

Introdução ao Programa R: Principais objetos e funcionalidades no R

Gilberto Rodrigues Liska; Josiane Rodrigues; Juliano Bortolini
`gilbertoliska@ufscar.br; josirodrigues@ufscar.br;`
`julianobortolini@ufmt.br`

Material de Apoio



Sumário

1 Principais objetos

- Vetores
- Listas
- Comandos úteis
- Matrizes
- Tabelas

2 Arquivos

3 Funções no R

Principais tipos de objetos no R

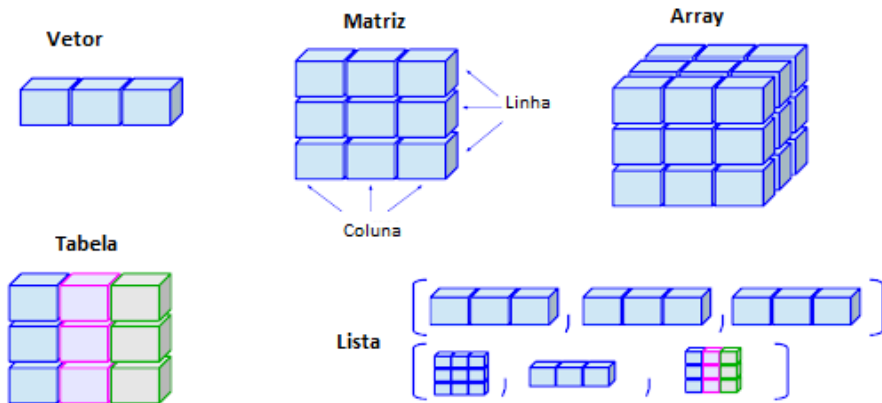


Figura 1: Principais objetos no R.

Vetores

- Em computação é comum manipular conjunto de valores como notas, preços, nomes etc. Para manipular um conjunto de dados de um mesmo tipo (inteiro, real, string) há uma estrutura de dados denominada de vetor;
- Vetores são “agregados homogêneos unidimensionais” que permitem agrupar um conjunto de valores de um mesmo tipo em uma única variável.

Vetor



Vetores

Exemplo 1

- Vamos definir um vetor “**nota**” de tamanho 5 de tipo inteiro, $\text{nota} = (60, 95, 80, 50, 98)$
- **OBS.:** O colchetes é o operador do objeto que está associado ao índice

Vetor



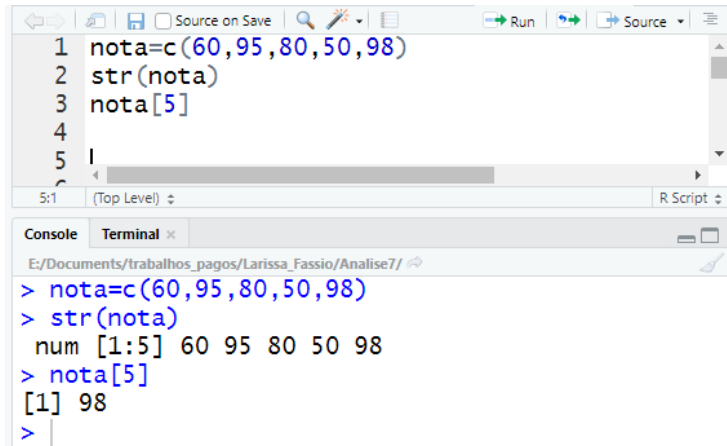
Índice	1	2	3	4	5
Valor	60	95	80	50	98
Vetor	nota[1]	nota[2]	nota[3]	nota[4]	nota[5]

Exemplo 1 no R

```
nota=c(60,95,80,50,98)
nota
str(nota) #ver estrutura do objeto
nota[5] #ver quinto elemento
```

Vetores

- Se quisermos imprimir a nota 50 do vetor **nota**:



The screenshot shows the R Studio interface. The top toolbar includes icons for navigation, saving, and running code. The source editor window contains the following R code:

```
1 nota=c(60,95,80,50,98)
2 str(nota)
3 nota[5]
4
5 |
```

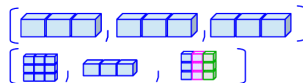
The console window at the bottom shows the output of the code:

```
> nota=c(60,95,80,50,98)
> str(nota)
 num [1:5] 60 95 80 50 98
> nota[5]
[1] 98
> |
```

Figura 2: Telas 1 e 3 no R do exemplo 1.

Listas

- **Lista** é um objeto consistindo de uma coleção ordenada de objetos conhecidos como seus componentes;
- Permite reunir em um só objeto componentes de diversos tipos, como por exemplo, vetor, valores lógicos, matriz, array etc.
- No R o comando para gerar uma lista é o *list*.



Listas

Exemplo 2

Vamos criar um objeto no R com as seguintes informações: Fred e sua esposa Mary tem 3 crianças, com idades 4, 7 e 9, respectivamente.

OBS.: Note que temos informação de número, nomes e um vetor.

Exemplo 2 no R

```
## definindo uma lista
lst=list(nome="Fred",
         esposa="Mary", criancas=3,
         idade=c(4,7,9))

# imprimir a lista
lst
$nome
[1] "Fred"
$esposa
[1] "Mary"
$criancas
[1] 3
$idade
[1] 4 7 9

# tamanho da lista
length(lst)
[1] 4
```


Observações

- Para acessar os elementos da lista pode-se utilizar os operadores `[xx]` ou `$`. Os `xx` representam a posição do elemento, ou elementos, que se deseja visualizar.
- Veja nos exemplos ao lado a diferença entre eles.
- **IMPORTANTE:** O uso desses operadores é bastante útil não somente em listas, mas também em arrays, matrizes e tabelas.
- O que a última linha de comando (`lst[[4]][-1]`) fez?

Exemplo 2 no R (cont.)

```
# ver o segundo objeto da lista
lst[2]
# ver o segundo e terceiro objeto da lista
lst[2:3]
# ver o segundo objeto da lista (diferente)
lst[[2]]
# se o componente for um vetor, para ver o
# segundo elemento desse vetor
lst[[4]][2]

## alternativamente
lst$esposa # ver o componente esposa

lst$idade[2] # semelhante ao lst[[4]][2]

lst[[4]][-1] # o que aconteceu?
```

Alguns comandos úteis

- Existem alguns comandos em R (e certamente em outras linguagens), que permitem adicionar e/ou remover elementos de uma lista, vetor etc.
- Ao adicionar ou remover um elemento, o tamanho da lista também é modificado
- Podemos adicionar um elemento ou vários elementos.
- Podemos ordenar os elementos de um vetor em ordem crescente ou decrescente automaticamente.
- Essas situações são resolvidas pelos comandos ***append*** e ***sort*** no R.



Fonte: <https://pixabay.com/images/id-2852153/>

Alguns comandos úteis

APPEND

- Se quisermos adicionar um elemento após o último valor do vetor `nota`
`= (60, 95, 80, 50, 98)`

Índice	1	2	3	4	5	6
Valor	60	95	80	50	98	20
Vetor	<code>nota[1]</code>	<code>nota[2]</code>	<code>nota[3]</code>	<code>nota[4]</code>	<code>nota[5]</code>	<code>nota[6]</code>

No R:

```
?append # pedindo ajuda
nota1=append(nota,20)
nota1
[1] 60 95 80 50 98 20
```

DICA

Veja o *help* da função ***append*** para mais esclarecimentos.

Alguns comandos úteis

APPEND

- Se quisermos adicionar um elemento após o primeiro valor do vetor
 $\text{nota} = (60, 95, 80, 50, 98)$

Índice	1	2	3	4	5	6
Valor	60	20	95	80	50	98
Vetor	<code>nota[1]</code>	<code>nota[2]</code>	<code>nota[3]</code>	<code>nota[4]</code>	<code>nota[5]</code>	<code>nota[6]</code>

No R:

```
nota2=append(nota,20,1)
nota2
[1] 60 20 95 80 50 98
```

Aguns comandos úteis

SORT

- Se quisermos organizar o vetor $\text{nota} = (60, 95, 80, 50, 98)$ em ordem crescente.

Original

Índice	1	2	3	4	5
Valor	60	95	80	50	98
Vetor	<code>nota[1]</code>	<code>nota[2]</code>	<code>nota[3]</code>	<code>nota[4]</code>	<code>nota[5]</code>

Ordenado em ordem crescente

Índice	1	2	3	4	5
Valor	50	60	80	95	98
Vetor	<code>nota[1]</code>	<code>nota[2]</code>	<code>nota[3]</code>	<code>nota[4]</code>	<code>nota[5]</code>

No R:

```
nota
[1] 60 95 80 50 98
nota4=sort(nota); nota4
[1] 50 60 80 95 98
```

Aguns comandos úteis

SORT

- Se quisermos organizar o vetor `nota = (60, 95, 80, 50, 98)` em ordem **decrecente**.

Original

Índice	1	2	3	4	5
Valor	60	95	80	50	98
Vetor	<code>nota[1]</code>	<code>nota[2]</code>	<code>nota[3]</code>	<code>nota[4]</code>	<code>nota[5]</code>

Ordenado em ordem decrecente

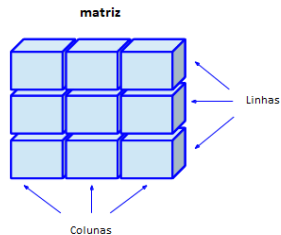
Índice	1	2	3	4	5
Valor	98	95	80	60	50
Vetor	<code>nota[1]</code>	<code>nota[2]</code>	<code>nota[3]</code>	<code>nota[4]</code>	<code>nota[5]</code>

No R:

```
nota
[1] 60 95 80 50 98
nota5=sort(nota, decreasing = TRUE); nota5
[1] 98 95 80 60 50
```

Matrizes

- Matrizes são equivalentes a vetores, contudo permitem a utilização de diversas dimensões acessadas via diferentes índices.
- Pode ser pensada como um vetor onde cada célula é outro vetor, recursivamente.
- Em diversas situações matrizes são necessárias para correlacionar informações.



Matrizes

Exemplo 3

Assumindo que uma turma tem três alunos e fossem anotadas as notas de 5 avaliações, seria necessária uma matriz bidimensional para guardar as notas de todos os alunos dessa turma.

Tabela 1: Dados do Exemplo.

Aluno	Avaliações				
	1	2	3	4	5
1	5	4.5	7	5.2	6.1
2	2.1	6.5	8	7	6.7
3	8.6	7	9.1	8.7	9.3

Matrizes - Exemplo 3 (cont.)

- Em notação matricial, os dados do exemplo 3 são dados por:

$$\begin{bmatrix} 5 & 4.5 & 7 & 5.2 & 6.1 \\ 2.1 & 6.5 & 8 & 7 & 6.7 \\ 8.6 & 7 & 9.1 & 8.7 & 9.3 \end{bmatrix}$$

que é uma matriz de 3 linhas e 5 colunas.

- No R utilizaremos o comando **matrix**, que tem como argumentos:
 - o vetor de dados
 - byrow=TRUE** significa que o preenchimento na matriz é sentido linha
 - nrow** e **ncol** são o número de linhas e colunas.

Exemplo 3 no R (cont.)

```
turma=matrix(c(5,4.5,7,5.2,6.1,
                2.1,6.5,8,7,6.7,
                8.6,7,9.1,8.7,9.3),
              byrow=TRUE,
              nrow=3,ncol=5)
```

```
turma
      [,1] [,2] [,3] [,4] [,5]
[1,]  5.0  4.5  7.0  5.2  6.1
[2,]  2.1  6.5  8.0  7.0  6.7
[3,]  8.6  7.0  9.1  8.7  9.3
```

```
# se quiser colocar os "nomes" nas
# linhas e colunas
rownames(turma)=c(1,2,3)
colnames(turma)=c(1,2,3,4,5)
```

Matrizes - Exemplo 3 (cont.)

- Com os dados do exemplo anterior, vamos calcular a média aritmética simples das notas dos alunos da turma.
- As notas configuram uma variável x do tipo numérica com índices i e j . Assim, x_{ij} refere-se a nota j do aluno i . Por exemplo, x_{23} refere-se a 3ª nota do segundo aluno.

Aluno	Nota				
	1	2	3	4	5
1	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
2	x_{21}	x_{22}	x_{23}	x_{24}	x_{25}
3	x_{31}	x_{32}	x_{33}	x_{34}	x_{35}

A média é dada pela seguinte fórmula

$$\text{Média} = \frac{x_{11} + x_{12} + x_{13} + \cdots + x_{34} + x_{35}}{15} \quad (1)$$

Matrizes - Exemplo 3 (cont.)

- Precisamos, primeiramente, somar os elementos. Como os dados estão dispostos em matriz, podemos utilizar a *estrutura de repetição FOR* para varrer as linhas e colunas da matriz.
- Além disso, precisaremos de um objeto, o qual será somado a cada iteração do **FOR**. Chamaremos esse objeto de **soma** e seu valor inicial é zero.
- Copie e cole os códigos a seguir no seu script.

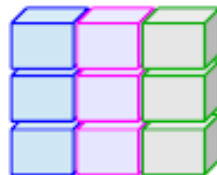
Exemplo 3 no R (cont.)

```
nl=dim(turma)[1] #numero de linhas
nc=dim(turma)[2] #numero de colunas
soma=0 #objeto com que receberá a soma
# for para percorrer as linhas (alunos)
for(i in 1:nl){
  #for para percorrer as colunas (notas)
  for(j in 1:nc){
    soma=soma+turma[i,j]
  }
}
soma
media=soma/(nl*nc)
media
```

Tabelas (Data frames)

- Uma tabela é um objeto que contém múltiplos vetores que são do mesmo tamanho. Esses vetores podem ser do tipo numérico, lógico ou de caracteres.
- É semelhante a uma matriz, com a vantagem de permitir a inclusão de caracteres não numéricos.
- É também semelhante a uma planilha ou base de dados e é útil para representar dados experimentais.
- no R o comando para criar uma tabela é o **data.frame**.

Tabela



Tabelas

Exemplo 4

- Um questionário foi aplicado aos cinco funcionários do setor administrativo de uma empresa fornecendo os dados apresentados na tabela 2.

Tabela 2: Respostas dos funcionários do setor administrativo da empresa ao questionário.

Funcionário	Curso	Idade	Salário (R\$)	Anos de empresa	Sexo
Lisa	superior	34	1100,00	5	F
Godofredo	superior	43	1450,00	8	M
João	fundamental	21	450,00	3	M
Joana	médio	37	960,00	8	F
Alba	médio	25	600,00	2	F

Tabelas - Exemplo 4 (cont.)

- Precisamos, primeiramente, criar os vetores que receberão cada coluna da tabela 2.
- **IMPORTANTE:** Não esquecer que os nomes ou letras devem estar entre aspas.

Exemplo 4 no R

```
nome=c("Lisa","Godofredo","João","Joana","Alba")
curso=c("superior","superior","fundamental","médio","médio")
idade=c(34,43,21,37,25)
salario=c(1100,1450,450,960,600)
anos=c(5,8,3,8,2)
sexo=c("F","M","M","F","F")
df=data.frame(nome,curso,idade,salario,anos,sexo,
               stringsAsFactors = FALSE)
```

```
df
```

	nome	curso	idade	salario	anos	sexo
1	Lisa	superior	34	1100	5	F
2	Godofredo	superior	43	1450	8	M
3	João	fundamental	21	450	3	M
4	Joana	médio	37	960	8	F
5	Alba	médio	25	600	2	F

Observações

- Os operadores [xx] ou \$, apresentados nas **Listas**, têm a mesma funcionalidade nas tabelas.
- Caso queiramos subdividir a tabela, o comando **subset** no R apresenta uma forma interessante de fazer isso. Seus argumentos são:
 - o objeto a ser subdividido
 - subset**: coluna e forma de subdivisão
 - select**: se indicada, retorna as colunas que satisfazem a subdivisão.
- Copie e cole os códigos no seu *script* e veja a funcionalidade do comando **subset**.

Exemplo 4 no R (cont.)

```
# subdividindo o dataframe  
df$curso #se quiser apenas uma coluna
```

```
# com o comando subset  
?subset #pedindo ajuda  
subset(df, curso=="médio")  
subset(df, anos>=3)  
subset(df, sexo=="M")  
subset(df, sexo=="F" & anos>2)  
subset(df, anos>2,  
       select=c(curso,nome))  
subset(df, sexo=="F",  
       select=c(curso,nome))  
subset(df, select=nome:anos)  
subset(df, select=-nome)
```

```
# armazenando num novo objeto  
# para ser usado  
df2=subset(df, select=-nome); df2
```

Trabalhando com matrizes e tabelas

- Voltando ao Exemplo 3, se quiséssemos calcular a média das notas de cada aluno, como deveríamos proceder?
- Uma alternativa seria utilizar a estrutura de repetição **FOR** em cada linha da matriz.
- Uma outra alternativa seria usar funções prontas no R que facilitam a manipulação de matrizes e tabelas, como é o caso das funções da família **apply** (*apply*, *tapply*, *sapply*, ...).
- Vamos mostrar como as funções *apply* e *tapply* podem ser úteis nos exemplos 3 e 4.



Fonte: <https://pixabay.com/images/id-294525/>

Trabalhando com matrizes e tabelas

APPLY

Aplica uma função (soma, máximo, média etc.) às linhas ou colunas de uma matriz ou array. Seus argumentos são:

- X: uma matriz (pode ser array);
- MARGIN: pode ser 1 ou 2. Se 1, a operação é feita nas linhas e 2 a operação é feita nas colunas;
- FUN: a função a ser utilizada.

Exemplo 5

Utilizando a matriz do **exemplo 3**, calcule a média da nota de cada aluno. Utilize a equação **1**, lembrando que deve-se somar apenas cinco notas e dividir por cinco.

Exemplo 5 (a) no R

```
## media para cada aluno
nl=dim(turma)[1] #numero de linhas
nc=dim(turma)[2] #numero de colunas
#objeto com que receberá a soma
soma=c(0,0,0)
# for para percorrer as linhas (alunos)
for(i in 1:nl){
  #for para percorrer as colunas (notas)
  for(j in 1:nc){
    soma[i]=soma[i]+turma[i,j]
  }
}
soma #devemos ter 3 valores
media=soma/(nc)
media
```

Exemplo 5 (b) no R

```
#ou simplesmente
apply(turma, 1, mean)
      1      2      3
5.56 6.06 8.54
```

Trabalhando com matrizes e tabelas

TAPPLY

Aplica uma função a cada grupo (não vazio) de valores fornecidos por uma combinação exclusiva dos níveis de certos fatores. Seus argumentos são:

- X: um objeto que pode ser subdividido, como um vetor.
- INDEX: um fator com mesmo comprimento de X.
- FUN: a função a ser aplicada aos elementos de X.

Exemplo 6

Utilizando a tabela **2** do **exemplo 4**, calcule a média salarial dos funcionários nos diferentes sexos. Utilize a equação **1**, lembrando que deve-se somar a quantidade de salários (que varia de acordo com o sexo) e dividir pelo número de pessoas de cada sexo.

Exemplo 6 no R

```
df #a tabela criada no exemplo 4
```

```
?tapply #pedindo ajuda da função  
tapply(salario, sexo, mean)
```

```
      F      M  
886.6667 950.0000
```

Sumário

1 Principais objetos

2 Arquivos

- Diretório
- read.table
- Calc
- Excel

3 Funções no R

Arquivos

- Até o momento trabalhamos com vetores, matrizes, tabelas e conjuntos de dados pequenos.
- É muito comum trabalhar com conjunto de informações grandes (grandes em extensão e/ou grandes em tamanho - bytes).
- Nesses casos é viável que esses arquivos externos sejam “chamados” para dentro de um programa, no nosso caso o R, ao invés de digitá-los dentro do programa.
- É recomendado que os dados estejam organizados como uma tabela (data.frame).



Fonte: <https://pixabay.com/images/id-28741/>

Arquivos - Exemplo

Exemplo 7

- Vamos utilizar o conjunto de dados que traz informações do sexo, idade, peso, altura, renda e número de faltas de 40 alunos de uma turma de uma universidade.
- Digite o conjunto de dados da tabela 3 em uma planilha.
- Abra o Bloco de Notas, copie os dados da planilha e cole no bloco de notas e salve com o nome “**Tab1**”.

Tab1 - Bloco de Notas

Arquivo	Editar	Formatar	Exibir	Ajuda		
Id	Sexo	Idade	Peso	Altura	Renda	Faltas
1	1	23	62	1.610	1	1
2	0	24	57	1.624	2	2
3	1	20	73	1.647	2	0
4	1	20	80	1.656	3	1
5	0	18	70	1.677	2	2
6	0	19	61	1.692	2	3
7	1	23	89	1.698	2	0
8	1	21	64	1.713	2	1
9	1	21	65	1.716	1	2
10	1	22	71	1.717	1	1
11	0	20	73	1.731	1	0
12	1	22	71	1.749	1	2
13	1	26	70	1.750	1	2
14	0	22	62	1.752	1	0
15	0	20	67	1.753	2	5
16	0	21	68	1.758	3	2
17	1	24	61	1.785	2	1

Figura 3: Bloco de notas do arquivo Tab1.

- Esse arquivo também está disponível em pasta virtual (POCA_R) por meio do link <https://1drv.ms/u/s!AvxsaQZPoPWdrxYOYHC1wUnvT2gW?e=qBws6w>

Dados do exemplo 7

Tabela 3: Informação do sexo (M=0,F=1), idade, peso, altura e renda (B=1, M=2 e A=3) e número de faltas de 40 alunos de uma turma de uma universidade.

Id	Sexo	Idade(anos)	Peso(Kg)	Altura(m)	Renda	Faltas	Id	Sexo	Idade(anos)	Peso(Kg)	Altura(m)	Renda	Faltas
1	1	23	62	1.610	1	1	21	0	18	54	1.803	2	1
2	0	24	57	1.624	2	2	22	0	19	65	1.811	3	2
3	1	20	73	1.647	2	0	23	1	19	75	1.816	2	0
4	1	20	80	1.656	3	1	24	1	26	74	1.826	3	1
5	0	18	70	1.677	2	2	25	0	21	66	1.827	1	2
6	0	19	61	1.692	2	3	26	0	19	67	1.828	2	3
7	1	23	89	1.698	2	0	27	0	22	83	1.829	1	0
8	1	21	64	1.713	2	1	28	0	18	84	1.841	3	1
9	1	21	65	1.716	1	2	29	1	25	72	1.842	2	2
10	1	22	71	1.717	1	1	30	1	24	74	1.853	1	1
11	0	20	73	1.731	1	0	31	0	26	66	1.861	2	0
12	1	22	71	1.749	1	2	32	0	23	70	1.887	2	2
13	1	26	70	1.750	1	2	33	0	19	72	1.889	3	2
14	0	22	62	1.752	1	0	34	1	23	73	1.891	1	0
15	0	20	67	1.753	2	5	35	1	26	86	1.898	2	5
16	0	21	68	1.758	3	2	36	0	27	71	1.904	3	2
17	1	24	61	1.785	2	1	37	0	27	77	1.915	2	1
18	0	19	68	1.786	2	3	38	1	18	57	1.921	2	3
19	0	23	59	1.799	2	2	39	1	24	67	1.929	2	2
20	1	19	66	1.802	3	3	40	0	25	73	1.977	2	3

Mudando diretório

Antes de chamar um arquivo externo no R, devemos mostrar para o R onde está o arquivo. Isso pode ser feito de duas formas:

Manualmente

Menu *Session* ⇒ *Set Working Directory* ⇒ *Choose directory* ⇒ procure a pasta onde está o arquivo ⇒ Clique em *open*

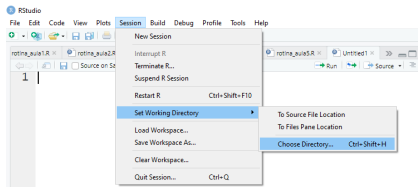


Figura 4: Mudando o diretório manualmente.

Por comando

Utilizando o comando `setwd("caminho de onde está o arquivo")`

No R:

```
# para mudar o diretório  
# por linha de comando  
setwd("Particao/Pasta/Nome do arquivo")
```

Importante: o caminho deve estar entre aspas.

Comando read.table

- Para ler um data frame diretamente, o arquivo externo normalmente apresenta uma forma especial.
- A primeira linha do arquivo geralmente tem um nome para cada variável (coluna) no arquivo de dados.
- Cada linha adicional do arquivo tem (geralmente) um rótulo de linha e os valores para cada variável.

Carregando o arquivo do exemplo 7 no R:

```
# depois de mudar o diretório, use
# o comando read.table
df=read.table("Tab1.txt")
df #imprime o arquivo na tela
```

```
10 # carregar o conjunto de dados
11 df=read.table("Tab1.txt")
12 df
13 |
```

13:1 (Top Level) ↕

Console Terminal x

E:/Documents/UFSCar/Fundamentos_e_Programacao_de_Computadores/ ↗

```
> # carregar o conjunto de dados
> df=read.table("Tab1.txt")
> df
```

	V1	V2	V3	V4	V5	V6	V7
1	Id	Sexo	Idade	Peso	Altura	Renda	Faltas
2	1	1	23	62	1.610	1	1
3	2	0	24	57	1.624	2	2
4	3	1	20	73	1.647	2	0
5	4	1	20	80	1.656	3	1

Figura 5: Arquivo do exemplo 7 carregado. ↻ ↺ ↻

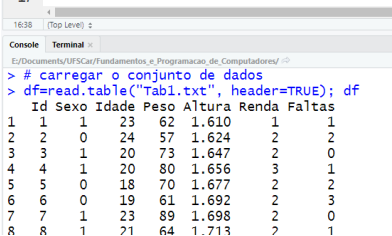
Comando read.table

- Note que aparecem como rótulos de colunas V_1, V_2, \dots, V_7 e na linha com rótulo "1" os nomes das colunas.
- Isso significa que a primeira linha é constituída de texto e as demais são números.
- Caso precisemos fazer operações com os valores de cada variável (que começam na linha com rótulo "2", os resultados podem ser equivocados ou o programa acusar erro.
- Pra contornar isso, basta acrescentar o argumento **header=TRUE** dentro do comando *read.table*.

No R:

```
# carregar o conjunto de dados
df=read.table("Tab1.txt", header=TRUE)
df
# ou simplesmente
df=read.table("Tab1.txt", h=TRUE); df
```

```
13 # carregar o conjunto de dados
14 df=read.table("Tab1.txt", header=TRUE); df
15 # ou simplesmente
16 df=read.table("Tab1.txt", h=TRUE); df
17
```



	Id	Sexo	Idade	Peso	Altura	Renda	Faltas
1	1	1	23	62	1.610	1	1
2	2	0	24	57	1.624	2	2
3	3	1	20	73	1.647	2	0
4	4	1	20	80	1.656	3	1
5	5	0	18	70	1.677	2	2
6	6	0	19	61	1.692	2	3
7	7	1	23	89	1.698	2	0
8	8	1	21	64	1.713	2	1

Figura 6: Arquivo do exemplo 7 carregado.

Lendo arquivo do Calc

- É possível carregar arquivos de dados externos com outras extensões, como csv, xls, xlsx etc.
- No caso de arquivo de dados feito no Calc do LibreOffice, é possível carregá-lo diretamente desse programa, ao invés de copiar o arquivo para um arquivo do tipo *.txt*.
- Para tal, é necessário instalar pacote no R chamado **readODS**.
- Na sequência, é necessário carregar o pacote para usufruir de suas funções.



Fonte: https://upload.wikimedia.org/wikipedia/commons/6/6a/LibreOffice_4.0_Calc_Icon.svg

Lendo arquivo do Calc

Exemplo 8

- Com os dados da tabela 3, digite-os em uma planilha do Calc.
- Esse arquivo também está disponível na pasta virtual (POCA_R), indicada no exemplo 7.
- O nome do arquivo é “Tab1.ods”. Faça o download desse arquivo e coloque-o na mesma pasta que está o arquivo Tab1.txt no seu computador.

Exemplo 8 no R:

```
## para instalar pacote por comando
install.packages("readODS")
library(readODS)# carregar pacote
df4=read_ods("Tab1.ods",sheet=1)
df4
```

```
36 library(readODS)
37
38 df4=read_ods("Tab1.ods",sheet=1)
39 df4 # ver o conjunto de dados
40
41
421 [Top Level] 5
```

```
Console Terminal
E:/Documents/UFSat/Fundamentos_e_Programacao_de_Computadores/
> df4=read_ods("Tab1.ods",sheet=1)
Parsed with column specification:
cols(
  Id = col_double(),
  Sexo = col_double(),
  Idade = col_double(),
  Peso = col_double(),
  Altura = col_double(),
  Renda = col_double(),
  Faltas = col_double()
)
> df4 # ver o conjunto de dados
  Id Sexo Idade Peso  Altura  Renda  Faltas
1  1  1    23   62  1.610    1    1
2  2  0    24   57  1.624    2    2
3  3  1    20   73  1.647    2    0
```

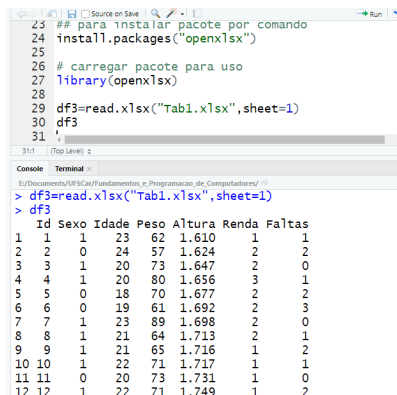
Figura 7: Arquivo do exemplo 7 carregado no Calc.

Lendo arquivo do Excel

- De maneira similar pode ser feito para arquivos feitos no Excel.
- Para tal, é necessário instalar pacote no R chamado **openxlsx**.
- Na sequência, é necessário carregar o pacote para usufruir de suas funções.
- Da mesma forma como foi feito com o Calc, faça no Excel. na pasta compartilhada tem o arquivo "Tab1.xlsx".

No R:

```
install.packages("openxlsx")
library(openxlsx)# carregar pacote
df3=read.xlsx("Tab1.xlsx",sheet=1)
df3
```



```
23 ## para instalar pacote por comando
24 install.packages("openxlsx")
25
26 # carregar pacote para uso
27 library(openxlsx)
28
29 df3=read.xlsx("Tab1.xlsx",sheet=1)
30 df3
31 '
```

```
> df3=read.xlsx("Tab1.xlsx",sheet=1)
> df3
```

	Id	Sexo	Idade	Peso	Altura	Renda	Faltas
1	1	1	23	62	1.610	1	1
2	2	0	24	57	1.624	2	2
3	3	1	20	73	1.647	2	0
4	4	1	20	80	1.656	3	1
5	5	0	18	70	1.677	2	2
6	6	0	19	61	1.692	2	3
7	7	1	23	89	1.698	2	0
8	8	1	21	64	1.713	2	1
9	9	1	21	65	1.716	1	2
10	10	1	22	71	1.717	1	1
11	11	0	20	73	1.731	1	0
12	12	1	22	71	1.749	1	2

Figura 8: Arquivo do exemplo 7 carregado no Excel.

Observações

- Nos dois últimos exemplos, utilizamos o comando ***install.packages*** para instalar dois pacotes. Uma vez executados esses comandos, não é necessário executá-los novamente, mesmo que o computador seja desligado.
- Uma vez instalado um pacote, precisamos carregá-lo para utilizar suas funções. Isso deve ser feito sempre quando o R for fechado, caso contrário os comandos *read_ods* e *read.xlsx* não funcionarão.
- Uma outra forma de instalar um pacote é por meio do menu **Packages** na tela 4 do R, clicando em *Install*, digitar o nome do pacote e em seguida em *Install* (Figura 9).

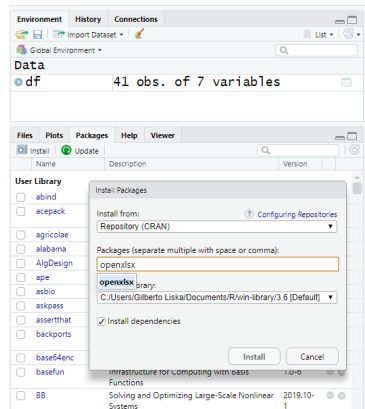


Figura 9: Instalação manual de pacotes no R.

Sumário

1 Principais objetos

2 Arquivos

3 Funções no R

Funcionalidades no R

Função SUM

Somar coisas é uma tarefa bastante comum. No R temos uma função pronta para fazer isso, que é a função **sum**. Vamos ilustrar essa função em exemplos de técnicas de somatório. O somatório é indicado pela letra grega sigma maiúscula (Σ). A figura 10 mostra seus principais elementos.

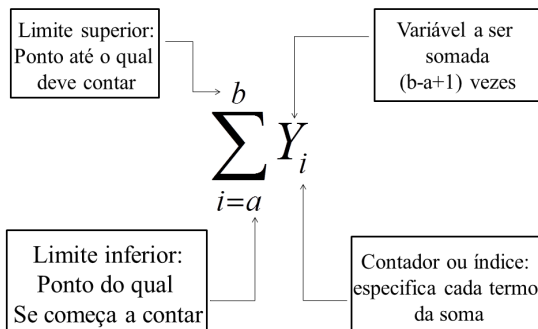


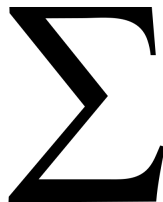
Figura 10: Esquema ilustrativo dos elementos típicos de um somatório.

Técnicas de Somatório

Teorema

Considere a , b e k constantes e X e Y variáveis. Então as seguintes propriedades envolvendo somatório são válidas:

- (i) $\sum_{i=1}^n aX_i = a \sum_{i=1}^n X_i$
- (ii) $\sum_{i=1}^n X_i Y_i \neq \sum_{i=1}^n X_i \sum_{i=1}^n Y_i$
- (iii) $\sum_{i=1}^n (aX_i + bY_i) = a \sum_{i=1}^n X_i + b \sum_{i=1}^n Y_i.$
- (iv) $\sum_{i=1}^n k = nk$
- (v) $\sum_{i=1}^n (X_i - \bar{X}) = 0$, em que $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i.$



Fonte: https://commons.wikimedia.org/wiki/File:Greek_uc_sigma.svg

Técnicas de Somatório

Com as técnicas anteriores, vamos fazer os seguintes exemplos no R.

Exemplo 9

Sejam os conjuntos $X = 2, 4, 4, 3, 2$ e $Y = 1, 2, 3, 6, 7$. Obtenha:

(a) $\sum_{i=1}^4 X_i$

(b) $\sum_{i=1}^5 Y_i$

(c) $\sum_{i=1}^4 4X_i^2$

(d) $\sum_{i=1}^5 X_i Y_i$

(e) $\left(\sum_{i=1}^5 X_i\right) \left(\sum_{i=1}^5 Y_i\right)$

(f) $\sum_{i=1}^5 (3X_i + 2Y_i)$

(g) $\sum_{i=2}^4 X_i Y_i + \sum_{i=1}^5 Y_i^2$

(h) $\sum_{i=1}^5 2$

Técnicas de Somatório

Copie e cole os códigos abaixo no seu script. Veja o que acontece.

Exemplo 9 no R

```
# Exemplo
X=c(2,4,4,3,2)
Y=c(1,2,3,6,7)

?sum #ajuda da funcao

# a)
sum(X[1:4])

# b)
sum(Y)

# c)
sum(4*X[1:4]^2)

# d)
sum(X*Y)

# e)
sum(X)*sum(Y)
```

Exemplo 9 no R

```
# f)
sum(3*X+2*Y)

# g)
sum(X[2:4]*Y[2:4])+sum(Y^2)

# h)
sum(rep(2,5))

## caso tivesse observacao perdida
## ou vazia
# colocando um observacao vazia
# como NA
W=c(1:5,NA)
W

sum(W)
sum(W, na.rm=TRUE)
```

Técnicas de Somatório

Exemplo 9 - Respostas

Sejam os conjuntos $X = 2, 4, 4, 3, 2$ e $Y = 1, 2, 3, 6, 7$. Obtenha:

$$(a) \sum_{i=1}^4 X_i = 13$$

$$(b) \sum_{i=1}^5 Y_i = 19$$

$$(c) \sum_{i=1}^4 4X_i^2 = 180$$

$$(d) \sum_{i=1}^5 X_i Y_i = 54$$

$$(e) \left(\sum_{i=1}^5 X_i \right) \left(\sum_{i=1}^5 Y_i \right) = 285$$

$$(f) \sum_{i=1}^5 (3X_i + 2Y_i) = 83$$

$$(g) \sum_{i=2}^4 X_i Y_i + \sum_{i=1}^5 Y_i^2 = 137$$

$$(h) \sum_{i=1}^5 2 = 10$$

Funcionalidades no R

Funções

- O R nos possibilita criar nossas próprias funções, que são genuinamente funções R, sendo armazenadas internamente de uma forma especial e podendo ser utilizadas em novas expressões.
- Desta forma a linguagem ganha grande poder, conveniência e elegância. O aprendizado em escrever funções úteis é uma das muitas maneiras de fazer com que o uso do R seja confortável e produtivo.
- A sintaxe geral de uma função é dada por:

```
nome <- function(arg 1, arg 2, ...){  
  expressão  
}
```

em que **expressão** é, normalmente, um grupo de comandos e **nome** é o objeto que receberá a função. Sua chamada se dará por um comando *nome(a1, a2, ...)*, em que *a1, a2* etc. são os valores que deverão ser passados como argumentos dos objetos (*arg 1, arg 2, ...*).

Funcionalidades no R

Exemplo 10

- Vamos criar uma função que faça a mesma coisa que o comando **sum** visto anteriormente, ou seja, dado um vetor de tamanho n , a função deverá somar seus elementos.
- Para construir essa função, devemos primeiramente pensar em quais serão seus argumentos. Veja que:
 - ① Precisaremos de um argumento referente ao tamanho do vetor (chamaremos de **n**).
 - ② Precisaremos fazer com que o programa percorra por todo o vetor. Uma estrutura de repetição pode ser útil (**FOR**)
 - ③ Precisaremos de um objeto auxiliar que servirá para armazenar os valores da soma. Chamaremos esse objeto de **s**.
- Tendo essas características em mente, podemos montar nossa função! Vamos chamá-la de **soma**.



Fonte:
<https://pixabay.com/images/id-3689669/>

Funcionalidades no R

Exemplo 10

Nossa função tem a seguinte estrutura de programação:

```

Início: x = vetor;
      n = comprimento de x;
      s = 0
for (i) in < 1:n >
      < s = s + x[i] >
fim
  
```

Exemplo 10 no R

```

X=c(2,4,4,3,2) #exemplo

soma=function(x){
  n=length(x)
  s=0 #objeto com que receberá a soma
  for(i in 1:n){
    s=s+x[i]
  }
  print(s)
}
soma(X*Y)
[1] 54
sum(X*Y) ## veja que é igual
[1] 54
  
```

Funcionalidades no R

Exemplo 11

Vamos criar uma função que calcule a média aritmética simples (\bar{X}), dada por

$$\bar{X} = \frac{\sum_{i=1}^n X_i}{n} \quad (2)$$

de um vetor qualquer.

Início: $x = \text{vetor};$

$n = \text{comprimento de } x;$

$s = 0$

for (i) in < 1:n >

< $s = s + x[i]$ >

< $\text{media} = s/n$ >

fim

Exemplo 11 (a) no R

```
X=c(2,4,4,3,2) #exemplo
```

```
media=function(x){
  n=length(x)
  s=0 #objeto com que receberá a soma
  for(i in 1:n){
    s=s+x[i]
  }
  m=s/n
  return(m)
}
media(X)
[1] 3
```

Exemplo 11 (b) no R

```
media2=function(x){
  n=length(x)
  m=soma(x)/n ## funcao soma
  return(m)
}
media2(X)
[1] 3
```

Observações

- Qual a diferença entre os procedimentos feitos no Exemplo 11 (a) e (b)?
- No **Exemplo 11 (a)**, criamos a função **media**, que utiliza a estrutura de repetição FOR. Tudo é feito em uma função apenas.
- No **Exemplo 11 (b)**, criamos a função **media2**, que faz a mesma coisa que a função **media**, com a diferença de que a função **media2** depende da função **soma** do exemplo 10.
- O procedimento usado em **media2** é bastante útil na construção de funções mais complexas e torna a programação mais dinâmica e fluída. Isso mostra também a versatilidade do R para construir funções próprias.



Fonte: <https://pixabay.com/images/id-294525/>

Funcionalidades no R

Exemplo 12

Vamos criar uma função que calcule a variância amostral (S^2), dada por

$$S^2 = \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{n - 1} \quad (3)$$

de um vetor qualquer. Em que \bar{X} é a média amostral como na equação 2.

Início: x = vetor;

n = comprimento de x;

m = média de x;

s2 = 0

for (i) in < 1:n >

< s2 = s2 + (x[i] - m)^2 >

< variancia = s2/(n - 1) >

fim

Exemplo 12 no R

```
X=c(2,4,4,3,2) #exemplo
## variancia
variancia=function(x){
  n=length(x)
  s=0 #objeto com que receberá a soma
  for(i in 1:n){
    s=s+x[i]
  }
  m=s/n
  s2=0
  for(i in 1:n){
    s2=s2 + (x[i] - m)^2
  }
  variancia=s2/(n-1)
  return(variancia)
}
variancia(X)
[1] 1
## ou
variancia2=function(x){
  n=length(x)
  m=media(x)
  s=soma((x-m)^2)
  variancia=s/(n-1)
  return(variancia)
}
variancia2(X)
[1] 1
```

Prática

Exercício

- O sistema de avaliação de uma disciplina em uma universidade consiste na realização de três provas e um trabalho. Além disso, o professor disponibiliza listas de exercícios que, se todas forem entregues, é somado 1 ponto na nota média do aluno. Se parte das listas é entregue, então é feito um cálculo proporcional.
- A Nota Média do aluno é calculada pela seguinte expressão

$$\text{Nota Média} = \frac{P1 + P2 + P3 + T}{4} + \text{Listas} \quad (4)$$

- Se a Nota Média do aluno for no mínimo 6 pontos, ele é aprovado, caso contrário é reprovado. Caso o resultado da equação 4 seja superior a 10 pontos, é atribuída nota 10 ao aluno.
- Considerando um aluno que obteve as seguintes notas: $P1 = 5$, $P2 = 3$, $P3 = 5$, $Trabalho = 6$ e $Listas = 1$, construa uma rotina que calcule a Nota Média e informe a situação do aluno.

Prática: Solução

Exercício no R

```
nm=function(p1,p2,p3,t,l){  
  n=(p1+p2+p3+t)/4 + 1  
  
  if(n>=10)  
    n=10  
  else n  
  
  if(n>=6)  
    cat(paste("Sua nota média é ", n, ". Aprovado"))  
  else  
    cat(paste("Sua nota média é ", n, ". Reprovado"))  
}  
nm(5,3,5,6,1)  
Sua nota média é  5.75 . Reprovado  
  
nm(10,10,10,10,1)  
Sua nota média é  10 . Aprovado
```



Fonte: <https://pixabay.com/images/id-4997565/>