

Introdução à Estatística Computacional

Fernando P. Mayer

Wagner H. Bonat

Laboratório de Estatística e Geoinformação (LEG)
Departamento de Estatística (DEST)
Universidade Federal do Paraná (UFPR)

2018-07-24

Última atualização: 2018-07-24

Sumário

| | | |
|----------|--------------------------------------------------|----------|
| 1 | Computação científica e interação com o R | 7 |
| 1.1 | Interagindo com o computador | 7 |
| 1.2 | Editores de texto | 7 |
| 1.2.1 | Editores para R | 8 |
| 1.3 | R | 8 |
| 1.3.1 | Configuração inicial | 9 |
| 1.3.2 | O R como uma calculadora | 9 |
| 1.3.3 | Para onde vão os resultados? | 9 |
| 1.3.4 | O editor de scripts | 10 |
| 1.3.5 | Operadores aritméticos | 10 |
| 1.3.6 | Ordens de execução | 10 |
| 1.3.7 | “Salvando” resultados | 11 |
| 1.4 | Finalizando o programa | 12 |

Prefácio

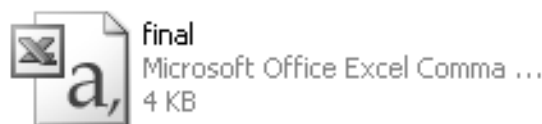
Alguma coisa aqui.

Capítulo 1

Computação científica e interação com o R

1.1 Interagindo com o computador

O que significa este ícone?



- É um documento do Microsoft Excel?
- É um arquivo de **texto pleno**, separado por vírgulas (CSV *comma separated values*)
- De fato, o nome do arquivo é `final.csv` e não `final`
- O Excel pode sim abrir este arquivo... assim como milhares de outros programas!

O que está acontecendo?

- O computador (leia-se, nesse caso, o sistema operacional Windows) “protege” o usuário dos detalhes sujos
- Isso é ruim? **Sim!**
- O usuário se acostuma com o computador ditando as regras
- É importante lembrar que é você quem deve dizer o que o computador deve fazer (nesse caso, com qual programa abrir certo arquivo)

O que deve acontecer?

- Para a maioria dos usuários, a interação com o computador se limita a clicar em links, selecionar menus e caixas de diálogo
- O problema com essa abordagem é que parece que o usuário é controlado pelo computador
- A verdade deve ser o oposto!
- É o usuário que possui o controle e deve dizer para o computador exatamente o que fazer
- Escrever código ainda tem a vantagem de deixar registrado tudo o que foi feito

1.2 Editores de texto

Uma característica importante de códigos de programação é que eles são em **texto puro**, por isso precisamos de um bom **editor de textos**

Características de um bom editor:

- **Identação automática**
- **Complementação de parênteses**
- **Destaque de sintaxe** (*syntax highlighting*)
- **Numeração de linhas**
- **Auto completar comandos**

1.2.1 Editores para R

Windows:

- Interface padrão: pouco recomendado
- Tinn-R

Linux:

- Vim-R-plugin
- Gedit-R-plugin

Todas as plataformas:

- Rstudio: recomendado para iniciantes
- Emacs + ESS: altamente recomendado

1.3 R

“The statistical software should help, by supporting each step from user to programmer, with as few intrusive barriers as possible.”

“... to turn ideas into software, quickly and faithfully.”

— John M. Chambers

O R é um dialeto do S e:

- *Ambiente* estatístico para análise de dados e produção de gráficos
- Uma completa linguagem de programação:
 - Interpretada (contrário de compilada)
 - Orientada a objetos:

Tudo no R é um objeto...

- Livre distribuição (código-aberto)
- Mais de 10000 pacotes adicionais

Pequeno histórico:

- 1980: Linguagem S: desenvolvida por R. Becker, J. Chambers e A. Wilks (AT&T Bell Laboratories)
- 1980: Versão comercial: S-Plus (Insightful Corporation)
- 1996: Versão livre: R desenvolvido por R. Ihaka e R. Gentleman (Universidade de Auckland)
- 1997: R Development Core Team
- Hoje: 20 desenvolvedores principais e muitos outros colaboradores em todo o mundo
- Estatísticos, matemáticos e programadores

1.3.1 Configuração inicial

- O **diretório de trabalho** é uma pasta onde o R será direcionado. Todos os arquivos que serão importados (base de dados, ...) ou exportados (base de dados, gráficos, ...) por ele ficarão nesta pasta.
- Existem duas maneiras de configurar o diretório de trabalho (suponha que vamos usar a pasta ~/estatcomp1):
- 1) Utilizando a função `setwd()` dentro do R:

```
setwd("~/estatcomp1")
```

- 2) Pelo menu do RStudio em Session > Set Working Directory > Choose Directory... Confira o diretório que está trabalhando com a função

```
getwd()
```

1.3.2 O R como uma calculadora

O símbolo > indica que o R está pronto para receber um comando:

```
> 2 + 2  
[1] 4
```

O símbolo > muda para + se o comando estiver incompleto:

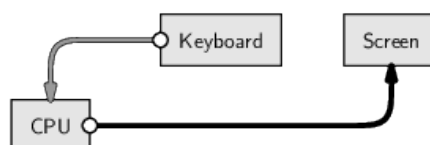
```
> 2 *  
+ 2  
[1] 4
```

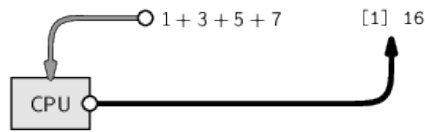
Espaços entre os números não fazem diferença:

```
> 2+      2  
[1] 4
```

1.3.3 Para onde vão os resultados?

```
> 1 + 3 + 5 + 7  
[1] 16
```





- Note que o resultado é apenas mostrado na tela, nada é salvo na memória (por enquanto)

1.3.4 O editor de scripts

- Para criar rotinas computacionais é necessário utilizar um editor de scripts.
- Clique em File > New file > R script. Salve com a extensão .R.
- Para enviar comandos diretamente para o console, selecione-os e aperte Ctrl + <Enter>.
- Para adicionar comentários ao script, utiliza-se o símbolo # antes do texto e/ou comandos. O que estiver depois do símbolo não será interpretado pelo R. Portanto:

```
2 + 2      # esta linha será executada
# 2 + 2    esta linha não será executada
```

1.3.5 Operadores aritméticos

| Operador | Significado |
|------------------------|---------------|
| + | adição |
| - | subtração |
| * | multiplicação |
| / | divisão |
| ^ | potência |
| exp() | exponencial |
| sqrt() | raiz quadrada |
| factorial() | fatorial |
| log(); log2(); log10() | logaritmos |

1.3.6 Ordens de execução

As operações são realizadas sempre seguindo as prioridades:

1. De dentro para fora de parênteses ()
2. Multiplicação e divisão
3. Adição e subtração

```
> 5 * 2 - 10 + 7
[1] 7
> 5 * 2 - (10 + 7)
[1] -7
> 5 * (2 - 10 + 7)
[1] -5
> 5 * (2 - (10 + 7))
[1] -75
```

Exercícios

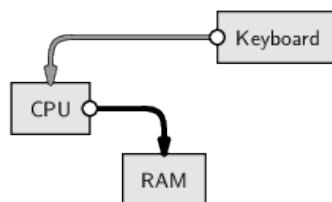
1. Calcule a seguinte equação: $32 + 16^2 - 25^3$
2. Divida o resultado por 345
3. Qual o resultado da expressão $\frac{e^{-2}2^4 - 1}{4!}$?
4. E do logaritmo desta expressão?

1.3.7 “Salvando” resultados

Do exercício anterior

```
> x <- 32 + 16^2 - 25^3
> x
[1] -15337
> x/345
[1] -44.45507
> (y <- (exp(-2) * 2^4 - 1)/factorial(4))
[1] 0.04855686
> log(y)
[1] -3.02502
```

Quando criamos uma variável (x, y), ela fica armazenada **temporariamente** na memória RAM.



Para saber quais objetos estão criados, usamos a **função** `ls()`

```
> ls()
[1] "x" "y"
```

Estas variáveis ficam armazenadas no chamado *workspace* do R

- O *workspace* consiste de tudo que for criado durante uma sessão do R, armazenado na memória RAM

Para efetivamente salvar essas variáveis, podemos armazenar esse *workspace* do R em disco, em um arquivo chamado `.Rdata`





- Quando o R é iniciado em um diretório com um arquivo `.Rdata`, as variáveis salvas são automaticamente carregadas
- No entanto, é sempre melhor salvar os dados e o **script**, assim é possível gerar os resultados novamente, sem salvar nada sem necessidade
- Veremos mais pra frente como salvar variáveis específicas, por exemplo, resultados de uma análise que leva muito tempo para ser executada
- O mais importante é salvar o **código**, assim sabemos **como** chegamos a determinado resultado, e podemos recriá-lo depois

1.4 Finalizando o programa

A qualquer momento durante uma sessão você pode usar o comando

```
> save.image()
```

No RStudio:

- File > Save As...
- Na janela que abrir, digite o nome do arquivo (por exemplo `script_aula1`) e salve
- Automaticamente o script será salvo com a extensão `.R` (nesse caso `script_aula1.R`) no diretório de trabalho que você configurou no início

Alternativamente, você pode também salvar toda sua área de trabalho, clicando em Workspace > Save As Default Workspace. Este processo irá gerar dois arquivos:

- `.Rdata`: contém todos os objetos criados durante uma sessão. Não é necessário (e nem recomendado) dar um nome antes do ponto. Dessa forma, a próxima vez que o programa for iniciado neste diretório, a área de trabalho será carregada automaticamente.
- `.Rhistory`: um arquivo texto que contém todos os comandos que foram digitados no console.

Referências

- Leek, J. [The Elements of Data Analytic Style](#). Leanpub, 2015.
- Murrell, P. [Introduction to data technologies](#). Boca Raton: Chapman & Hall/CRC, 2009.
- Peng, RD. [R programming for data science](#). Leanpub, 2015.