Estatística

Introdução à Aplicação R

2008/2009

O que é o R?

- É um conjunto integrado de ferramentas computacionais que permitem a manipulação e análise de dados, o cálculo numérico e a produção de gráficos.
- É uma linguagem de programação simples (uma variante da linguagem S).
- É uma linguagem interpretada:
  - os comandos são imediatamente executados.
- É uma linguagem orientada por objectos:
  - os dados manipulados são armazenados na memória activa do computador na forma de objectos, que têm um nome e aos quais se podem aplicar acções.

Instalar o R

- O R é uma aplicação de distribuição gratuita e de código público (http://cran.r-project.org/), existindo versões já compiladas para execução nos principais sistemas operativos (Windows, Linux e Macintosh).
- Depois de fazer o *download* da versão adequada ao sistema operativo do computador (por ex. R-2.7.2-win32.exe, para o Windows), para instalar o R basta executar esse ficheiro.

Iniciar uma sessão de R

- Criar na área de trabalho uma nova pasta (por exemplo, aulasR) onde irão ser guardados os ficheiros de dados – pasta de trabalho.
- Iniciar o R:

Start → All Programs → R → R 2.7.2

(*em alternativa, pode-se criar um atalho no desktop e alterar as suas propriedades de modo a iniciar a aplicação na pasta de trabalho*)

RGui (Graphical user interface)

- O R funciona fundamentalmente no modo "pergunta-resposta":
  - Os comandos (frases que se escrevem numa certa sintaxe) introduzem-se a seguir à prompt (>) e são executados após pressionar Enter (↵)
- Edição de comandos:
  - Seleccionar comandos executados anteriormente: ↑ ou ↓
  - Percorrer a linha de comandos: ← ou →
  - Colocar o cursor no início / fim da linha de comandos: Home / End
- Para terminar o R:
  - executar o comando `q()` ou fechar a janela da aplicação.

Comandos elementares

- Expressão - o resultado é apresentado no visor e não é registado em memória.

```
> 2+3/4*7^2
[1] 38.75
> exp(-2)/log(sqrt(2))
[1] 0.3904951
```

- Atribuição – o resultado da expressão é atribuído à variável e não é apresentado no visor.

```
> x <- 2      # <- é o operador de atribuição
> x
[1] 2
> x <- 2+3/4*7^2
> x
[1] 38.75
```

Designações das variáveis

- Os nomes das variáveis podem conter letras (maiúsculas são diferentes de minúsculas), algarismos e pontos. O 1º caracter deve ser uma letra.

  Exemplo: `Pb.ISA`

- Alguns nomes são utilizados pelo sistema, pelo que devem ser evitados (por ex., `c, q, t, C, D, F, I, T, diff, df, pt`).

Help

- O comando `help` serve para obter informação sobre uma função específica. Por exemplo, para obter informação sobre a função `sin()` pode utilizar-se qualquer uma das opções:

```
> help(sin)
> help("sin")
> ?sin
```

- O comando `help.search()` permite pesquisar uma sequência de caracteres. Por exemplo,

```
> help.search("solve system")
```

- O comando `help.start()` acede a uma página com informação diversa sobre o R.

(ver também o menu Help no ambiente de trabalho.)

Objectos

- Os objectos são as entidades que o R cria e manipula e que são guardadas em memória.
  - Para listar os objectos disponíveis pode utilizar-se qualquer uma das opções:

    ```
    > ls()
    > objects()
    ```

  - Para mostrar alguma informação sobre os objectos:

    ```
    > ls.str()
    ```

  - Para eliminar objectos:

    ```
    > rm(objecto1, objecto2,...)
    ```

  - Para eliminar todos os objectos:

    ```
    > rm(list=ls())
    ```

Objectos (*Workspace*)

- *Workspace* é a colecção dos objectos disponíveis numa sessão, que pode ser guardada num ficheiro com vista à sua utilização em futuras sessões de R.
  - No final de cada sessão é perguntado ao utilizador se quer guardar o *workspace.* Caso a resposta seja afirmativa, todos os objectos disponíveis em memória são guardados no ficheiro ".RData", na pasta de trabalho em uso, podendo ser carregados na próxima sessão de R.
    - Especificação da pasta de trabalho:

      menu File → Change dir...
    - Importar um ficheiro de objectos (por exemplo o ficheiro ".RData"):

      menu File → Load Workspace...

Alguns objectos

- *Vector*
- *Matrix*
- *List*
- *Data Frame*

Vector

- Um vector é uma colecção ordenada de elementos do mesmo tipo (*mode*) (valores numéricos, lógicos, alfanuméricos, ...).
- Uma das maneiras de criar um vector no R é através da função `c()`.

```
> x <- c(5.4, -3.7, 11.2, 0.78, 21.6)
> x
[1]  5.40 -3.70 11.20  0.78 21.60
> y <- c(FALSE,  TRUE,  TRUE,  TRUE, FALSE)
> y
[1] FALSE  TRUE  TRUE  TRUE FALSE
> nomes <- c("Ana","Paulo","Zé")
> nomes
[1] "Ana"   "Paulo" "Zé"
```

Vector

- Um vector pode conter os símbolos especiais `NA` e `NaN`, que designam um valor desconhecido e um valor não definido, respectivamente.

```
> y <- c(NA, 53, 31, 15, 62)
> sqrt(c(-1,1,2))
[1]       NaN 1.000000 1.414214
Warning message:
NaNs produced in: sqrt(c(-1, 1, 2))
```

Vector

- `c()` é a função concatenação. Os argumentos desta função podem ser eles próprios vectores, pelo que `c(v1,v2)` devolve a concatenação dos vectores `v1` e `v2`. Esta função pode ter vários argumentos.

```
> c(x,0,x)
[1] 5.40 -3.70 11.20 0.78 21.60 0.00
  5.40 -3.70 11.20 0.78 21.60
> nomes <- c(nomes,"Manel")
> nomes
[1] "Ana"   "Paulo" "Zé"    "Manel"
```

Vector numérico (operações aritméticas)

- As operações entre dois vectores são realizadas elemento a elemento.
- Alguns operadores aritméticos: +, −, *, /, ^

```
> v1 <- c(5,7)

> v2 <- c(10,11,12,13)

> 1/v1
[1] 0.2000000 0.1428571

> v1+v2 # o vector de menor dimensão é concatenado
> v1*v2 # consigo mesmo(total ou parcialmente) até
          ficar com o mesmo número de elementos do
          maior

> v1*4 # cada elemento de v1 é multiplicada por 4
```

Vector numérico (algumas funções)

- `length(x)` devolve o número de elementos do vector `x`,
- `sum(x)` devolve a soma dos elementos do vector `x`,
- `prod(x)` devolve o produto dos elementos do vector `x`,
- `cumsum(x)` devolve um vector cujos elementos são a soma acumulada dos elementos do vector `x`,
- `cumprod(x)` devolve um vector cujos elementos são o produto acumulado dos elementos do vector `x`,
- `sort(x)` devolve um vector com os elementos do vector `x` ordenados por ordem crescente,
- `choose(n,k)` devolve $\binom{n}{k}$.

Vector numérico (algumas funções)

- Quando o argumento é o vector `x`, as funções

  `log`, `exp`, `sin`, `cos`, `tan`, `atan`, `sqrt`, `abs`, ...

  devolvem o vector em que cada elemento é o resultado de aplicar a função ao elemento homólogo de `x`.

Geração de sequências regulares

- O R permite gerar sequências de valores numéricos

```
> 1:15     # define o vector c(1,2,...,14,15)
> 15:1     # define o vector c(15,14,...,2,1)
> 2*1:15   # define o vector c(2,4,...,28,30)
> (2*1):15 # define o vector c(2,3,...,14,15)
> seq(1,15)
> seq(to=15,from=1)
> seq(-5,-1,by=0.5)
 [1] -5.0 -4.5 -4.0 -3.5 -3.0 -2.5 -2.0 -1.5 -1.0
> seq(length=9, from=-5, by=0.5)
> rep(1:3,2) # define o vector c(1,2,3,1,2,3)
> rep(1:3,each=2) # é o vector c(1,1,2,2,3,3)
```

Vector lógico (operações)

- Os resultados das operações que envolvem os operadores relacionais:

  `<, <=, >, >=, ==, !=`

  são valores lógicos.

```
> 2==sqrt(4)
[1] TRUE
> 2!=sqrt(4)
[1] FALSE
```

- Quando aplicados a vectores, produzem vectores lógicos.

```
> x
[1]  5.40 -3.70 11.20  0.78 21.60
> x>11
[1] FALSE FALSE  TRUE FALSE  TRUE
```

Vector lógico (operações)

- Os operadores lógicos conjunção, disjunção e negação são, respectivamente:

```
                &     |     !

> x
[1]  5.40 -3.70 11.20  0.78 21.60
> x>13 | x<5
[1] FALSE TRUE  FALSE TRUE  TRUE
```

Vector lógico (função `is.na`)

- A função `is.na(x)` devolve um vector lógico em que cada elemento é `TRUE` se o elemento homólogo em `x` é do tipo `NA` ou `NaN`, e ou `FALSE` caso contrário.

```
> z <- c(5.4,-3.7,11.2,NA,21.6)
> is.na(sqrt(z))
[1] FALSE  TRUE FALSE  TRUE FALSE
Warning message:
NaNs produced in: sqrt(z)
> !is.na(z)
[1]  TRUE  TRUE  TRUE FALSE  TRUE
```

Selecção de componentes de vectores

- Cada elemento de um vector pode ser seleccionado indicando a sua posição entre parêntesis rectos a seguir ao nome do vector.

```
> letras<-c("a","b","c","d","e","f","g")
> letras[3]
[1] "c"
```

- Pode também criar-se um vector constituído por alguns dos elementos do vector x indicando as suas posições num vector `ind` colocado entre parêntesis rectos:

```
> ind<- c(1,2,4)
> letras[ind]
[1] "a" "b" "d"
> letras[-ind] # devolve os elementos de letras
                 que não estão nas posições
                 indicadas em ind
[1] "c" "e" "f" "g"
```

Selecção de componentes de vectores (cont.)

- Pode também seleccionar-se elementos de um vector escrevendo uma condição entre parêntesis rectos a seguir ao nome do vector.

```
> x
[1]  5.40 -3.70 11.20  0.78 21.60
> x[x>0]
[1]  5.40 11.20  0.78 21.60
> x[x>0 & x<10]
[1] 5.40 0.78
> notas <- c(1,2,0.75,0.5,1.75,1,0.75,1,0.5)
> teste <- c("I1","I2","Iac","II1","II2","IIac",
+ "III1","III2","IIIac")
> teste[notas>1]
[1] "I2" "II2"
```

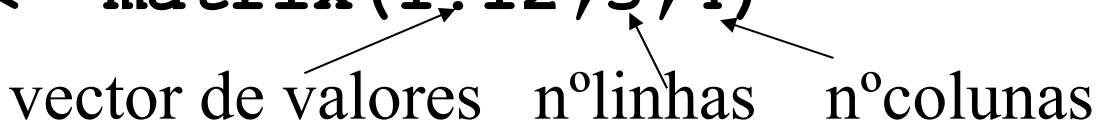
Matrix

- Uma matriz é uma colecção de dados, todos do mesmo tipo, referenciados por dois índices. É uma generalização para duas dimensões de um vector.
- Uma matriz é definida pelo número de linhas (n), número de colunas (m) e um conjunto de ($n \times m$) valores.

Matrix (cont.)

- A função `matrix()` serve para criar matrizes.

```
> M <- matrix(1:12,3,4)
```




```
> M
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

- A opção `byrow=TRUE` faz com que os elementos sejam dispostos por linha.

```
> M <- matrix(1:12,3,4,byrow=TRUE)
```

Matrix (cont.)

- Um objecto do tipo *matrix* tem associado o atributo `dim` que indica a sua dimensão, i.e., nº de linhas e nº de colunas.

```
> dim(M) # ver a dimensão de M
[1] 3 4
```

- Um vector pode ser transformado numa matriz.

```
> A <- c(3,2,-4,0,-1,8)  # A é um vector
> dim(A) <- c(3,2) # transforma A numa
                   matriz com 3 linhas e 2
                   colunas
> A
     [,1] [,2]
[1,]    3    0
[2,]    2   -1
[3,]   -4    8
```

Matrix (cont.)

- Uma matriz pode ser transformada num vector.

```
> A
     [,1] [,2]
[1,]    3    0
[2,]    2   -1
[3,]   -4    8

> as.vector(A) # devolve um vector cujos
                 elementos são os da
                 matriz

[1]  3  2 -4  0 -1  8
```

Matrix (cont.)

- Pode criar-se uma matriz por concatenação de vectores e/ou matrizes:
  - A função `cbind` faz a concatenação dos objectos por coluna

```
> u <- 1:3 ; v <- 4:6
> U <- cbind(u,v)
```

  - A função `rbind` faz a concatenação dos objectos por linha

```
> V <- rbind(u,v)
```

Matrix (indexação e selecção de componentes)

- `M[i,j]` é o elemento de `M` que está na linha `i` e na coluna `j`

```
> M[3,1]
```

- `M[i,]` é a linha `i` e `M[,j]` é a coluna `j` da matriz `M`. Estes objectos são do tipo *vector*

```
> M[2,]
> M[,1]
```

- Para seleccionar parte de uma linha/coluna indicam-se os índices correspondentes

```
> M[c(1,3),1]  # igual a M[-2,1]
```

- Se a selecção incluir mais do que uma linha e mais do que uma coluna, o resultado é uma submatriz

```
> M[2:3,c(1,3,4)]
```

List

- Uma lista é uma colecção ordenada de objectos, que podem ser de tipos diferentes (vectores numéricos, vectores lógicos, matrizes, listas, funções, ...).
- Vamos chamar componente a cada um dos objectos da lista.
- As componentes são numeradas e podem ter um nome associado.

```
> estudante<-list(nr=12345,nome="José Silva",
+ notas=c(2.5,3.4,2.1,4.3),"Arq P")
```

- O resultado de muitas funções é uma lista.

List (selecção de componentes)

- As componentes de uma lista são indexadas e podem ser referidas pelo seu índice, utilizando parêntesis rectos duplos; se tiverem nome, também podem ser referidas pelo nome.

```
> estudante[[2]]
> estudante$nome
```

- Se forem utililizados parentesis rectos simples, o resultado é uma sub-lista formada pelas componentes referidas.

```
> estudante[2:3]
```

Data frame

- Uma *data frame* é semelhante a uma matriz em que as colunas podem ser de tipos diferentes.

```
> resultados <- data.frame(numA=12345:12350,
+ turma=c(1,1,2,2,3,3),nota=c(10.1,8.2,12.5,
+ 13.1,9.7,7.3))
```

- As *data frames* são casos especiais de listas em que as componentes têm o mesmo nº de elementos.

Data frame (selecção de componentes)

- Os elementos de uma *data frame* podem ser referidos (como numa matriz) indicando a linha e a coluna:

```
> resultados[3,1]
> resultados[,2]
```

- A notação `nomeDataFrame$nomeComponente` (como numa lista) também pode ser usada:

```
> resultados$turma
> resultados[[2]]
```

Data frame (função `attach`)

- As consultas a *data frames* podem ser simplificadas utilizando a função `attach()`
- A função `attach()` permite aceder directamente às colunas de uma *data frame*, sem necessidade de referir o nome da *data frame*.

```
> turma
Error: object "turma" not found
> attach(resultados)
> turma
[1] 1 1 2 2 3 3
> detach(resultados)
> turma
Error: object "turma" not found
```

Leitura de ficheiros

- Uma tabela de valores, registada num ficheiro de texto (ASCII), pode ser atribuída a uma *data frame*, através da função

```
read.table(ficheiro,header=FALSE,sep="", dec =
  ".",row.names, col.names, as.is = FALSE,
  na.strings = "NA")
```

em que,

`ficheiro` é o nome do ficheiro de dados; se este não se encontrar na pasta de trabalho, é necessário indicar o endereço completo (utilizando / ou \\ como separador de pastas);

`header` é uma variável lógica que indica se o ficheiro tem ou não os nomes das variáveis na primeira linha. Por omissão o valor é FALSE;

`sep` é o caracter que separa os valores em cada linha; por omissão é `""` que corresponde a um ou mais espaços em branco;

Leitura de ficheiros (cont.)

`dec` é o caracter utilizado como separador decimal. Por omissão é o ponto;

`row.names` é um vector com os nomes das linhas. Por omissão `1,2,3,...`;

`col.names` é um vector com os nomes das colunas. Por omissão `V1,V2,V3,...`;

`as.is` é uma variável lógica que controla a conversão das variáveis alfanuméricas em factores (objectos do R para variáveis qualitativas – vectores de categorias). Se o seu valor é TRUE a leitura mantém o tipo original dos dados;

`na.strings` é o valor usado para os dados desconhecidos no ficheiro e que será convertido em `NA`.

Leitura de ficheiros (exemplo)

- O ficheiro de texto (com valores separados por vírgulas e com cabeçalho) \\prunus\home\cadeiras\Estatistica\EstacMeteo.csv contem dados referentes às estações meteorológicas automáticas do Instituto de Meteorologia de Portugal.
  Criar a *data frame* `estac.meteo` com o conteúdo deste ficheiro.

```
> estac.meteo <- read.table("//prunus/home/
+ cadeiras/Estatistica/EstacMeteo.csv",
+ header=TRUE, as.is=TRUE,sep=",")
```

Escrita em ficheiros

- Para escrever num ficheiro o conteúdo de uma *data frame* ou de uma matriz usa-se a função

```
write.table(x,file="",quote=TRUE,sep=" ",
na="NA",dec=".",row.names=TRUE,col.names=TRUE)
```

em que,

`x` é o nome do objecto a guardar;

`file` é o nome do ficheiro, incluindo o caminho de pastas;

`quote` é uma variável lógica que indica se as variáveis alfanuméricas são ou não escritas entre aspas;

`sep` é o caracter que separa os valores em cada linha; por omissão é um espaço em branco;

Escrita em ficheiros (cont.)

`na` é a representação no ficheiro dos NA's de `x`;

`dec` é o caracter utilizado como separador decimal. Por omissão é o ponto;

`row.names`, **`col.names`** ou são variáveis lógicas que indicam se o ficheiro vai ou não conter os nomes das linhas/colunas de `x`, ou são vectores alfanuméricos com os nomes a atribuir às linhas/colunas no ficheiro;

- Exemplo: Guardar no ficheiro "resultados.csv" a *data frame* `resultados`, utilizando como separador de campos a vírgula.

```
> write.table(resultados,"resultados.csv",
+ row.names=FALSE,sep=",")
```

Indicadores numéricos no R

- `mean(x)` devolve a média dos elementos do vector `x`
- `median(x)` devolve a mediana dos elementos do vector `x`
- `quantile(x,probs=p)` devolve o quantil de ordem `p` dos elementos do vector `x`; por omissão `p=seq(0, 1, 0.25)`, isto é, devolve os extremos e os quartis de `x`
- `max(x)/min(x)` devolve o máximo/mínimo dos elementos do vector `x`
- `range(x)` devolve o vector `c(min(x),max(x))`
- `IQR(x)` devolve a amplitude inter-quartil dos elementos do vector `x`

Indicadores numéricos no R (cont.)

- `var(x)` devolve a variância dos elementos do vector `x`; é igual a `sum((x-mean(x))^2)/(length(x)-1)`
- `sd(x)` devolve o desvio padrão dos elementos do vector `x`; é igual a `sqrt(var(x))`
- `summary(x)` devolve os extremos, os quartis e a média do vector `x`

Funções e valores NA

- Muitas das funções estatísticas anteriores podem ser usadas com um argumento adicional, na seguinte forma

```
função(x, na.rm=FALSE)
```

em que `na.rm` é uma variável lógica que indica se os valores desconhecidos devem ser ou não ignorados.

Caso o vector `x` inclua elementos `NA`,

- se `na.rm=FALSE` (o que ocorre por omissão) a função devolve `NA` ou uma mensagem de erro,
- se `na.rm=TRUE` os valores `NA` são ignorados.

```
> y <- c(NA, 53, 31, 15, 62)
> mean(y)
> mean(y,na.rm=TRUE)
```