

Ciência de dados em R

Curso-R

2020-05-17

Contents

Sobre

O R é uma linguagem de programação *open source* para análise de dados que fornece uma grande variedade de ferramentas estatísticas e gráficas.

Chapter 1

Instalação

Nesta seção, abordaremos como instalar o R e o RStudio no Linux e no Windows. Também discutiremos sobre a instalação de pacotes no R.

1.1 Instalação do R

A instalação padrão do R é feita a partir do CRAN, uma rede servidores espalhada pelo mundo que armazena versões idênticas e atualizadas de códigos e documentações para o R.

Sempre que for instalar algo do CRAN, utilize o servidor (*mirror*) mais próximo de você.

1.1.1 No Windows

Para instalar o R no Windows, siga os seguintes passos:

1. Acesse o CRAN: <https://cran.r-project.org/bin/windows/base/>
2. Clique em “Download R x.x.x for Windows”, sendo x.x.x o número da versão mais recente disponível.
3. Salve o arquivo em qualquer pasta do seu computador.
4. Clique no arquivo duas vezes com o botão esquerdo e siga as instruções para instalação.

Na etapa de escolher a pasta de destino da instalação, se você escolher um local que não esteja dentro da sua pasta de usuário, você precisará de acesso de administrador. Se escolher uma pasta dentro da sua pasta de usuário, não precisará.

Pronto! O R está instalado no seu computador!

1.1.2 No Linux

Como a instalação no Linux depende da distribuição utilizada e, em geral, usuário de Linux são mais experientes, vamos informar apenas as coordenadas até as instruções/arquivos de instalação para cada distribuição. Se você tiver alguma dificuldade durante o processo, por favor envie a sua dúvida para a nossa comunidade ou para o e-mail duvidas@curso-r.com. Faremos o possível para ajudar.

1. Acesse o CRAN: <https://cran.r-project.org/>
2. Clique em *Download R for Linux*.
3. Clique no link referente à distribuição que você utiliza.
4. Siga as instruções contidas na página para instalar o R.

1.2 Instalação do RStudio

Agora vamos instalar a versão *open source* do RStudio, a IDE que utilizaremos para escrever e executar códigos em R.

1.2.1 No Windows

Para instalar o RStudio no Windows, siga os seguintes passos:

1. Entre no site da Rstudio: <https://rstudio.com>
2. No topo da página, clique em download.
 - 2a. Se você tiver acesso administrador, baixe a versão que está na lista de *All Installers*.
 - 2b. Se você não tiver acesso de administrador, faça o download da versão que está na lista *Zip/Tarballs*.

Instalando se você for administrador

3. Clique duas vezes no arquivo que você baixou da página do RStudio e siga as instruções de instalação.

Pronto! O RStudio está pronto para ser utilizado.

Instalação se você não for administrador

3. Clique com o botão direito no arquivo baixado e depois em *Extrair Tudo* conforme a imagem.
4. Após a descompactação do arquivo ter sido finalizada, você terá uma pasta chamada: **RStudio-x.x.x**, em que x.x.x é o número da versão baixada. Abra essa pasta e entre na subpasta com nome **bin**.
5. Procure pelo arquivo chamado **rstudio** e clique duas vezes. Isso abrirá o RStudio. Recomendo fixar o programa na barra de tarefas para não precisar repetir essa etapa sempre que for abrir o programa.

Observação: se você excluir a pasta que extraímos, o RStudio irá parar de funcionar.

1.2.2 No Linux

1. Entre no site da Rstudio: <https://rstudio.com>
2. No topo da página, clique em download.
3. Clique no link referente à distribuição que você utiliza para fazer o download do arquivo de instalação.
4. A depender da sua distribuição do Linux, instale o arquivo baixado.

1.3 Instalação de pacotes

Um pacote é um conjunto de funções que têm como objetivo resolver um problema específico. São eles que deixam o R poderoso, capaz de enfrentar qualquer tarefa de análise de dados. Assim, fique bastante à vontade para instalar e atualizar muitos e muitos pacotes ao longo da sua experiência com o R.

O legal é que qualquer pessoa pode fazer um novo pacote e disponibilizar para a comunidade, o que acelera bastante o desenvolvimento da ferramenta. Dificilmente você vai fazer uma análise apenas com as funções básicas do R e dificilmente não vai existir um pacote com as funções que você precisa.

Existem três principais maneiras de instalar pacotes. Em ordem de frequência, são:

- Via CRAN (Comprehensive R Archive Network): `install.packages("nome-do-pacote")`.
- Via Github: `devtools::install_github("nome-do-repo/nome-do-pacote")`.
- Via arquivo .zip/.tar.gz: `install.packages("C:/caminho/nome-do-pacote.zip", repos = NULL)`.

Para conseguir instalar alguns pacotes no Linux, você pode precisar instalar dependências do sistema manualmente. Por exemplo, se você quer instalar o pacote `devtools` no R, será necessário ter as bibliotecas `curl`, `openssl`, `httr` e `git2r`.

Essas dependências geralmente podem ser instaladas no terminal por meio do comando `apt-get install nome-da-biblioteca`. Caso você não consiga instalar um pacote devido a ausência de uma dependência, uma maneira de saber quais bibliotecas você precisa instalar é observar as mensagens que aparecem no console durante a tentativa da instalação do pacote.

1.3.1 Via CRAN

Instale pacotes que não estão na sua biblioteca usando a função `install.packages("nome_do_pacote")`. Por exemplo:

```
install.packages("magrittr")
```

E, de agora em diante, não precisa mais instalar. Basta carregar o pacote com `library(magrittr)`.

Escreva `nome_do_pacote::nome_da_funcao()` se quiser usar apenas uma função de um determinado pacote. O operador `::` serve para isso. Essa forma também é útil quando se tem duas funções com o mesmo nome e precisamos garantir que o código vá usar a função do pacote correto.

1.3.2 Via Github

Desenvolvedores costumam disponibilizar a última versão de seus pacotes no Github, e alguns deles sequer estão no CRAN. Mesmo assim ainda é possível utilizá-los instalando diretamente pelo github. O comando é igualmente simples:

```
devtools::install_github("rstudio/shiny")
```

Apenas será necessário o username e o nome do repositório (que geralmente tem o mesmo nome do pacote). No exemplo, o username foi “rstudio” e o repositório foi “shiny”.

Se você não é familiar com o github, não se preocupe! Os pacotes disponibilizados na plataforma geralmente têm um README cuja primeira instrução é sobre a instalação. Se não tiver, provavelmente este pacote não te merece! =)

1.3.3 Via arquivo .zip ou .tar.gz

Se você precisar instalar um pacote que está zipado no seu computador (ou em algum servidor), utilize o seguinte comando:

```
install.packages("C:/caminho/para/o/arquivo/zipado/nome-do-pacote.zip", repos = NULL)
```

É semelhante a instalar pacotes via CRAN, com a diferença que agora o nome do pacote é o caminho inteiro até o arquivo. O parâmetro `repos = NULL` informa que estamos instalando a partir da máquina local.

A aba **Packages** do RStudio também ajuda a administrar os seus pacotes.

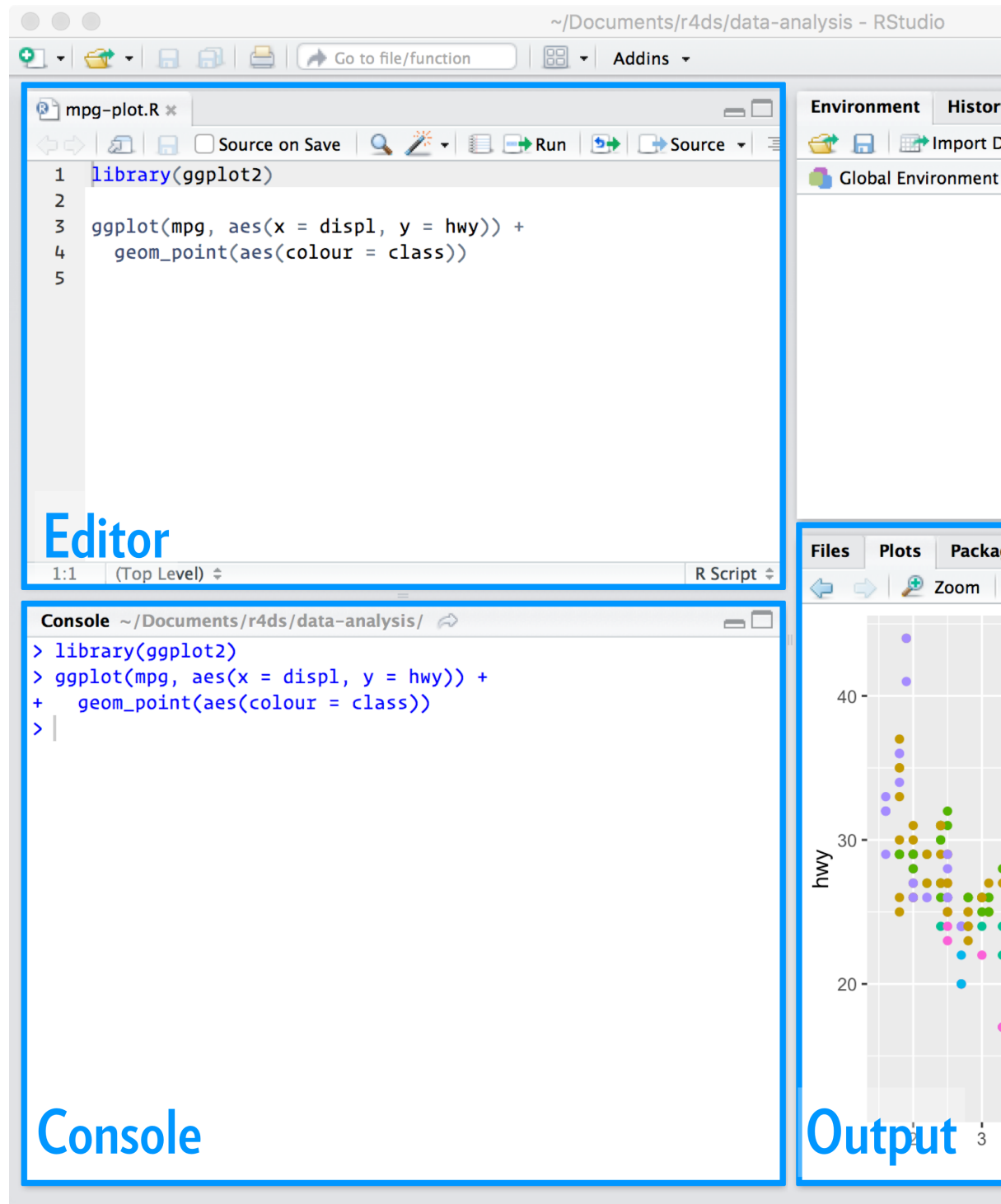
Chapter 2

RStudio

O RStudio é sem dúvidas o mais completo ambiente de desenvolvimento para programação em R. Descubra aqui as funcionalidades do RStudio que nos permitem escrever códigos e analisar resultados de forma muito mais eficiente.

2.1 Telas

Ao abrir o RStudio, você verá 4 quadrantes. Observe a figura abaixo.



Esses quadrantes representam o **editor**, o **console**, o **environment** e o **output**. Eles vêm nesta ordem, mas você pode organizá-los da forma que preferir acessando a seção *Pane Layout* da opção **Global options...** no menu **Tools**.

O editor e o console são os dois principais painéis do RStudio. Passaremos a maior parte do tempo neles.

- **Editor/Scripts**: é onde escrevemos nossos códigos. Repare que o RStudio colore algumas palavras e símbolos para facilitar a leitura do código.
- **Console**: é onde rodamos o código e recebemos as saídas. O R vive aqui!

Os demais painéis são auxiliares. O objetivo deles é facilitar pequenas tarefas que fazem parte tanto da programação quanto da análise de dados, como olhar a documentação de funções, analisar os objetos criados em uma sessão do R, procurar e organizar os arquivos que compõem a nossa análise, armazenar e analisar os gráficos criados e muito mais.

- **Environment**: painel com todos os objetos criados na sessão.
- **History**: painel com um histórico dos comandos rodados.
- **Files**: mostra os arquivos no diretório de trabalho. É possível navegar entre diretórios.
- **Plots**: painel onde os gráficos serão apresentados.
- **Packages**: apresenta todos os pacotes instalados e carregados.
- **Help**: janela onde a documentação das funções serão apresentadas.
- **Viewer**: painel onde relatórios e dashboards serão apresentados.

2.2 Atalhos

Conhecer os atalhos do teclado ajuda bastante quando estamos programando no RStudio. Veja os principais:

- **CTRL+ENTER**: avalia a linha selecionada no script. O atalho mais utilizado.
- **ALT+-**: cria no script um sinal de atribuição (`<-`). Você o usará o tempo todo.
- **CTRL+SHIFT+M**: (`%>%`) operador *pipe*. Guarde esse atalho, você o usará bastante.
- **CTRL+1**: altera cursor para o script.
- **CTRL+2**: altera cursor para o console.
- **CTRL+ALT+I**: cria um chunk no R Markdown.
- **CTRL+SHIFT+K**: compila um arquivo no R Markdown.
- **ALT+SHIFT+K**: janela com todos os atalhos disponíveis.

No MacBook, os atalhos geralmente são os mesmos, substituindo o **CTRL** por **command** e o **ALT** por **option**.

2.3 Projetos

Uma funcionalidade muito importante do RStudio é a possibilidade de criar **projetos**.

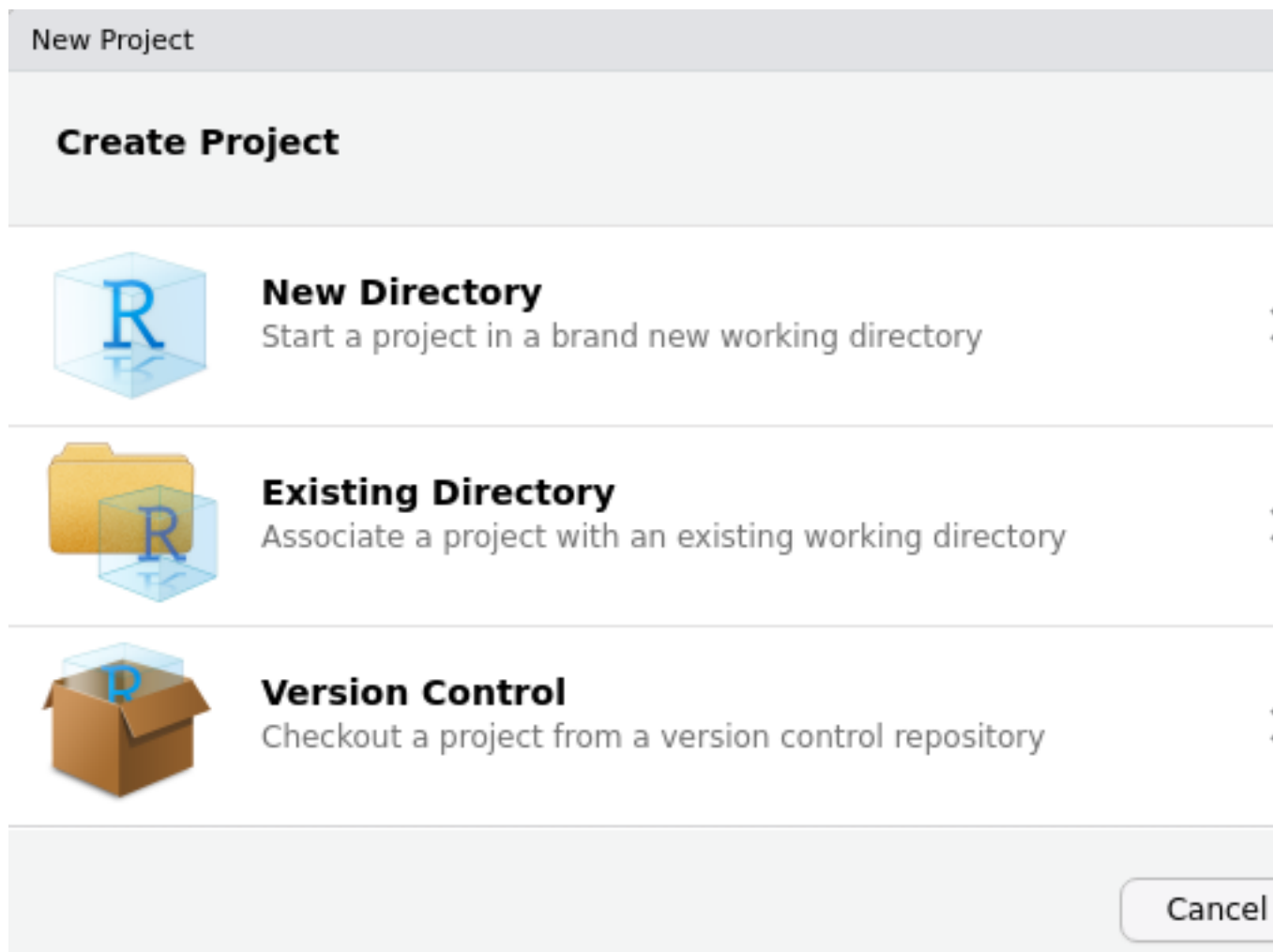
Um projeto nada mais é do que uma pasta no seu computador. Nessa pasta, estarão todos os arquivos que você usará ou criará na sua análise.

A principal razão de utilizarmos projetos é **organização**. Com eles, fica muito mais fácil importar bases de dados para dentro do R, criar análises reproduutíveis e compartilhar o nosso trabalho.

Você que está começando agora no R, já se habitue a criar um novo projeto para cada nova análise que for fazer.

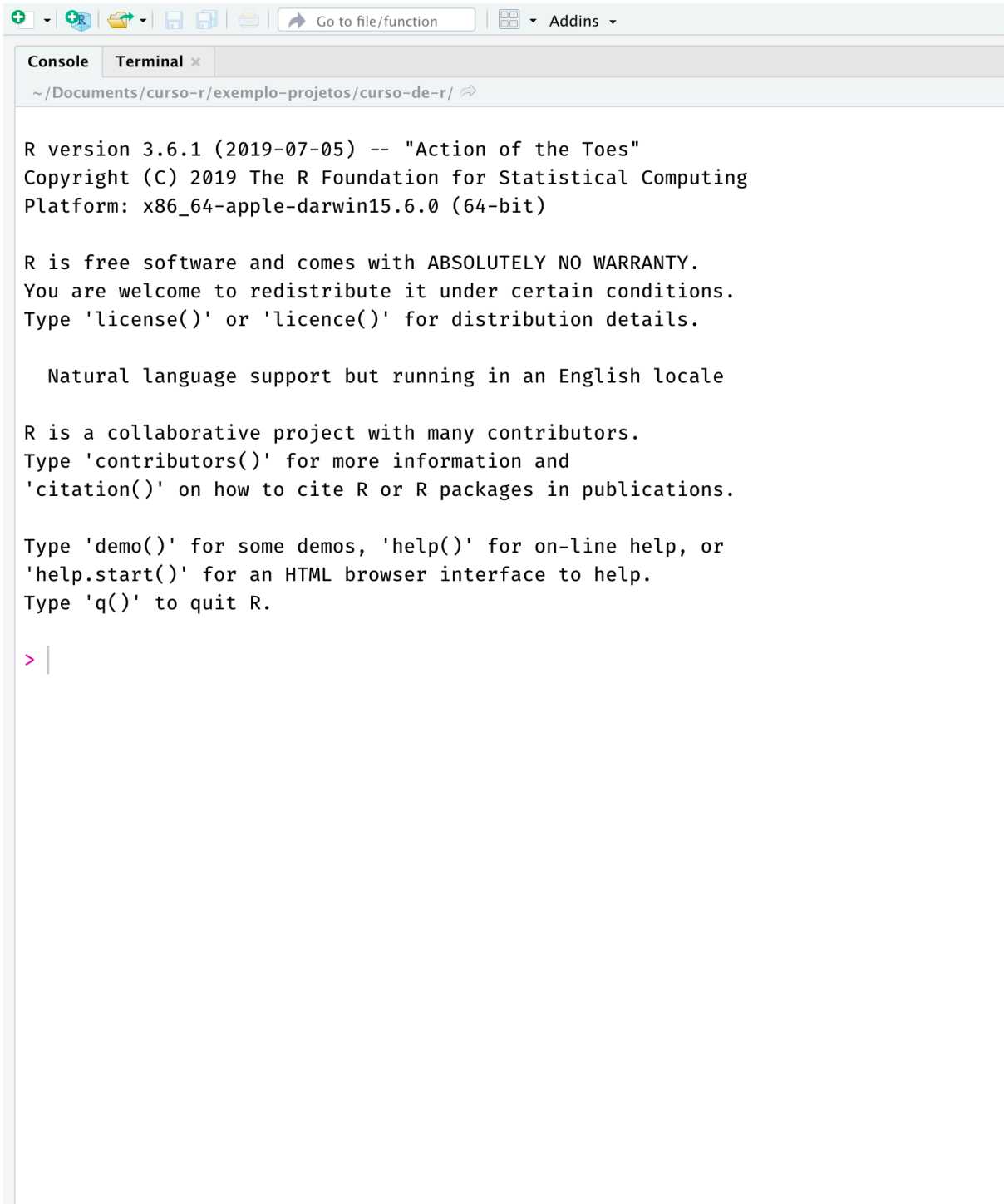
Para criar um projeto, clique em **New Project...** no Menu **File**. Na caixa de diálogo que aparecerá, clique em **New Directory** para criar o projeto em uma nova pasta ou **Existing Directory** para criar em uma pasta existente.

Se você tiver o **Git** instalado, você também pode usar projetos para conectar com repositórios do Github e outras plataformas de desenvolvimento. Para isso, basta clicar em **Version Control**.



Criando um projeto, o RStudio criará na pasta escolhida um arquivo `nome-do-projeto.Rproj`. Você pode usar esse arquivo para iniciar o RStudio já com o respectivo projeto aberto.

Quando um projeto estiver aberto no RStudio, o seu nome aparecerá no canto superior direito da tela. Na aba **Files**, aparecerão todos os arquivos contidos no projeto.



The screenshot shows the RStudio interface with the Terminal pane active. The terminal displays the R startup message, including the version (3.6.1), copyright (© 2019 The R Foundation for Statistical Computing), and platform (x86_64-apple-darwin15.6.0). It also includes a disclaimer about the warranty and a list of useful commands like 'license()', 'contributors()', 'citation()', 'demo()', 'help()', and 'q()'.

```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

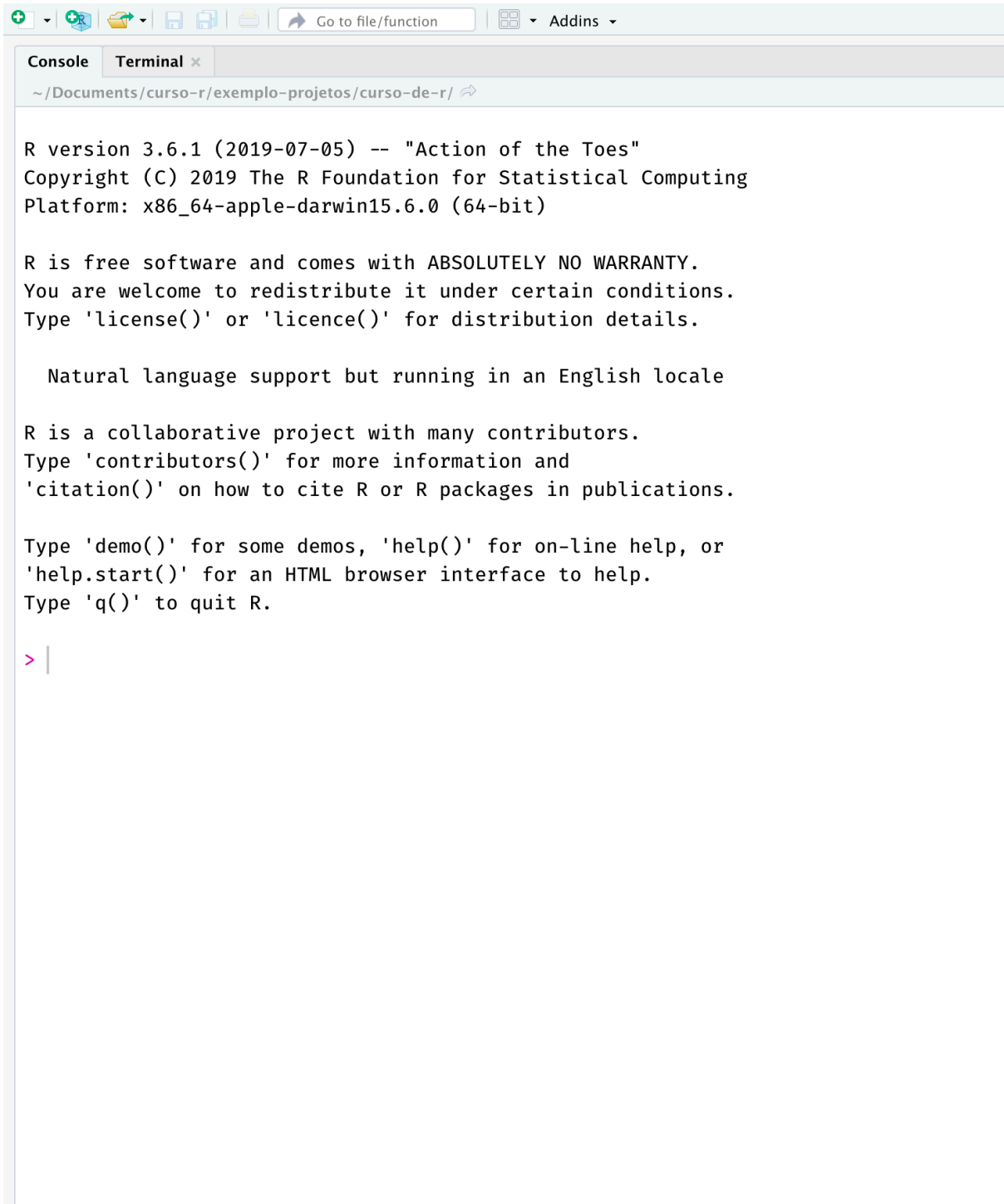
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

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.

> |
```

Você pode criar livremente novas pastas dentro da pasta do projeto. Por padrão, o R sempre começará a procurar arquivos na pasta raiz do projeto (é a pasta que contém o `nome-do-projeto.Rproj`).

Uma maneira fácil de navegar entre projetos é utilizar o menu disponibilizado quando clicamos no nome do projeto. Veja a figura a seguir.



The screenshot shows the RStudio interface with the Terminal pane active. The terminal displays the R startup message, including the version (3.6.1), copyright (© 2019 The R Foundation for Statistical Computing), and platform (x86_64-apple-darwin15.6.0). It also includes a disclaimer about the warranty and a list of useful commands like `license()`, `contributors()`, `demo()`, and `q()`. The prompt `>` is visible at the bottom of the terminal.

```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

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.

> |
```

Nesse menu, além de podermos criar novos projetos ou abrir projetos já existentes, também temos um acesso rápido a projetos abertos recentemente. Basta clicar em qualquer um deles para trocar de projeto, isto é, deixar de trabalhar em uma análise e começar a trabalhar em outra.

A seguir, apresentamos algumas estruturas de organização de projetos no RStudio.

Estrutura 1. Por extensão de arquivo.

```
nome_do_projeto/
- .Rprofile      # códigos para rodar assim que abrir o projeto
- R/             # Código R, organizado com a-carrega.R, b-prepara bd.R, c-vis.R, d-modela, ...
- RData/        # Dados em formato .RData
- csv/          # Dados em .csv
- png/          # gráficos em PNG
- nome_do_projeto.Rproj
```

Estrutura 2. Típico projeto de análise estatística.

```
project/
- README.Rmd     # Descrição do pacote
- set-up.R       # Pacotes etc
- R/             # Código R, organizado com 0-load.R, 1-tidy.R, 2-vis.R, ...
- data/          # Dados (estruturados ou não)
- figures/        # gráficos (pode ficar dentro de output/)
- output/        # Relatórios em .Rmd, .tex etc
- project.Rproj
```

Estrutura 3. Pacote do R.

```
project/
- README.md      # Descrição do pacote
- DESCRIPTION    # Metadados estruturados do pacote e dependências
- NAMESPACE     # importações e exportações do pacote
- vignettes/     # Relatórios em .Rmd
- R/             # Funções do R
- data/          # Dados estruturados (tidy data)
- data-raw/      # Dados não estruturados e arqs 0-load.R, 1-tidy.R, 2-vis.R, ...
- project.Rproj
```