# Leitura, Manipulação e Armazenamento de Dados com o R

Juliana Freitas de Mello e Silva  Luciane Ferreira Alcoforado

Departamento de Estatística, Universidade Federal Fluminense

# Sumário

**1 Leitura** **4**

1.1 Arquivo tipo .txt . . . . . 4

1.2 Arquivo tipo .csv . . . . . 5

1.3 Arquivos tipo .xls . . . . . 6

**2 Manipulação** **7**

2.1 Comandos nrow, ncol e dim . . . . . 7

2.2 Comando names . . . . . 8

2.3 Comando factor . . . . . 9

2.4 Comando head . . . . . 13

2.5 Comandos attach, detach e $ . . . . . 14

2.6 Comando length . . . . . 15

2.7 Comando rm . . . . . 16

2.8 Comando table . . . . . 16

2.9 Comando which . . . . . 17

2.10 Comando subset . . . . . 18

2.11 Comando summary . . . . . 19

2.12 Comando substr . . . . . 19

**3 Armazenamento** **20**

3.1 Comando write.table . . . . . 20

3.2 Comando write.csv e write.csv2 . . . . . 21

3.3 Comando write.xls . . . . . 21

# 1 Leitura

Para ler algum arquivo no R, deve-se primeiramente saber a sua extensão. As mais utilizadas são: .txt, .csv e .xls.

O arquivo com a extensão .txt é proveniente de um bloco de notas, e os demais de uma planilha do excel.

## 1.1 Arquivo tipo .txt

Para ler esse tipo de arquivo, usa-se o comando read.table. Ele pode ser descrito da seguinte forma:

read.table(file, header, sep, dec)

- file

  É o nome do arquivo entre aspas e, com o seu diretório incluso (cada pasta do diretório deve estar sepadado por duas barras, \\).

  Por exemplo:
  `file = "E:\\UFF\\ ... \\Dados\\dados.txt"`

- header

  É um argumento que assume valor lógico. Portanto, pode ser igual a "TRUE"ou "FALSE". Header em inglês significa cabeçalho, e é exatamente isso que se deseja saber: se há (TRUE) ou não (FALSE) um cabeçalho em seus dados.

  Por exemplo: header = FALSE

- sep

  Indica o que separa as colunas, que pode ser espaço, tab, barras.

  Exemplo: sep = ' '. Significa que as colunas do banco de dados estão separadas por espaços.

- dec

  Dec informa como a casa decimal será indicada.

  Exemplo: dec = "."ou dec = ",".

Para ler o documento de texto (.txt), deve-se juntar as informações acima, da seguinte forma:

```
> # Leitura de um arquivo .txt
> dados<- read.table(file =
+ "E:\\UFF\\2012.1\\Monitoria\\Projeto\\Apostila\\Dados\\dados.txt",
+ header = TRUE, sep = " ", dec = ",")
```

Observação: o arquivo do exemplo acima se encontra disponível em `http://www.datafilehost.com/download-562f1a39.html`.

## 1.2 Arquivo tipo .csv

"Csv"significa *comma separated values* ou valores separados por vírgula. Se o arquivo possui essa extensão, ele deve ser lido pelo comando read.csv ou read.csv2. A diferença entre esses dois está em como as casas decimais estão indicadas. Se a vírgula foi utilizada, deve-se usar o comando read.csv2, caso contrário, read.csv.

Esse comando tem os mesmos argumentos que o read.table. Segue um exemplo:

```
> # Leitura de um arquivo .csv
> dados<- read.csv2(file =
+ "E:\\UFF\\2012.1\\Monitoria\\Projeto\\Apostila\\Dados\\dados.csv",
+ header = TRUE, sep = ",", dec = ".")
```

Observação: o arquivo utilizado nesse exemplo está disponível em: `http://www.datafilehost.com/download-b31de167.html`

## 1.3 Arquivos tipo .xls

Para ler esse tipo de arquivo é necessário ter o pacote xlsReadWrite. Nesse pacote há o comando read.xls, que é usado da seguinte forma:

read.xls(file, colNames, sheet, from)

- file

  Como foi mencionado acima, é o nome do arquivo.

  Exemplo: `file = "E:\\UFF\\ ... \\Dados\\dados.xls"`

- colNames

  Tem a mesma utilidade de header. Se a primeira linha representar o nome das colunas, o valor lógico que deverá ser utilizado é o TRUE, caso contrário, FALSE.

  Exemplo: colnames = TRUE

- sheet

  É o número da planilha com a qual se deseja trabalhar.

  Exemplo: sheet = 1

- from

  Indica a linha pela qual se deseja o começo da leitura dos dados.

  Exemplo: from = 1

```
> # Carrega o pacote xlsReadWrite
> library(xlsReadWrite)

xlsReadWrite version 1.5.4 (826aa0)
Copyright (C) 2010 Hans-Peter Suter, Treetron, Switzerland.

This package can be freely distributed and used for any
```

```
purpose. It comes with ABSOLUTELY NO GUARANTEE at all.
xlsReadWrite has been written in Pascal and contains binary
code from a proprietary library. Our own code is free (GPL-2).

Updates, issue tracker and more info at http://www.swissr.org.
```

```
> # Lê o arquivo dados.xls
> dados<- read.xls(file =
+ "E:\\UFF\\2012.1\\Monitoria\\Projeto\\Apostila\\Dados\\dados.xls",
+ colNames = TRUE, sheet = 1)
```

Observação 1: o link do arquivo lido nesse exemplo se encontra disponível em `http://www.datafilehost.com/download-27f84a06.html`

Observação 2: é possível editar os dados a qualquer momento, através do comando edit(nome do objeto). Uma janela abrirá e assim o usuário pode fazer as modificações de interesse.

# 2 Manipulação

Nas seções abaixo serão apresentados alguns comandos que podem ser realizados após carregados os dados.

Os exemplos se referem ao arquivo dados.xls. O link de acesso para esse arquivo se encontra acima, na seção 1.3 (assim como o passo a passo para a leitura).

## 2.1 Comandos nrow, ncol e dim

Esses comandos indicam os números de linhas e colunas existentes. ”Row”significa linha, e ”col”representa column, que significa coluna em inglês. Ou seja, nrow é o número de linhas

e ncol, o de colunas.

```
> # Número de linhas de "dados"
> nrow(dados)

[1] 29

> # Número de colunas de "dados"
> ncol(dados)

[1] 13
```

Por sua vez, dim se refere à dimensão dos dados, mostrando as linhas e as colunas (nesta ordem).

```
> # Dimensão de "dados" (número de linhas e colunas)
> dim(dados)

[1] 29 13
```

## 2.2 Comando names

Esse comando serve para nomear ou renomear as colunas de um data frame. Seus argumentos são os mesmos que os da criação de um vetor, como mostrado no exemplo abaixo.

```
> names(dados)<- c("Numero", "A1", "A2", "A3", "A4", "A5", "A6", "A7", "A8",
+ "Idade", "Escolaridade", "Profissao", "InteresseFormacao")
```

## 2.3 Comando factor

O comando factor pode ser utilizado para renomear as observações de uma variável. Por exemplo, na variável "Escolaridade"há duas respostas diferentes: "medio completo"e "superior incompleto". Essas informações podem ser reduzidas para abreviações ou siglas, com a intenção de facilitar a leitura e/ou diminuir a quantidade de informações na tela.

Esse comando é composto pelos seguintes argumentos:

factor(x, levels, labels)

- x é o vetor de interesse;
- em levels, que significa "níveis"ou "categorias", deve-se colocar todas as diferentes observações da variável;
- em labels, que significa "rótulo"ou "legenda", deve-se colocar (como o nome já diz") a legenda. Ou seja a modificação que se deseja nas observações.

Por exemplo:

```
> # Como foi dito acima, a variável "Escolaridade" possui respostas muito
> # extensas, ocasinando uma "poluição visual".
> dados$Escolaridade

 [1] medio completo      superior incompleto -
 [4] medio completo      superior incompleto superior incompleto
 [7] superior incompleto medio completo      superior incompleto
[10] medio completo      medio completo      superior incompleto
[13] superior incompleto superior incompleto medio completo
[16] medio completo      medio completo      superior incompleto
[19] superior incompleto superior incompleto superior incompleto
[22] medio completo      medio completo      medio completo
```

```
[25] medio completo      superior incompleto superior incompleto
[28] superior incompleto superior incompleto
Levels: - medio completo superior incompleto

> # Para rezudir essas observações, duas siglas serão criadas: "mc",
> # representando "medio completo" e "si", representando "superior incompleto".
> dados$Escolaridade<- factor(dados$Escolaridade, levels = c("medio completo",
+ "superior incompleto", "-"), labels = c("mc", "si", "-"))
> dados$Escolaridade

 [1] mc si -  mc si si si mc si mc mc si si si mc mc mc si si si si mc mc mc mc
[26] si si si si
Levels: mc si -

> # Na variável Profissão, as observações "estudante" e "funcionário publico"
> # são extensas. Segue o comando para reduzi-las:
> dados$Profissao

 [1] estudante           estudante           -
 [4] estudante           -                   estudante
 [7] -                   estudante           funcionario publico
[10] estudante           estudante           estudante
[13] -                   -                   estudante
[16] estudante           nenhuma             -
[19] estudante           -                   estudante
[22] estudante           -                   estudante
[25] nenhuma             estudante           estudante
[28] estudante           estudante
Levels: - estudante funcionario publico nenhuma
```

```
> dados$Profissao<- factor(dados$Profissao, levels = c("estudante", "-",
+ "funcionario publico", "nenhuma"), labels = c("est", "-", "func pub",
+ "nenhuma"))
> dados$Profissao

 [1] est      est      -        est      -        est      -        est
 [9] func pub est      est      est      -        -        est      est
[17] nenhuma  -        est      -        est      est      -        est
[25] nenhuma  est      est      est      est
Levels: est - func pub nenhuma

> # Por último, na variável InteresseFormacao também há observações grandes,
> # como "ciencias exatas e da terra" e "engenharias".
> dados$InteresseFormacao

 [1] ciencias exatas e da terra
 [2] ciencias exatas e da terra
 [3] ciencias exatas e da terra
 [4] ciencias exatas e da terra
 [5] ciencias exatas e da terra
 [6] ciencias exatas e da terra
 [7] ciencias exatas e da terra
 [8] ciencias exatas e da terra
 [9] ciencias exatas e da terra
[10] ciencias exatas e da terra e engenharias
[11] ciencias exatas e da terra
[12] ciencias exatas e da terra
[13] ciencias exatas e da terra
[14] engenharias
[15] outros
```

```
[16] ciencias exatas e da terra
[17] ciencias exatas e da terra
[18] ciencias exatas e da terra
[19] ciencias exatas e da terra
[20] ciencias exatas e da terra
[21] ciencias exatas e da terra
[22] ciencias exatas e da terra
[23] ciencias exatas e da terra
[24] ciencias exatas e da terra
[25] engenharias
[26] ciencias exatas e da terra
[27] outros
[28] ciencias exatas e da terra
[29] ciencias exatas e da terra
4 Levels: ciencias exatas e da terra ...
```

```
> dados$InteresseFormacao<- factor(dados$InteresseFormacao,levels =
+ c("ciencias exatas e da terra", "ciencias exatas e da terra e engenharias",
+   "outros", "engenharias"), labels = c("cet", "cet e eng", "outros", "eng"))
> dados$InteresseFormacao
```

```
 [1] cet       cet       cet       cet       cet       cet       cet
 [8] cet       cet       cet e eng cet       cet       cet       eng
[15] outros    cet       cet       cet       cet       cet       cet
[22] cet       cet       cet       eng       cet       outros    cet
[29] cet
Levels: cet cet e eng outros eng
```

## 2.4 Comando head

Com este comando é possível ver as primeiras observações dos dados. Pode-se definir quantas linhas se deseja ver, por exemplo:

```
> # Sem especificações, esse comando mostra as seis primeiras linhas
> head(dados)
```

```
  Numero A1 A2 A3 A4 A5 A6 A7 A8 Idade Escolaridade Profissao InteresseFormacao
1      1  9  5  3  4  1  3  1  1    17           mc       est               cet
2      2  7  3  2  0  4  7  0  0    18           si       est               cet
3      3  5  7  2  5  7  5  5  2    20            -         -               cet
4      4 10  0  7  1  6  4  1  2    18           mc       est               cet
5      5  5  3  7  3  6  2  2  2    19           si         -               cet
6      6  4  7  6  2  8  9  2  3    19           si       est               cet
```

```
> # Mostra as dez primeiras linhas
> head(dados, 10)
```

```
   Numero A1 A2 A3 A4 A5 A6 A7 A8 Idade Escolaridade Profissao
1       1  9  5  3  4  1  3  1  1    17           mc       est
2       2  7  3  2  0  4  7  0  0    18           si       est
3       3  5  7  2  5  7  5  5  2    20            -         -
4       4 10  0  7  1  6  4  1  2    18           mc       est
5       5  5  3  7  3  6  2  2  2    19           si         -
6       6  4  7  6  2  8  9  2  3    19           si       est
7       7  8  8  6  6  0 10  5  3    19           si         -
8       8  9  6  0  8  0  9  5  5    19           mc       est
9       9  9  2  7  5  1  8  0  0    27           si  func pub
10     10 10  7  0  6  8 10  5  2    20           mc       est
```

```
   InteresseFormacao
1                cet
2                cet
3                cet
4                cet
5                cet
6                cet
7                cet
8                cet
9                cet
10         cet e eng
```

```
> # As três primeiras linhas
> head(dados, 3)
```

```
  Numero A1 A2 A3 A4 A5 A6 A7 A8 Idade Escolaridade Profissao InteresseFormacao
1      1  9  5  3  4  1  3  1  1    17           mc       est               cet
2      2  7  3  2  0  4  7  0  0    18           si       est               cet
3      3  5  7  2  5  7  5  5  2    20            -         -               cet
```

## 2.5 Comandos attach, detach e $

Há duas maneiras de selecionar uma variável de um data frame. Uma delas é usando o símbolo $, e a outra é pelo comando attach.

O attach faz com que seja possível acessar a variável apenas pelo seu nome, bastando apenas digitá-lo. Esse modo tem uma desvantagem: caso haja alguma variável que possua o mesmo nome que alguma variável do data frame, ela será substituída.

Para que isso não ocorra, pode-se utilizar o $, da seguinte forma: nome do data frame$variável.

Por sua vez, o detach desfaz o que o attach faz. Por exemplo:

```
> # Para acessar a variável "Escolaridade"
> dados$Escolaridade

 [1] mc si -  mc si si si mc si mc mc si si si mc mc mc si si si si mc mc mc mc
[26] si si si si
Levels: mc si -

> # Ou ainda
> attach(dados)
> Escolaridade

 [1] mc si -  mc si si si mc si mc mc si si si mc mc mc si si si si mc mc mc mc
[26] si si si si
Levels: mc si -

> detach(dados)
```

Observação: note que não é mais possível acessar nenhuma variável desse banco de dados sem o $.

## 2.6 Comando length

Length significa comprimento. Esse comando é muito útil, ele retorna o tamanho (ou comprimento) de um vetor. Por exemplo:

```
> # Para saber quantas observações há no variável Idade
> length(dados$Idade)

[1] 29

> # Há 29 observações
```

## 2.7 Comando rm

Este comando é usado quando se deseja remover um objeto qualquer, que pode ser desde um simples vetor até um conjunto de dados.

Para exemplificar esse comando, será criado um vetor de dados (pois ele sera removido).

```
> # Criando o vetor "objeto"
> objeto<- c(3, 4, 1, 23, 3, 43, 5, 6, 42, 100, 0, 38)
> objeto

 [1]   3   4   1  23   3  43   5   6  42 100   0  38

> # Removendo-o
> rm(objeto)
> # Observe que não existe mais o "objeto".
```

## 2.8 Comando table

Através desse comando é possível construir uma tabela de freqüência simples para a variável de interesse. Por exemplo:

```
> table(dados$Idade)

16 17 18 19 20 27
 1  3  9  9  6  1

> # Ou seja, há uma pessoa com 16 anos; 3 pessoas com 17 anos e assim por
> # diante
```

## 2.9 Comando which

Which significa ”o qual”ou ”a qual”, e é útil quando o intuito é saber quais linhas de um conjunto de dados tem uma determinada característica. Esse comando é simples, sendo necessário a condição que se deseja testar, por exemplo:

```
> # Linhas nas quais os valores da variável Idade assume o valor 10
> which(dados$Idade == 10)

integer(0)

> # Linhas nas quais os valores da variável Idade assume o valor 18
> which(dados$Idade == 18)

[1]  2  4 11 14 16 19 22 24 27
```

Note que para visualizar as observações das demais variáveis das linhas de interesse, pode-se fazer da seguinte forma:

```
> # O objeto posições contém os números das linhas nas quais Idade = 18
> posicoes<- which(dados$Idade == 18)
> # Exibe as observações de todas as variáveis, porém somente as linhas nas
> # quais o indivíduo possui Idade = 18.
> dados[posicoes,]

   Numero A1 A2 A3 A4 A5 A6 A7 A8 Idade Escolaridade Profissao
2       2  7  3  2  0  4  7  0  0    18           si       est
4       4 10  0  7  1  6  4  1  2    18           mc       est
11     11  6  3  5  0  5  8  0  0    18           mc       est
14     14  9  3  4  7  5  5  0  2    18           si         -
```

```
16     16  0  3  2  0  0  2  0  0    18          mc       est
19     19  7  0  9  6  3  8  2  0    18          si       est
22     22  5  2  7  0  5  9  0  0    18          mc       est
24     24  3  6  1  2  2  8  0  0    18          mc       est
27     27  3  7  8  0  7  5  0  0    18          si       est
   InteresseFormacao
2                cet
4                cet
11               cet
14               eng
16               cet
19               cet
22               cet
24               cet
27             outros
```

## 2.10 Comando subset

Subset em inglês significa subconjunto, portanto é isso que esse comando faz: um subconjunto dos dados. Os argumentos necessários são os dados e em seguida a condição desejada para formar o subconjunto, como no exemplo abaixo.

```
> # Se o interesse for selecionar em Idade, os valores que são maiores que 18
> subset(dados$Idade, dados$Idade > 18)

 [1] 20 19 19 19 19 27 20 19 20 20 19 20 19 19 19 20
```

## 2.11 Comando summary

Retorna um resumo dos dados. Neste resumo estão contidos (nesta ordem) o valor mínimo, o primeiro quartil, a mediana ou segundo quartil, a média, o terceiro quartil e o valor máximo.

Exemplo:

```
> summary(dados$Idade)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  16.00   18.00   19.00   18.86   19.00   27.00

> summary(dados$A1)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.000   5.000   7.000   6.552   9.000  10.000
```

## 2.12 Comando substr

Com esse comando é possivel selecionar ”partes”de vetor de caracteres. A sintaxe desse comando é bastante simples:

substr(x, start, stop)

Onde:

- x é o vetor do qual se deseja selecinar alguma parte;
- start se refere à primeira posição do caracter que será selecionado;
- stop se refere à última posição.

O exemplo a seguir.

```
> subs<- substr(dados$InteresseFormacao, start = 1, stop = 3)
> subs

 [1] "cet" "cet" "cet" "cet" "cet" "cet" "cet" "cet" "cet" "cet" "cet" "cet"
[13] "cet" "eng" "out" "cet" "cet" "cet" "cet" "cet" "cet" "cet" "cet" "cet"
[25] "eng" "cet" "out" "cet" "cet"

>
> # No caso da variável InteresseFormação, na linha 15 tem-se a palava
> # "outros", com 6 posições de caracteres. Ou seja, o caracter da posição 1
> # é o "o", na posição 2, o "u", na posição 3, um "t".
> # Como pode ser observado, esse comando selecionou da variável
> # InteresseFormacao, a informação que estava contida entre o 1º e o 3º
> # caracteres, em cada linha.
```

# 3 Armazenamento

Os dados já foram lidos, modificados e agora falta salvá-los, uma vez que todas as modificações feitas até agora não alteraram o arquivo original, mas sim ”dentro”do R. Portanto três formas de exportar arquivos serão apresentadas: write.table, write.csv e write.xsl. A diferença entre essas formas está no arquivo que será gerado.

O write.table gera um arquivo .txt, o write.csv gera uma planilha do excel com a extensão .csv e o write.xls também gera uma planilha do excel, porém com a extensão .xls.

## 3.1 Comando write.table

Se o intuito for gerar um arquivo .txt este comando deve ser utilizado, seus argumentos são:

write.table(x, file)

Onde:

- x é o objeto que se deseja exportar;
- file é pra onde o arquivo será exportado (seu diretório com nome e extensão)

Segue um exemplo:

```
> # Para exportar o objeto "dados" para um arquivo .txt com o nome "export"
> write.table(x = dados,
+ file = "E:\\UFF\\2012.1\\Monitoria\\Projeto\\Apostila\\export1.txt")
```

## 3.2 Comando write.csv e write.csv2

Esses comando gera um arquivo .csv e seus argumentos são exatamente os mesmos do comando write.table. A diferença entre esses dois é como a casa decimal será representada. Em write.csv ela será representada como ponto (.), em write.csv2, como vírgula (,).

```
> # Para exportar o objeto "dados" para um arquivo .csv com o nome "export2"
> write.csv(x = dados,
+ file = "E:\\UFF\\2012.1\\Monitoria\\Projeto\\Apostila\\export2.csv")
> # Para exportar o objeto "dados" para um arquivo .csv com o nome "export3"
> write.csv2(x = dados,
+ file = "E:\\UFF\\2012.1\\Monitoria\\Projeto\\Apostila\\export3.csv")
```

## 3.3 Comando write.xls

Por último, para exportar um arquivo .xls utiliza-se o comando write.xls cujos argumentos são os mesmo dos citados acima.

```
> # Para exportar o objeto "dados" para um arquivo .xls com o nome "export4"
> write.csv(x = dados,
+ file = "E:\\UFF\\2012.1\\Monitoria\\Projeto\\Apostila\\export4.xls")
```

# Referências


[R Development Core Team(2012)] R Development Core Team (2012) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URLhttp://www.R-project.org/. ISBN 3-900051-07-0.

[Suter(2011)] Suter, H.-P. (2011) *xlsReadWrite: Read and write Excel files (.xls)*. URLhttp://CRAN.R-project.org/package=xlsReadWrite. R package version 1.5.4.