Linguagem de Programação S: Fundamentos e Aplicações em Recursos Florestais

Linguagem de Programação S: Fundamentos e Aplicações em Recursos Florestais

1. INTRODUÇÃO

1.3. Comandos/Funções

1.3.1. A Linha de Comando

O R é um ambiente interativo, ou seja, que permite ao usuário enviar um comando por vez e receber o resultado[1)]. No **console**, a linha de comando se inicia com o sinal “>”, indicando que o R está pronto para receber um comando.

Os outros dois estados da linha de comando são o de *execução* e o *comando incompleto*. No modo de

execução não é exibido nenhum sinal e não é possível digitar outro comando. Você só perceberá isso se der um comando que tenha um tempo de execução muito longo.

Já o estado no comando incompleto, surge o sinal “+”:

| Console do R

```
>
> log(2
+ )
[1] 0.6931472
>
```

Na primeira linha, não fechamos os parênteses da função “log” e teclamos <RETURN>. O R responde com o sinal de comando incompleto (+) e fica à espera da sua conclusão. Digitando o parêntese que falta e apertando a tecla <RETURN> novamente o R retorna o resultado do comando, precedido de um índice numérico[2)].

Cada linha do **script** deve conter uma linha de comando válida no console, inclusive as linhas com comentários.

```
> log(2)
[1] 0.6931472
> # log(2)
>
> sin(pi/2)
[1] 1
> # sin(pi/2)
>
```

1.3.2. Sintaxe Básica dos Comandos

Um comando no R em geral inclui uma ou mais funções, que seguem a seguinte sintaxe:

função(argumento1 = valor , argumento2 = valor , ...)

Como nos exemplos abaixo:

```
> plot(x=area,y=riqueza)
> plot(area, riqueza)
> plot(area,riqueza,xlab="Área (ha)", ylab="Riqueza")
>
```

- No primeiro caso, o valor de cada argumento usado está explicitado. O argumento x da função `plot` é a variável independente, e o argumento y é a variável dependente.
- Se o nome dos argumentos é omitido, como no segundo caso, o **R** usa o critério de posição: o primeiro valor é atribuído ao primeiro argumento, o segundo valor ao segundo argumento, e assim por diante. Como os dois primeiros argumentos da função plot são x e y, o segundo

comando acima equivale ao primeiro.
- Os dois critérios podem ser combinados, como no terceiro comando: como x e y são os dois primeiros argumentos, não é preciso declará-los. Como os outros dois argumentos que se deseja usar (`xlab` e `ylab`) não vêm em seguida, é preciso declará-los.

Um função útil para se conhecer os argumentos de qualquer função do **R** é a função `args`, pois ela lista os principais argumentos da função de interesse.

```
> args(plot)
function (x, y, ...)
NULL
> args(lm)
function (formula, data, subset, weights, na.action, method = "qr",
    model = TRUE, x = FALSE, y = FALSE, qr = TRUE, singular.ok = TRUE,
    contrasts = NULL, offset, ...)
NULL
>
```

1.3.3. Criação de Objetos: Atribuição

Você pode "guardar" o resultado de um comando com a operação de *atribuição*, que tem a sintaxe:

objeto recebe valor

Há dois operadores que atribuem valores a um objeto dessa maneira:

- Sinal de menor seguido de hífen, formando uma seta para a esquerda: "**<-**"
- Sinal de igual: "**=**"

Console do R

```
> a <- sqrt(4)
> b = sqrt(4)
```

Uma forma de atribuição menos usada é:

valor atribuído a **objeto**

Nesse caso, o sinal de atribuição é o hífen seguido de sinal de maior, formando uma seta para direita: "**->**"

Console do R

```
> sqrt(4) -> d
```

Para exibir o conteúdo de um objeto, basta digitar seu nome

```
> a
[1] 2
> b
[1] 2
> d
[1] 2
```

Se a atribuição é para um objeto que não existe, esse objeto é criado. Mas cuidado: se já há um objeto com o mesmo nome na sua área de trabalho, seus valores serão substituídos:

```
> a <- sqrt(4)
> a
> a <- 10^2
> a
```

1.3.4. Mensagens de Erro e de Avisos

Como em qualquer linguagem, o R tem regras de sintaxe e grafia. Mas contrário das linguagens humanas, mesmo um pequeno erro torna a mensagem incompreensível para o R, que então retorna uma mensagem de erro:

```
> logaritmo(2)
Error: could not find function "logaritmo"
> log(2))
Error: unexpected ')' in "log(2))"
> log(2, basse=10)
Error: unused argument(s) (basse = 10)
> log(2, base=10)
[1] 0.30103
```

Em outros casos, o comando pode ser executado, mas com um resultado que possivelmente você não desejava. O R cria mensagens de alerta para os casos mais comuns desses resultados que merecem atenção :

```
> log(-2)
[1] NaN
Warning message:
In log(-2) : NaNs produced
```

Observação:

Embora seja possível escolher o português como a língua no R, nem todas as mensagens de erro ou aviso aparecerão em português. Como a língua mãe do R é o inglês, nessa disciplina todas mensagens (error/warnnings) serão apresentadas em inglês.

1.3.5. Gerenciando a Área de Trabalho com Comandos

Listando Objetos

O comando `ls` lista todo o conteúdo da área de trabalho, se não é fornecido argumento:

```
> ls()
[1] "consoantes" "CONSOANTES" "vogais" "VOGAIS"
```

A função `ls` possui argumentos que podem refinar seus resultados, consulte a ajuda para os detalhes.

A função **objects** realiza praticamente as mesmas operações da função `ls`.

RStudio: nessa interface, existe a janela "*Environment*" que mostra os objetos existentes na área de trabalho.

Apagando Objetos

O comando `rm` apaga objetos da área de trabalho:

```
> ls()
[1] "consoantes" "CONSOANTES" "vogais"     "VOGAIS"
> rm(consoantes)
> ls()
[1] "CONSOANTES" "vogais"     "VOGAIS"
```

Consulte a ajuda da função `rm` para seus argumentos.

Existe a função `remove` que apaga objetos do seu ***workspace*** da mesma forma que a função `rm`, mas seus argumentos são ligeiramente diferentes.

1.3.5. Particularidades dos Comandos no R

Como em toda linguagem, o **R** tem algumas regras básicas de sintaxe e grafia:

1. O nome de comandos e objetos no **R** pode ser compostos de:
   - letras (minúsculas ou maiúsculas),
   - números, e
   - o ponto (.).
2. Evite qualquer outro caracter especial, incluindo *o espaço em branco*.
3. O nome *não pode ser iniciado* por um número.
4. O **R** é sensível à caixa: o comando `q` é diferente do comando `Q` (que não existe!!!).

Para que um comando seja executado pelo R é necessário ser acompanhado de parênteses `()`. Compare esse comando

```
> q()
Save workspace image? [y/n/c]: c
>
```

com o comando

```
> q
function (save = "default", status = 0, runLast = TRUE)
.Internal(quit(save, status, runLast))
<environment: namespace:base>
>
```

O comando sem os parênteses é na verdade o nome do comando. Sendo o **R** um software de código aberto, toda vez que se digita o nome de um comando, ele não executa o comando mas mostra o conteúdo do comando (o código).

1)
é possível também de executar um lote de comandos, mas neste wiki trabalharemos apenas com o modo interativo.
2)
o significado deste índice ficará claro na seção sobre sub-amostragem. Por enquanto basta saber que os resultados do R são precedidos por um indicador numérico entre colchetes