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

Data frames, qualR, e plots

Mario Gavidia-Calderón, Rafaela Squizzato, Thiago Nogueira

18/02/2025

Universidade de São Paulo

Resumo aula passada

Exemplo usando dados CETESB

Pacotes

Do Excel para R

qualR

Plots

Resumo aula passada

Exemplo usando dados CETESB

Rede de monitoramento de qualidade do ar CETESB

- Uma das melhores da região (Riojas-Rodríguez et al., 2016)
- $>$ 60 estações **automáticas**.
- 22 estações manuais
- 2 estações movies
- O3, NO, NO2, CO, PM2.5, PM10, SO2
- Dados abertos!

Sistema QUALAR

- A plataforma permite **baixar e visualizar** a informação da rede de monitoramento.
- `https://qualar.cetesb.sp.gov.br/qualar/home.do`
- Exercicio 1: Criar conta

Sistema QUALAR

- Vamos baixar dados de **Novembro de 2023**.
  - Estação Pinheiros.
  - Parâmetros: O3, MP2.5, Velocidade do vento.
  - DEMO.

Leer arquivo

```r
pin <- read.table(
  "~/Downloads/1739840283882.csv",
  header = F, # Não tem nome das colunas
  sep = ";",  # O separador das colunas é ;
  dec = ",",  # O separador decimal é ,
  skip = 8).  # Vai ler o arquivo desde a linha 8
```

Leer arquivo

- Exploramos usando `head(pin)`

```
##           V1    V2 V3 V4  V5
## 1 01/11/2023 01:00 NA  1 1.0
## 2 01/11/2023 02:00 NA 12 1.1
## 3 01/11/2023 03:00 NA 14 1.0
## 4 01/11/2023 04:00 NA 22 1.3
## 5 01/11/2023 05:00 NA 18 1.2
## 6 01/11/2023 06:00 NA 21 1.5
```

Trocar nomes das colunas

```
names(pin) <- c("day", "hour", "pm25", "o3", "ws")
head(pin)
```

```
##          day  hour pm25 o3  ws
## 1 01/11/2023 01:00   NA  1 1.0
## 2 01/11/2023 02:00   NA 12 1.1
## 3 01/11/2023 03:00   NA 14 1.0
## 4 01/11/2023 04:00   NA 22 1.3
## 5 01/11/2023 05:00   NA 18 1.2
## 6 01/11/2023 06:00   NA 21 1.5
```

AED

- Vamos calcular a média, max, min, sd e mediana do **ozônio**.

```
mean(pin$o3, na.rm = T)
```

```
## [1] 46.70475
```

```
median(pin$o3, na.rm = T)
```

```
## [1] 39
```

AED

- Vamos calcular a média, max, min, sd e mediana do **ozônio**.

```
max(pin$o3, na.rm = T)
```

```
## [1] 183
```

```
min(pin$o3, na.rm = T)
```

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

Funções úteis

- print()

```
o3_mean <- mean(pin$o3, na.rm = T)
print(o3_mean)
```

```
## [1] 46.70475
```

Funções úteis

- paste()

```
pin_o3_mean <- mean(pin$o3, na.rm = T)
print(paste("A media horaria de O3:",  pin_o3_mean))
```

```
## [1] "A media horaria de O3: 46.7047477744807"
```

Funções úteis

- round()

```
print(paste("A media horaria de O3:",  round(pin_o3_mean)))
```

```
## [1] "A media horaria de O3: 47"
```

```
print(paste("A media horaria de O3:",  round(pin_o3_mean, 2)))
```

```
## [1] "A media horaria de O3: 46.7"
```

Funções úteis

- unique()

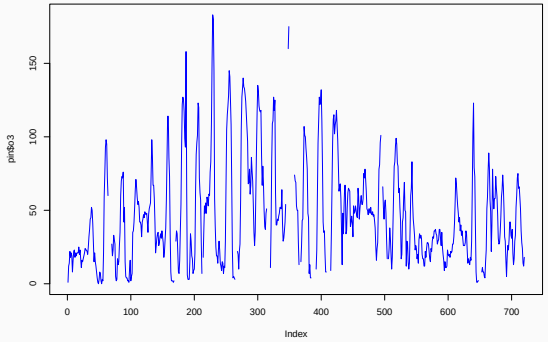
```
horas <- unique(pin$hour)
print(horas)
```

```
##  [1] "01:00" "02:00" "03:00" "04:00" "05:00" "06:00" "07:00" "08:0
## [10] "10:00" "11:00" "12:00" "13:00" "14:00" "15:00" "16:00" "17:0
## [19] "19:00" "20:00" "21:00" "22:00" "23:00" "24:00"
```

Plots simples

Série temporal

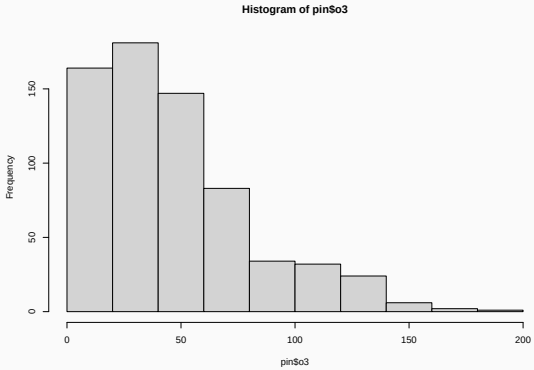
```
plot(pin$o3, t = "l", lwd = 1.2, col = "blue")
```



Plots simples

Histograma

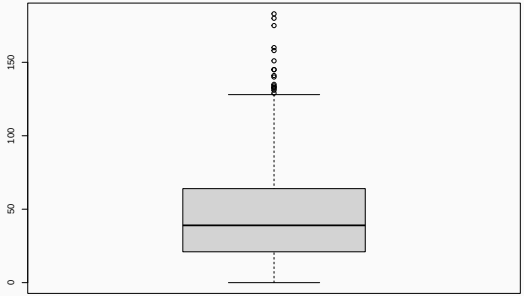
```
hist(pin$o3)
```



Plots simples

boxplot

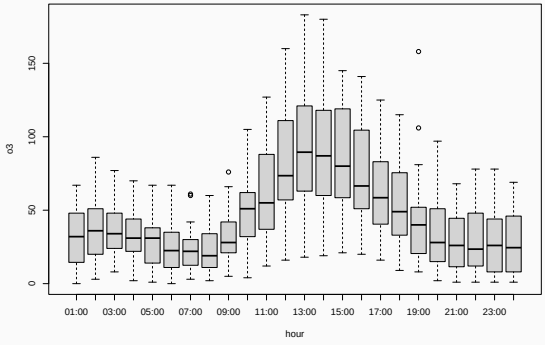
```
boxplot(pin$o3)
```



Plots simples

boxplot

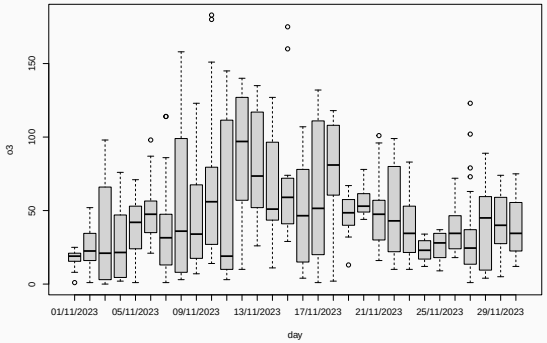
```
boxplot(o3~hour, data=pin)
```



Plots simples

boxplot

```
boxplot(o3~day, data=pin)
```



Filtrar dados

- Usamos o comando `subset(df, subset = <condição>)`

```r
# Horas com [o3] maior que a mediana
pin_o3_med <- subset(
  pin,
  subset = o3 >= mean(pin$o3, na.rm=T)
)
```

Filtrar dados

- Usamos o comando `subset(df, subset = <condição>)`

```
##           day  hour pm25 o3  ws
## 38 02/11/2023 14:00   NA 52 2.7
## 39 02/11/2023 15:00   NA 50 3.1
## 58 03/11/2023 10:00    6 59 3.5
## 59 03/11/2023 11:00    6 75 3.5
## 60 03/11/2023 12:00   11 93 3.9
## 61 03/11/2023 13:00   11 98 3.8
```

Data!

- É muito importante falar para R que temos dados tipo data (`POSIXct`).

```
pin$date <- paste(pin$day, pin$hour)
head(pin)
```

```
##          day  hour pm25 o3  ws             date
## 1 01/11/2023 01:00   NA  1 1.0 01/11/2023 01:00
## 2 01/11/2023 02:00   NA 12 1.1 01/11/2023 02:00
## 3 01/11/2023 03:00   NA 14 1.0 01/11/2023 03:00
## 4 01/11/2023 04:00   NA 22 1.3 01/11/2023 04:00
## 5 01/11/2023 05:00   NA 18 1.2 01/11/2023 05:00
## 6 01/11/2023 06:00   NA 21 1.5 01/11/2023 06:00
```

Data!

- É muito importante falar para R que temos dados tipo data (`POSIXct`).

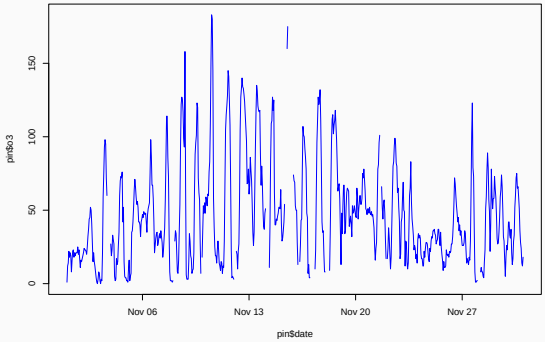
```
pin$date <- as.POSIXct(
  strptime(pin$date,
           format="%d/%m/%Y %H:%M")
)
```

- Informação sobre formato data

Data

Série temporal

```
plot(pin$date, pin$o3, t = "l", lwd = 1.2, col = "blue")
```



Pacotes

Pacotes

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

Bibliotecas

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

```
# Instalando Rmarkdown
install.packages("raster")
```

Bibliotecas

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

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

Bibliotecas

- Por exemplo, para usar a base de dados do Brasil, usamos o pacote `geobr`.

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

Do Excel para R

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`

qualR

Limitações do sistema QUALAR

Plataforma é limitada:

- Um poluente por vez (3 no modo avançado).
- Uma estação por vez.
- Baixar dados anuais fica devagar

Dados tem problemas.

- Trocar separador decimal.
- *Missing dates*: Datas faltantes
- *Wrong date format*: Arrumar o formato das datas

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 ter cadastro na plataforma QUALAR da CETESB.
- Referência qualR

Instalação qualR

- O jeito mais fácil é o seguinte:

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

Instalação qualR

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 parârmetros `cetesb_param` na coluna `code`.

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

```
##                             name units code
## 17    TEMP (Temperatura do Ar)    ºC   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 em `cetesb_aqs`) e o nome do parâmetro (igual 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 poluentes de uma estação

- Também podemos baixar vários poluentes de uma estação. Só precisamos definir os poluentes para baixar em um `vetor`. Vamos baixar O_3, $PM_{2.5}$, e NO_X.

```
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 precise usar outro software (e.g. PMF). Então você pode 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 mesmo poluente de várias estações

- As vezes você precisa comparar valores de várias estações. Neste exemplo vamos baixar valores de NO_X 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 mesmo poluente de várias estações

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

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

Um mesmo poluente de 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 mesmo poluente de várias estações

- Vamos comparar as estações

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

```
## [1] 48.12185
```

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

```
## [1] 21.54269
```

- 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 funçãoes que não precisam do argumento `parameters` pois foram desenvolvidas 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

Plots

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

Plots

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