Introdução ao R para análise de dados e construção de gráficos

Alessandro Samuel-Rosa

2020-11-23

Contents

Chapter 1

Apresentação

Chapter 2

R & RStudio

A página do projeto R na Internet pode ser acessada pelo endereço https://www.r-project.org/. Nela você encontra grande quantidade de informação sobre o R. Estão disponíveis diversos manuais de uso (*Documentation* > *Manuals*), bem como livros (*Documentation* > *Books*) e artigos publicados na revista do projeto (*Documentation* > *The R Journal*). Também há uma página com respostas às perguntas mais frequentes (*Documentation* > *FAQs*) e uma página inteira explicando como é possível conseguir ajuda sozinho antes de recorrer a terceiros (*Help With R* > *Getting Help*)[1].

O procedimento de instalação do R depende do sistema operacional (OS, do inglês *operating system*) de seu computador:

- Linux: https://cloud.r-project.org/bin/linux/
- (Mac) OS X: https://cloud.r-project.org/bin/macosx/
- Windows: https://cloud.r-project.org/bin/windows/base/

A página de cada OS possui as instruções necessárias para descarregar e instalar o R em seu computador. Em geral, o processo de instalação é muito parecido com aquele de outros programas de computador que você está acostumado a usar.

Depois de completada a instalação do R, é hora de instalar o RStudio. A última versão gratuíta[2] do instalador do RStudio para o OS de seu computador pode ser descarregada do seguinte endereço na Internet:

- https://www.rstudio.com/products/rstudio/download/

[1] Como a documentação do R é extensa e a maioria dos colaboradores do projeto não são pagos pelo trabalho desenvolvido, recomendo que você sempre procure, primeiro, resolver qualquer dúvida sozinho.

[2] Licença AGPL v3.

Assim como para o R, você não encontrará maiores dificuldades no processo de instalação do RStudio.

Depois de instalados R e RStudio, inicie o RStudio em seu computador. A interface do RStudio deve se parecer mais ou menos com aquela mostrada na figura abaixo[3].

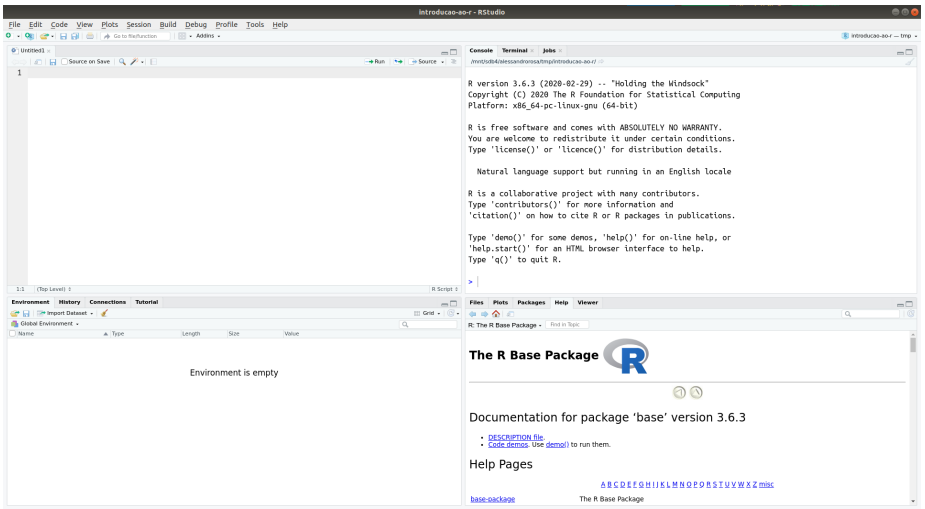


Figure 2.1: RStudio: ambiente de desenvolvimento integrado (IDE) para R em sua versão para Linux.

O RStudio é composto por quatro painéis retangulares que ocupam seus quadrantes:

- Painel superior esquerdo: usado para preparar o roteiro (*script*) de análise de dados.
- Painel superior direito: interface de linha de comando (CLI), o console do R.
- Painel inferior direito: serve à visualização de gráficos e páginas de ajuda do R.
- Painel inferior esquerdo: apresenta informações sobre a sessão de trabalho.

Dentre os quatro painéis, é no painel superior esquerdo que passamos a maior parte do tempo quando analisamos dados no R. É ali que redigimos aquilo que queremos que o R faça com nossos dados na forma de comandos usando a linguagem do R. Para nos comunicamos com o R, enviamos esses comandos para o console do R, localizado no painel superior direito. Os resultados produzidos pelo R podem ser emitidos, tanto no console (tabelas), como no painel inferior direito (gráficos). Se não soubermos como nos comunicarmos com o R para que

[3] A disposição dos painéis do ambiente de trabalho pode ser alterada acessando *Tools* > *Global Options* > *Pane Layout*.

execute determinada função, podemos visitara aba de ajuda localizada no painel inferior direito.

Chapter 3

Diretório de arquivos

Um dos passos mais importatantes de qualquer projeto é a criação de uma estrutura racional de diretórios de arquivos. Isso pode ser feito diretamente a partir do RStudio. Para isso, basta acessar *File* > *New Project...* > *New Directory* > *New Project*. Na janela que abrir, são definidos o nome do diretório que armazenará os arquivos do projeto[1] e o local do sistema de arquivos do computador onde esse diretório ficará localizado. Feito isso, o RStudio será reinicializado e o painel inferior direito (aba *Files*) mostrará o interior do diretório que acabou de ser criado.

A seguinte estrutura de diretórios de arquivos deve ser criada no interior do diretório recém criado:

```
nome-do-projeto
|- code/
|- data/
|- docs/
|- res/
|  |- fig/
|  |- tab/
|- tmp/
|- README.md
```

São cinco subdiretórios, cada um deles com o propósito de armazenar arquivos com conteúdos específicos:

- `code`: roteiros de análise de dados escritos usando a linguagem R.
- `data`: dados das variáveis que serão estudadas no projeto.

[1] Deve ser dada preferência a um nome curto e fácil de lembrar e relacionar com o escopo do projeto (mnemônico). Além disso, devem ser utilizadas apenas letras minúsculas e sem acentuação, substituindo os espaços por traços. Isso facilitará a gestão programática dos arquivos do projeto.

- `docs`: documentos de texto relacionados ao projeto.
- `res`: resultados do projeto exportados como figuras e tabelas—figuras podem ser armazenadas no subdiretório `fig`, enquanto o subdiretório `tab` pode ser usado para armazenar tabelas.
- `tmp`: arquivos temporários irrelevantes.

Diretórios podem ser criados diretamente no painel inferior direito do RStudio. Para isso, basta acessar *New Folder* e definir o nome do diretório desejado conforme listado acima.

O próximo passo é criar um arquivo com a descrição do projeto. Isso é importante para garantir que, da próxima vez que você visitar esse diretório de arquivos, não tenha que depender inteiramente da sua memória. O arquivo `README.md` tem essa finalidade. Para criar esse arquivo, acesse *File* > *New File* e selecione a opção *Markdown File*. O arquivo que abrir no painel superior esquerdo do RStudio deve ser salvo na raiz do diretório do projeto usando o nome `README.md` acessando *File* > *Save As...*. Depois disso, registre no `README.md` informações essenciais do projeto como nome, equipe participante e data de início. O arquivo `README.md` também pode ser usado para descrever o conteúdo de cada um dos subdiretórios do projeto. Ao longo do desenvolvimento do projeto, ele será bastante útil para anotas as decisões tomadas e resumir os resultados alcançados.

Para concluir a construção da estrutura do diretório de arquivos, precisamos criar apenas mais um arquivo, agora no interior do diretório `code`. Esse arquivo será usado para redigir e armazenar o roteiro de análise de dados usando a linguagem R. Para criar esse arquivo, acesse *File* > *New File* e selecione a opção *R Script*. Salve o arquivo que abrir no painel superior esquerdo do RStudio acessando *File* > *Save As...*—você pode atribuir um nome curto como `main.R`.

Chapter 4

Operações matemáticas

Vamos fazer algumas operações matemáticas para nos familiarizar com o R. Inicie copiando e colando a linha de comando abaixo no console do R localizado no painel superior direito do RStudio. Em seguida, pressione a tecla *Enter*.

```
# Operações matemáticas no R: soma
2 + 3
```

O console do R deve ter retornado o valor `5` como resultado da operação realizada. Agora realize a operação de subtração `2 - 3`. O valor retornado pelo console do R deve ser `-1`[1].

Os símbolos da linguagem R para realizar as operações matemáticas básicas são os mesmos encontrados em qualquer calculadora científica: `+` para adição, `*` para multiplização, `/` para divisão e `-` para subtração. São também os mesmos símbolos utilizados em planilhas eletrônicas de edição de dados. Isso significa que podemos deduzir algumas coisas sobre o funcionamento do R a partir daquilo que conhecemos de outras ferramentas dedicadas à análise e manipulação de dados.

Tarefa: *Realize todas as quatro operações matemáticas fundamentais utilizando diferentes valores para conhecer melhor o funcionamento básico do R. Registre essas operações no arquivo* ***`main.R`****, inserindo comentários textuais sobre cada uma delas.*

Um elemento importante presente no bloco acima é o comentário precedido pelo símbolo `#` (cerquilha). Na linguagem R, a cerquilha tem o papel de indicar

[1] O espaçamento entre número e operador matemático não tem importância do ponto de vista da operação matemática. Contudo, do ponto de vista estético, para facilitar a leitura dos comandos, costuma-se usar a formatação com espaços `2 + 3` em vez de `2+3`.

o início de um comentário textual que serve de instrução à pessoa que está escrevendo ou lendo o roteiro de análise do dados. Esses comentários podem ser incluídos, tanto em uma linha própria do roteiro, como na mesma linha após um comando, mas nunca antes de um comando. A inclusão de comentários textuais ao logo do roteiro serve para documentarmos a atividade que estamos realizando. Isso permite que outros, e nós mesmos, algumas semanas ou meses mais tarde, possamos entender o propósito de cada uma das linhas de comando redigidas.

O bloco abaixo apresenta mais algumas operações matemáticas.

```
2^2      # Potenciação (ou exponenciação)
log(4)   # Função logarítmica
sqrt(25) # Raiz quadrada
```

As duas últimas operações apresentadas no bloco acima são representadas por identificadores, especificamente, as "palavras" `log` e `sqrt`, seguidas por dois parênteses contendo um valor numérico em seu interior. As expressões `log()` e `sqrt()` identificam funções do R, nesse caso, as funções logarítmica e raiz quadrada.

Tarefa: *Identifique as funções do R utilizadas para realizar as operações de soma, multiplicação e subtração. Registre essas funções no arquivo* `main.R`*, inserindo comentários textuais sobre cada uma delas.*

Chapter 5

Funções e objetos

A gramática da linguagem R possui dois elementos principais: as palavras reservadas e as palavras-chave. As ***palavras reservadas*** constituem signos com significado especial para a linguagem. Elas não podem ser utilizadas para outros fins que não aqueles especificados internamente no R. Algumas dessas palavras reservadas são `if`, `else`, `while`, `TRUE` e `FALSE`[1].

As ***palavras-chave*** são aquelas utilizadas para identificar funções e objetos. ***Funções*** nada mais são do que operações matemáticas e lógicas[2]. Assim, as palavras-chave que identificam funções são aquelas que acionam tais operações. São elas que possibilitam, por exemplo, enviar ao R o comando para que realize determinada análise estatística de determinado conjunto de dados. A maioria das operações utilizadas na análise de dados já está definida na linguagem do R, como é o caso de `log`, `sqrt`, `sum`, `prod`, `diff`, `abs`, `cos`, `tan`, `det`, `exp`, `max`, `min`, `mean`, `median`, entre muitas outras.

Tarefa: *A aba de ajuda do RStudio possui, em seu canto superior direito, uma caixa de busca. Digite a primeira letra de seu nome e selecione, na lista suspensa que aparecer, uma função que lhe chamar a atenção. Qual é a operação matemática ou lógica realizada por essa função?*

Para acessar uma função no R, precisamos especificar seu nome e os dados que serão processados na pela operação matemática ou lógica que ela identifica. Esses dados são sempre especificados entre dois parênteses que seguem o nome da

[1] O editor de comandos do RStudio destaca as palavras reservadas em azul para facilitar sua identificação. Busque pelo termo *reserved* no painel de ajuda do RStudio para conhecer todas as palavras reservadas da linguagem R.

[2] Como diria John Chambers, tudo o que "acontece" no R acontece pela ação de uma função.

função. Por exemplo, para calcular a raiz quadrada de 25, precisamos especificar que a função usada é `sqrt` e o valor numérico a ser operado é 25.

```
x <- 25
y <- sqrt(x)
print(y)
```

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

O bloco acima apresenta outro elemento importante do R: os objetos. Um ***objeto*** nada mais é do que uma estrutura de dados[3]. Essas estruturas de dados podem ser vetores, matrizes, listas, entre muitas outras. Elas são inseridas ou criadas pelo próprio usuário ou produzidas como resultado do processamento dos dados. No exemplo anterior, temos dois objetos cujos nomes são `x` e `y`. O objeto `x` armazena o valor numérico 25, enquanto o resultado da operação raiz quadrada é armazenado no objeto `y`. Para vermos o conteúdo do objeto `y`, basta usar a função `print`.

Tarefa: *Pelo teorema de Pitágoras, o comprimento da hipotenusa de um triângulo retângulo pode ser calculado em função do comprimento de seus catetos. Calcule o comprimento da hipotenusa de um triângulo retângulo cujos catetos possuem medidas $b = 3$ e $c = 4$. Registre as operações no arquivo* ***`main.R`*** *usando objetos para armazenar os valores numéricos.*

Um tipo de objeto bastante útil na análise de dados é o vetor, ou seja, uma sequência de dados de mesmo tipo. Por exemplo, um vetor pode ser usado para armazenar os dados sobre os meses do ano, tanto no formato numérico, como no formato textual. Quando dois vetores possuem o mesmo comprimeiro e seus elementos possuem relação direta, podemos reunir os mesmos em uma matriz ou tabela. Com a função `str` podemos conhecer a estrutura de um objeto, o que nos ajuda a decidir como usar o mesmo nas análises subsequentes.

```
numero <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)
nome <- c("Jan", "Fev", "Mar", "Abr", "Mai", "Jun",
          "Jul", "Ago", "Set", "Out", "Nov", "Dez")
meses <- data.frame(numero, nome)
str(meses)
```

```
## 'data.frame':    12 obs. of  2 variables:
##  $ numero: num  1 2 3 4 5 6 7 8 9 10 ...
##  $ nome  : Factor w/ 12 levels "Abr","Ago","Dez",..: 5 4 9 1 8 7 6 2 12 11 ...
```

Tarefa: *Analise o bloco acima, descreva o conteúdo de cada objeto criado e*

[3] Como diria John Chambers, tudo o que "existe" no R é um objeto.

explique o propósito de cada função usada. Registre suas observações no arquivo `main.R`*, reproduzindo as operações acima.*

Chapter 6

Estatísticas descritivas

```
dados <- read.table(file = "data/iris.csv", dec = ",", header = TRUE)
str(dados)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
summary(dados)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

```
r <- cor(dados[, c("Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width")])
print(r)
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
```

```
## Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
## Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
## Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
## Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

```
r <- round(r, 2)
write.table(r, file = "res/tab/correlacao.csv", dec = ",")
```

Chapter 7

Gráficos univariados

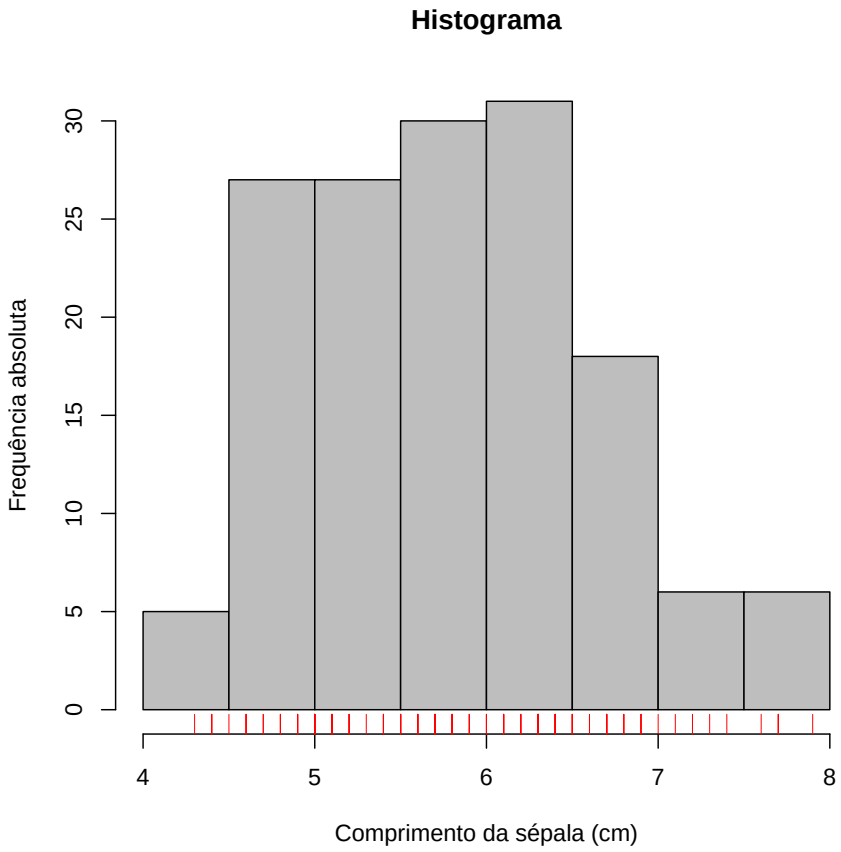
```
dados <- read.table(file = "data/iris.csv", dec = ",", header = TRUE)
str(dados)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
hist(dados[, "Sepal.Length"])
```

Tarefa. *No painel inferior direito do RStudio, acesse a aba* Packages *e encontre o pacote chamado* `graphics`*. Navegue pelo índice de funções do pacote* `graphics` *e escolha a função gráfica que mais chamar a sua atenção. Descreva o propósito dessa função e tente replicar os exemplos mostrados em sua página de ajuda.*

```
hist(dados[, "Sepal.Length"],
     main = "Histograma",
     xlab = "Comprimento da sépala (cm)",
     ylab = "Frequência absoluta",
     col = "gray")
rug(dados[, "Sepal.Length"], col = "red")
```




```
dev.off()
png("res/fig/histograma.png")
hist(dados[, "Sepal.Length"],
     main = "Histograma",
     xlab = "Comprimento da sépala (cm)",
     ylab = "Frequência absoluta",
     panel.first = grid(),
     col = "gray")
rug(dados[, "Sepal.Length"], col = "red")
dev.off()
```

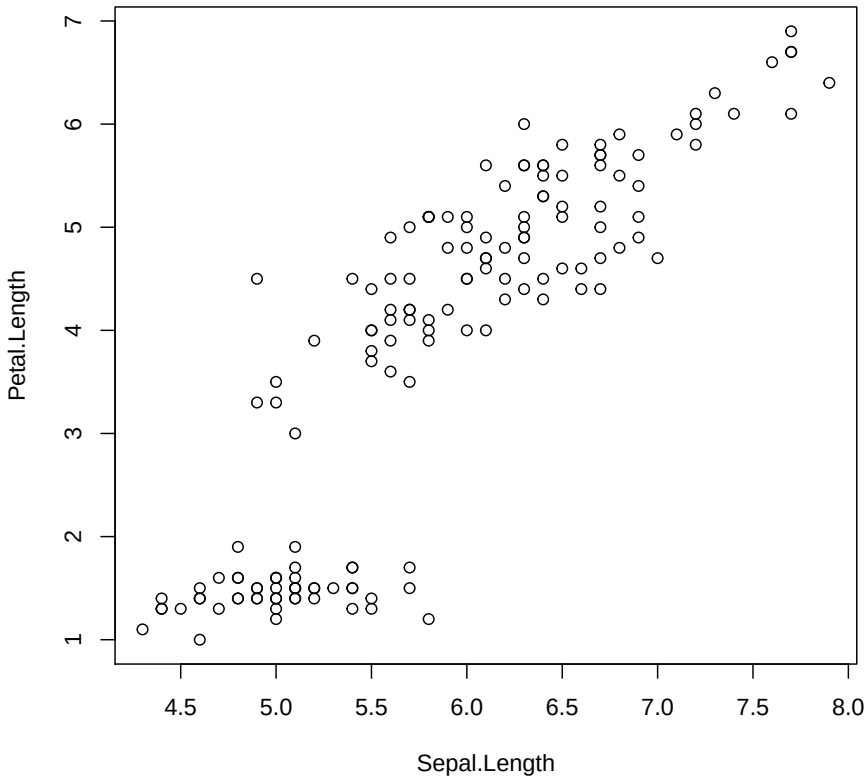
Chapter 8

Gráficos bivariados

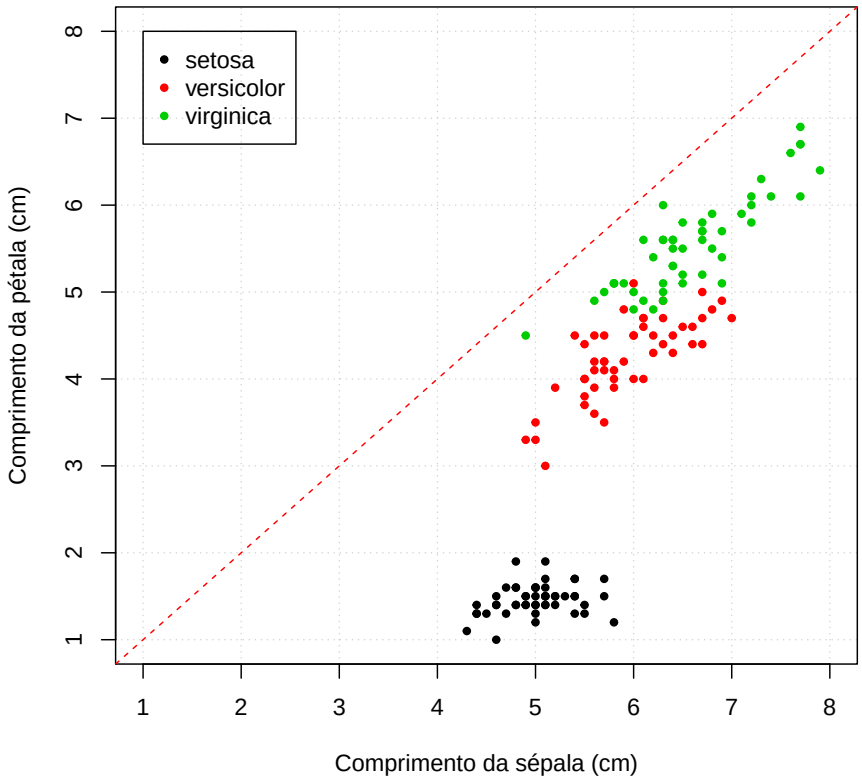
```
dados <- read.table("data/iris.csv", dec = ",", header = TRUE)
str(dados)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
plot(dados[, c("Sepal.Length", "Petal.Length")])
```




```
plot(dados[, c("Sepal.Length", "Petal.Length")],
     xlab = "Comprimento da sépala (cm)",
     ylab = "Comprimento da pétala (cm)",
     xlim = c(1, 8), ylim = c(1, 8), pch = 20,
     panel.first = grid(), col = dados[, "Species"])
abline(a = 0, b = 1, col = "red", lty = "dashed")
legend(x = 1, y = 8, legend = levels(dados[, "Species"]),
       col = 1:3, pch = 20, )
```


setosa
versicolor
virginica
Comprimento da pétala (cm)
Comprimento da sépala (cm)
1
2
3
4
5
6
7
8