

Trabalhando com arquivos

Vitor Rios

11 de novembro de 2017

Salvando sua vida seu código

Se você fechar o R, vai perder tudo que você fez

Solução: salve seu código em um script!

Script é um arquivo de texto sem formatação, com a extensão `.R` ou `.r`, ao invés de `.txt`. Pense no script como a receita, o R como a cozinha, e a análise o jantar.

Você pode editar no bloco de notas, Notepad++, Gedit, Vim, EMACS, etc, mas evite usar Word, ele insere formatações que atrapalham tudo

Use o RStudio!

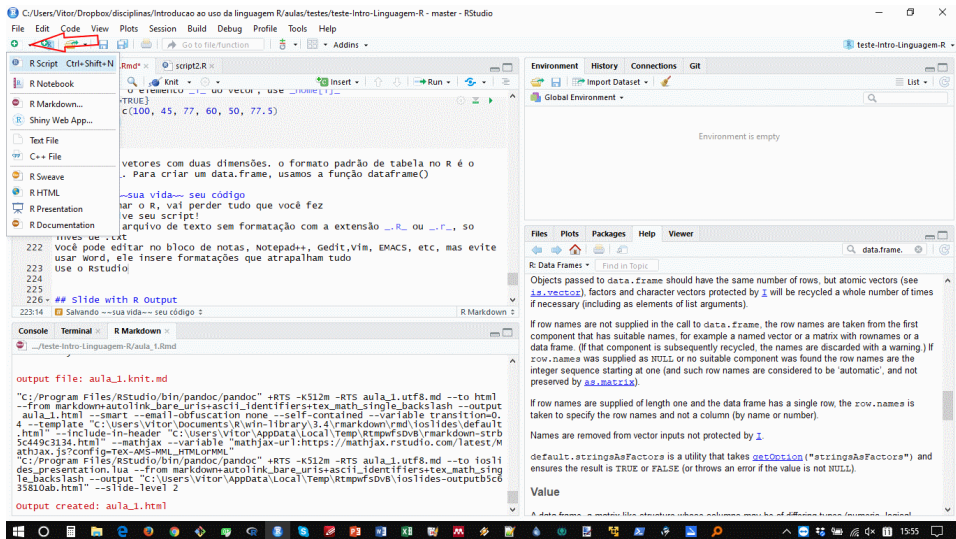


Figure 1:

Mas salvar onde?

O R usa uma coisa chamada diretório de trabalho. Ele encontra automaticamente tudo que esteja nesse diretório e salva as coisas nesse diretório. É recomendado que cada projeto seu no R viva no seu próprio diretório, e os scripts mudem o diretório de trabalho para o diretório da análise logo no começo

```
getwd() ##exibe o diretório de trabalhoda sessão atual
```

```
## [1] "C:/Users/Vitor/Dropbox/disciplinas/Introducao ao uso da linguagem R"
```

Para mudar o diretório de trabalho, use a função `setwd()`

```
setwd("C:/Curso_R/aula1") ##muda o diretório de trabalho da sessão atual
```

ATENÇÃO

O endereço deve vir sempre em aspas, e use `/` ao invés de `\` no Windows

Cada coisa, um script!

Separe suas funções e análises em scripts diferentes, vai facilitar *muito* sua vida.

Use nomes curtos e descritivos. Funcao1.R, funcao2.R, regressao.R, regressaoLogisticaMultivaridaAmostra1BTSversão1.ré pedir pra sofrer.

Uma pasta para cada coisa:

- ~/doutorado/
- /dados
- /scripts - /outputs - /figuras
- /manuscrito

Você pode chamar um script de dentro do outro, usando o comando `source(endereço/script.R)`. O endereço pode inclusive ser um site.

ATENÇÃO

`source(script.R)` executa todos os comandos que estão dentro de `script.R`, cuidado

Crie um script com os codigos que usamos até aqui, comentando cada linha com um `#` após o comando, dizendo o que ele faz

```
source("https://raw.githubusercontent.com/vrios/Intro-Linguagem-R/master/a
```

dica: isso também funciona com as funções `read.csv()` e `read.table()`

```
caixeta = read.csv("https://raw.githubusercontent.com/vrios/Intro-Linguagem-R/master/a  
head(caixeta)
```

##	local	parcela	arvore	fuste	cap	h	especie
## 1	chauas	1	1	1	210	80	Myrcia sulfiflora
## 2	chauas	1	3	1	170	80	Myrcia sulfiflora
## 3	chauas	1	4	1	720	70	Syagrus romanzoffianus
## 4	chauas	1	5	1	200	80	Tabebuia cassinoides
## 5	chauas	1	6	1	750	170	indet.1
## 6	chauas	1	7	1	320	80	Myrcia sulfiflora

Dica de estruturar seu script:

Comece sempre com um cabeçalho, dizendo para que serve o script, e se ele faz parte de algum projeto

Carregue todos os pacotes necessários, com `library()` no começo do script. Se quiser, use comentários para explicar porque o pacote é necessário

Liste cada passo do script em português

verbo - objeto

- ▶ Ler dados do arquivo
- ▶ Verificar dados ausentes / errados
 - ▶ verificar NAs
 - ▶ verificar erros de digitação
 - ▶ verificar cabeçalhos
- ▶ Selecionar colunas para analisar
- ▶ Rodar análise
 - ▶ Passo 1
 - ▶ Passo 2 _

Pseudocódigo

Substitua a ação pela função adequada do R

Se necessário, quebre em sub itens

- ▶ Ler dados do arquivo
 - ▶ `file = read.table("preçosComida.csv")`
- ▶ Verificar dados ausentes / errados
 - ▶ verificar NAs
 - ▶ `is.na(file)`
 - ▶ verificar erros de digitação
 - ▶ `unique(file$species)`
 - ▶ verificar cabeçalhos
 - ▶ `head(file)`
- ▶ Selecionar colunas para analisar
 - ▶ `file$species[file$species!="Chuchu"]`
- ▶ Rodar análise
 - ▶ Passo 1
 - ▶ `lm(file$price ~ file$species)`

Salvando tudo junto

Se você quiser salvar tudo que está na memória do R, inclusive objetos use a função `save.image()` com a extensão `.RData`

```
save.image(file="tudoJunto.RData")
```

Isto é útil quando temos uma análise demorada, que gera objetos grandes, e queremos acessar o resultado dela rapidamente no futuro. Podemos também salvar objetos específicos com a função `save()`:

```
save(file="resultadosSalvos.RData",list = list(resultado1, resultado2, resu
```

Em geral, é melhor salvar os scripts do que o workspace ou os objetos

Lendo arquivos do R

Abrir um arquivo, carregar um arquivo e executar um arquivo são coisas diferentes.

Abrir -> ver os conteúdos

- pode ser no editor do R, RStudio, ou externo
- não executa nem carrega nada

Executar -> rodar os comandos dentro do arquivo

- função `source()`

Carregar -> colocar o conteúdo do arquivo para dentro de um objeto

- funções `load()`, `read.table()`, `read.csv()` e etc

Função `list.files()`

Lista todos os arquivos no diretório de trabalho. Pode ser usada para selecionar um grupo de arquivos que serão lidos sequencialmente, por exemplo, com o argumento `pattern` para selecionar arquivos que contenham uma determinada sequência no nome

Atenção: apenas lista, não carrega nem abre os arquivos

```
list.files()
```

```
## [1] "arquivos"          "aula_10.Rmd"       "aula_1.html"
## [4] "aula_1.pdf"        "aula_1.R"          "aula_1.Rmd"
## [7] "aula_2.html"       "aula_2.pdf"        "aula_2.R"
## [10] "aula_2.Rmd"        "aula_3.html"       "aula_3.pdf"
## [13] "aula_3.R"          "aula_3.Rmd"        "aula_4.html"
## [16] "aula_4.Rmd"        "aula_5.Rmd"        "aula_6.Rmd"
## [19] "aula_7.Rmd"        "aula_8.Rmd"        "aula_9.Rmd"
## [22] "figuras"           "LICENSE"            "README.md"
## [25] "script.R"          "teste-aulas.Rproj"
```

```
aulas = list.files(pattern = "aula") # arquivos que possuam "aula" em qual  
aulas
```

```
## [1] "aula_10.Rmd"      "aula_1.html"      "aula_1.pdf"  
## [4] "aula_1.R"         "aula_1.Rmd"       "aula_2.html"  
## [7] "aula_2.pdf"       "aula_2.R"         "aula_2.Rmd"  
## [10] "aula_3.html"      "aula_3.pdf"       "aula_3.R"  
## [13] "aula_3.Rmd"       "aula_4.html"      "aula_4.Rmd"  
## [16] "aula_5.Rmd"       "aula_6.Rmd"       "aula_7.Rmd"  
## [19] "aula_8.Rmd"       "aula_9.Rmd"       "teste-aulas.Rproj"
```

```
scripts = list.files(pattern = "\\\\.R$") # "\\\\.R$" significa arquivos terminando em .R  
scripts # objeto contendo os nomes dos arquivos, não os conteúdos
```

```
## [1] "aula_1.R" "aula_2.R" "aula_3.R" "script.R"
```

Executar arquivos

Usamos a função `source()` para executar o conteúdo de um arquivo de código. Especialmente útil para carregar funções específicas ou scripts de análise.

```
ls()
```

```
## [1] "aulas"      "caixeta" "scripts"
```

```
source("arquivos/toroidal.distance.R")
```

```
ls()
```

```
## [1] "aulas"                "caixeta"                "scripts"
```

```
## [4] "toroidal.distance"    "toroidal.distances"
```

```
toroidal.distances
```

```
## function (object, tam)
## {
##     toroid.dist.matrix = matrix(NA, nrow = length(object[, 1]),
##     ncol = length(object[, 1]))
##     for (i in 1:length(object[, 1])) {
##         for (j in 1:length(object[, 1])) {
##             x1 = object[i, 1]
##             x2 = object[j, 1]
##             y1 = object[i, 2]
##             y2 = object[j, 2]
##             dMin = sqrt(min(abs(x1 - x2), tam - abs(x1 - x2)) *
##             min(abs(x1 - x2), tam - abs(x1 - x2)) + min(abs(y1 -
##             y2), tam - abs(y1 - y2)) * min(abs(y1 - y2),
##             tam - abs(y1 - y2)))
##             toroid.dist.matrix[i, j] = dMin
##         }
##     }
## }
```

Carregando arquivos

Usamos a função `load()` para carregar arquivos criados com as funções `save()`, `save.image()`

```
load(file="resultadosSalvos.RData")  
ls()
```

Lendo arquivos quaisquer

O R consegue ler arquivos externos, se você disser a ele o que esperar

```
help("read.table")
```

`read.table()` é especializada em ler tabelas em formato texto

Salvar os dados em formato texto facilita leitura, backup e compatibilidade com versões anteriores e posteriores de programas (.xls / .xlsx)

- arquivos de texto são entendidos automaticamente por software de versionamento (aula 9)

Existem algumas funções com opções padrão que facilitam o uso da `read.table()`

`read.csv()` lê arquivos usando como padrão o ponto (.) como sepador de decimais e a vírgula (,) como separador de colunas.

`read.csv2()` usa a vírgula como separador de decimais e o ponto-e-vírgula (;) como separador de colunas

Argumentos importantes

```
#argumentos são sempre separados por vírgulas na chamada da função
#no começo, é bom colocar cada argumento em uma linha para facilitar o entendimento
#TRUE habilita a opção, FALSE desabilita
arquivoLido =read.csv(file = "/arquivos/caixeta.csv",
                      header = TRUE, #interpreta a primeira linha como sendo o cabeçalho
                      as.is = TRUE , # não altera interpretação do tipo de dado em função dos pontos decimais
                      sep = ",", # define o caractere que deve ser o separador de colunas
                      dec = ".", #define o caractere que deve ser o separador de casas decimais
                      )
```

O comando acima cria um objeto chamado `arquivoLido`, com o conteúdo de "arquivo.csv"

O arquivo original não é alterado, não importa o que você fizer dentro do R

Se você quiser salvar suas alterações em um arquivo de texto, use `write.table()`, `write.csv()` ou `write.csv2()`

Importantíssimo

Nunca jamais use o Word ou Docs para editar arquivos de texto puro

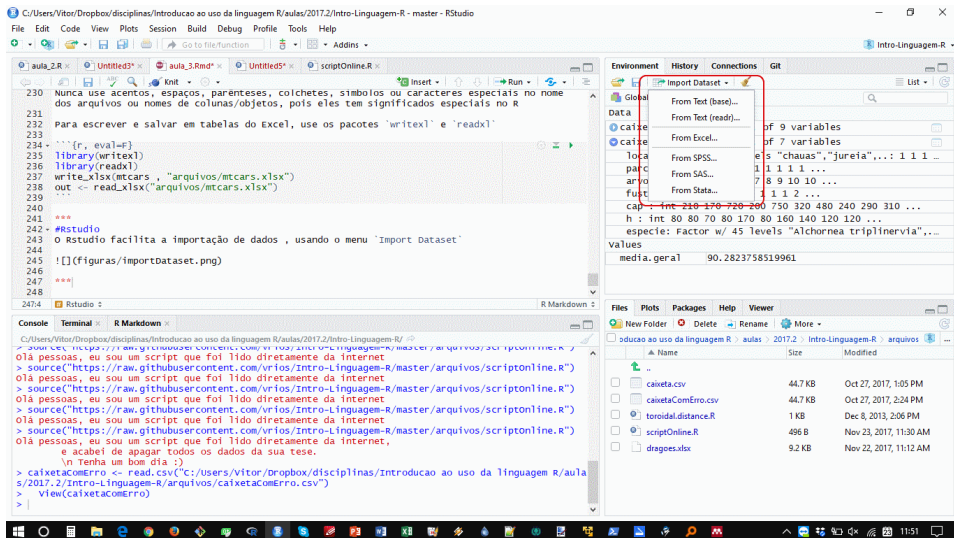
Nunca use acentos, espaços, parênteses, colchetes, símbolos ou caracteres especiais no nome dos arquivos ou nomes de colunas/objetos, pois eles tem significados especiais no R

Para escrever e salvar em tabelas do Excel, use os pacotes `writexl` e `readxl`

```
library(writexl)
library(readxl)
write_xlsx(mtcars , "arquivos/mtcars.xlsx")
out <- read_xlsx("arquivos/mtcars.xlsx")
```

Rstudio

O Rstudio facilita a importação de dados , usando o menu Import Dataset



The screenshot displays the RStudio environment with the 'Import Dataset' menu open. The menu options are: From Text (base)..., From Text (readr)..., From Excel..., From SPSS..., From SAS..., and From Stata... The 'From Text (base)...' option is highlighted. The background shows an R script with comments in Portuguese and R code for writing and reading Excel files. The Environment pane on the right shows a data frame with columns 'media.geral' and '90.2823758519961'. The Files pane at the bottom shows a directory structure with files like 'caixeta.csv', 'caixetaComErro.csv', 'toroidal.distance.R', 'scriptOnline.R', and 'dragoes.xlsx'.

```
230 Nunca use acentos, espaços, parênteses, colchetes, símbolos ou caracteres especiais no nome
231 dos arquivos ou nomes de colunas/objetos, pois eles tem significados especiais no R
232
233 Para escrever e salvar em tabelas do Excel, use os pacotes 'writexl' e 'readxl'
234
235 ```{r, eval=F}
236 library(writexl)
237 library(readxl)
238 write_xlsx(mtcars, "arquivos/mtcars.xlsx")
239 out <- read_xlsx("arquivos/mtcars.xlsx")
240
241 ***
242 #Rstudio
243 O Rstudio facilita a importação de dados , usando o menu 'Import Dataset'
244
245 
246
247 ***
248
```

Environment

Object	Class	Attributes
Global	Environment	
Data	Environment	
caixe	data.frame	of 9 variables
caixe	data.frame	of 7 variables
loca	data.frame	es "chauas", "jureia",...: 1 1 1 ...
parC	data.frame	1 1 1 1 1 ...
arvo	data.frame	7 8 9 10 10 ...
fust	data.frame	1 1 1 2 ...
cap	data.frame	int 210 170 720 200 750 320 480 240 290 310 ...
h	data.frame	int 80 80 70 80 170 80 160 140 120 120 ...
especie	Factor	w/ 45 levels "Alchornea triplinervia",...
Values		
media.geral		90.2823758519961

Files

Name	Size	Modified
..		
caixeta.csv	44.7 KB	Oct 27, 2017, 1:05 PM
caixetaComErro.csv	44.7 KB	Oct 27, 2017, 2:24 PM
toroidal.distance.R	1 KB	Dec 8, 2013, 2:06 PM
scriptOnline.R	496 B	Nov 23, 2017, 11:30 AM
dragoes.xlsx	9.2 KB	Nov 22, 2017, 11:12 AM

