

T2 – Lectura de datos

Leitura e Manipulação de Dados

Entrada de Dados Diretamente no R

Função “c()” (concatenate ou combine)

As funções de criação de vetores já foram detalhadas na seção anterior. Basta lembrar aqui que todas elas são usadas para entrar diretamente dados em vetores no R:

```
meu.vetor <- c(4.3,8.9,18.2,6.5)
meu.vetor
```

```
## [1] 4.3 8.9 18.2 6.5
```

```
vetor.vazio <- c()
vetor.vazio
```

```
## NULL
```

Função “matrix()”

A função matrix cria uma matriz com os valores do argumento data. O números de linhas e colunas são definidos pelos argumentos nrow e ncol:

```
minha.matriz <- matrix(data=1:9,nrow=3,ncol=3)
minha.matriz
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

Como o default do argumento data é NA, se ele é omitido o resultado é uma matriz vazia:

```
minha.vazia <- matrix(nrow=3,ncol=3)
minha.vazia
```

```
##      [,1] [,2] [,3]
## [1,]   NA   NA   NA
## [2,]   NA   NA   NA
## [3,]   NA   NA   NA
```

Também por default, os valores são preenchidos por coluna. Para preencher por linha basta o alterar o argumento byrow para TRUE:

```
minha.matriz.lin <- matrix(data=1:12,nrow=3,ncol=4,byrow=T)
minha.matriz.lin
```

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

Note que: (byrow=T) se for FALSE (the default) a matriz é preenchida por colunas, caso contrário a matriz é preenchida por linhas.

```
minha.matriz2.Col <- matrix(data=1:12,nrow=3,ncol=4,byrow=F)
minha.matriz2.Col
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
```

```
## [2,] 2 5 8 11
## [3,] 3 6 9 12
```

Se o argumento data tem menos elementos do que a matriz, eles são repetidos até preenchê-la:

```
elementos <- matrix(c("ar","água","terra","fogo","Leeloo"),ncol=4,nrow=4)
```

```
## Warning in matrix(c("ar", "água", "terra", "fogo", "Leeloo"), ncol = 4, :
## comprimento dos dados [5] não é um submúltiplo ou múltiplo do número de
## linhas [4]
```

```
elementos
```

```
##      [,1] [,2] [,3] [,4]
## [1,] "ar"  "Leeloo" "fogo" "terra"
## [2,] "água" "ar"    "Leeloo" "fogo"
## [3,] "terra" "água" "ar"    "Leeloo"
## [4,] "fogo" "terra" "água" "ar"
```

```
elementos <- matrix(c("fogo","água"),ncol=4,nrow=4)
elementos
```

```
##      [,1] [,2] [,3] [,4]
## [1,] "fogo" "fogo" "fogo" "fogo"
## [2,] "água" "água" "água" "água"
## [3,] "fogo" "fogo" "fogo" "fogo"
## [4,] "água" "água" "água" "água"
```

Função “data.frame()”

Com a função data.frame reunimos vetores de mesmo comprimento em um só objeto:

```
nomes <- c("Beethoven","José de San Martin","Helena Blavatsky","Ruy Barbosa")
ano.nasc <- c(1770,1778,1891,1849)
vive <- c("F","F","F","F")
Personalidades <- data.frame(nomes,ano.nasc,vive)
Personalidades
```

```
##              nomes ano.nasc vive
## 1      Beethoven    1770     F
## 2 José de San Martin    1778     F
## 3   Helena Blavatsky    1891     F
## 4     Ruy Barbosa    1849     F
```

O mesmo, em um só comando:

```
Personalidades.exemplo2 <- data.frame(nomes=c("Beethoven","José de San Martins","Helena Blavasky","Ruy B
Personalidades.exemplo2
```

```
##              nomes ano.nasc vive
## 1      Beethoven    1770     F
## 2 José de San Martins    1778     F
## 3   Helena Blavasky    1891     F
## 4     Ruy Barbosa    1849     F
```

Dados que já Estão em Arquivos

Antes vou até a pasta onde os arquivos estão.

```
#setwd("Documentos/Cursos/SAR-PolSAR-Course/Code/")
```

Leitura e Exportação de Arquivos-Texto: “read.table()” e “write.table()”

Para conjuntos de dados grandes, é mais prático gerar um arquivo de texto (ASCII) a partir de uma planilha ou banco de dados, e usar a função read.table para ler os dados para um objeto no R.

Para criar um objeto com os dados do arquivo gbmam93.csv (apagar extensão .pdf), por exemplo, digitamos:

```
gbmam93 <- read.table(file="gbmam93.csv",header=T,row.names=1,sep=",")
gbmam93
```

```
##      a b c d e f g h i j k l m n o p q r s
## 1  1 1 1 0 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1
## 2  1 1 0 1 1 0 1 0 1 0 0 1 1 0 1 1 1 1 1
## 3  1 0 0 1 1 0 1 0 1 0 0 1 1 1 1 1 1 1 1
## 4  1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1
## 5  1 0 0 1 1 0 1 0 1 0 0 1 1 0 0 0 1 0 0
## 6  1 1 1 0 1 0 1 1 0 0 0 1 1 1 1 1 1 1 1
## 7  1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
## 8  1 0 1 1 1 0 0 0 1 0 0 1 1 0 0 1 1 0 1
## 9  1 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0
## 10 1 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0
## 11 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 1
## 12 1 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1
## 13 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1
## 14 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
```

Com a função read.table podemos carregar arquivos com extensão .txt, .csv, e outros.

O argumento header=T indica que a primeira linha são os nomes das variáveis, assim como row.names=1 indica que a primeira coluna deve ser usada para os nomes das linhas. O argumento sep indica qual é o sinal de separação de cada registro, no caso vírgulas.

Esses e os outros argumentos da função a tornam extremamente flexível para ler dados em arquivos texto. Consulte a ajuda para mais informações, e também para conhecer as variantes read.csv e read.delim.

Para exportar um objeto para um arquivo texto, use a função write.table, que tem a mesma lógica.

Conjuntos de Dados Distribuídos com os Pacotes do R

Muitos pacotes do R incluem conjuntos de dados para exemplos, treinamento e verificação de análises. Se o pacote já está carregado (funções library ou require) todos os seus objetos estão disponíveis, inclusive os objetos de dados. Incluindo as séries temporais de número de peles de lincas caçados no Canadá, analisadas pelo ecólogo Charles Elton obtém-se:

```
lynx
```

```
## Time Series:
## Start = 1821
## End = 1934
## Frequency = 1
##      [1] 269 321 585 871 1475 2821 3928 5943 4950 2577 523 98 184 279
##     [15] 409 2285 2685 3409 1824 409 151 45 68 213 546 1033 2129 2536
##     [29] 957 361 377 225 360 731 1638 2725 2871 2119 684 299 236 245
```

```
## [43] 552 1623 3311 6721 4254 687 255 473 358 784 1594 1676 2251 1426
## [57] 756 299 201 229 469 736 2042 2811 4431 2511 389 73 39 49
## [71] 59 188 377 1292 4031 3495 587 105 153 387 758 1307 3465 6991
## [85] 6313 3794 1836 345 382 808 1388 2713 3800 3091 2985 3790 674 81
## [99] 80 108 229 399 1132 2432 3574 2935 1537 529 485 662 1000 1590
## [113] 2657 3396
```

```
#Time Series:
```

```
Start = 1821
```

```
End = 1934
```

```
Frequency = 1
```

Como qualquer objeto de um pacote, lynx tem um arquivo de ajuda, que é exibido com o comando `help(lynx)` ou `?lynx`:

```
help(lynx)
```

T2 – Lectura de datos PolSAR

Carregar o arquivo `.rdata`, nestes arquivo temos a covariância (C3) da imagem que salvamos, por exemplo San Francisc.

Click no arquivo `AIRSAR_SanFrancisc_Enxu.rdata` que esta dentro da pasta raiz.

```
#read("AIRSAR_SanFrancisc_Enxu.RData")
```