

## **Estatística para Computação**

**Professora Priscila Ferraz Franczak**

Engenheira Ambiental – UNIVILLE

Especialista em Emergências Ambientais - PUCPR

Mestre em Ciência e Engenharia de Materiais - UDESC

Doutora em Ciência e Engenharia de Materiais - UDESC

[priscila.franczak@gmail.com](mailto:priscila.franczak@gmail.com)

# Plano de Aula

## **1. Sobre o R**

1.1 Introdução

1.2 O programa

1.3 Como instalar

## **2. Iniciando o RStudio**

2.1 Símbolos ou comandos importantes

2.2 Ajuda

2.3 Atribuição de valores

2.4 Tipos de dados

2.5 Operações aritméticas

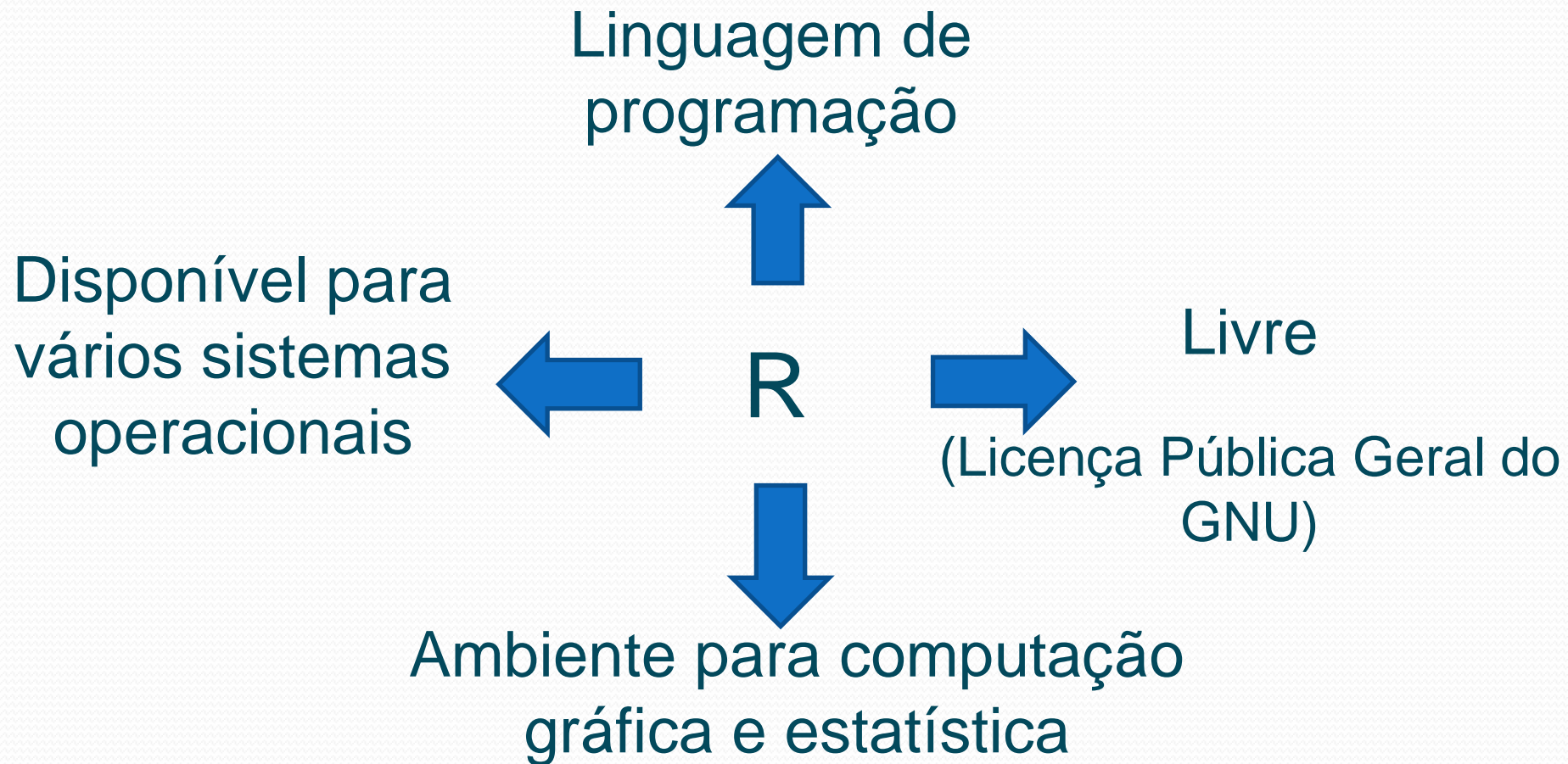
2.6 Manipulando objetos e seus atributos.

## **3. Exercícios**

# 1. Sobre o R

## 1.1 Introdução

- O R é uma linguagem orientada a objetos
- Foi criada em 1996 originalmente por Ross Ihaka e por Robert Gentleman na Universidade de Auckland, Nova Zelândia.
- O nome R provém em parte das iniciais dos criadores e também de um jogo figurado com a linguagem S.



## 1.2 O programa

- É uma série integrada de instalações de softwares para manipulação de dados, cálculo e exibição gráfica. **Possui:**
- Manipulação de dados eficaz;
- Facilidade de armazenamento;

- Extensa, coerente e integrada coleção de ferramentas intermediárias para **análise de dados**;
- Instalações gráficas para análise de dados e exibição tanto direta no computador quanto para cópia permanente.

- Várias pessoas utilizam o R como um **sistema estatístico**.
- Proporciona um ambiente interior com várias **técnicas estatísticas**, clássicas e modernas, que foram implementadas dentro do software.
- Algumas estão compiladas dentro da base do ambiente R ou como pacotes.
- Há em torno de 25 pacotes disponíveis com R.

# 1.3 Como instalar

## Através do site: <http://www.r-project.org>

<https://www.r-project.org>



[\[Home\]](#)

**Download**

[CRAN](#)

**R Project**

[About R](#)

[Logo](#)

[Contributors](#)

[What's New?](#)

[Reporting Bugs](#)

[Development Site](#)

[Conferences](#)

[Search](#)

**R Foundation**

[Foundation](#)

[Board](#)

[Members](#)

[Donors](#)

[Donate](#)

# 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 MacOS. 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

- **R version 3.4.3 (Kite-Eating Tree)** has been released on 2017-11-30.
- **The R Journal Volume 9/1** is available.
- **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



## CRAN Mirrors

The Comprehensive R Archive Network is available at the following URLs, please choose a location close to you. Some statistics on the status of the mirrors can be found here: [main page](#), [windows release](#), [windows old release](#).

If you want to host a new mirror at your institution, please have a look at the [CRAN Mirror HOWTO](#).

### 0-Cloud

<https://cloud.r-project.org/>  
<http://cloud.r-project.org/>

Automatic redirection to servers worldwide, currently sponsored by Rstudio  
Automatic redirection to servers worldwide, currently sponsored by Rstudio

### Algeria

<https://cran.usthb.dz/>  
<http://cran.usthb.dz/>

University of Science and Technology Houari Boumediene  
University of Science and Technology Houari Boumediene

### Argentina

<http://mirror.fcaglp.unlp.edu.ar/CRAN/>

Universidad Nacional de La Plata

### Australia

<https://cran.csiro.au/>  
<http://cran.csiro.au/>  
<https://mirror.aarnet.edu.au/pub/CRAN/>  
<https://cran.ms.unimelb.edu.au/>  
<https://cran.curtin.edu.au/>

CSIRO  
CSIRO  
AARNET  
School of Mathematics and Statistics, University of Melbourne  
Curtin University of Technology

### Belgium

<http://www.freeststatistics.org/cran/>  
<https://lib.ugent.be/CRAN/>  
<http://lib.ugent.be/CRAN/>

K.U.Leuven Association  
Ghent University Library  
Ghent University Library

### Brazil

<http://nbcgib.uesc.br/mirrors/cran/>  
<https://cran-r.c3sl.ufpr.br/>  
<http://cran-r.c3sl.ufpr.br/>  
<https://cran.fiocruz.br/>  
<http://cran.fiocruz.br/>  
<https://vps.fmvz.usp.br/CRAN/>  
<http://vps.fmvz.usp.br/CRAN/>  
<https://brieger.esalq.usp.br/CRAN/>  
<http://brieger.esalq.usp.br/CRAN/>

Center for Comp. Biol. at Universidade Estadual de Santa Cruz  
Universidade Federal do Parana  
Universidade Federal do Parana  
Oswaldo Cruz Foundation, Rio de Janeiro  
Oswaldo Cruz Foundation, Rio de Janeiro  
University of Sao Paulo, Sao Paulo  
University of Sao Paulo, Sao Paulo  
University of Sao Paulo, Piracicaba  
University of Sao Paulo, Piracicaba

### Bulgaria

<https://ftp.uni-sofia.bg/CRAN/>  
<http://ftp.uni-sofia.bg/CRAN/>

Sofia University  
Sofia University

# Interface do R

- Ao iniciar o R abrirá automaticamente o Console que é janela onde os comandos são digitados.
- Internamente ao console, se encontra o ***prompt***, que é um sinal indicador de que o programa está pronto para receber comando.



R Console

R version 3.4.2 (2017-09-28) -- "Short Summer"  
Copyright (C) 2017 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)

R é um software livre e vem sem GARANTIA ALGUMA.  
Você pode redistribuí-lo sob certas circunstâncias.  
Digite 'license()' ou 'licence()' para detalhes de distribuição.

R é um projeto colaborativo com muitos contribuidores.  
Digite 'contributors()' para obter mais informações e  
'citation()' para saber como citar o R ou pacotes do R em publicações.

Digite 'demo()' para demonstrações, 'help()' para o sistema on-line de ajuda,  
ou 'help.start()' para abrir o sistema de ajuda em HTML no seu navegador.  
Digite 'q()' para sair do R.

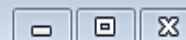
[Área de trabalho anterior carregada]

&gt; |

- Contudo, alguns usuários não têm tanta facilidade em trabalhar com janelas de *prompt*, especialmente quando é necessário utilizar diversas linhas de comando.
- Uma alternativa é usar **editores de texto para digitar os comandos** e depois transferir os comandos do editor para o console do R.
- No Windows um novo arquivo pode ser aberto acessando o menu “Arquivo” → Novo Script”.



R Console



R version 3.4.2 (2017-09-28) -- "Short Summer"  
Copyright (C) 2017 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)

R é um software livre e vem sem GARANTIA A  
Você pode redistribuí-lo sob certas circun  
Digite 'license()' ou 'licence()' para det

R é um projeto colaborativo com muitos con  
Digite 'contributors()' para obter mais in  
'citation()' para saber como citar o R ou

Digite 'demo()' para demonstrações, 'help(  
ou 'help.start()' para abrir o sistema de  
Digite 'q()' para sair do R.

[Área de trabalho anterior carregada]

> |

Sem nome - Editor R



# RStudio

- O RStudio oferece uma interface de usuário mais completa para o R.
- IDE (Integrated Development Environment) ou Ambiente Integrado de Desenvolvimento.
- Software que oferece a **integração entre o R e um avançado editor de textos** voltado para a edição de comandos.

- O RStudio pode ser baixado em

<http://www.rstudio.com>

- Ele deve ser instalado em computadores em que o R já esteja instalado.
- Durante a instalação, o RStudio integra-se à última versão do R instalada no seu computador.

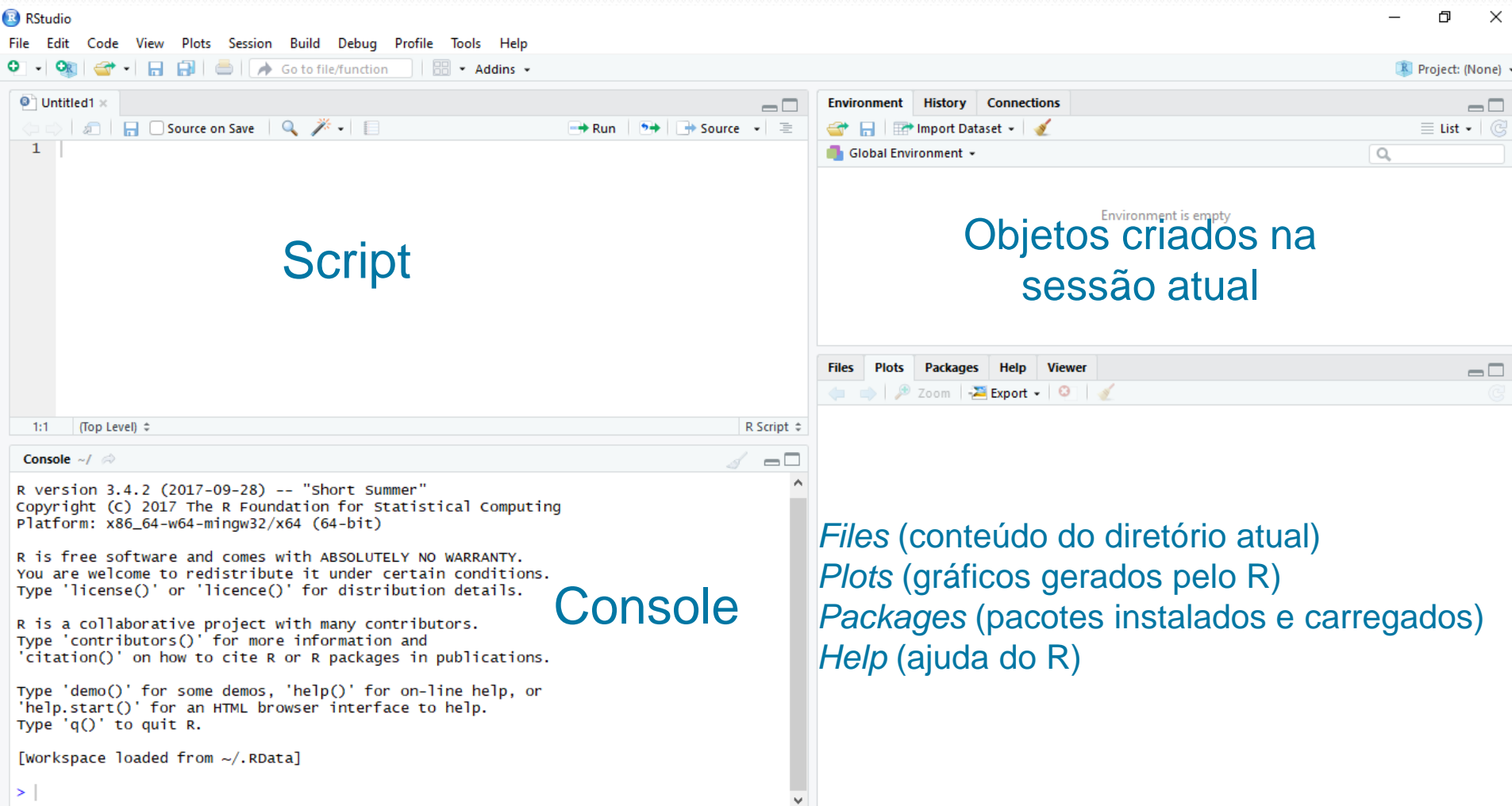
- O RStudio pode ser usado on-line em

<http://www.rstudio.cloud>

- Pode ser usado o log in da conta google ou GitHub.



# A janela do RStudio é dividida em 4 partes:



## 2. Iniciando o RStudio

### 2.1 Símbolos ou comandos importantes

- O R é *case-sensitive* (diferencia maiúscula de minúscula);
- O separador de casas decimais é o ponto (.);
- A vírgula é usada para separar argumentos (informações);
- Não é recomendado o uso de acentos em palavras.

## Alguns símbolos e comandos importantes:

- `#` : tudo que for digitado após esse símbolo na mesma linha de comando será ignorado pelo programa. Ideal para inserir comentários a respeito do código.
- `;` : separa dois comandos na mesma linha.
- `,` : separa elementos, como os números dentro de um comando.
- `NA` : dado ausente.
- `rm(x)` : remove o objeto `x`.

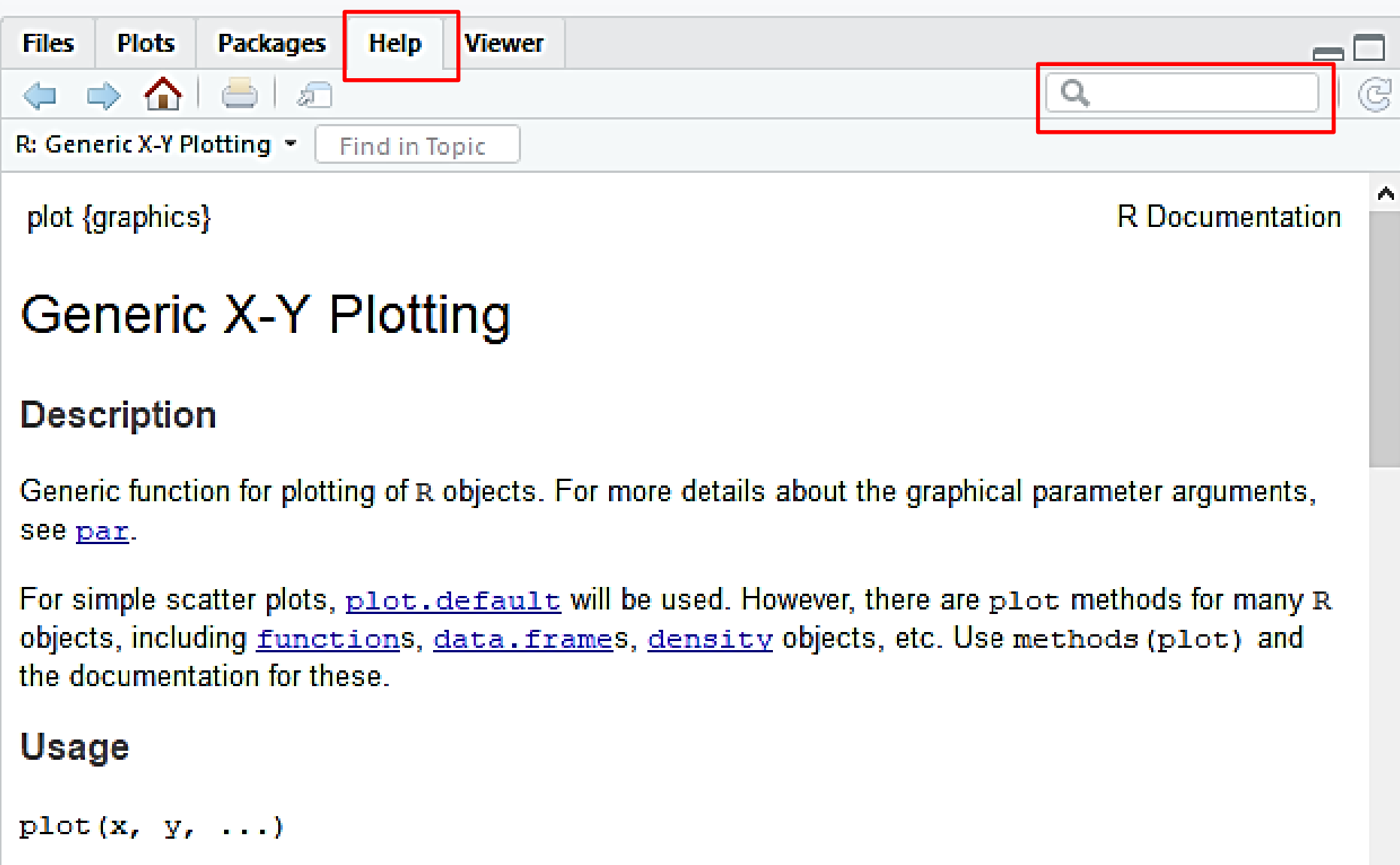
## 2.2 Ajuda

- É praticamente impossível memorizar todos os comandos do R.
- Assim, a ajuda pode ser útil quando se deseja saber qual comando utilizar e como utilizá-lo.
- Existem diversas formas de buscar ajuda no R.
- Exemplo: como plotar um gráfico?

- Um exemplo de busca é você deve digitar:

??plot

- O R irá exibir a ajuda do comando `plot( )`, fornecendo diversas informações.
- Pode-se digitar `plot` direto na aba de pesquisa em “help” também.



Files Plots Packages **Help** Viewer

← → Home Print Help

R: Generic X-Y Plotting Find in Topic

plot {graphics} R Documentation

# Generic X-Y Plotting

## Description

Generic function for plotting of R objects. For more details about the graphical parameter arguments, see [par](#).

For simple scatter plots, [plot.default](#) will be used. However, there are `plot` methods for many R objects, including [functions](#), [data.frames](#), [density](#) objects, etc. Use `methods(plot)` and the documentation for these.

## Usage

```
plot(x, y, ...)
```

Executar no R!

## Arguments

- x** the coordinates of points in the plot. Alternatively, a single plotting structure, function or *any R object with a plot method* can be provided.
- y** the y coordinates of points in the plot, *optional* if **x** is an appropriate structure.
- ... Arguments to be passed to methods, such as [graphical parameters](#) (see [par](#)). Many methods will accept the following arguments:

**type**

what type of plot should be drawn. Possible types are

- "p" for points,
- "l" for lines,
- "b" for both,
- "c" for the lines part alone of "b",
- "o" for both 'overplotted',
- "h" for 'histogram' like (or 'high-density') vertical lines,
- "s" for stair steps,
- "S" for other steps, see 'Details' below,
- "n" for no plotting.

All other types give a warning or an error; using, e.g., `type = "punkte"` being equivalent to `type = "p"` for S compatibility. Note that some methods, e.g. [plot.factor](#), do not accept this.

## 2.3 Atribuição de valores

- Como todo tipo de programação, é comum que tenhamos que atribuir valores para algumas variáveis antes de utilizá-las.
- Podemos atribuir de várias formas:

```
> x <- 10           #x é a variável que recebe o valor 10;
> 0.56 -> x         #x é a variável que recebe o valor 0.56;
> x = -8            #x é a variável que recebe o valor -8;
> assign("x", 2i)   #x é a variável que recebe o imaginário 2i;
```

Usaremos o símbolo <- para fazer as atribuições



## 2.4 Tipos de dados

Executar no R!

- Temos 4 tipos de dados no R:

```
> #Numérico
> valor <- 605
> valor
[1] 605
>
> #Caracteres
> string <- "Olá, mundo!"
> string
[1] "Olá, mundo!"
>
```

```
> #Lógicos
> 2 < 6
[1] TRUE
>
> #Números complexos
> nc <- 2 + 3i
> nc
[1] 2+3i
```

Esses tipos serão revistos em “atributos dos objetos”

## 2.5 Operações aritméticas

- Expressões aritméticas podem ser construídas através dos operadores usuais e das regras de precedência:

Símbolo	Função
$\wedge$ ou $**$	Potenciação
$/$	Divisão
$*$	Multiplicação
$+$	Adição
$-$	Subtração

```
1 #usando a linha de comandos, podemos:
2
3 #somar:
4 1+1
5
6 #subtrair:
7 1-1
8
9 #multiplicar:
10 5*3
11
12 #dividir:
13 6/2
14
15 #realizar cálculos mais complexos:
16 sin(5)|
```

16:7

(Top Level) ↕

R Script ↕

Console C:/Users/prisc/OneDrive/Estatística/Univille/Aulas/Análise de dados com R/Aulas/Exercício

```
> #usando a linha de comandos, podemos:
>
> #somar:
> 1+1
[1] 2
> #subtrair:
> 1-1
[1] 0
> #multiplicar:
> 5*3
[1] 15
> #dividir:
> 6/2
[1] 3
> #realizar cálculos mais complexos:
> sin(5)
[1] -0.9589243
> |
```

# Executar no R!

- Algumas operações têm prioridade sobre outras, como a multiplicação sobre a adição.
- Parênteses podem ser usados para isolar partes da expressão para evitar erros.

#isolando partes de uma expressão:

```
> (4+2)*3
```

```
[1] 18
```

```
> 4+2*3
```

```
[1] 10
```

- Além das operações básicas, o R possui outras operações disponíveis:

Função	Descrição
<i>abs(x)</i>	valor absoluto de $x$
<i>log(x, b)</i>	logaritmo de $x$ com base $b$
<i>log(x)</i>	logaritmo natural de $x$
<i>log10(x)</i>	logaritmo de $x$ com base 10
<i>exp(x)</i>	exponencial elevado a $x$
<i>sin(x)</i>	seno de $x$
<i>cos(x)</i>	cosseno de $x$
<i>tan(x)</i>	tangente de $x$
<i>round(x, digits = n)</i>	arredonda $x$ com $n$ decimais
<i>ceiling(x)</i>	arredondamento de $x$ para o maior valor
<i>floor(x)</i>	arredondamento de $x$ para o menor valor
<i>length(x)</i>	número de elementos do vetor $x$
<i>sum(x)</i>	soma dos elementos do vetor $x$
<i>prod(x)</i>	produto dos elementos do vetor $x$
<i>max(x)</i>	seleciona o maior elemento do vetor $x$
<i>min(x)</i>	seleciona o menor elemento do vetor $x$
<i>range(x)</i>	retorna o menor e o maior elemento do vetor $x$

# Arredondamentos e aproximações:

- Em determinadas análises estatísticas, precisamos apresentar os resultados com casas decimais e/ou fazer arredondamentos.
- Comandos úteis:

`round ( )`  
`signif ( )`

# round ( )

- É usado quando se deseja arredondar um valor ou conjunto de valores em um número pré-estabelecido **de casas decimais**.
- Dentro dos parênteses coloca-se quem será arredondado e em quantas casas decimais após a vírgula.

- Exemplo:

```
> x <- pi #x recebe o valor da constante pi
> x       #exibindo x
[1] 3.141593
> round(x,3) #o 3 representa o número de casas
decimais
[1] 3.142
```

# Observação importante!

- O número 1 entre colchetes `[1]` é um comando implícito do comando `print ( )`.
- Ou seja, escrevendo `print (x)` obteríamos o mesmo resultado que escrevendo apenas `x`.
- Porém, dentro de funções, esse comando deve ser usado explicitamente.



# signif ( )

- É usado quando se deseja arredondar um valor ou conjunto de valores em um número pré-estabelecido **de algarismos significativos**.
- Dentro dos parênteses coloca-se quem será arredondado e em quantos algarismos.
- Exemplo:

```
> x #exibindo x, que possui o valor de pi  
[1] 3.141593  
> signif(x, 3)  
[1] 3.14
```

## 2.6 Manipulando objetos e seus atributos

Todas as variáveis (escalares, vetores, matrizes, etc.) criadas pelo R são chamadas de **objetos**, ou seja, não são apenas valores sequenciais ou não.

## ❖ Criando objetos (variáveis)

- Uma variável pode ser criada com a operação de atribuição (<-).
- Deve começar com uma letra.
- Exemplos:

`x <- 7`    `#x` receberá o valor 7

`t <- 2*3`    `#t` irá receber o valor da  
operação

## ❖ Listando objetos (variáveis)

- Após listar, pode-se ter o controle sobre eles.
- A função `ls( )` mostra todas as variáveis que foram criadas.
- Exemplo:

```
>X <- 7 #x receberá o valor 7
```

```
>t <- 2*3 #t irá receber o valor da  
operação
```

```
>ls() #lista todas as variáveis existentes  
na memória.
```

```
[1] "t" "x"
```

## ❖ Removendo objetos (variáveis)

- `remove( )` ou `rm( )`
- Para usá-lo, basta fornecer o nome da variável a ser removida:

```
A <- 7      #cria A
```

```
B <- 2      #cria B
```

```
rm(A, B)    #remove A e B
```

- Se a variável que foi removida for digitada, o console aponta o erro:

```
>A
```

```
>Error: object 'A' not found
```

## ❖ Atributos dos objetos

- Como o R trabalha com objetos, eles possuem **nome, conteúdo e um atributo associado** que especifica qual o tipo de dados estão representados pelo objeto.
- Em uma análise estatística, mesmo que dois objetos **contenham o mesmo valor**, os resultados se **diferem pelos seus atributos**.

## ❖ Atributos dos objetos

- Todo objeto possui atributos intrínsecos: tipo e tamanho.

Tipo	Descrição
character	Textos ou caracteres
numeric	Números inteiros ou reais
logical	Verdadeiro ou falso (TRUE/FALSE)
complex	Números complexos
list	Combina diferentes tipos num mesmo objeto
function	Comandos

## Exemplo:

```
>x<-c(1,3,5,7,11)
>mode(x); length(x) #mostra o tipo e o
tamanho do objeto
[1] "numeric"
[1] 5
```



```
>a<-"Angela"; b<-TRUE; c<-8i  
>#objetos com tipos diferentes
```

```
>mode(a); mode(b); mode(c)  
>#exibe os atributos "tipo" dos  
objetos
```

```
[1] "character"  
[1] "logical"  
[1] "complex"
```

Outro caminho é usando “is”:

```
>a<-"Angela"; b<-TRUE; c<-8i  
#objetos com tipos diferentes
```

```
>is.numeric(a)
```

```
[1] FALSE
```

```
>is.character(b)
```

```
[1] FALSE
```

```
>is.complex(c)
```

```
[1] TRUE
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Project: (None)

Operações aritméticas.R

```
41 x<-c(1,3,5,7,11)
42 mode(x); length(x) #mostra o tipo e o tamanho do objeto
43
44 a<-"Angela" b<-TRUE; c<-8i #objetos com tipos diferentes
45 mode(a); mode(b); mode(c) #exibe os atributos "tipo" dos
46
47 a<-"Angela"; b<-TRUE; c<-8i #objetos com tipos diferentes
48 is.numeric(a)
49 is.character(b)
50 is.complex(c)
51
52
```

Environment History Connections

Global Environment

Values

a	"Angela"
b	TRUE
c	0+8i
x	num [1:5] 1 3 5 7 11

O RStudio lista todas as variáveis no “ambiente”,  
no lado direito do programa.

- Além de atributos intrínsecos (tipo e tamanho), os objetos podem ter diversos atributos: nomes, dimensão, etc.
- Os nomes dos elementos contidos num objeto, por exemplo, é um atributo controlado pelo `names( )`:

```
>escola<-c(100, 45, 55)      #criando um objeto  
>escola                      #exibindo "escola"  
[1] 100 45 55
```

```
>names(escola)<-c("alunos", "masc", "fem")  
#nomes
```

```
>escola    #exibindo escola
```

```
alunos  masc  fem  
   100    45   55
```

# ❖ Infinito, NaN e NA

- Inf e -Inf: infinito.
- NaN: “não número” (do inglês *Not a Number*).
- NA: **valores ausentes** (do inglês *Not Available*, também conhecido como *missing data*).

>#infinito, NaN e NA:

>2/0 #dividindo 2 por zero

[1] Inf

>-2/0 #dividindo -2 por zero

[1] -Inf

>x<-Inf #x recebe infinito

>x #exibe x

[1] Inf

>is.infinite(x) #x é infinito?

[1] TRUE

Quando o resultado de uma operação é indefinido, ele é representado no R por NaN:

```
>0/0          #dividindo zero por zero  
[1] NaN  
>Inf - Inf    #infinito menos infinito  
[1] NaN
```



O NA é usado para representar um dado ausente.

**Exemplo:** mensuração do peso das pessoas da sala. Caso não colete o peso de alguém, este será representado no seu conjunto de dados por um NA.

```
>pesos<-c(62,NA,76,93,49) #o 2º não pode  
ser observado
```

```
>pesos #exibe pesos  
[1] 62 NA 76 93 49
```

```
>is.na(pesos) #os elementos de pesos são  
NA?  
[1] FALSE TRUE FALSE FALSE FALSE
```

- Alguns comandos, como o `mean( )`, que calcula a média aritmética de um conjunto de dados, não operam com elementos NA.
- Pode-se usar o argumento `na.rm=TRUE` como alternativa.
- Os valores ausentes serão descartados antes do processamento.

```
>pesos<-c(62,NA,76,93,49)
```

```
#o 2º não pode ser observado
```

```
>mean(pesos)
```

```
Error in mean(pesos) : could not find  
function "mean"
```

```
>mean(pesos, na.rm = TRUE)
```

```
[1] 70
```



# **3. Exercícios**