

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

---

Mario Gavidia-Calderón, Rafaela Squizzato, Thiago Nogueira

06/02/2024

Universidade de São Paulo

Pacotes

Do Excel para R

Subset data frames

qualR

Plots

# Pacotes

---

- Pacotes são como as **extensões** no browser.
- Conjunto de funções específicas para tarefas específicas.

- Para instalar pacotes usamos a função  
`install.packages("nome_do_biblioteca")`

# Instalando Rmarkdown

```
install.packages("rmarkdown")
```

- Por exemplo, para ler arquivos `.xls` podemos instalar a biblioteca `readxls`

```
# Instalando readxl  
install.packages("readxl")
```

## Do Excel para R

---

- Um jeito de ler arquivos `.xls` é abrir os arquivos em Excel ou Google Sheet e salvar como `.csv`.
- Depois ler a tabela usando a função `read.table()`
- As vezes vale a pena abrir no Excel ou Google Sheet e mudar o nome das colunas.
- Também podemos usar a função `read_excel()` do pacote `readxl`



- Vamos usar a base de dados de qualidade do ar da WHO

```
library(readxl)
who <- read_excel("../..//data/who_aap_2021_v9_11august2022.xlsx",
                  sheet = 2) # Nome o numero do sheet no arquivo
```

## Base de dados da WHO

- Este é um exemplo de nomes de coluna *complicados*.

```
str(who)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    32191 obs. of  15 variables:
## $ WHO Region                               : chr  "Eastern Mediterranean"
## $ ISO3                                       : chr  "AFG" "ALB" "ALB" "ALB"
## $ WHO Country Name                         : chr  "Afghanistan" "Albania"
## $ City or Locality                         : chr  "Kabul" "Durres" "Durres"
## $ Measurement Year                        : num  2019 2015 2016 2015
## $ PM2.5 (µg/m3)                          : num  119.8 NA 14.3 NA NA
## $ PM10 (µg/m3)                           : num  NA 17.6 24.6 NA NA
## $ NO2 (µg/m3)                             : num  NA 26.6 24.8 24 26.3
## $ PM25 temporal coverage (%)              : num  18 NA NA NA NA NA
```

## Subset data frames

---

# Subset

- As vezes precisamos selecionar um sub-conjunto de dados do data frame.
- Um jeito simples é usar a função `subset()`

```
novo_df <- subset(df_original,  
                  subset = condição baseado nas filas,  
                  select = vetor com as colunas)
```

## Subset dados da who para Brasil

- Vamos selecionar os dados do Brasil.

```
who_br <- subset(who,  
                 `WHO Country Name` == "Brazil")
```

- Olha como o nome da coluna que tem espaços é escrita usando ' '.

## Subset dados da who para Brasil

- Vamos selecionar os dados do Brasil.

```
# Examinando o novo data frame
```

```
nrow(who_br)
```

```
## [1] 479
```

```
head(unique(who_br$`City or Locality`))
```

```
## [1] "Americana" "Aracatuba" "Araraquara" "Araucaria" "Barra M"
```

```
## [6] "Bauru"
```

- Olha como o nome da coluna que tem espaços é escrita usando ' '.

## Subset dados da WHO para Sao Paulo

```
who_sp <- subset(who,  
  subset = `City or Locality` == "Sao Paulo" ,  
  select = c("Measurement Year",  
    "PM2.5 (µg/m3)",  
    "PM10 (µg/m3)",  
    "NO2 (µg/m3)" ))
```

## Subset dados da WHO para Sao Paulo

- Vamos arrumar um pouco o data frame

```
names(who_sp)
```

```
## [1] "Measurement Year" "PM2.5 (µg/m3)"    "PM10 (µg/m3)"     "NO2 (µg/m3)"
```

```
names(who_sp) <- c("year", "pm25", "pm10", "no2")
```

```
names(who_sp)
```

```
## [1] "year" "pm25" "pm10" "no2"
```



## Anos acima do padrão da WHO

```
plot(who_sp$year, who_sp$no2, t = "l", lwd = 1.5,  
     xlab = "Anos", ylab = "NO2 (ug/m3)")  
abline(h = 40, lwd = 1.5, col = "red")  
legend("topright", col = "red", lwd = 1.5, legend = "NO2 WHO AQS")
```



- Que cidades ultrapassarm o Padrão de qualidade do ar de NO<sub>2</sub> no ano 2019?

qualR

---

- Pacote desenvolvido para baixar os dados da CETESB dentro do R.
- Gera dataset prontos e completos para análise:
  - Horários faltantes preenchidos com **NA**.
  - Coluna **date** tipo **POSIXct** para usar diretamente com **openair**.
- Precisa de ter cadastro na plataforma QUALAR da CETESB.
- Referencia qualR

- O jeito mais fácil é o seguinte:

```
install.packages('qualR',  
                 repos = c('https://ropensci.r-universe.dev',  
                           'https://cloud.r-project.org'))
```

## Atenção

Para usar 'qualR' você precisa ter uma conta no sistema QUALAR da CETESB. Você precisa do **usuário** e da **senha**.

## Códigos dos parâmetros e das estações

- Para saber os códigos das estações usamos `cetesb_aqs` na coluna `code`.

```
library(qualR)
head(cetesb_aqs, 4)
```

##		name	code	lat	lon	loc
## 1		Americana	290	-22.72425	-47.33955	Interior
## 2		Araçatuba	107	-21.18684	-50.43932	Interior
## 3		Araraquara	106	-21.78252	-48.18583	Interior
## 4		Bauru	108	-22.32661	-49.09276	Interior

## Códigos dos parâmetros e das estações

- Para saber os códigos dos parâmetros `cetesb_param` na coluna `code`.

```
library(qualR)
tail(cetesb_param, 4)
```

##		name	units	code
## 17	TEMP (Temperatura do Ar)	°C	25	
## 18	TOL (Tolueno)	ug/m3	62	
## 19	UR (Umidade Relativa do Ar)	%	28	
## 20	VV (Velocidade do Vento)	m/s	24	



## Baixar um poluente de uma estação

- Vamos baixar dados de **ozônio** da **primeira semana de janeiro de 2024** na estação **Pinheiros**. Usamos a função `cetesb_retrieve_param()`.

```
pin_o3 <- cetesb_retrieve_param(  
  username = Sys.getenv("QUALAR_USER"), # Trocar pelo seu usuário  
  password = Sys.getenv("QUALAR_PASS"), # Trocar pela sua senha  
  parameters = 63, # de cetesb_param  
  aqs_code = 99, # de cetesb_aqs,  
  start_date = "01/01/2024", # Formato dd/mm/yyyy  
  end_date = "07/01/2024" # Formato dd/mm/yyyy  
)
```

## Baixar um poluente de uma estação

- Também aceita o nome da estação (igual do que em `cetesb_aqs`) e o nome do parâmetro (igual do que em `cetesb_param`).

```
pin_o3 <- cetesb_retrieve_param(  
  username = Sys.getenv("QUALAR_USER"), # Trocar pelo seu usuário  
  password = Sys.getenv("QUALAR_PASS"), # Trocar pela sua senha  
  parameters = "O3", # de cetesb_param  
  aqs_code = "Pinheiros", # de cetesb_aqs,  
  start_date = "01/01/2024", # Formato dd/mm/yyyy  
  end_date = "07/01/2024" # Formato dd/mm/yyyy  
)
```

## Baixar vários poluente de uma estação

- Também podemos baixar vários poluentes de uma estação. Só precisamos definir os poluentes para baixar em um **vetor**. Vamos baixar O<sub>3</sub>, PM<sub>2.5</sub>, e NO<sub>x</sub>.

```
pols <- c("O3", "MP2.5", "NOx") # Olhar usando cetesb_aqs
pin_o3 <- cetesb_retrieve_param(
  username = Sys.getenv("QUALAR_USER"), # Trocar pelo seu usuário
  password = Sys.getenv("QUALAR_PASS"), # Trocar pela sua senha
  parameters = pols, # vetor com poluentes
  aqs_code = "Pinheiros", # de cetesb_aqs,
  start_date = "01/01/2024", # Formato dd/mm/yyyy
  end_date = "07/01/2024" # Formato dd/mm/yyyy
)
```

## Baixar vários poluentes e meteorologia de uma estação

- Só precisamos definir os parâmetros para baixar em um vetor. Vamos baixar  $PM_{2.5}$  e velocidade e direção do vento.

```
params <- c("MP2.5", "VV", "DV") # Olhar usando cetesb_aqs
pin_pm25 <- cetesb_retrieve_param(
  username = Sys.getenv("QUALAR_USER"), # Trocar pelo seu usuário
  password = Sys.getenv("QUALAR_PASS"), # Trocar pela sua senha
  parameters = params, # vetor com poluentes
  aqs_code = "Pinheiros", # de cetesb_aqs,
  start_date = "01/01/2024", # Formato dd/mm/yyyy
  end_date = "07/01/2024" # Formato dd/mm/yyyy
)
```

## Baixar vários poluentes e meteorologia de uma estação

- Você não precisa colocar os argumentos das funções **mas precisa seguir a ordem dos argumentos.**

```
params <- c("MP2.5", "VV", "DV")
pin_pm25 <- cetesb_retrieve_param(
  Sys.getenv("QUALAR_USER"), # username
  Sys.getenv("QUALAR_PASS"), # password
  params, # parameters
  "Pinheiros", # aqs_code
  "01/01/2024", # start_date
  "07/01/2024" # end_date
)
```

## Salvar dados em .csv

- Pode ser que você precisar usar outro software (e.g. PMF). Então você pode usar exportar os dados em .csv. Só adicionar o argumento `to_csv = TRUE`.

```
params <- c("MP2.5", "VV", "DV") # Olhar usando cetesb_aqs
pin_pm25 <- cetesb_retrieve_param(
  username = Sys.getenv("QUALAR_USER"), # Trocar pelo seu usuário
  password = Sys.getenv("QUALAR_PASS"), # Trocar pela sua senha
  parameters = params, # vetor com poluentes
  aqs_code = "Pinheiros", # de cetesb_aqs,
  start_date = "01/01/2024", # Formato dd/mm/yyyy
  end_date = "07/01/2024", # Formato dd/mm/yyyy
  to_csv = TRUE
)
```

## Salvar dados em .csv

- O arquivo salvo tem o nome `Pinheiros_MP2.5_VV_DV_01-01-2024_07-01-2024.csv` e ficará na pasta de trabalho (conferir usando `getwd( )`).

## Um poluente várias estações

- As vezes você precisa de comparar valores de várias estações. Neste exemplo vamos baixar valores de NO<sub>x</sub> da estação Ibirapuera e Pinheiros.

```
aqs <- c(99, 83) # de cetesb_aqs, Pinheiros é 99 e Ibirapuera 83
nox_pin_ibi <- lapply(
  aqs,
  cetesb_retrieve_param,
  username = Sys.getenv("QUALAR_USER"),
  password = Sys.getenv("QUALAR_PASS"),
  parameters = "NOx",
  start_date = "01/01/2024",
  end_date = "07/07/2024"
)
```



## Um poluente várias estações

- O resultado de usar `lapply` é uma lista. Vamos a transformar em um `data.frame`.

```
nox_all <- do.call(rbind, nox_pin_ibi)
```

## Um poluente várias estações

- Usando subset podemos separar os data frames.

```
nox_pin <- subset(nox_all, subset = aqs == "Pinheiros")  
nox_ibi <- subset(nox_all, subset = aqs == "Ibirapuera")
```

# Um poluente várias estações

- Vamos comparar as estações

```
mean(nox_pin$nox, na.rm = TRUE)
```

```
## [1] 24.1
```

```
mean(nox_ibi$nox, na.rm = TRUE)
```

```
## [1] 13.50479
```

- Pinheiros tem maior concentração de NOX do que Ibirapuera. Por que?

## Outras funções.

- `cetesb_retrieve_param()` é a função mais importante e a mais usada.
- Existem outras funções que não precisam do argumento **parameters** pois foram desenhados para baixar parâmetros específicos:
  - `cetesb_retrieve_pol()`: Baixa todos os **poluentes**.
  - `cetesb_retrieve_met()`: Baixa todos os **parâmetros meteorológicos**.
  - `cetesb_retrieve_met_pol()`: Baixa todos os **parâmetros da estação**.

# Plots

---

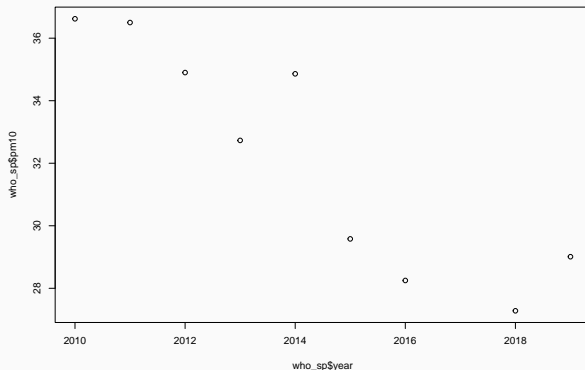
- *Uma figura vale mais do que 100 palavras.*
- Vamos aprofundizar como usar **R Base Graphics** para criar os plots.
- Seguires as *10 simple rules for better figures* do **Rougier et al. (2014)**

- A principal função é **plot()**. Podemos usar **plot()** para criar **séries de tempo** e **gráfico de dispersão**.
- Outras função são:
  - **hist()**: Cria histogramas.
  - **barplot()**: Diagrama de barras.
  - **boxplot()**: Diagrama de caixas.

# Séries temporais

- Vamos a plotar a série temporal da concentrações de pm10.

```
plot(who_sp$year, who_sp$pm10)
```

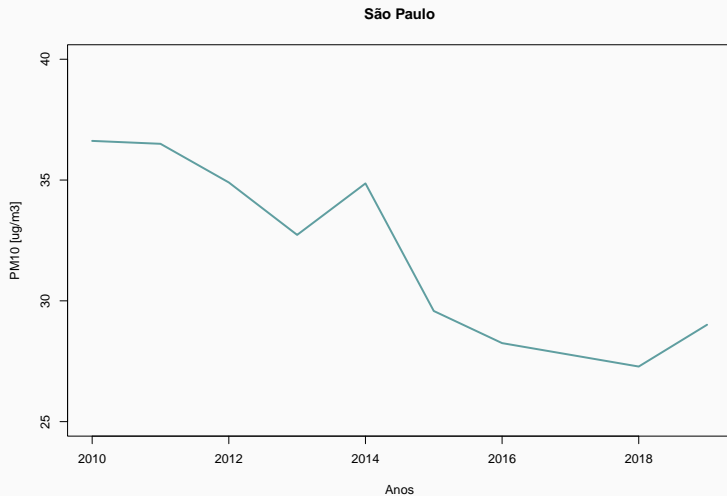




## Séries temporais

```
plot(who_sp$year, who_sp$pm10, # Dado eixo x, dado eixo y
     t = "l", # tipo linha
     lwd = 2.5, # largura da linha
     col = "cadetblue", # color da linha
     ylim = c(25, 40), # limite do eixo y
     xlab = "Anos", # nome eixo x
     ylab = "PM10 [ug/m3]", # nome eixo y
     main = "São Paulo" # titulo do plot
)
```

# Séries temporais



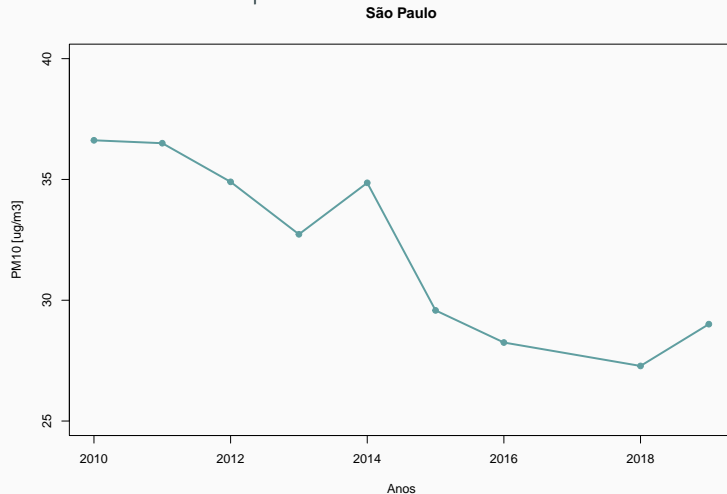
# Séries temporais

- Podemos adicionar pontos

```
plot(who_sp$year, who_sp$pm10, # Dado eixo x, dado eixo y
     t = "l", # tipo linha
     lwd = 2.5, # largura da linha
     col = "cadetblue", # color da linha
     ylim = c(25, 40), # limite do eixo y
     xlab = "Anos", # nome eixo x
     ylab = "PM10 [ug/m3]", #
     main = "São Paulo"
)
points(who_sp$year, who_sp$pm10,
       pch = 19, # tipo do ponto
       col = "cadetblue") # color do ponto
```

# Séries temporais

- Podemos adicionar pontos



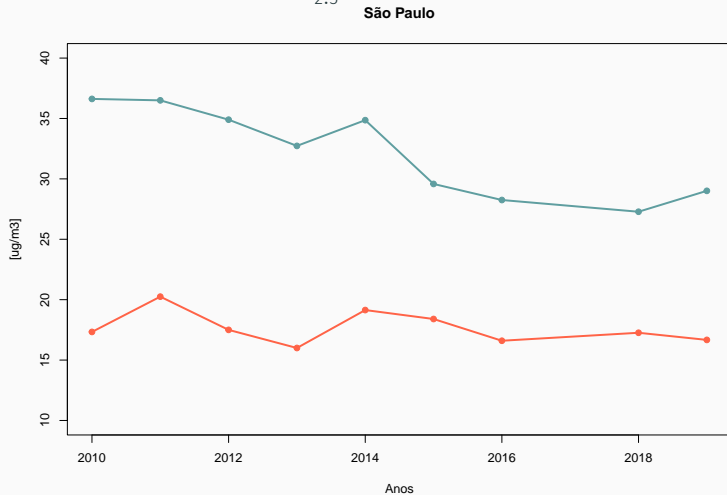
# Séries temporais

- Adicionemos também  $PM_{2.5}$

```
plot(who_sp$year, who_sp$pm10, # Dado eixo x, dado eixo y
     t = "l", lwd = 2.5, col = "cadetblue",
     ylim = c(10, 40), # para visualizar valores de PM2.5
     xlab = "Anos", main = "São Paulo",
     ylab = "[ug/m3]") # So unidades
points(who_sp$year, who_sp$pm10, col = "cadetblue", pch = 19 )
# Adicionamos PM.25
lines(who_sp$year, who_sp$pm25, col = "tomato", lwd = 2.5)
points(who_sp$year, who_sp$pm25, col = "tomato", pch = 19 )
```

# Séries temporais

- Adicionemos também  $PM_{2.5}$



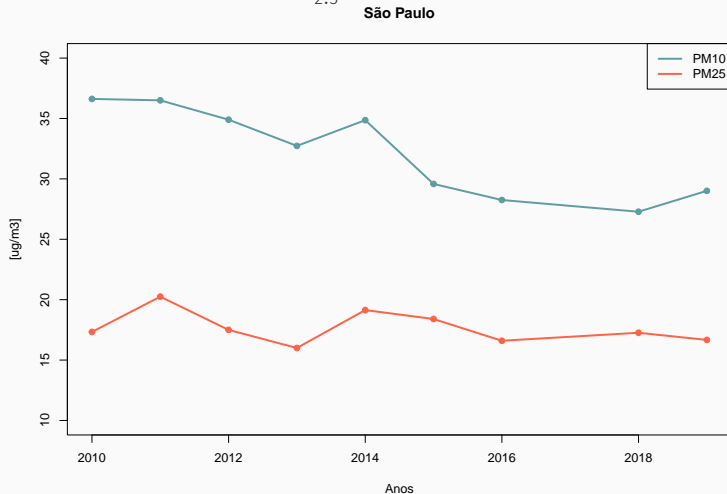
# Séries temporais

- Adicionemos também  $PM_{2.5}$

```
plot(who_sp$year, who_sp$pm10, t = "l", lwd = 2.5, col = "cadetblue",  
     ylim = c(10, 40), xlab = "Anos", main = "São Paulo", ylab = "[ug/m3]",  
     points(who_sp$year, who_sp$pm10, col = "cadetblue", pch = 19 )  
     lines(who_sp$year, who_sp$pm25, col = "tomato", lwd = 2.5)  
     points(who_sp$year, who_sp$pm25, col = "tomato", pch = 19 )  
     legend("topright", col = c("cadetblue", "tomato"), lwd = 2.5,  
           legend = c("PM10", "PM25"))
```

# Séries temporais

- Adicionemos também  $PM_{2.5}$

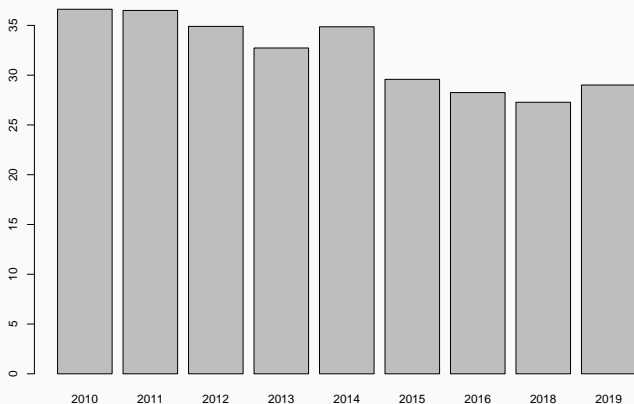




## Diagrama de barras.

- Usamos a função `barplot()`.

```
barplot(who_sp$pm10, names.arg = who_sp$year)
```



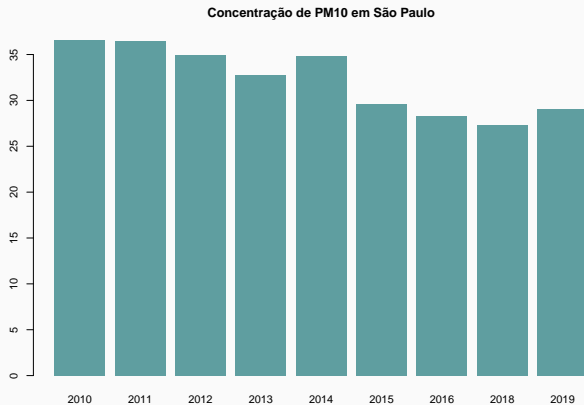
## Diagrama de barras.

- Usamos a função `barplot()`.

```
barplot(who_sp$pm10,  
        names.arg = who_sp$year, # Nome das barras  
        col = "cadetblue", # Color das barras  
        border = FALSE, # Sem borde das barras  
        main = "Concentração de PM10 em São Paulo")
```

# Diagrama de barras.

- Usamos a função `barplot()`.



## Diagrama de barras.

- Comparemos com  $PM_{2.5}$ .
- Primeiro temos que transformar nosso dato em matriz.

```
pm <- subset(who_sp, select = c("pm10", "pm25"))  
pm_m <- t(as.matrix(pm))
```

## Diagrama de barras.

- Comparemos com  $PM_{2.5}$ .
- Primeiro temos que transformar nosso dato em matriz.

```
barplot(pm_m, beside = TRUE, border = TRUE,  
        col = c("cadetblue", "tomato"),  
        names.arg = who_sp$year,  
        main = "Concentrações de PM em São Paulo")  
legend("topright", legend = c("PM10", "PM2.5"),  
       pch = 15, col = c("cadetblue", "tomato"))
```

# Diagrama de barras.

- Comparemos com  $PM_{2.5}$ .
- Primeiro temos que transformar nosso dato em matriz.

