

## Aula 3 - Lendo dados no R

Marcelo Prudente e Rafael Giacomini

13 de março de 2018

- 1 **Introdução**
- 2 **Importando Dados no R**
- 3 **Arquivos de texto: .csv e .txt**
- 4 **xls e xlsx**
- 5 **EXERCÍCIO 1**
- 6 **Exportando dados**

# Introdução

- Nesta aula, apresentaremos:
  - ▶ Como ler dados no **R**
  - ▶ Como exportar dados no R
  - ▶ Os fundamentos da manipulação de dados no **R**
    - ★ Foco na sintaxe
  - ▶ Como mesclar dados no R

## Importando Dados no R

- Duas funções importantes para a organização do trabalho são a **getwd()** e a **setwd()**.
  - ▶ *digite*: **getwd()** em seu console. O que você obtêm?
- Elas informam e definem as pastas onde os arquivos estão.

- É possível criar uma pasta para guardar seus dados e definí-la como a seu diretório de trabalho.

```
# Estabelecer diretorio de trabalho  
setwd("C:/curso_r_enap")
```

- Esse comando permite baixar arquivos com maior facilidade

- Em geral, as bases governamentais estão disponibilizadas em:
  - ▶ formatos de texto (*.csv* ou *.txt*) ou
  - ▶ em excel (*.xls* ou *.xlsx*).
- Há ainda outros formatos relacionados aos diversos programas estatísticos do mercado:
  - ▶ *.dta*, do STATA
  - ▶ *.sav*, do SPSS
  - ▶ *.sas7bdat*, do SAS
  - ▶ *.fwf*, formato fixo
  - ▶ entre outros



- Apesar dos diversos tipos de dados, a leitura deles no programa é bastante simples.
- Há inúmeros pacotes desenhados para essa função:
  - ▶ o pacote de base do sistema, com `read.csv()`
  - ▶ o pacote *tidyverse*, com o `read_csv()` e outros
  - ▶ o pacote *data.table*, com o `fread()`
  - ▶ o pacote *foreign*, com `read.dta()`, `read.spss()`, entre outros

- Na prática, a leitura de dados em todos os pacotes tem uma fórmula comum para a leitura dos dados:
  - 1 Determinar o local do arquivo (ex: "C:/minha\_pasta/meu\_arquivo")
  - 2 Determinar o separador do arquivo (";", ",")
  - 3 Determinar os separadores dos números decimais (".", ",")
  - 4 Determinar se a primeira linha contém o nome das variáveis (**header == TRUE** )

## Arquivos de texto: .csv e .txt

- Antes de executar os comandos, é importante verificar a pasta em que está o arquivo.
- Depois, fazer uma inspeção visual nos dados do arquivo *exemplo1.csv*.
  - ▶ identificar o separador do arquivo
  - ▶ identificar o separador dos decimais
- **Note:** o separador das colunas do arquivo *exemplo1.csv* é uma ,
  - ▶ Esse formato é muito comum nos dados americanos

```
# Base do sistema
ex1_base <- read.csv("C:/curso_r_enap/exemplo1.csv")

# Tidyverse - readr
ex1_tdv <- read_csv("C:/curso_r_enap/exemplo1.csv")

# Ou ainda
ex1_tdv <- read_csv(file.choose())
```

- Os comandos acima assumem que:
  - ▶ o **sep** = “,”
  - ▶ o **dec** = “.”
  - ▶ o **header** == **TRUE**

- Agora, vamos inspecionar o arquivo **exemplo2.csv**

```
# Base do sistema
ex2_base <- read.csv2("C:/curso_r_enap/dados/exemplo2.csv")

# Tidyverse
ex2_tdv <- read_csv2("C:/curso_r_enap/dados/exemplo2.csv")
```

- Os comandos acima assumem que:
  - ▶ o **sep** = ";"
  - ▶ o **dec** = ","
  - ▶ o **header** == **TRUE**

- o **R** fornece o **read.csv** como programa de base para ler arquivos de texto (*.txt*).
- Todavia, o **fread()** traz as mesmas funções com maior velocidade.

```
# Base
ex3_base <- read_csv("C:/curso_r_enap/dados/exemplo1a.txt")
# readr
ex3_tdv <- read_csv("C:/curso_r_enap/dados/exemplo1a.txt")
# fread
ex3_fread <- fread("C:/curso_r_enap/dados/exemplo1a.txt")
```

- A abordagem do readr permite importar dados no formato tbl\_df (tibble), cuja estrutura é mais moderna para leitura de dados.
- É mais rápido que a base do sistema e bastante intuitivo.

| função                    | o que processa   |
|---------------------------|--|
| <code>read_csv()</code>   | arquivos csv separados por ,                             |
| <code>read_tsv()</code>   | arquivos separados por tab                               |
| <code>read_delim()</code> | arquivos separados por delimitador geral (espaço ou tab) |
| <code>read_fwf()</code>   | arquivos com largura fixa                                |
| <code>read_table()</code> | arquivos separados por um espaço em branco               |



- As funções do readr são versáteis e produzem `tbl_df`, mas o `fread()`, do pacote **data.table** é mais rápido entre todas as soluções.
  - ▶ Melhor solução para arquivos com **1gb ou +**
- Além disso, o `fread()` deduz qual o separador do arquivo.
- Ainda, permite:
  - ▶ ler apenas algumas linhas (**nrow = 10**, por exemplo) do arquivo para uma rápida inspeção
  - ▶ também, ler algumas colunas desejadas (`select(c("col1", "col5"))`, por exemplo))
- No entanto, às vezes apresenta erros para a leitura dos decimais.

- Ao baixar dados no R no Windows, muitas vezes a leitura de dados de texto pode ser problemática.
  - ▶ especialmente com as soluções do **readr**

```
# leitura com problema
problema <- read_csv("C:/curso_r_enap/dados/exemplo1a.txt")
head(problema, 10)
```

- Observe que a leitura dos acentos é problemática, pois o programa é focado em dados produzidos nos EUA.

- Perceba que o problema de leitura ocorrerá com texto. Se o banco tem apenas números, não há problema.
- Esse problema da leitura de dados no windows pode ser resolvido com a especificação do encoding.

```
resolvido <- read_csv("C:/curso_r_enap/dados/exemplo1a.txt",  
                      locale = locale(encoding = "Latin1"))  
head(resolvido, 15)
```

## xls e xlsx

- Há duas formas práticas de ler dados do *Excel* no **R**.
- Vamos observar o arquivo **exemplo4.xlsx** e aplicar a leitura por comando.

```
library(readxl)
# Lendo os dados do excel
ex4  <- read_excel("C:/curso_r_enap/dados/exemplo4.xlsx")

# Importar a segunda planilha
ex4_s2  <- read_excel("C:/curso_r_enap/dados/exemplo4.xlsx", s
```

- Há também a possibilidade de ler os dados com o cursor.

- É possível ler outros arquivos, como os de SPSS, SAS ou STATA. Também, dados *.json*, *.xml*, entre outros.
- Todos os comandos acima podem ser executados facilmente com o uso do mouse.
- Siga as instruções:
  - ▶ *File*
  - ▶ *Import Dataset*
  - ▶ *From ...*
- Apesar disso, recomendamos aprender os comandos. É mais rápido!!

# EXERCÍCIO 1

- Baixe os dados da planilha **exemplo2.csv**. Depois verifique:
  - ▶ Qual a classe do banco de dados `exemplo1_base`? E do `exemplo1_tdv`?
  - ▶ Quais as classes das variáveis do banco?
  - ▶ Como visualizar esses dados?
  - ▶ Quais os nomes das variáveis do banco? Crie um vetor com esses nomes.
  - ▶ Como é possível extrair a média da expectativa de vida?
  - ▶ Qual o máximo e o mínimo dessa variável?
    - ★ Crie um `data.frame` com cada um desses valores.



## Exportando dados

- É possível utilizar as funções **write** para exportar dados do **R** para a sua pasta.
- Como os dados de texto podem ser lidos em qualquer programa (EXCEL, SPSS, SAS, STATA, entre outros), vamos aprender a exportar arquivos nessa forma.

```
# Você pode determinar a pasta na qual o arquivo será salvo
setwd("C:/curso_r_enap/meus_dados/")

# Exportando arquivos de texto
write.table(exemplo1_base, "meu_exemplo.csv")
```

- Essa função permite determinar:

```
# Separador
write.table(exemplo1_base, "meu_exemplo2.csv",
            sep = ";")

# Decimal
write.table(exemplo1_base, "meu_exemplo2.csv",
            sep = ";", dec = ",")

# Omitir o número das linhas
write.table(exemplo1_base, "meu_exemplo2.csv",
            sep = ";", dec = ",", row.names = FALSE)

# mais?
?write.table; help(write.table)
```