Introdução ao R: aplicações na linguística com corpos

Diana Santos

EBraLC 2012: Material para ler e aplicar antes do curso

1 Primeiros passos: Entrar e sair

Carregue na imagem do R, ou invoque da linha de comandos o comando "R".

O interpretador de R tem em geral o "pronto" >. Aí pode começar a escrever comandos.

O comando mais importante é como sair do interpretador:

```
> q()
```

Ao executá-lo, o R pergunta se pretende guardar o seu ambiente de trabalho – o que pode ser conveniente se estiver a meio de um estudo ou projeto complexo, mas que não é necessário se estiver apenas a experimentar ou se já tiver obtido os resultados que pretende.

O segundo comando mais importante é pedir ajuda:

```
> help()
```

Se quiser saber mais informação sobre um comando, por exemplo a raiz quadrada(sqrt), use `help(sqrt)`.

Também pode ser interessante e importante saber onde está, ou seja, como é que o R interpreta o ambiente em que está instalado:

```
> getwd()
```

Conforme o sistema operacional em que trabalha, poderá obter respostas como

```
[1] "e:/work/dssantos"
[1] "/hf/sokrates/ilos-u1/dssantos"
```

Presumo que saberá interpretá-las, e se quiser que a diretoria de trabalho com o R seja outra, poderá instruir o R para modificar essa variável:

```
> setwd("m:Rwork/")
```

ou

```
> setwd("/linguateca/cursos/R/")
```

Para se familiarizar com a forma como o R funciona e responde, pode começar por experimentar alguns comandos do tipo "máquina de calcular" e de manipulação de estruturas de dados mais complexas.

a) aritméticos e cadeias de caracteres

```
> 3+4
[1] 7
> sqrt(49)
[1] 7
> valor <- 8
> valor
[1] 8
> (7+6)*3 - valor
[1] 31
> "olá"
[1] "olá"
> algo
Error: object 'algo' not found
> 'desculpe'
[1] "desculpe"
> mas()
Error: could not find function "mas"
> help
function (topic, package = NULL, lib.loc = NULL, verbose = getOption("verbose"),
    try.all.packages = getOption("help.try.all.packages"), help_type = getOption
{
    types <- c("text", "html", "pdf")
    if (!missing(package))
 [...]
>
```

b) vetores Note que a função `c` concatena, ou seja, junta vários objetos num vetor:

```
> c("23","1","batatas","cenouras")
[1] "23"       "1"        "batatas"  "cenouras"
> bobagem <- c("23","1","batatas","cenouras")
> bobagem
[1] "23"       "1"        "batatas"  "cenouras"
> 24:100
 [1]  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42
[20]  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61
[39]  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
[58]  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99
[77] 100
```

A forma como o R respondeu dá para compreender, espero, um conjunto de realidades básicas sobre esta linguagem, nomeadamente, que

- cada linha de resposta vem numerada, começando com 1, e a ordem do primeiro caso dessa linha é sempre impressa no lado esquerdo, entre []
- quando são dadas instruções, o R executa o comando correspondente e devolve o resultado
- quando se digita simplesmente um nome, o R vai ver se a esse nome já foi dado algum sentido. Se foi, dá-nos o valor atribuído (que até pode ser uma descrição de código), senão dá uma mensagem de erro
- quando se lhe comunica um valor, ele ecoa esse valor
- há diferença entre nomes de variáveis, e cadeias de carateres (marcadas por aspas ou plicas)
- as funções no R são designadas por NOME(argumentos). Quando não têm argumentos, como a função `help`, têm de ser marcadas por () a seguir ao nome, ou seja `help()`

2 Comunicar com o exterior

Se quisermos trabalhar a sério, e não usar somente o R como um oráculo ou máquina de calcular, teremos forçosamente de interagir com dados pré-existentes e produzir resultados e figuras. Para isso, é preciso saber como ler e escrever arquivos.

A maneira mais fácil para começar é ler de um ficheiro existente na rede, por isso os primeiros exercícios irão proceder de um conjunto de dados que

pus na minha página. Mas depois vão progressivamente usando arquivos dentro do vosso computador.

```
> CondivSimples <- read.table("http://folk.uio.no/dssantos/cursoR/condiv.txt")
> CondivSimples
               V1     V2       V3     V4      V5
1         dataset colour    theme decade variety
2     fashion50BR   1512  fashion     50      BR
3     fashion50PT   2351  fashion     50      PT
[...]
```

Ao inspecionar o conteúdo do que lemos é imediatamente aparente que a primeira linha é especial, porque é um cabeçalho. Existe uma forma simples de dizer isso ao R – mas antes disso, quero chamar a atençao para o comportamento do R quando os comandos são maiores do que uma linha: o R fica à espera do resto, e mostra que está à espera, escrevendo + em vez de >.

```
> CondivSimples <- read.table("http://folk.uio.no/dssantos/cursoR/condiv.txt",
+ header=TRUE)
> CondivSimples
          dataset colour    theme decade variety
1     fashion50BR   1512  fashion     50      BR
2     fashion50PT   2351  fashion     50      PT
[...]
```

O conteúdo deste arquivo ilustra a forma mais usada para fazer análises estatísticas: uma observação por linha, contendo o valor de várias variáveis, cada variável correspondendo a uma coluna. Em inglês, um conjunto de dados assim é chamado "dataframe", e proponho chamar-lhe em português uma "folha de registo".

Uma folha de registo é como uma matriz em que tanto às linhas como às colunas se podem dar nomes. A maior diferença é que uma matriz pressupõe que todos os elementos sejam do mesmo tipo, enquanto numa folha de registo não.

Uma forma de obter rapidamente as dimensões (número de linhas e de colunas) é usar a função `dim`:

```
> dim(CondivSimples)
[1] 18  5
```

Para obter apenas uma coluna, pode-se simplesmente usar o nome da folha de registo seguida de cifrão e do nome da coluna:

```
CondivSimples$colour
 [1] 1512 2351 1342 6481 1525  669  719  694  468  318 1167  614  136  583  127
[16]  993   97  265
```

No caso deste exemplo, também podemos ver que a primeira coluna é sempre diferente, e que presumivelmente foi usada para identificar de forma única cada linha. Por isso, é possível informar o R disso também, através da indicação `rownames=número da coluna`. Esta situação não é contudo tão comum, porque as folhas de registo costumam ter milhares de linhas e nem sempre faz sentido dar-lhes um identificador único.

```
> CondivSimples <- read.table("http://folk.uio.no/dssantos/cursoR/condiv.txt",
+ header=TRUE, row.names=1)
> CondivSimples
               colour    theme decade variety
fashion50BR      1512  fashion     50      BR
fashion50PT      2351  fashion     50      PT
[...]
```

Agora existem portanto apenas quatro colunas na folha de registo, e não cinco como anteriormente.

```
> dim(CondivSimples)
[1] 18  4
```

Podemos estar interessados em criar uma nova folha de registo só com uma parte das informações de outra. Para isso existe a função `subset`. No exemplo seguinte, selecionamos apenas as linhas relativas ao futebol e as colunas com cor e variante:

```
> SoFutebol<-subset(CondivSimples,theme=="football",c(1,4))
> SoFutebol
               colour variety
football50BR      719      BR
football50PT      694      PT
football70BR      468      BR
football70PT      318      PT
football2000BR   1167      BR
football2000PT    614      PT
```

Vamos agora escrever esta folha de registo na vossa diretoria de trabalho.

```
> write.table(SoFutebol, file="futebol.txt")
```

Além de poder ir ver, através do seu sistema operativo, o que está nesse arquivo, outra forma é tornar a lê-lo para o R:

```
read.table("futebol.txt")
```

Vamos agora criar uma tabela à mão, e depois escrevê-la num arquivo externo. Embora esta não seja uma maneira natural de trabalhar, é uma maneira adequada de nos familiarizarmos com o R.

```
> tabela<-matrix(c(38,14,11,51),nrow=2)
> tabela
> colnames(tabela)<-c("OlhosAzuis","OlhosCastanhos")
> rownames(tabela)<-c("CabeloLoiro","CabeloCastanho")
> write.table(tabela, file="CabelosOlhos.txt")
```

3 Figuras e gráficos

Naturalmente, até agora o R não se destacou de qualquer outra linguagem de programação ou de manipulação de texto, de matrizes ou de outras estruturas de dados.

A questão das capacidades gráficas é talvez o primeiro caso em que os usuários se podem aperceber da diferença e vantagem em usar o R.

Em primeiro lugar, é preciso compreender que o R dispõe de várias janelas, e que por omissão abre uma janela para todos os gráficos. Ou seja, ao desenharmos um gráfico ou figura, deixamos de ver e de poder usar a anterior. A não ser que a guardemos, ou que explicitamente indiquemos que queremos uma nova janela.

Os seguintes são os comandos mais simples de criação de figuras, em que o til ~ pode ser lido "em função de".

```
> plot(colour ~ variety, data=CondivSimples)
> hist(CondivSimples$colour)
> plot(CondivSimples$colour)
```

Se estivermos a trabalhar com uma dada folha de registo, dá jeito não ter sempre que estar a escrever o nome todo, e usar apenas os nomes associados a ela. Isso consegue-se com o comando `attach`:

```
> attach(CondivSimples)
```

Refaça as figuras anteriores depois de se ter associado à folha de registo, para ver como é mais simples.

Uma das coisas que convém compreender em relação ao R é que tem um mecanismo de simplificação (ou omissão) que permite várias formas diferentes de invocar uma "mesma" função, especificando ou não um número variável de coisas e de argumentos. (Tal com em linguagem natural um verbo n-ário pode ser usado com menos argumentos...) Assim, a função `plot` pode dar resultados, ou melhor gráficos, completamente diferentes conforme os argumento e seus tipos.

Além disso podem especificar-se muitíssimas escolhas diferentes, como por exemplo o título da figura, a forma e a cor dos pontos, o nome das abcissas e das ordenadas, a escala, etc etc etc. Alguns exemplos simples:

```
> plot(colour, col="green", pch=3, type="b")
> plot(colour, col="red", pch=2, las="1", tcl="0.2", xlab="casos",
+ ylab="Quantas palavras de cor", type="h")
> plot(colour, col="red", pch=2, las="1", xlab="casos",
+ legend(4,6481,"Máximo")
```

Mas para explorar bem esta vertente, nada como fazer `help(plot)` e ler as várias opções acessíveis, ou consultar o cartão de referência do R [17].

De qualquer maneira, é importante explicar como obter as figuras num formato que dê para incluir noutras aplicações, quer artigos, quer apresentações. Isso consegue-se simplesmente direcionando o lugar para onde se imprime através de comandos com o nome do formato pretendido. Se quiser um pdf, basta fazer:

```
> pdf("simple.pdf")
> plot(colour~theme)
> dev.list()
```

Se quiser voltar à sua janela, basta ver quais as janelas através do comando `dev.list`, e depois selecionar a que quer através do respetivo número

```
> dev.list()
X11cairo      pdf
       2        3
> dev.set(2)
```

Enquanto não fechar a janela em que está a escrever, não pode geralmente abri-la com outros programas, por isso é importante que, quando estiver pronta, a feche com o comando `dev.off()`. Este comando tem como argumento o número da janela em questão. Se for invocado sem argumentos, fecha a última janela aberta, como é o caso seguInte:

```
> dev.off()
X11cairo
       2
```

O mesmo se passa para formatos como `png`, `bmp`, etc.

Se quiser abrir várias janelas no seu computador ao mesmo tempo, e não escrever por cima, pode usar `dev.new()`.

4 Preservar o trabalho entre sessões

Embora não seja apropriado para principiantes, dá muito jeito pder preservar o trabalho entre sessões quando se trabalha intensivamente e temos de interromper no meio de um percurso cheio de ingredientes e tarefas feitas.

Nesses casos, pode-se instruir o R para guardar tudo para quando nós voltarmos. O R tem um formato especial, do tipo `.Rdata`, e os seguintes comandos para guardar e recuperar o que se guardou.

```
> save.image("tudo.Rdata")
```

Saia do R e volte. Repare que o R pergunta sempre se quer guardar, mas nesse caso guarda num geral (chamado simplesmente `.Rdata`), e pode guardar mais do que queremos. Por exemplo, podemos guaradr a parte importante, e depois ficarmos a fazer algumas experiências ou contas, que não são para guardar...

Esta é a altura para aprender alguns comandos de limpeza e organização da "casa". `ls` permite listar o que está no ambiente, e `rm` apagar.

```
> ls()
character(0)
> load("tudo.Rdata")
> ls()
[1] "bobagem"       "CondivSimples" "onlyfb"        "SoFutebol"
```

Outra coisa que pode dar jeito, e que ocupa menos espaço, é manter uma história dos comandos, para a qual o R também tem um formato especial, `.Rhistory`. Pode dar-se um nome:

```
> savehistory("comandosTese.Rhistory")
```

Como deverá ser fácil de adivinhar, para os ir buscar numa futura sessão, basta invocar o comando dual `loadhistory`.

```
> loadhistory("comandosTese.Rhistory")
```

Em todos estes casos, é preciso reparar que, se nada diferente for dito, o R assume que estes objetos estão na diretoria de trabalho. Se não estiverem, tem de se incluir o nome completo do arquivo dentro das aspas, ou mudar para o sítio certo (com `setwd`) antes de o carregar.

5 Exercícios de estatística

(Estes exercícios são para ser feitos, explicados, e depois resolvidos, durante o próprio curso, por isso a informação aqui é mínima, propositadamente.)

5.1 Referência ao medo em português

Leia o ficheiro `http://folk.uio.no/dssantos/cursoR/medo.txt`, para uma folha de registo chamada `medo`. (A partir de agora, todos os nomes dos arquivos pressupõem o mesmo caminho, `http://folk.uio.no/dssantos/cursoR/`, que não será mais repetido.)

Familiarize-se com essa folha de registo, e explore-a graficamente.

Crie uma nova coluna com a informação número de palavras de medo por 10.000 palavras, e crie a figura correspondente.

Existe diferença entre os gêneros textuais? Use métodos estatísticos para responder a esta pergunta. Para saber mais sobre o medo em português e em inglês, veja [10].

5.2 Estudo das cores em português

Vá buscar o arquivo `CondivCompleto.txt`.

Investigue se existem diferenças entre o número de palavras de cor utilizadas por década, por variante do português, e por tema.

Obtenha um modelo dos dados, e observe se existem dependências entre as variáveis independentes.

Verifique, no corpo Vercial, o que pode dizer ou descobrir acerca da evolução das cores, ou da sua cor preferida, usando o material de `vercial.txt`.

5.3 Estudo de diferenças lexicais entre as variantes

Para que casos é possível demonstrar estatisticamente que as palavras são usadas de forma diferente nas duas variantes?

Exemplos reais, retirados de [14] e dos corpos do AC/DC, e presentes em `DifsLex.txt` e `DifsLex2.txt`.

Medida da incerteza e do intervalo de confiança.

5.4 Diferença entre tamanhos de frases entre originais e traduções

As frases nos textos traduzidos têm sempre um tamanho superior ao das frases originais? Ou depende da língua? Ou doutra coisa qualquer?

Os dados são os tamanhos das frases (em termos de número de palavras) no CorTrad jornalístico, em `tamanhosCorTradJorn.txt`.

Nota: Quando uma frase é traduzida por várias, conta-se a soma das várias. Quando mais do que uma frase é convertida para uma só, conta-se a soma das várias.

Experimente retirar, dos seus testes, os casos de frases não traduzidas, de forma a não contarem em cálculos de tamanho.

5.5 Estudo de diferenças sintáticas entre mulheres e homens

Leia o ficheiro `austen.txt`, com dados retirados de [16].

Investigue se há diferenças no uso de adjetivos entre mulheres e homens, em geral, e nos casos de conversas entre géneros, nos livros de Jane Austen. Para esta última questão, use `austenCompleto.txt`.

5.6 Estudo de diferenças semânticas entre autores

Dados os valores sobre cores obtidos com o COMPARA e apresentados em [18], podemos afirmar que há diferenças estatisticamente significativas entre os autores nessa matéria?

Os arquivos relevantes chamam-se `autCoresCOMPARA.txt`, com a distribuição de cores pro autor, e `autAdjCOMPARA.txt`, com a distribuição de adjetivos por autor.

5.7 O uso dos tempos em português

Passemos agora para métodos exploratórios. Existem contextos em que certos tempos são muito mais frequentes do que outros? Existem verbos que atraem certos tempos e rejeitam outros?

Estude o problema usando as listas de frequências de tempo e aspeto que se encontram no arquivo `tempos.txt`, inicialmente apresentadas em [15].

5.8 Deteção de tradutês

Dado um conjunto de características obtidas para uma série de textos traduzidos de várias línguas para inglês, é possível detetar automaticamente se são traduzidos ou não, e de que língua provêm?

[9] usaram o corpo criado e documentado em [8] para investigar esta questão. Os dados da ocorrência de palavras gramaticais estão em `iht.txt`, para 261 textos, em inglês original, e em inglês traduzido do coreano, do grego e do hebraico.

6 Leituras recomendadas

Na secção de referências incluo vários livros escritos para apresentar o R a pessoas que trabalham com língua, do qual destaco o livro de Baayen [1] como o mais completo, assim como outros livros que são interessantes quer para aprender estatística (com R) quer para aprender a aplicação da estatística à linguística ou ao processamento computacional da língua.

Também incluo alguma documentação de referência do R, e materiais de apoio criados por outros, que se encontrem acessíveis na rede.

Faço também uma referência especial à tese do Stefan Evert [5], por ser extremamente clara e didática no que se refere ao estudo de co-ocorrências e de colocações.

O caminho de cada um através desta floresta de documentação, não o pretendo predeterminar. Bom passeio, e boa colheita... de ideias, métodos, e programas!

Agradecimentos

Agradeço a Noam Ordan a extensa correspondência que culminou com o envio do corpo IHT e dos materiais usados, assim como a Jean Piton e Ana Engh os comentários pertinentes em relação a estas instruções.

Referências

[1] Harald Baayen. *Analyzing Linguistic Data: A practical introduction to Statistics using R*, Cambridge University Press, 2008.

[2] R. H. Baayen. *languageR: Data sets and functions with "Analyzing Linguistic Data: A practical introduction to statistics*, 2011. `http://CRAN.R-project.org/package=languageR`. R package version 1.2.

[3] Christopher S. Butler. *Statistics in Linguistics*, Oxford: Blackwell, 1985. `http://www.uwe.ac.uk/hlss/llas/statistics-in-linguistics/bkindex.shtml`.

[4] Michael J. Crawley. *Statistics: An Introduction using R*, John Wiley and Sons, 2005.

[5] Stefan Evert. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Tese de doutoramento, Institut für maschinelle Sprachverarbeitung, University of Stuttgart, 2004.

[6] Stefan Th. Gries. *Statistics for Linguistics with R: A Practical Introduction*, Mouton de Gruyter, 2009.

[7] Keith Johnson. *Quantitative Methods in Linguistics*, Wiley-Blackwell, 2008.

[8] Ido Kanter, Haggai Kfir, Brenda Malkiel e Miriam Shlesinger. Identifying universals of text translation. *Journal of Quantitative Linguistics*, 13(1):35–43, 2006.

[9] Moshe Koppel e Noam Ordan. Translationese and its dialects. Em *ACL'11*, 2011. p. 1318–1326.

[10] Belinda Maia e Diana Santos. Who is afraid of... what? - In English and in Portuguese. Em Signe Oksefjell Ebeling, Jarle Ebeling e Hilde Hasselgård, editores, *Proceedings of ICAME 32*, 2012.

[11] Michael P. Oakes. *Statistics for Corpus Linguistics*, Edinburgh University Press, 1998.

[12] Michael P. Oakes e Meng Ji, editores. *Quantitative methods in corpus-based translation studies : a practical guide to descriptive translation research*, John Benjamims, 2012.

[13] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2010. `http://www.R-project.org`. ISBN 3-900051-07-0.

[14] Tommaso Raso e Heliana Melo, editores. *C-oral-Brasil I Corpus de referência do português brasileiro falado informal*, Editora UFMG, 2012.

[15] Diana Santos. The next step for the translation network. Em Diana Santos, Krister Lindén e Wanjiku Ng'ang'a, editores, *Shall We Play the Festschrift Game? Essays on the occasion of Lauri Carlson's 60th birthday*, 2012. p. 35–52.

[16] Kari Anne Rand Schmidt. Male and female language in Jane Austen's novels. Em Stig Johansson e Bjørn Tysdahl, editores, *Papers from the First Nordic Conference for English Studies (Oslo, 19-19 September, 1980)*, 1980. p. 198–210.

[17] Tom Short. *R Reference Card*, 2004. `http://cran.r-project.org/doc/contrib/Short-refcard.pdf`.

[18] Rosário Silva, Susana Inácio e Diana Santos. Colouring COMPARA: contrastive and monolingual colour studies in English and Portuguese. Apresentação na American Association for Corpus Linguistics (AACL) (Provo, USA, March 2008), 2008. `http://www.linguateca.pt/Diana/download/Silvaetal_AACL2008_final.pdf`.

[19] Luís Torgo. *Acetatos usados nas aulas*, sem data. `http://www.liaad.up.pt/~ltorgo/Ensino/FEP/`. Faculdade de Economia, Universidade do Porto.