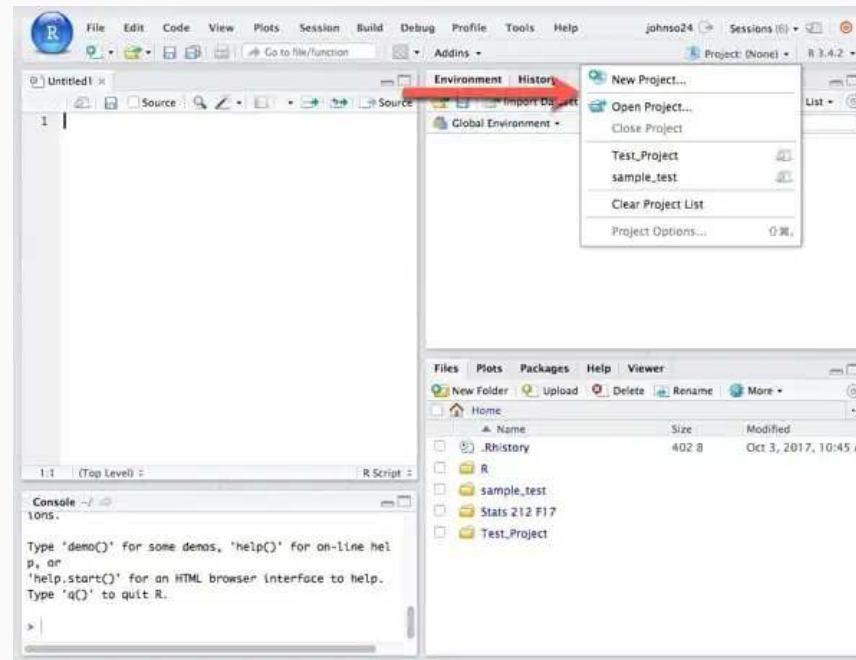


2. RStudio

Projeto R (.Rproj) (Tesseract)

Sempre **abram o RStudio** pelo arquivo **.Rproj**

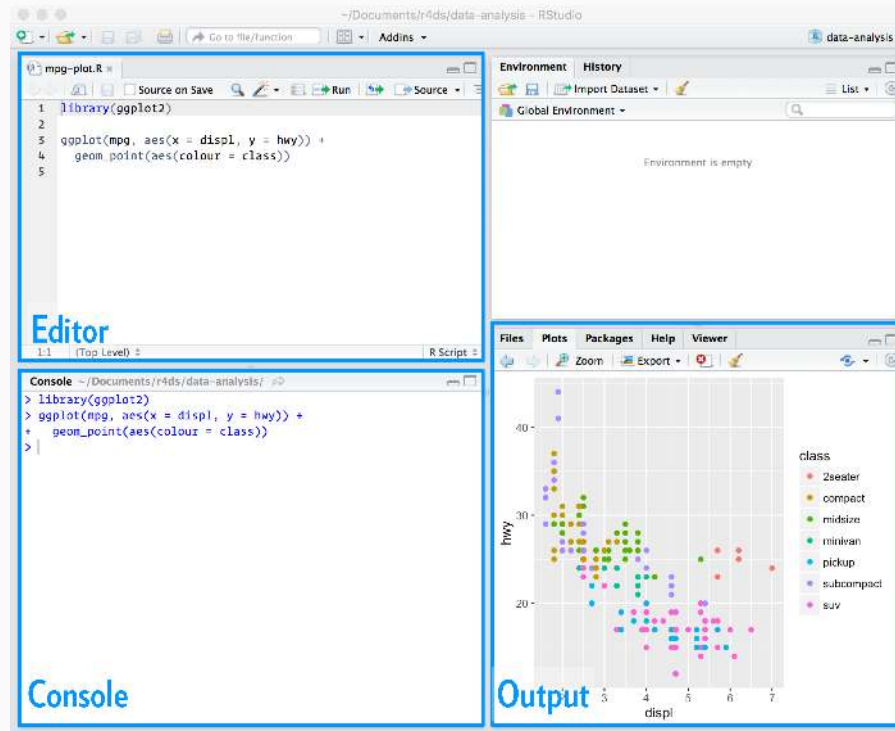
Ou **definem o projeto** depois de abrir o RStudio



3. Console

Console

O console é onde a **linguagem R instalada é carregada** para executar os códigos



3. Console

Console

O console é onde a **linguagem R instalada é carregada** para executar os códigos

```
10 + 2
```

```
## [1] 12
```

```
1:42
```

```
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 3
```

Alguém notou alguns colchetes a mais?

3. Console

Colchetes

Indicam a **posição** em uma sequência de elementos

```
10 + 2
```

```
## [1] 12
```

```
1:42
```

```
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
```

```
10:60
```

```
## [1] 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
```

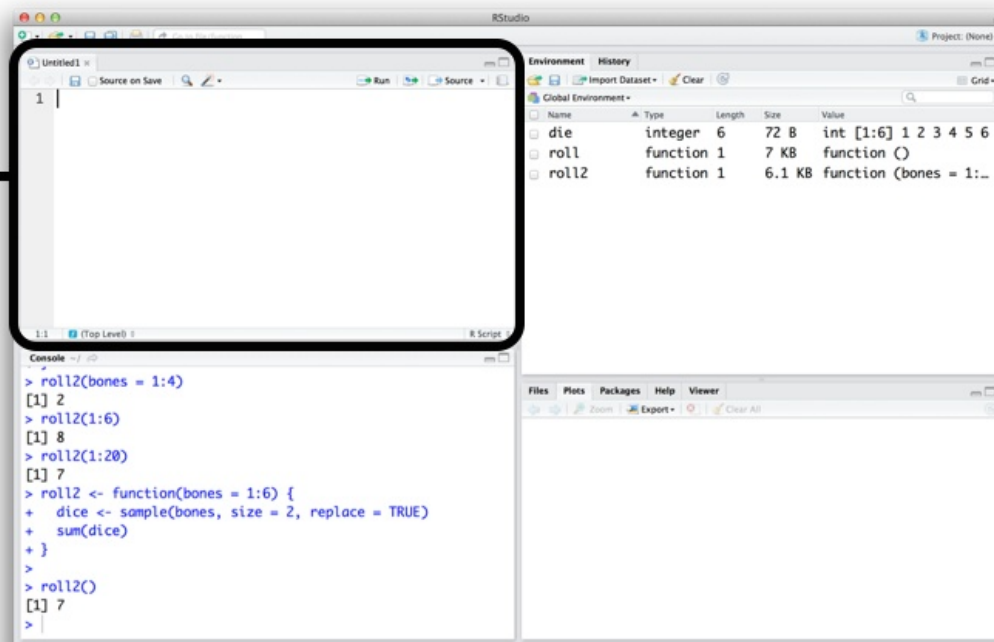
4. Scripts

Script (.R)

Atalho: Ctrl + Shift + N

- São **rascunhos** dos códigos
- Será neles que os **códigos serão escritos** e depois **enviados ao console do R**
- São **arquivos de texto simples**, que serão salvos no formato .R

New script



4. Scripts

Script (.R)

Todos os **códigos** devem ser digitados preferencialmente no **script**

Deixem o **cursor** em **qualquer local da linha** e executem essa linha utilizando essa **combinação**:

Atalho: Ctrl + Enter

Vamos testar:

```
1
```

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

```
1 + 2
```

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

E é isso que faremos pelo resto de nossas vidas...

4. Scripts

Esclarecimentos

Isso é texto, não digite no R!

Digitar no script

```
print("Isso é o resultado que deve aparecer no console")
```

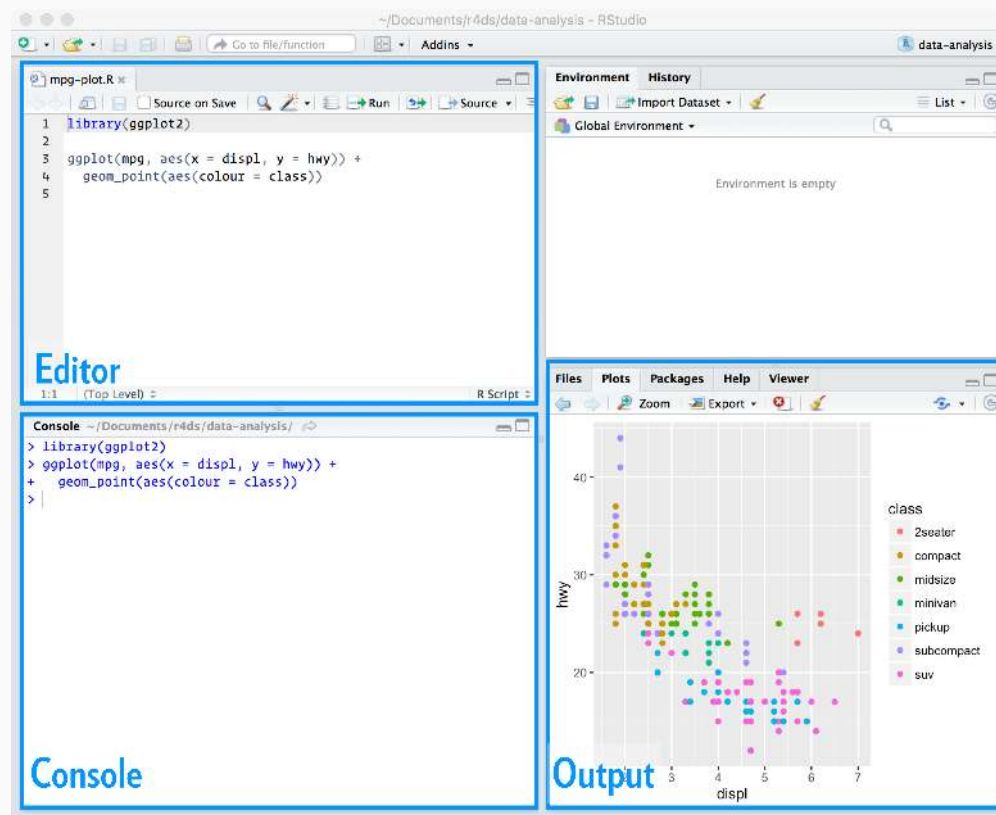
Resultado no console

```
## [1] "Isso é o resultado que deve aparecer no console"
```


4. Scripts

Salvar um script

Atalho: `ctrl + S`



4. Scripts

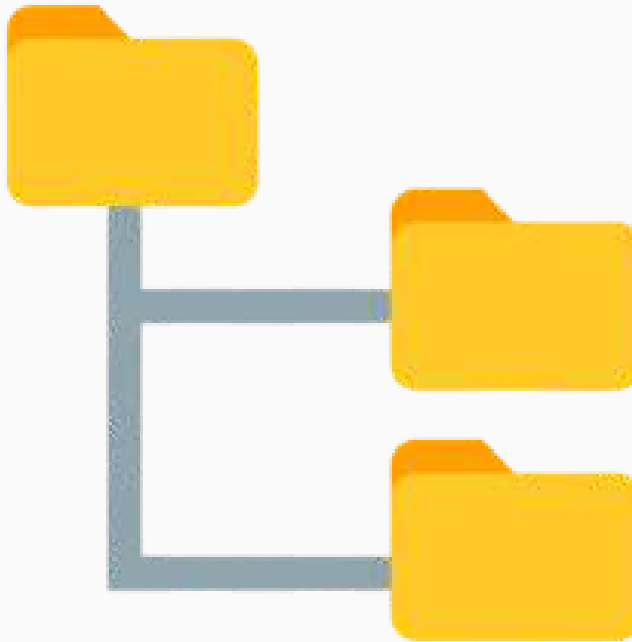
Salvar um script

Pasta do diretório `~/course-geospatial-data-r/`:

- 00_plano_ensino
- 01_slides
- **02_scripts**
- 03_dados

Nome do arquivo

`02_script_intro_geocomp_r.R`



4. Scripts

Comentários (#)

Comentários **não são lidos** pelo R e **descrevem informações** em nosso script

São representados pelo **#** (hash) ou **#'** (hash-linha)

Informações sobre os códigos

```
# comentarios  
# o r nao le o codigo depois do # (hash)  
  
42 # essas palavras nao sao executadas, apenas o 42
```

```
## [1] 42
```

4. Scripts

Comentários (#)

Comentários **não são lidos** pelo R e **descrevem informações** em nosso script

São representados pelo **#** (hash) ou **#'** (hash-linha)

Cabeçalho

```
#' ---  
#' titulo: Introdução ao uso de dados geoespaciais no R  
#' autor: seu nome  
#' data: 25-10-2021  
#' ---
```

5. Operadores

Operadores

Operadores aritméticos (Números)

Operador	Descrição	Uso
+	Adição	$a + b$
-	Subtração	$a - b$
*	Multiplicação	$a * b$
/	Divisão	a / b
%%	Resto da divisão	$a \% b$
/%	Quociente da divisão	$a \div b$
^	Potenciação	a^b

Operadores relacionais (TRUE|FALSE)

Operador	Descrição	Uso
<	Menor	$a < b$
>	Maior	$a > b$
==	Igual	$a == b$
<=	Menor ou igual	$a \leq b$
>=	Maior ou igual	$a \geq b$
!=	Não igual (diferente)	$a != b$

5. Operadores

Ordem das operações aritméticas

`^ >> * ou / >> + ou -`

```
# sem especificar - segue a ordem
```

```
1 * 2 + 2 / 2 ^ 2
```

```
## [1] 2.5
```

```
# especificando - segue a ordem dos parênteses
```

```
((1 * 2) + (2 / 2)) ^ 2
```

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

Exercícios

Exercício 01

Resolvam essa equação...

Escolha a alternativa correta:

$$7 + 7 \div 7 + 7 \times 7 - 7 = ?$$

a) 00

b) 08

c) 50

d) 56

03:00

Exercício 01

Resposta

```
# exercicio 01  
7 + 7 / 7 + 7 * 7 - 7
```

```
## [1] 50
```

6. Objetos

Atribuição (<-)

Objetos são palavras que **atribuímos** dados

A atribuição possibilita a **manipulação** de dados ou resultados de análises

Utilizaremos o símbolo "<" (**menor**) seguido de "-" (**menos**), **sem espaço!!!**

palavra <- dados

Atalho: Alt + -

Using = instead of <- for assignment



6. Objetos

Vamos atribuir o **valor 10** à palavra **obj10**

```
# atribuicao - simbolo (←)  
obj10 ← 10
```

Agora a palavra **obj10** vale **10**

Mas não aconteceu nada....



6. Objetos

Sempre **confira** a atribuição!!!

Chame o objeto **novamente!!!**

```
# atribuicao - simbolo (←)  
obj10 ← 10  
obj10
```

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

Outro exemplo

```
# atribuicao - simbolo (←)  
obj2 ← 2  
obj2
```

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

6. Objetos

CUIDADO!

O R **sobrescreve** os valores dos objetos com o **mesmo nome**!

```
# sobrescreve o valor dos objetos  
obj ← 100  
obj
```

```
## [1] 100
```

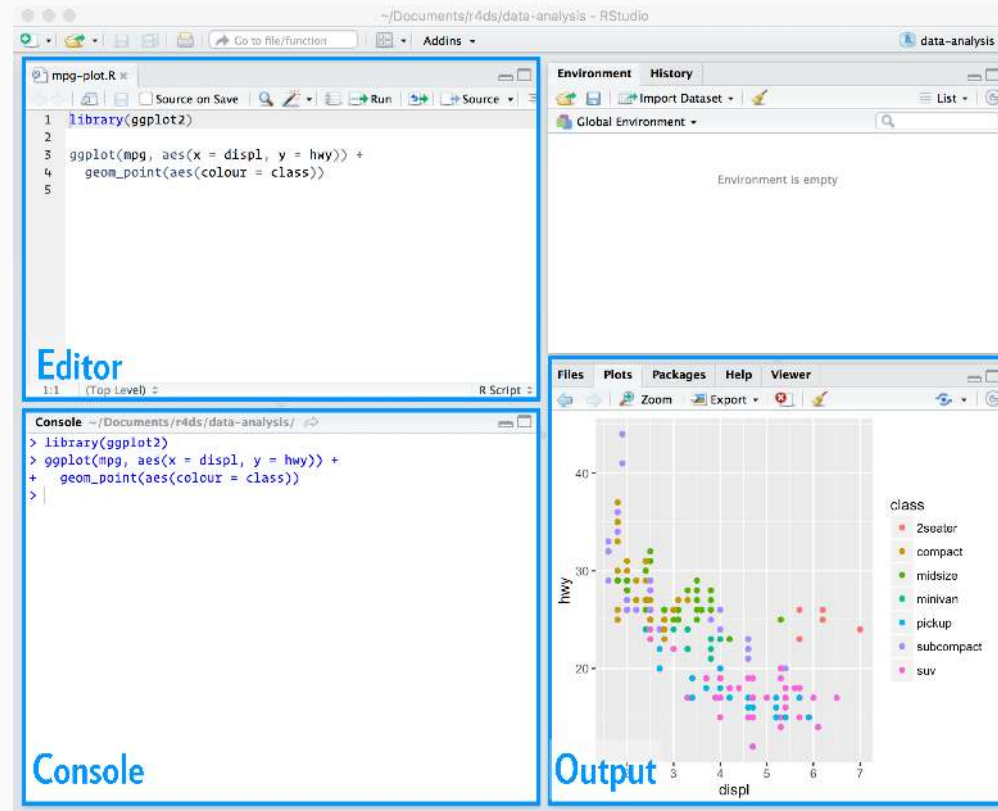
```
# obj agora vale 2  
obj ← 2  
obj
```

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

Os objetos podem ser visualizados no painel
Environment

6. Objetos

Ambiente (*Environment*)



6. Objetos

O R tem **limitações** ao nomear objetos!

1. Nome de objetos só podem **começar por letras (a-z ou A-Z) ou pontos (.)**
2. Nome de objetos só podem **conter letras (a-z ou A-Z), números (0-9), underscores (_) ou pontos (.)**
3. R é *case-sensitive*, i.e., ele difere **letras maiúsculas** de **minúsculas**. Assim, um objeto chamado "*resposta*" é diferente do objeto "*RESPOSTA*"
4. Evitem utilizar **letras maiúsculas, acentos** ou **cedilha (ç)**
5. Nome de objetos não podem ser iguais a **nomes especiais**:

```
break, else, FALSE, for, function, if, Inf, NA, NaN, next, repeat, return, TRUE, while
```


6. Objetos

Podemos utilizar **objetos** para fazer operações

```
# definir dois objetos  
va1 ← 10  
va1
```

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

```
va2 ← 2  
va2
```

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

6. Objetos

Podemos utilizar **objetos** para fazer operações

```
# operacoes com objetos  
va1 + va2 # adicao
```

```
## [1] 12
```

```
va1 - va2 # subtracao
```

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

```
va1 * va2 # multiplicacao
```

```
## [1] 20
```

```
va1 / va2 # divisao
```

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

6. Objetos

Podemos utilizar **objetos** para fazer operações

```
# operacoes com objetos e atribuicao  
adi ← va1 + va2 # adicao  
adi
```

```
## [1] 12
```

```
sub ← va1 - va2 # subtracao  
sub
```

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

```
mul ← va1 * va2 # multiplicacao  
mul
```

```
## [1] 20
```

```
div ← va1 / va2 # divisao  
div
```

Exercícios

Exercício 02

Verifique se 3×2^3 é maior que 2×3^2

03:00

Exercício 02

Resposta

```
# exercicio 02
```

```
3 * 2 ^ 3 >= 2 * 3 ^ 2
```

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

7. Funções

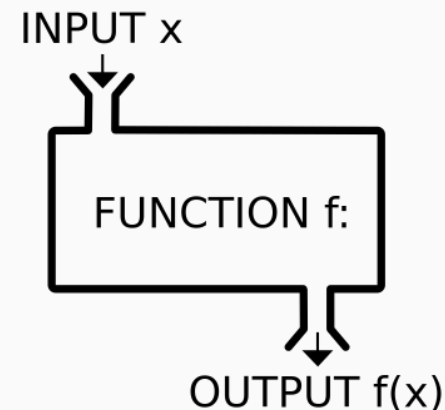
Funções

Códigos que realizam **operações** em **argumentos**

Estrutura de uma função:

```
nome_da_funcao(argumento1, argumento2)
```

1. **Nome da função:** remete ao que ela faz
2. **Parênteses:** limitam a função
3. **Argumentos:** onde a função atuará
4. **Vírgulas:** separam os argumentos



A diagram showing the code `print("hello world")` with red dashed boxes and text annotations. A box around the opening and closing parentheses is labeled "Parentheses contain parameters". A box around the entire `print` statement is labeled "Function Name". A box around the string `"hello world"` is labeled "Parameters separated by commas".

7. Funções

Exemplos

```
# soma  
sum(10, 2)
```

```
## [1] 12
```

```
# soma de objetos  
sum(obj10, obj2)
```

```
## [1] 12
```

```
# soma de objetos atribuidos a objetos  
obj_sum ← sum(obj10, obj2)
```


7. Funções

Argumentos

Os **argumentos** de uma função podem ser de **dois tipos**:

1. **Valores** ou **Objetos**: a função irá **alterar os valores** em si ou os valores **atribuídos** aos objetos
2. **Parâmetros**: valores fixos que informam um **método** ou a realização de uma **operação**. Informa-se o **nome desse argumento**, seguido de "=" e um *número*, *texto* ou *TRUE* ou *FALSE*

Exemplo:

```
sum(1, 2, 3, NA)
```

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

```
sum(1, 2, 3, NA, na.rm = TRUE)
```

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

7. Funções

1. Argumentos como **valores**

```
# funcoes - argumentos como valores  
# soma  
sum(10, 2)
```

```
## [1] 12
```

```
# produto  
prod(10, 2)
```

```
## [1] 20
```

7. Funções

1. Argumentos como **objetos**

```
# funcoes - argumentos como objetos  
# soma  
sum(va1, va2)
```

```
## [1] 12
```

```
# produto  
prod(va1, va2)
```

```
## [1] 20
```

7. Funções

2. Argumentos como **parâmetros**

```
# funcoes - nome dos argumentos  
# repeticao - todos  
rep(x = 1:5, times = 10)
```

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

```
# repeticao - cada  
rep(x = 1:5, each = 10)
```

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

7. Funções

Atribuição de resultados a objetos

```
# atribuicao dos resultados
# repeticao - todos
rep_times ← rep(x = 1:5, times = 10)
rep_times
```

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

```
# atribuicao dos resultados
# repeticao - todos
rep_each ← rep(x = 1:5, each = 10)
rep_each
```

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

7. Funções

Atribuição, função e linha temporal

Criar dois objetos

```
# resposta  
# criar dois objetos  
foo ← 2  
bar ← 3
```

Somar esses objetos e **atribuição** ao objeto *su*

```
# somar e atribuir  
su ← sum(foo, bar)  
su
```

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

Raiz quadrada do *su* e **atribuição** ao *sq*

```
# raiz e atribuir  
sq ← sqrt(su)  
sq
```

7. Funções

Atribuição, função e linha temporal

Esse é o processo de programação no R:

1. **Atribuição** de dados a objetos
2. **Funções** que **operam e mudam** esses dados
3. Nova **atribuição** desses resultados a novos objetos

Exercícios

Exercício 03

Criem dois objetos (qualquer nome) com os valores 100 e 300

Multipliquem esses objetos (função **prod**) e atribuem ao objeto *mult*

Façam o logaritmo natural (função **log**) do objeto *mult* e atribuem ao objeto *ln*

04:00

Exercício 03

Resposta

```
# exercicio 03  
# criar dois objetos  
foo ← 100  
bar ← 300
```

```
# multiplicar e atribuir  
mult ← prod(foo, bar)  
mult
```

```
## [1] 30000
```

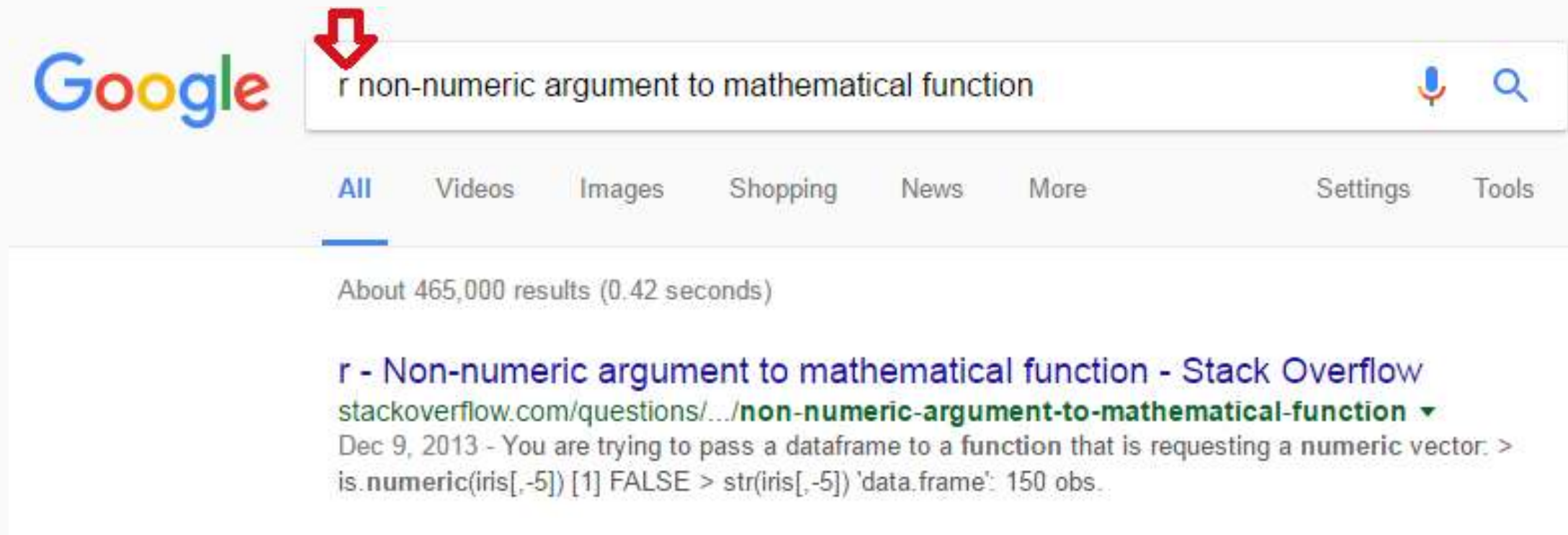
```
# raiz e atribuir  
ln ← log(mult)  
ln
```

```
## [1] 10.30895
```

Nesse momento vocês devem estar se perguntando:
como raio vou saber o nome das funções?!



Uma **maracutaia** para ajudar!



E de onde vêm as funções?!

7. Funções

Funções vêm de **duas fontes**:

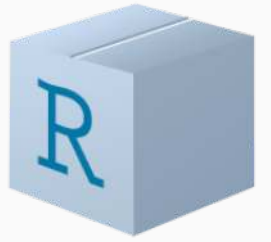
1. Pacotes já **instalados por padrão** e que são **carregados** quando abrimos o R
2. Pacotes que **instalamos** e **carregamos** com códigos



E o que são pacotes afinal?!

8. Pacotes

Coleção de funções para executar tarefas específicas



Duas fontes:

- **CRAN** (*finalizados*)
- **GitHub** (*em desenvolvimento*)

Verificar pacotes carregados

```
# verificar pacotes carregados  
search()
```

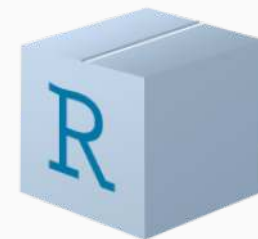
```
## [1] ".GlobalEnv"          "package:countdown"  "package:R.utils"    "package:R.oo"       "package:R.me  
## [6] "package:here"        "package:xaringan"   "package:pagedown"   "package:plotly"     "package:tmap  
## [11] "package:RStoolbox"    "package:fasterize"  "package:rnaturalearth" "package:geobr"      "package:forc  
## [16] "package:stringr"     "package:dplyr"      "package:purrr"      "package:readr"      "package:tidy  
## [21] "package:tibble"      "package:ggplot2"    "package:tidyverse"  "package:sf"         "package:rast  
## [26] "package:sp"          "tools:rstudio"      "package:stats"      "package:graphics"   "package:grDe  
## [31] "package:utils"       "package:datasets"   "package:methods"    "Autoloads"          "package:base
```

8. Pacotes

Coleção de funções para executar **tarefas específicas**

Duas fontes:

- **CRAN** (*finalizados*)
- **GitHub** (*em desenvolvimento*)



Verificar **pacotes instalados**

```
# verificar pacotes instalados  
library()
```

8. Pacotes

Ex.: pacote `vegan`

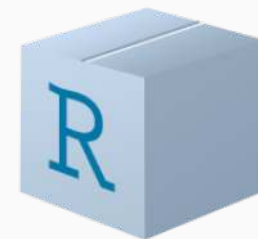
Fontes:

Pacotes do CRAN

<https://cran.r-project.org/web/packages/vegan/index.html>

Pacotes do GitHub

<https://github.com/vegandevs/vegan>



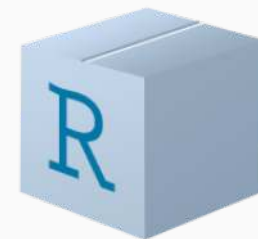
8. Pacotes

Instalar pacotes

1. Instala-se apenas **uma vez**
2. **Precisa** estar conectado à **internet**
3. O **nome do pacote precisa** estar entre **aspas**
4. Função (CRAN):

```
install.packages()
```

```
# instalar pacotes  
install.packages("vegan")
```



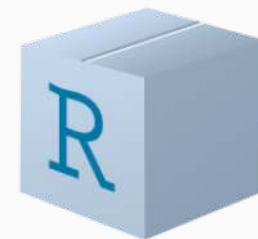
8. Pacotes

Carregar pacotes

1. Carrega-se **toda vez** que se abre **uma nova sessão do R**
2. **Não precisa** estar conectado à **internet**
3. O **nome do pacote não precisa** estar entre **aspas**
4. Funções:

`library()` ou `require()`

```
# carregar pacotes  
library(vegan)
```



8. Pacotes

Instalar pacotes do GitHub

1. Instalar pacote **devtools**



```
# instalar pacote devtools
install.packages("devtools")

# carregar pacote devtools
library(devtools)
```

Instalar usando a função `install_github()`

Atentar para usar essa forma **usuário/repositório**

```
# instalar pacote do github
install_github("vegandevs/vegan")

# carregar pacote do github
library("vegan")
```

8. Pacotes

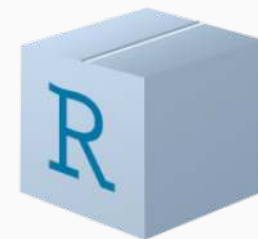
Atualização de pacotes

Pacotes são **atualizados com frequência**

Pacotes **não se atualizam sozinhos**

A instalação de um pacote pode depender da **versão** de outros pacotes

Geralmente é uma função que **demora** para rodar



```
# atualizacao dos pacotes instalados  
update.packages(ask = FALSE, checkBuilt = TRUE)
```

E onde ficam esses pacotes no meu notebook?

8. Pacotes

Diretório dos pacotes instalados

```
# diretorios de instalacao dos pacotes  
.libPaths()[1]
```

```
## [1] "/home/mude/R/x86_64-pc-linux-gnu-library/4.1"
```

Windows

C:/Users/**nome_do_computador**/Documentos/R/win-library/**numero_da_versao_r**

Unix (Linux e MacOS):

/home/**nome_do_computador**/R/**tipo_do_computador**/**numero_da_versao_r**

8. Pacotes

Exemplos:

vegan – análises de comunidades

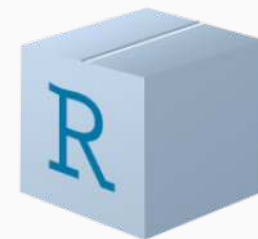
sf – manipulação de vetores

raster – manipulação de rasters

ggplot2 – gráficos

bblme – seleção de modelos (AIC)

tidyverse – data science



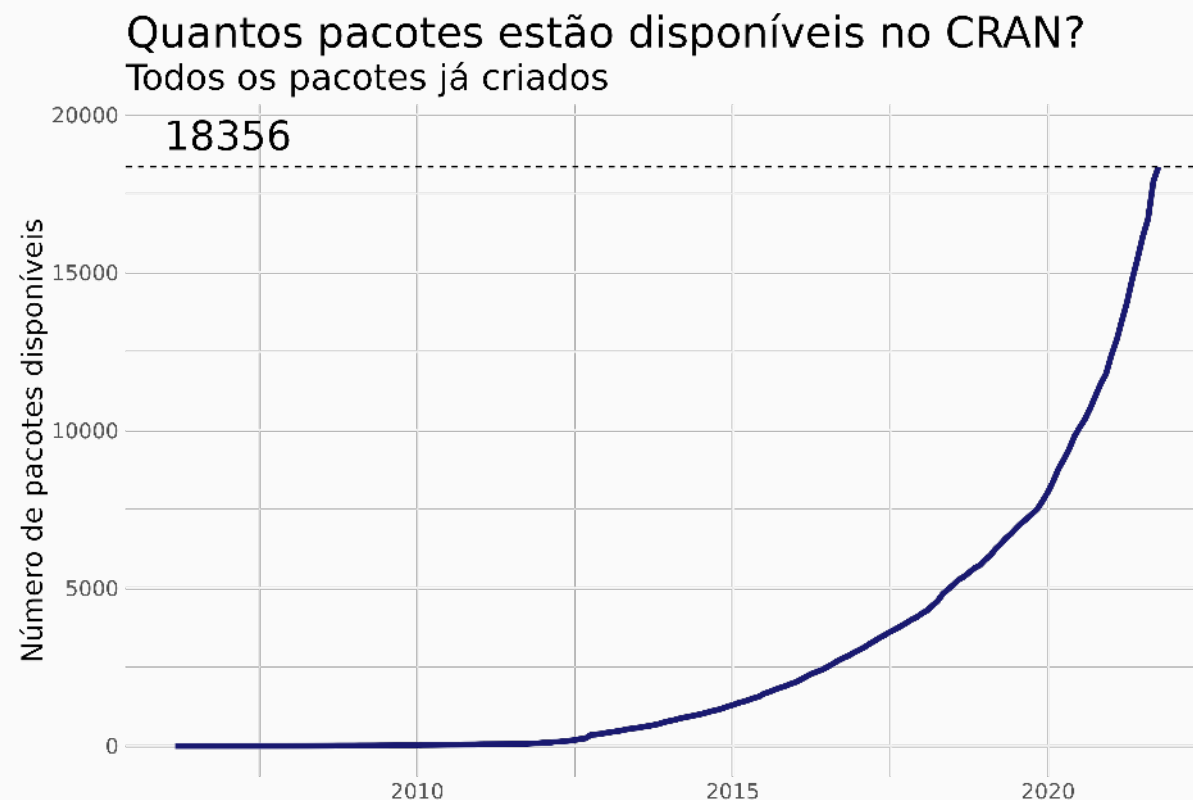
E quantos pacotes existem?

8. Pacotes

Número de pacotes no CRAN (atualmente)

```
nrow(available.packages())
```

```
## [1] 18293
```



Exercícios

Exercício 04

Instalem o pacote **tidyverse** do CRAN

02:00

Exercício 04

Resposta

```
# exercicio 04  
install.packages("tidyverse")
```

9. Ajuda

Descreve as informações de uma função

```
# ajuda  
# descreve as informacoes de uma funcao  
help("mean") # arquivo .html  
?mean
```

- **Description:** resumo da função
- **Usage:** como utilizar a função e quais os seus argumentos
- **Arguments:** detalha os argumentos e como os mesmos devem ser especificados
- **Details:** detalhes importantes para se usar a função
- **Value:** mostra como interpretar a saída (*output*) da função (os resultados)
- **Note:** notas gerais sobre a função
- **Authors:** autores da função
- **References:** referências bibliográficas para os métodos usados para construção da função
- **See also:** funções relacionadas
- **Examples:** exemplos do uso da função. Às vezes pode ser útil copiar esse trecho e colar no R para ver como funciona e como usar a função

9. Ajuda

Descreve as informações de uma função

mean (base) R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)
```

```
## Default S3 method:  
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

x An R object. Currently there are methods for numeric/logical vectors and [date](#), [date-time](#) and [time interval](#) objects. Complex vectors are allowed for `trim = 0`, only.

trim the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.

na.rm a logical value indicating whether NA values should be stripped before the computation proceeds.

... further arguments passed to or from other methods.

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

[weighted.mean](#), [mean.POSIXct](#), [colMeans](#) for row and column means.

Examples

```
x <- c(0:10, 50)  
xm <- mean(x)  
c(xm, mean(x, trim = 0.10))
```

9. Ajuda

Detalhes de um pacote

```
# detalhes de um pacote  
library(help = "vegan")
```

- Descrição
- Versão
- Autores
- Dependências
- Sites
- Repositório
- Índice de funções
- Diretório

```
Information on package 'vegan'  
  
Description:  
Package:      vegan  
Title:        Community Ecology Package  
Version:      2.5-6  
Author:       Jari Oksanen, F. Guillaume Blanchet, Michael Friendly, Roeland Kindt, Pierre Legendre, Dan  
              McGlinn, Peter R. Minchin, R. B. O'Hara, Gavin L. Simpson, Peter Solymos, M. Henry H. Stevens,  
              Eduard Szoecs, Helene Wagner  
Maintainer:   Jari Oksanen <jhoksane@gmail.com>  
Depends:      permute (>= 0.9-0), lattice, R (>= 3.4.0)  
Suggests:     parallel, tcltk, knitr  
Imports:      MASS, cluster, mgcv  
VignetteBuilder:  utils, knitr  
Description:  Ordination methods, diversity analysis and other functions for community and vegetation  
              ecologists.  
License:      GPL-2  
BugReports:   https://github.com/vegandevs/vegan/issues  
URL:          https://cran.r-project.org, https://github.com/vegandevs/vegan  
NeedsCompilation: yes  
Packaged:     2019-08-31 07:21:37 UTC; jarioksa  
Repository:   CRAN  
Date/Publication: 2019-09-01 14:30:02 UTC  
Built:        R 4.0.2; x86_64-pc-linux-gnu; 2020-08-21 17:53:22 UTC; unix  
  
Index:
```

Todos se lembram da atribuição e criação de objetos?

```
palavra <- dados
```

10. Ambiente

Tamanho dos objetos

```
# tamanho de um objeto
object.size("bar")
```

```
## 112 bytes
```

Listar todos os objetos criados

```
# listar objetos
ls()
```

```
## [1] "adi"
## [4] "bio01_laea"
## [7] "bioclim_bio01"
## [10] "dem_rc"
## [13] "dem_rc_crop"
## [16] "dem_rc_dec"
## [19] "dem_rc_log10"
## [22] "dem_rc_rcl"
## [25] "dem_rc_utm23s_agre_media_linhas"
## [28] "dem_rc_zonal"

"b"
"bio01_moll"
"biomas"
"dem_rc_abaixo_500"
"dem_rc_crop_mask"
"dem_rc_focal_sd"
"dem_rc_mask"
"dem_rc_utm23s"
"dem_rc_utm23s_agre_media_pontos"
"dem_rc2"

"bar"
"bioclim"
"dem"
"dem_rc_acima_600"
"dem_rc_crop_mask_inv"
"dem_rc_global_dist"
"dem_rc_prod"
"dem_rc_utm23s_agre_m
"dem_rc_utm23s_dis_bi
"div"
93/124
```

10. Ambiente

CUIDADO!

Toda a vez que **fechamos o R**, os objetos criados são **apagados**!



10. Ambiente

Salvar todos os objetos criados (.RData)

Session -> Save Workspace As... -> meus_objetos.RData

```
# exportar objetos  
save.image("todos_meus_objetos.RData")
```

Carregar os objetos criados e salvos

Session -> Load Workspace... -> meus_objetos.RData

```
# importar objetos  
load("todos_meus_objetos.RData")
```

10. Ambiente

Salvar todos os objetos criados (.RData)

```
# salvar apenas um objeto  
save(bar, file = "meu_obj.RData")  
  
# salvar apenas um objeto  
save(bar, adi, file = "meus_objs.RData")
```

Carregar os objetos criados e salvos

```
# carregar os objetos  
load("meus_objs.RData")
```


10. Ambiente

Salvar todos os objetos criados (.rds)

```
# salvar um objeto para um arquivo  
saveRDS(obj, file = "meu_obj.rds")
```

Carregar os objetos criados e salvos

```
# carregar esse objeto  
readRDS(file = "meu_obj.rds")
```

10. Ambiente

Remover um objeto

```
# listar objetos  
ls()
```

```
## [1] "adi"  
## [4] "bio01_laea"  
## [7] "bioclim_bio01"  
## [10] "dem_rc"  
## [13] "dem_rc_crop"  
## [16] "dem_rc_dec"  
## [19] "dem_rc_log10"  
## [22] "dem_rc_rcl"  
## [25] "dem_rc_utm23s_agre_media_linhas"  
## [28] "dem_rc_zonal"  
## [31] "files"  
## [34] "laea"  
## [37] "ln"  
## [40] "moll"  
## [43] "obj"  
## [46] "obj2"  
## [49] "ra_layer1"  
## [52] "ra_layer4"  
  
"b"  
"bio01_moll"  
"biomas"  
"dem_rc_abaixo_500"  
"dem_rc_crop_mask"  
"dem_rc_focal_sd"  
"dem_rc_mask"  
"dem_rc_utm23s"  
"dem_rc_utm23s_agre_media_pontos"  
"dem_rc2"  
"foo"  
"landsat_rc"  
"map_biomass_plotly_int"  
"mul"  
"obj_sum"  
"ra_brick"  
"ra_layer2"  
"ra_stack"  
  
"bar"  
"bioclim"  
"dem"  
"dem_rc_acima_600"  
"dem_rc_crop_mask_inv"  
"dem_rc_global_dist"  
"dem_rc_prod"  
"dem_rc_utm23s_agre_m"  
"dem_rc_utm23s_dis_bi"  
"div"  
"i"  
"landsat_rc_ndvi"  
"map_rc_2020_plotly_i"  
"mult"  
"obj10"  
"ra_layer"  
"ra_layer3"  
"rc_2020"
```

10. Ambiente

Remover todos os objetos

```
# listar objetos  
ls()
```

```
## [1] "adi"  
## [4] "bio01_moll"  
## [7] "biomas"  
## [10] "dem_rc_abaixo_500"  
## [13] "dem_rc_crop_mask"  
## [16] "dem_rc_focal_sd"  
## [19] "dem_rc_mask"  
## [22] "dem_rc_utm23s"  
## [25] "dem_rc_utm23s_agre_media_pontos"  
## [28] "dem_rc2"  
## [31] "foo"  
## [34] "landsat_rc"  
## [37] "map_biomas_plotly_int"  
## [40] "mul"  
## [43] "obj_sum"  
## [46] "ra_brick"  
## [49] "ra_layer2"  
## [52] "ra_stack"  
  
"b"  
"bioclim"  
"dem"  
"dem_rc_acima_600"  
"dem_rc_crop_mask_inv"  
"dem_rc_global_dist"  
"dem_rc_prod"  
"dem_rc_utm23s_agre_media"  
"dem_rc_utm23s_dis_bil"  
"div"  
"i"  
"landsat_rc_ndvi"  
"map_rc_2020_plotly_int"  
"mult"  
"obj10"  
"ra_layer"  
"ra_layer3"  
"rc_2020"  
  
"bio01_laea"  
"bioclim_bio01"  
"dem_rc"  
"dem_rc_crop"  
"dem_rc_dec"  
"dem_rc_log10"  
"dem_rc_rcl"  
"dem_rc_utm23s_agre_m"  
"dem_rc_zonal"  
"files"  
"laea"  
"ln"  
"moll"  
"obj"  
"obj2"  
"ra_layer1"  
"ra_layer4"  
"rc_2020_utm23s"
```

10. Ambiente

Carregar os objetos criados e salvos

Session -> Load Workspace... -> meus_objetos.RData

```
# rodem para verificar
```

```
ls()
```

```
## [1] "adi"      "bar"      "foo"      "lo"       "mu"       "obj"      "obj_10"   "obj_2"    "rep_each" "  
## [12] "su"       "sub"      "va1"      "va2"
```

11. Citações

Como citar o R e os pacotes em trabalhos?

```
## citacao do r e dos pacotes
```

```
# citacao do R  
citation()
```

```
##
```

```
## To cite R in publications use:
```

```
##
```

```
## R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing  
## Austria. URL https://www.R-project.org/.
```

```
##
```

```
## A BibTeX entry for LaTeX users is
```

```
##
```

```
## @Manual{,
```

```
## title = {R: A Language and Environment for Statistical Computing},
```

```
## author = {{R Core Team}},
```

```
## organization = {R Foundation for Statistical Computing},
```

```
## address = {Vienna, Austria},
```

```
## year = {2021},
```

```
## url = {https://www.R-project.org/},
```

```
## }
```

11. Citações

Como citar o R e os pacotes em trabalhos?

```
# citacao dos pacotes  
citation("vegan")
```

```
##  
## To cite package 'vegan' in publications use:  
##  
##   Jari Oksanen, F. Guillaume Blanchet, Michael Friendly, Roeland Kindt, Pierre Legendre, Dan McGlinn, Peter R. Min  
##   O'Hara, Gavin L. Simpson, Peter Solymos, M. Henry H. Stevens, Eduard Szoecs and Helene Wagner (2020). vegan: Com  
##   Package. R package version 2.5-7. https://CRAN.R-project.org/package=vegan  
##  
## A BibTeX entry for LaTeX users is  
##  
##   @Manual{,  
##     title = {vegan: Community Ecology Package},  
##     author = {Jari Oksanen and F. Guillaume Blanchet and Michael Friendly and Roeland Kindt and Pierre Legendre an  
##     year = {2020},  
##     note = {R package version 2.5-7},  
##     url = {https://CRAN.R-project.org/package=vegan},  
##   }  
##
```

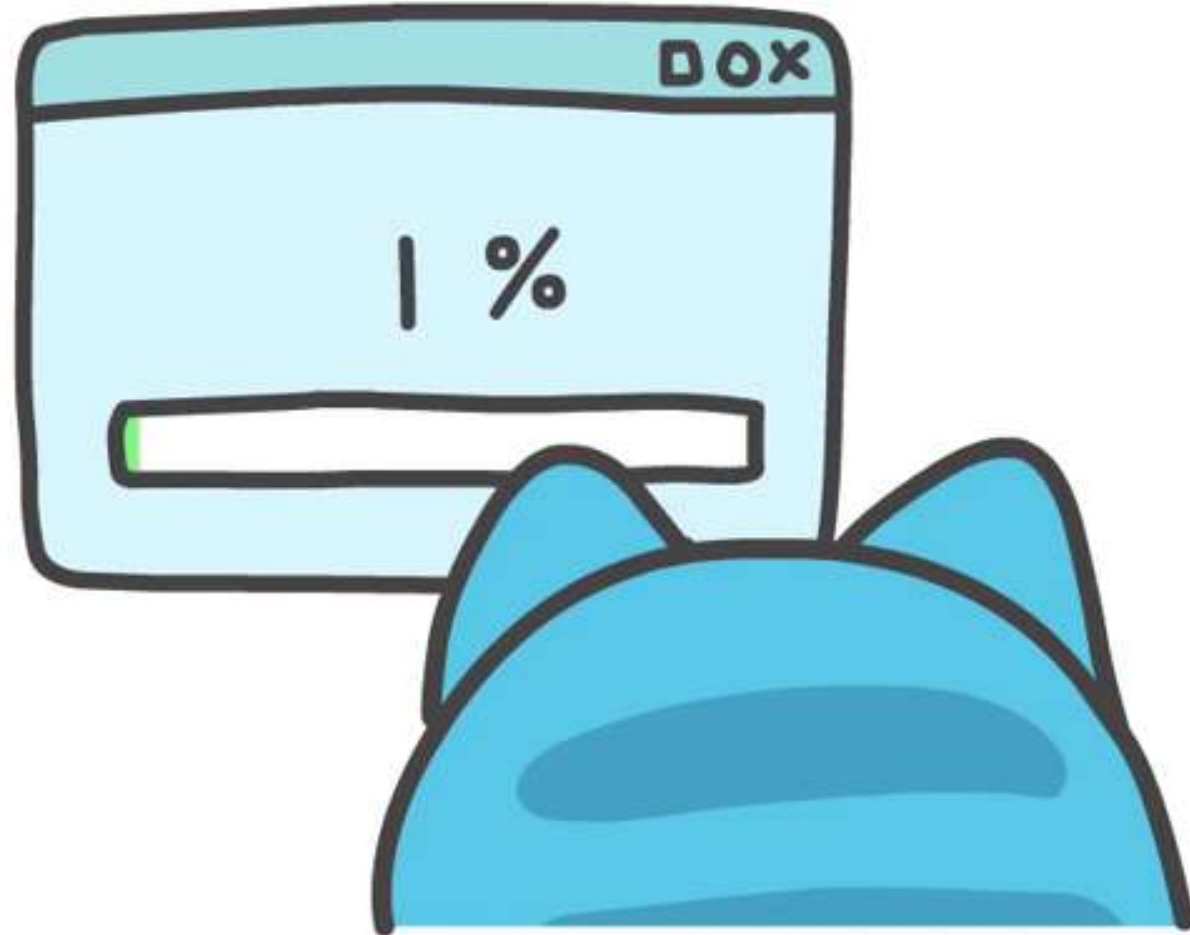
Erros!!!

Se seu script rodou sem erros, tem algo errado...
(Maurício Vancine)



@INSTADEVANEIOS

YORHÁN ARAÚJO



I'm fine



12. Principais erros

1. Esquecer de completar um código (+)

Parênteses

```
sum(1, 2  
+
```

```
## Error: <text>:3:0: unexpected end of input  
## 1: sum(1, 2  
## 2: +  
##      ^
```

Aspas

```
"string  
+
```

```
## Error: <text>:1:1: unexpected INCOMPLETE_STRING  
## 1: "string  
## 2: +  
##      ^
```

12. Principais erros

2. Esquecer da vírgula

```
sum(1 2)
```

```
## Error: <text>:1:7: unexpected numeric constant
```

```
## 1: sum(1 2
```

```
##           ^
```

12. Principais erros

3. Chamar um objeto errado

```
obj ← 10  
OBJ
```

```
## Error in eval(expr, envir, enclos): object 'OBJ' not found
```

12. Principais erros

4. Esquecer de carregar um pacote

```
detach("package:vegan", unload = TRUE)
```

```
# carregar dados  
data(dune)  
  
# funcao do pacote vegan  
decostand(dune, "hell")
```

```
## Error in decostand(dune, "hell"): could not find function "decostand"
```

12. Principais erros

4. Esquecer de carregar um pacote

```
# carregar o pacote
library(vegan)

# carregar dados
data(dune)

# funcao do pacote vegan
decostand(dune, "hell")
```

##		Achimill	Agrostol	Airaprae	Alopgeni	Anthodor	Bellpere	Bromhord	Chenalbu	Cirsarve	Comapalu	Eleopal
## 1		0.2357023	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
## 2		0.2672612	0.0000000	0.0000000	0.2182179	0.0000000	0.2672612	0.3086067	0.0000000	0.0000000	0.0000000	0.0000000
## 3		0.0000000	0.3162278	0.0000000	0.4183300	0.0000000	0.2236068	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
## 4		0.0000000	0.4216370	0.0000000	0.2108185	0.0000000	0.2108185	0.2581989	0.0000000	0.2108185	0.0000000	0.0000000
## 5		0.2156655	0.0000000	0.0000000	0.0000000	0.3049971	0.2156655	0.2156655	0.0000000	0.0000000	0.0000000	0.0000000
## 6		0.2041241	0.0000000	0.0000000	0.0000000	0.2500000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
## 7		0.2236068	0.0000000	0.0000000	0.0000000	0.2236068	0.0000000	0.2236068	0.0000000	0.0000000	0.0000000	0.0000000
## 8		0.0000000	0.3162278	0.0000000	0.3535534	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.3162278
## 9		0.0000000	0.2672612	0.0000000	0.2672612	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
## 10		0.3049971	0.0000000	0.0000000	0.0000000	0.3049971	0.2156655	0.3049971	0.0000000	0.0000000	0.0000000	0.0000000
## 11		0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000

12. Principais erros

5. Usar o nome da função de forma errônea

```
colsums(dune)
```

```
## Error in colsums(dune): could not find function "colsums"
```



12. Principais erros

5. Usar o nome da função de forma errônea

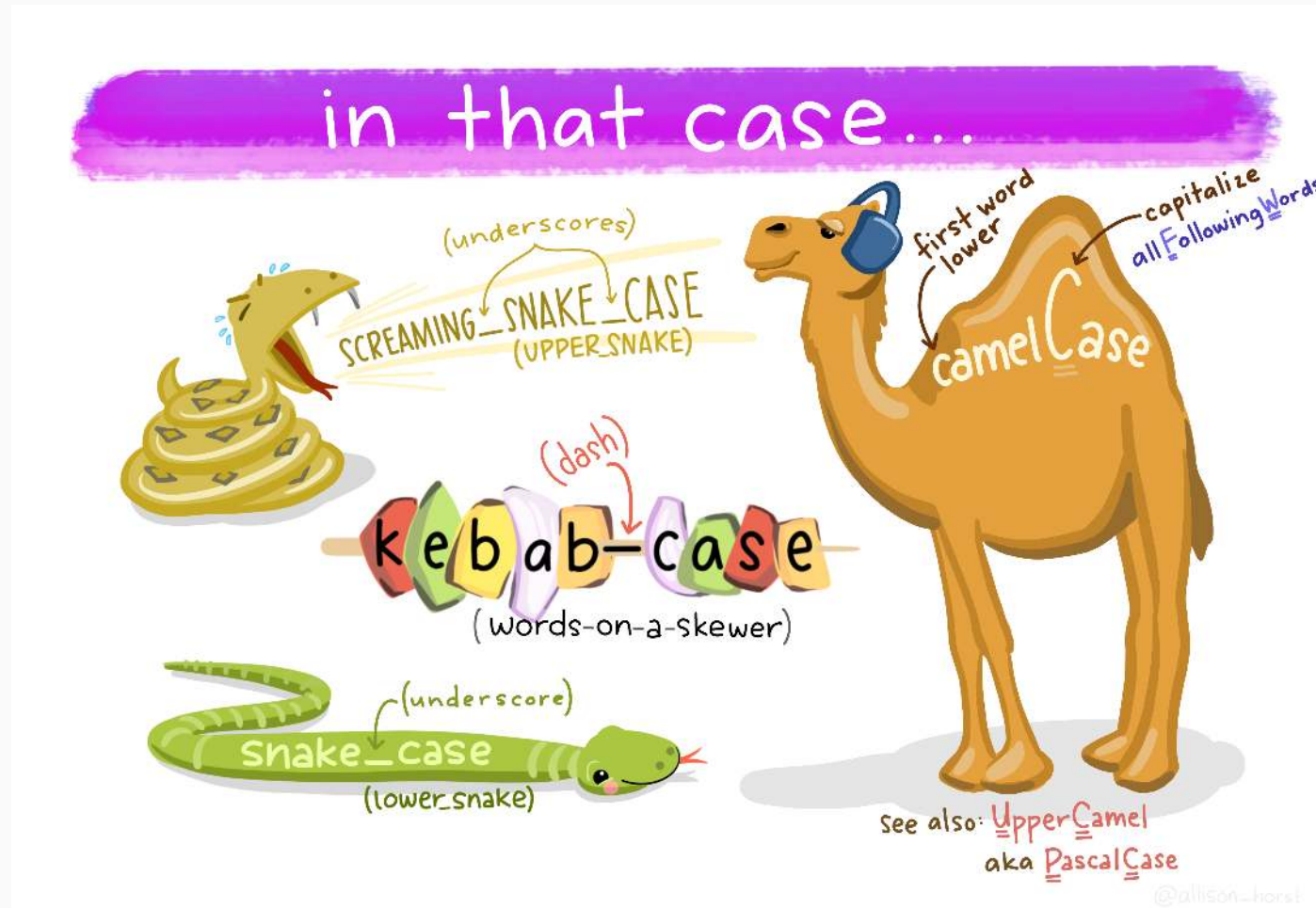
```
colSums(dune)
```

```
## Achimill Agrostol Airaprae Alop geni Anthodor Bellpere Bromhord Chenalbu Cirsarve Comapalu Eleopal u Elymrepe Empeni  
##      16      48       5      36      21      13      15       1       2       4      25      26  
## Lolipere Planlanc Poaprat Poatriv Ranuflam Rumeacet Sagiproc Salirepe Scorautu Trifprat Trifrepe Vicilath Bracru  
##      58      26      48      63      14      18      20      11      54       9      47       4
```



12. Principais erros

Cases



12. Principais erros

6. Atentar para o diretório correto

```
# listar os arquivos do diretorio definido  
dir()[1:4]
```

```
## [1] "00_slides_intro_geoespacial_r.html" "00_slides_intro_geoespacial_r.pdf" "00_slides_intro_geoespacial_r.Rmd"  
## [4] "01_slides_intro_geoespacial_r.html"
```

```
# listar os arquivos do diretorio definido  
list.files()[1:4]
```

```
## [1] "00_slides_intro_geoespacial_r.html" "00_slides_intro_geoespacial_r.pdf" "00_slides_intro_geoespacial_r.Rmd"  
## [4] "01_slides_intro_geoespacial_r.html"
```

```
# listar os arquivos do diretorio definido por um padrão  
dir(pattern = ".Rmd")
```

```
## [1] "00_slides_intro_geoespacial_r.Rmd" "01_slides_intro_geoespacial_r.Rmd" "02_slides_intro_geoespacial_r.Rmd"  
## [4] "03_slides_intro_geoespacial_r.Rmd" "04_slides_intro_geoespacial_r.Rmd" "05_slides_intro_geoespacial_r.Rmd"  
## [7] "06_slides_intro_geoespacial_r.Rmd" "07_slides_intro_geoespacial_r.Rmd" "08_slides_intro_geoespacial_r.Rmd"  
## [10] "09_slides_intro_geoespacial_r.Rmd"
```

13. Principal material de estudo

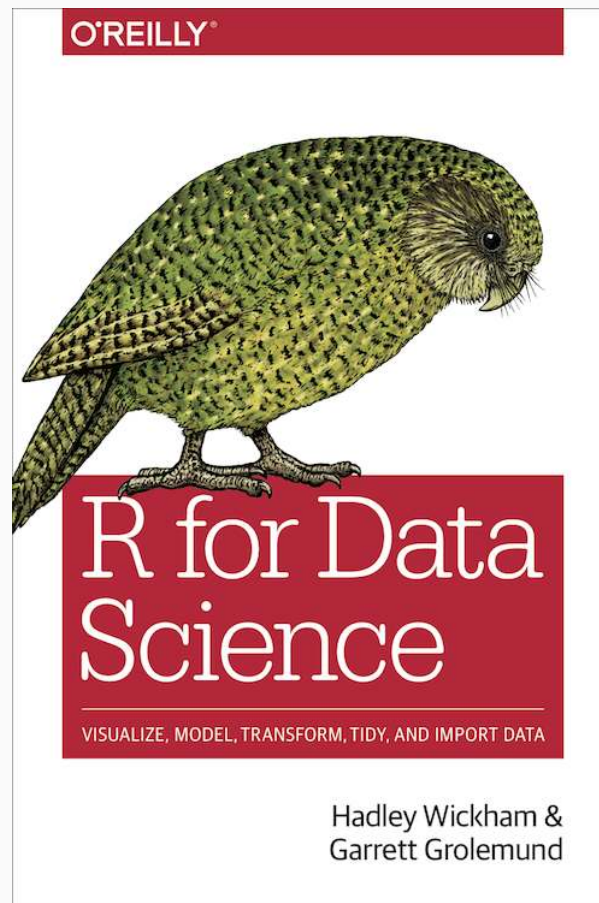
Ciência de Dados com R: introdução (2018)



[Guerra et al. \(2020\)](#)

13. Principal material de estudo

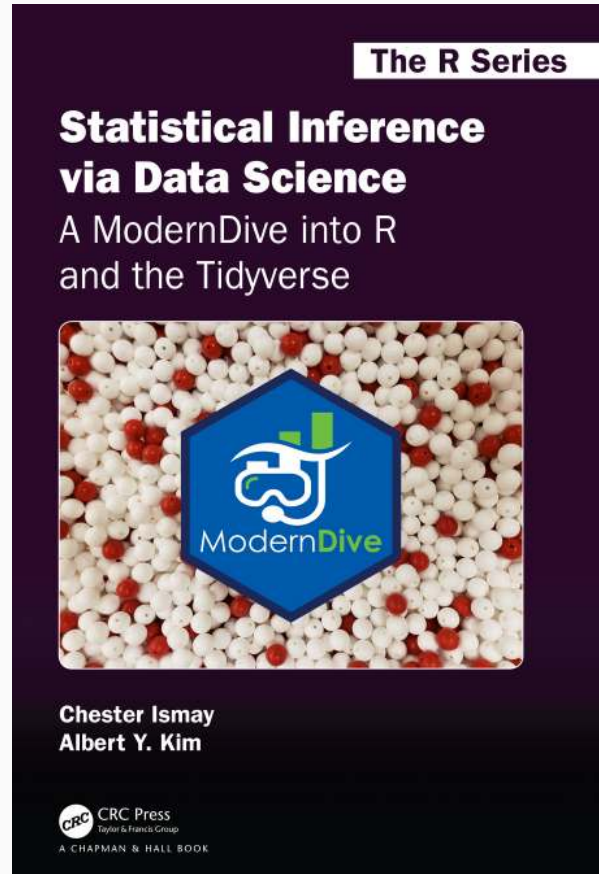
R for Data Science (2017)



[Wickham & Grolemund \(2017\)](#)

13. Principal material de estudo

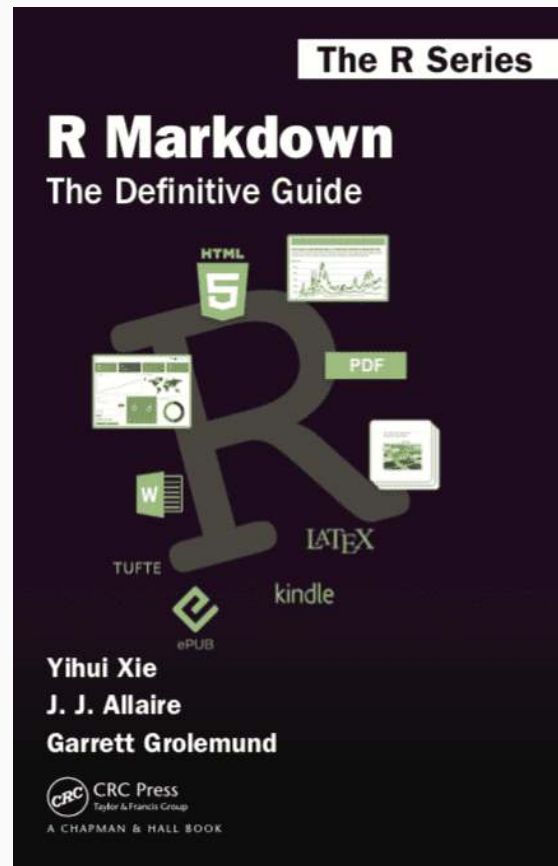
Statistical Inference via Data Science (2019)



[Ismay & Kim \(2019\)](#).

13. Principal material de estudo

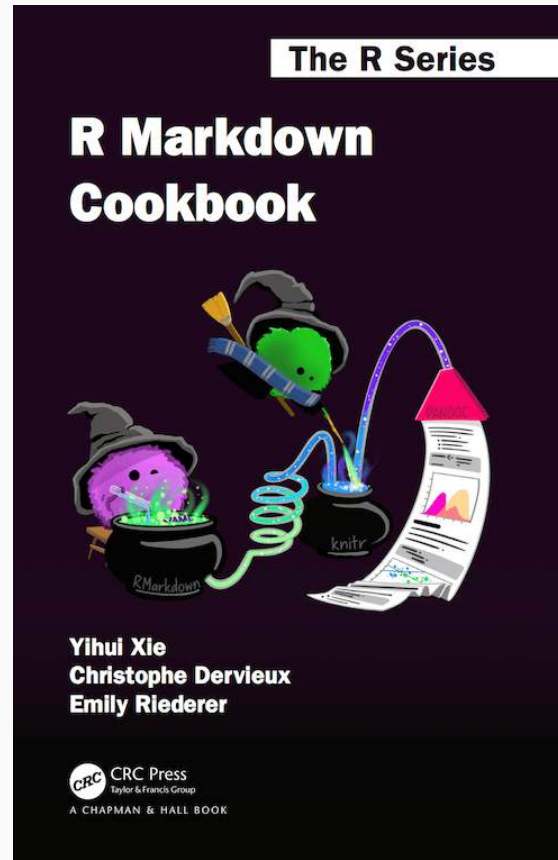
R Markdown: The Definitive Guide (2018)



[Xie et al. \(2019\)](#).

13. Principal material de estudo

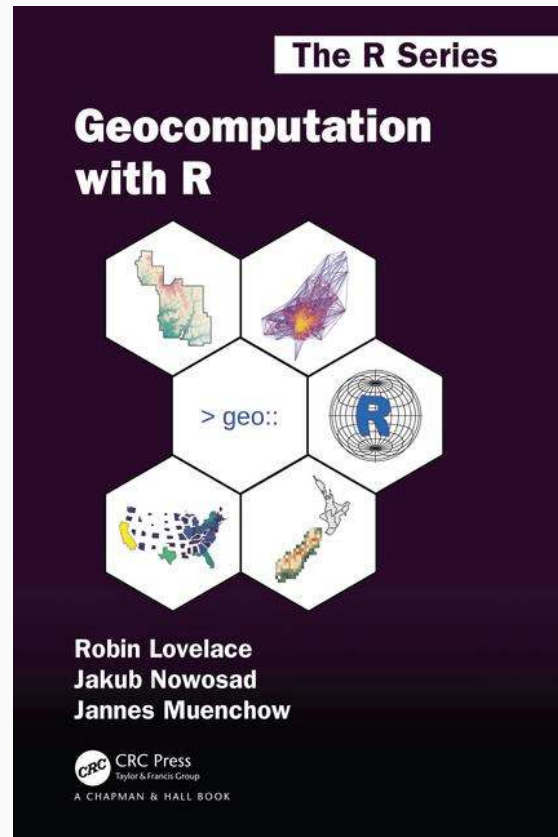
R Markdown Cookbook (2021)



[Xie et al. \(2021\)](#)

13. Principal material de estudo

Geocomputation with R (2019)



[Lovelace et al. \(2019\).](#)

Dúvidas?

Maurício Vancine

Contatos:

✉ mauricio.vancine@gmail.com

🐦 [@mauriciovancine](https://twitter.com/mauriciovancine)

🌀 [mauriciovancine](https://github.com/mauriciovancine)

🔗 mauriciovancine.github.io



Slides criados via pacote [xaringan](https://github.com/jjallaire/xaringan) e tema [Metropolis](https://github.com/jjallaire/metropolis). Animação dos sapos por [@probzz](https://twitter.com/probzz).