

Introdução à Linguagem de Programação em R para tratamento de dados de poluição do ar

R básico e data frames

Mario Gavidia-Calderón, Rafaela Squizzato, Thiago Nogueira

05/02/2024

Universidade de São Paulo

Introdução

RStudio

Sintaxe Básica

Data Frames

Introdução

Por que o curso?

Efeitos adversos à saúde



Por que o curso?

Carga global de doenças da poluição do ar em 2015:

- 19 % de todas as mortes cardiovasculares.
- 23 % de todas as mortes por câncer de pulmão.

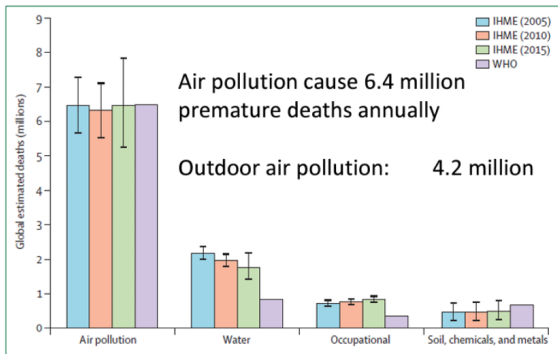
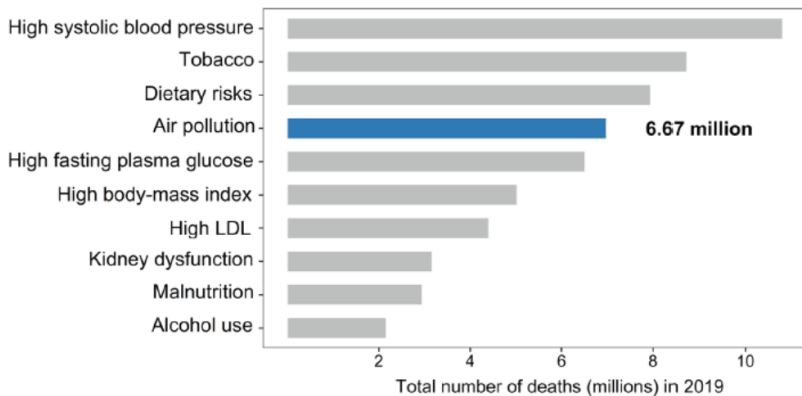


Figure 4: Global estimated deaths (millions) by pollution risk factor, 2005–15
Using data from the GBD study⁴² and WHO.³⁹ IHME=Institute for Health Metrics and Evaluation.

Por que o curso?

Ranking global de fatores de risco por número total de mortes em 2019

FIGURE 1 Global ranking of risk factors by total number of deaths from all causes in 2019.



Por que o curso?

Então

- Precisamos conhecer o nível da poluição.
- Para isso precisamos de medições: dados.
- Este curso tem como objetivo mostrar como trabalhar com dados de poluição do ar no R.

Por que o curso?

- Analizadores de poluentes medem concentrações com maior frequência.
- R é uma ótima ferramenta para mexer com dados.
- É importante conhecer uma linguagem de programação.

Um questionário



Por que R?

- R é uma **Linguagem de programação** para a análise de dados.
 - Um sistema para **estatística**.
 - Um sistema de computação gráfica e **estatística**.
 - Um ambiente para a análise de dados e **estatística**.



Por que R?

- É *open source* (é **livre**).
- Funciona em qualquer **sistema operacional**.
- Podemos trabalhar **com muitos dados e tipos**.
- Grande comunidade de usuários: **Muita ajuda on-line**.
- **Reprodutibilidade** das ciências.



Por que R?

- Muitos **pacotes** para muitas áreas das ciências.
 - **openair** → poluição do ar.
 - **sf** e **raster** → GIS.
 - **Rmarkdown** → Documentos e apresentações.
 - **shiny** → aplicações.
 - etc, etc, etc



RStudio

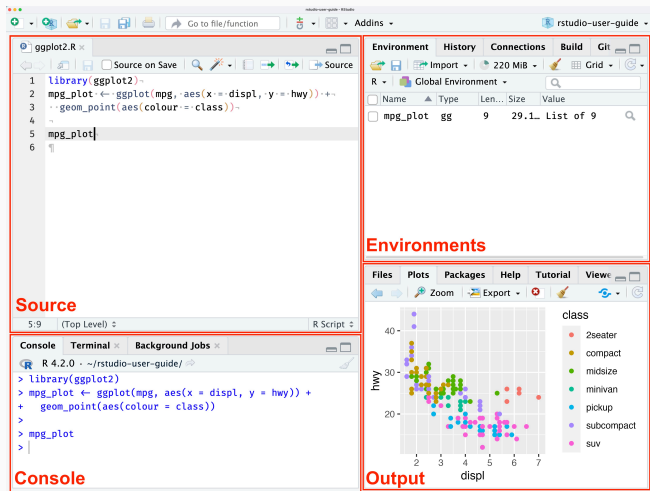


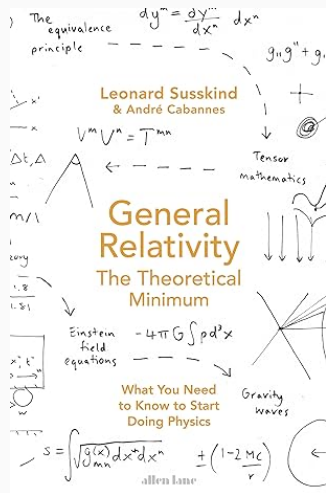
Figure 1: Distribuição das janelas do RStudio. Fonte: RStudio User Guide

- É importante aprender os *keyboard shortcuts*.
 - `Ctrl + 1`: Janela scripts.
 - `Ctrl + 2`: Janela console.
 - `Alt + - : <-`
 - `TAB`: Autocompleta nome de funções e diretórios de arquivos.
- Na console, `↑` e `↓` procura comandos anteriores.

Sintaxe Básica

The theoretical minimum

- What you need to know to start doing R



R como calculadora

- R é uma calculadora.
- Segue a ordem das operações

```
(5 + 10 * 2 / 4) ^ 2 - 5
```

```
## [1] 95
```

Declarar variáveis

- No R usamos `<-` em vez de `=` para definir variáveis.

```
R <- 8.314
```

```
R
```

```
## [1] 8.314
```

- Para comentar usamos `#`. O código após o `#` não é lido.

```
R <- 8.314 # Constante universal dos gases (J K / mol)  
R
```

```
## [1] 8.314
```

- Para usar funções: **nome_da_função()**.
- Dentro dos () colocamos os **argumentos**.

```
class(R)
```

```
## [1] "numeric"
```

Objetos

- No R existem diversos tipos de objetos.
- **character**

```
o3_nome <- "ozônio"  
class(o3_nome)
```

```
## [1] "character"
```

- **numeric**

```
this_year <- 2024  
g <- 9.81 # m/s2  
class(this_year)
```

```
## [1] "numeric"
```

Objetos

- booleans

```
verdade <- TRUE  
verdade
```

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

```
falso <- 5 > 10  
falso
```

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

```
muito_falso <- "cinco" == "5"  
muito_falso
```


Vetores

- É definido usando a função `c()`. Só podem ter um único tipo de objeto.

```
pontos_cardeais <- c("N", "E", "S", "W")  
pontos_cardeais # só character
```

```
## [1] "N" "E" "S" "W"
```

```
pontos_cardeais_graus <- c(0, 90, 180, 270)  
class(pontos_cardeais_graus) # só numeric
```

```
## [1] "numeric"
```

Vetores

- Uma **sequência** é definida `seq(início, final, intervalo)`

```
de_1ate5 <- seq(1, 5)
```

```
de_1ate5
```

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

```
pares_ate10 <- seq(0, 10, 2)
```

```
pares_ate10
```

```
## [1] 0 2 4 6 8 10
```

```
sec_float <- seq(0, 1, 0.2)
```

```
sec_float
```

- Para saber quantos elementos tem um vetor usamos a função `length`.

```
length(de_1ate5)
```

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

```
length(sec_float)
```

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

Vetores

- Para **selecionar** elementos do vetor: **nome_vetor[posição]**:

```
# Primeiro elemento
```

```
pontos_cardeais_graus[1]
```

```
## [1] 0
```

```
# Último elemento
```

```
pontos_cardeais_graus[4]
```

```
## [1] 270
```

Vetores

- Podemos **selecionar** vários elementos usando **outro vetor**

```
# Segundo y tercero  
pontos_cardeais[c(2, 3)]
```

```
## [1] "E" "S"
```

- Podemos **eliminar** elementos usando **nome_vetor[-posição]**

```
GEE <- c("H2O", "CO2", "O2", "CH4")  
GEE
```

```
## [1] "H2O" "CO2" "O2"  "CH4"
```

```
GEE[-3] # Oxigênio não é GEE
```

- Podemos Substituir um elemento do vetor assim:

```
# Substituímos Oxigênio por Ozônio
```

```
GEE[3] <- "O3"
```

```
GEE
```

```
## [1] "H2O" "CO2" "O3"  "CH4"
```

Exercício 1

Criar três vetores. Um vetor chamado **pol_sp** com os poluentes que tem padrão de qualidade do ar no Estado de São Paulo. Outro vetor chamado **pol_amostra** com o **menor tempo de amostragem** em horas. Finalmente, um vetor chamado **pol_pqa** com o respectivo padrão de qualidade do ar.

Referência: Padrões de qualidade do ar CETESB

Exercicio 1

```
pol_sp <- c("MP10", "MP2.5", "SO2", "NO2", "O3", "CO", "FMC", "PTS", "P")
pol_amostra <- c(24, 24, 24, 1, 8, 8, 24, 24, 365*24)
pol_pqa <- c(100, 50, 40, 240, 130, 9, 100, 240, 0.5)
```


Operações *Element-wise*

```
tempC <- c(27, 32, 28, 26)
tempK <- tempC + 273.15
tempK
```

```
## [1] 300.15 305.15 301.15 299.15
```

```
tempk_chr <- as.character(tempK)
str(tempk_chr)
```

```
## chr [1:4] "300.15" "305.15" "301.15" "299.15"
```

Operações *Element-wise*

```
pol_atr <- c("nome", "pm", "conc", "unit")  
paste("o3", pol_atr, sep = "_")
```

```
## [1] "o3_nome" "o3_pm"   "o3_conc" "o3_unit"
```

Data Frames

- Um **data frame** é uma **tabela**
- Uma matriz **indexada**: tem nomes das **colunas** e **linhas**.
- Cada **coluna** é uma **variável**.
- Cada **linha** é uma **observação**.
- Um conjunto de vetores.

data frame

- Criamos um **data frame** usando a função **data.frame()**

```
gases <- c("N2", "O2", "Ar", "CO2")
massa_molar <- c(28, 32, 40, 12 + 2 * 16)
percentagem <- c(78.08, 20.95, 0.9, 0.04)

ar <- data.frame(gas = gases,
                  W = massa_molar,
                  per = percentagem)
```

data frame

```
ar
```

```
##   gas  W   per  
## 1  N2 28 78.08  
## 2  O2 32 20.95  
## 3  Ar 40  0.90  
## 4 CO2 44  0.04
```

data frame

- Criamos um **data frame** usando a função **data.frame()**

```
ar <- data.frame(gas = c("N2", "O2", "Ar", "CO2"), # Ou diretamente  
                 W = c(28, 32, 40, 12 + 2 * 16),  
                 per = c(78.08, 20.95, 0.9, 0.04))
```

data frame

```
ar
```

```
##   gas  W   per  
## 1  N2 28 78.08  
## 2  O2 32 20.95  
## 3  Ar 40  0.90  
## 4 CO2 44  0.04
```


Selecionar colunas \$

- Seleccionamos uma **coluna** de um **data frame** como um **vetor**
- Sintaxis: `df$nome_coluna`
- E.g. Nome dos componentes do ar

```
ar$gas
```

```
## [1] "N2" "O2" "Ar" "CO2"
```

```
class(ar$gas)
```

```
## [1] "character"
```

Selecionar filas e colunas []

- Seleccionamos uma **coluna** de um **data frame** como um **data frame**
- Sintaxis: `df[intereiro]` ou `df[nome_coluna]`
- E.g. Nome dos componentes do ar

```
ar[1] # ou ar["gas"]
```

```
##      gas  
## 1    N2  
## 2    O2  
## 3    Ar  
## 4   CO2
```

```
class(ar[1])
```

Selecionar filas e colunas []

- Algumas funções precisam **vetores** como **input**
- e.g. média massa molar

```
mean(ar["W"])
```

```
## Warning in mean.default(ar["W"]): argument is not numeric or logical  
## NA
```

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

```
mean(ar$W)
```

```
## [1] 36
```

Criando novas colunas

- Usamos `$`: `df$nova_coluna <- nova_coluna`

```
# Adicionamos o nome completo dos gases.
```

```
ar$name <- c("Nitrogênio",  
            "Oxigênio",  
            "Argônio",  
            "Dióxido de Carbono")
```

```
ar
```

```
##   gas  W   per          name  
## 1  N2 28 78.08      Nitrogênio  
## 2  O2 32 20.95       Oxigênio  
## 3  Ar 40  0.90       Argônio  
## 4 CO2 44  0.04 Dióxido de Carbono
```

Algumas funções importantes

- Número de linhas: `nrow()`
- Número de colunas: `ncol()`

```
nrow(ar)
```

```
## [1] 4
```

```
ncol(ar)
```

```
## [1] 4
```

Algumas funções importantes

- Tipo de objeto de cada coluna: **str()**

```
str(ar)
```

```
## 'data.frame':    4 obs. of  4 variables:  
## $ gas : chr  "N2" "O2" "Ar" "CO2"  
## $ W   : num  28 32 40 44  
## $ per : num  78.08 20.95 0.9 0.04  
## $ name: chr  "Nitrogênio" "Oxigênio" "Argônio" "Dióxido de Carbono"
```

- nome das colunas

```
names(ar)
```

```
## [1] "gas" "W" "per" "name"
```

Algumas funções importantes

- Primeiras observações: **head()**
- Últimas observações: **tail()**

head(ar)

```
##      gas  W   per                name
## 1   N2 28 78.08          Nitrogênio
## 2   O2 32 20.95          Oxigênio
## 3   Ar 40  0.90          Argônio
## 4  CO2 44  0.04 Dióxido de Carbono
```

tail(ar)

```
##      gas  W   per                name
```

Algumas funções importantes

- Primeiras observações: **head()**
- Últimas observações: **tail()**

```
head(ar, 2)
```

```
##   gas  W  per      name
## 1  N2 28 78.08 Nitrogênio
## 2  O2 32 20.95  Oxigênio
```

```
tail(ar, 2)
```

```
##   gas  W  per      name
## 3  Ar 40 0.90      Argônio
## 4 CO2 44 0.04 Dióxido de Carbono
```


Substituição de coluna

- Para substituir uma coluna, ela tem que ter o mesmo número de filas.

```
# Em espanhol
```

```
ar$nombrs <- c("Nitrógeno",  
               "Oxígeno",  
               "Argón",  
               "Dióxido de carbono")
```

```
ar
```

| ## | gas | W | per | name | nombrs |
|------|-----|----|-------|--------------------|--------------------|
| ## 1 | N2 | 28 | 78.08 | Nitrogênio | Nitrógeno |
| ## 2 | O2 | 32 | 20.95 | Oxigênio | Oxígeno |
| ## 3 | Ar | 40 | 0.90 | Argônio | Argón |
| ## 4 | CO2 | 44 | 0.04 | Dióxido de Carbono | Dióxido de carbono |

Exercicio 2

- Criar um `data.frame` chamado `pqa_sp` usando os vetores do Exercício 1.
- Adiciona uma coluna chamada `pm` com o peso molecular de cada poluente. Para MP colocar 1.
- Adiciona uma coluna chamada `nome` com o nome completo de cada poluente.

Exercicio 2

```
pqa_sp <- data.frame(  
  pol_sp,  
  pol_amostra,  
  pol_pqa  
)  
pqa_sp$pm <- c(1, 1, 32 + 16 * 2, 14 + 16 * 2, 16 * 3, 12 + 16, 1, 1, 2)
```

data frames: Ler arquivos .csv

- Vamos ler dados do ano 2023 do aeroporto de Guarulhos.
- Os dados são baixados do site **ASOS Network da Iowa State University**.
- No R para ler tabelas em csv (e para outros formatos) usamos a função `'read.table()'`.
- Esta função precisa saber o **diretório do arquivo**. Para isso podemos usar a função `file.choose()`.

data frames Ler arquivos .csv

```
gru <- read.table("../..data/SBGR.csv", # ou file.choose()
                  header = TRUE, # se as colunas tem nome
                  sep = ",", # o separador de colunas
                  dec = ".") # o separador decimal
```

data frames Ler arquivos .csv

- Exploramos o nome das colunas do data frame:

```
names(gru)
```

```
## [1] "station"          "valid"             "tmpf"
## [4] "dwpf"             "relh"              "drct"
## [7] "sknt"             "p01i"              "alti"
## [10] "mslp"             "vsby"              "gust"
## [13] "skyc1"            "skyc2"              "skyc3"
## [16] "skyc4"            "skyl1"              "skyl2"
## [19] "skyl3"            "skyl4"              "wxcodes"
## [22] "ice_accretion_1hr" "ice_accretion_3hr" "ice_accretion_6hr"
## [25] "peak_wind_gust"   "peak_wind_drct"    "peak_wind_time"
## [28] "feel"             "metar"              "snowdepth"
```

data frames Ler arquivos .csv

- Exploramos a estrutura do data frame.

```
str(gru)
```

```
## 'data.frame':      8736 obs. of  30 variables:
## $ station      : chr  "SBGR" "SBGR" "SBGR" "SBGR" ...
## $ valid        : chr  "2023-01-01 00:00" "2023-01-01 01:00" "20
## $ tmpf         : num  69.8 68 69.8 68 66.2 64.4 64.4 64.4 62.6
## $ dwpf         : num  66.2 66.2 68 66.2 66.2 64.4 64.4 64.4 62
## $ relh         : num  88.3 94 94 94 100 ...
## $ drct         : num  80 110 80 70 60 80 80 100 0 0 ...
## $ sknt         : num  3 5 6 6 6 7 4 3 0 0 ...
## $ p01i         : num  0 0 0 0 0 0 0 0 0 0 ...
## $ alti         : num  30 1 30 1 30 1 30 1 30
```

- A temperatura está em Farenheit e a velocidade de vento está em nós.

```
gru$tc <- (gru$tmpf - 32) * 5 / 9
```


- A temperatura está em Farenheit e a velocidade de vento está em nós.

```
gru$ws <- gru$sknt * 0.51
```

Análise exploratória de dados (AED)

- Vamos olhar as estatísticas básicas usando `summary()` da temperatura, velocidade do vento, umidade do ar.

```
summary(gru[c("tc", "relh", "ws")])
```

| ## | tc | relh | ws |
|----|---------------|----------------|----------------|
| ## | Min. : 7.00 | Min. : 18.33 | Min. : 0.000 |
| ## | 1st Qu.:17.00 | 1st Qu.: 65.54 | 1st Qu.: 1.530 |
| ## | Median :20.00 | Median : 83.09 | Median : 2.550 |
| ## | Mean :20.71 | Mean : 78.27 | Mean : 2.763 |
| ## | 3rd Qu.:24.00 | 3rd Qu.: 93.79 | 3rd Qu.: 3.570 |
| ## | Max. :37.00 | Max. :100.00 | Max. :11.730 |

Análise exploratória de dados (AED)

- Ou podemos calcular *manualmente*:

```
mean(gru$tc, na.rm = TRUE) # na.rm = TRUE não considera NA
```

```
## [1] 20.71028
```

```
median(gru$tc, na.rm = TRUE)
```

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

Análise exploratória de dados (AED)

```
max(gru$tc, na.rm = TRUE)
```

```
## [1] 37
```

```
min(gru$tc, na.rm = TRUE)
```

```
## [1] 7
```

```
sd(gru$tc, na.rm = TRUE)
```

```
## [1] 4.903718
```

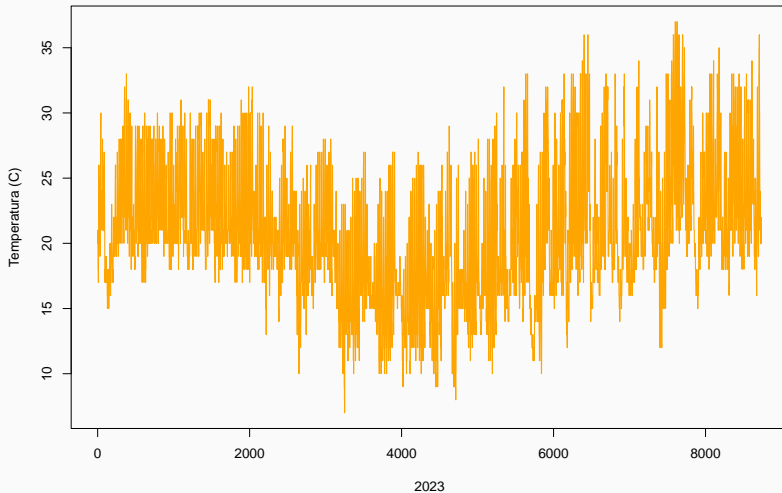
Exercio 3

- Baixe os dados do aeroporto de campo de marte para o ano 2023.
- Transforme a Temperatura para Celcius, e a velocidade do vento para m s^{-1} .
- Existen datos faltantes?
- Todos as columnas foram lidas corretamente?

Um plot simples

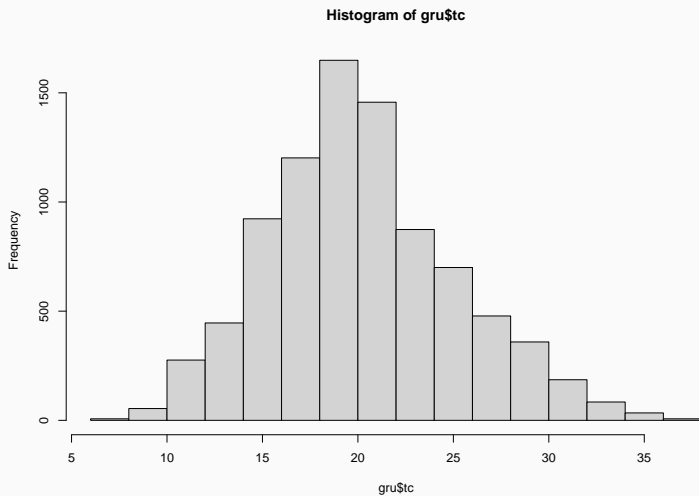
```
plot(gru$tc, # vetor para plotar  
      t = "l", # tipo de plot, l = linha  
      xlab = "2023", # nome do eixo x  
      ylab = "Temperatura (C)", # nome eixo y  
      col = "orange", # color da linha  
      lwd = 1.25 # largura linha  
    )
```

Um plot simples



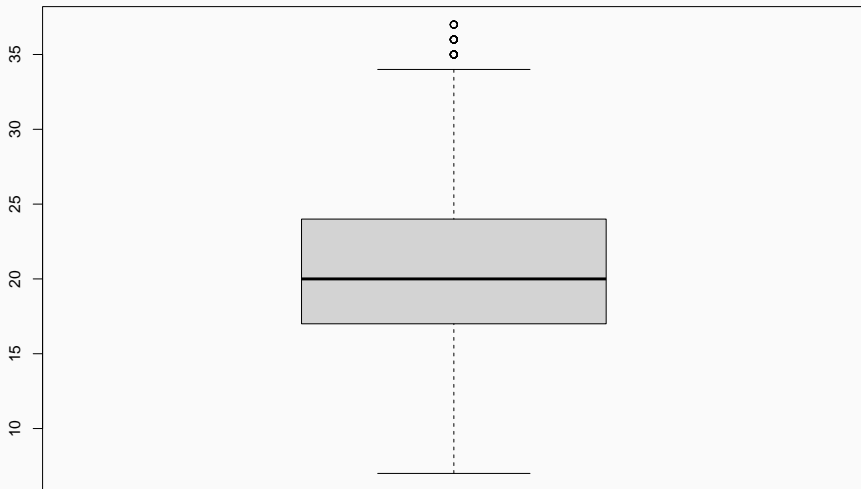
Histograma

```
hist(gru$tc)
```



Um diagrama de caixa

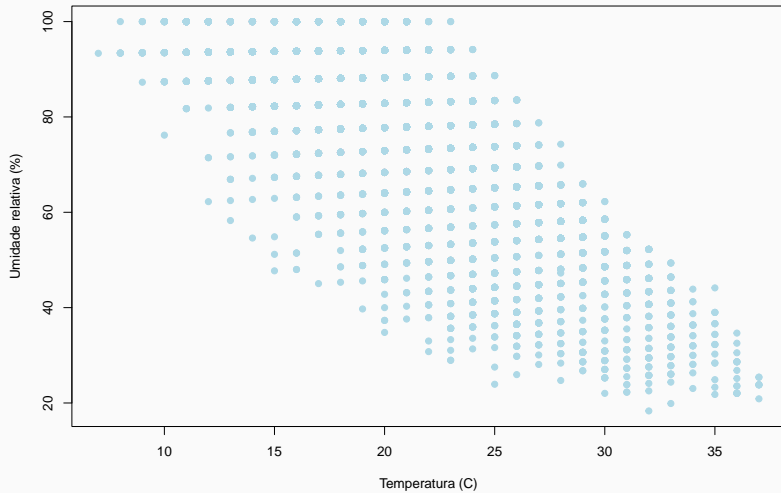
```
boxplot(gru$tc)
```



Um scatter plot

```
plot(gru$tc, # valores eixo x
     gru$relh, # valores eixo y
     col = "lightblue", # color dos pontos
     xlab = "Temperatura (C)", # nome eixo x
     ylab = "Umidade relativa (%)" # nome eixo y
     )
```

Um scatter plot



Coeficiente de correlação

```
cor(gru$tc, # variavel x  
     gru$relh) # variavel y
```

```
## [1] -0.7436237
```

Exercício 4

- Calcule a média, mediana, min, max e sd, da Temperatura, Umidade relativa, Velocidade do vento no Aeroporto Campo de Marte.
- Faça uma série temporal, um histograma e um boxplot destes parâmetros.