CURSO
INTRODUÇÃO AO R

Aula 2

Linguagem R

Luis Iván Ortiz Valencia

IESC - UFRJ

2011

Tópicos

1. Operações básicas com números.
2. Estruturas básicas do R.
3. Objetos de estrutura Vetor.
4. Objetos de estrutura Matriz

Operações básicas com números

- **A linha de comando funciona como uma calculadora.**

Operações binárias	Nome
x + y , x – y	Soma, Diferencia
x * y , x / y	Produto, Divisão
x ** y ou x ^ y	Potência

↑ **Ordem das operações**

```
> 2+3*2-4/5^2
[1] 7.84
```

- **Usar o parêntese para agrupar operações.**

```
> (2+3)*(2-4)/5^2
[1] -0.4
```

- **O R usa o ponto (.) como separador dos decimais.**

Operações básicas com números

Muda o separador decimal para a vírgula nos resultados de operações:
options(OutDec = ",")

```
> options(OutDec=",")

> 2/3
[1] 0,6666667

> 3.4 - 1
[1] 2,4
```

Operações básicas com números

Função	R
Seno	sin(x)
Cosseno	cos(x)
Tangente	tan(x)
Raiz Quadrada	sqrt(x)
Valor Absoluto	abs(x)
Logaritmo natural	log(x)
Logaritmo base 10	log10(x)
Logaritmo base n	log(x,n)
Exponencial	exp(x)
Fatorial	factorial(x)

Operações básicas com números

- Por *default*, o R mostra 7 dígitos.

```
> 343.544332223445
[1] 343.5443
> 3213131233133
[1] 3.213131e+12
> 1/3
[1] 0.3333333
> 100/3
[1] 33.33333
```

Operações básicas com números

Define o número de dígitos significativos visíveis nos resultados de operações: options(digits = número)

```
> 1/3
[1] 0.3333333
options(digits=2)
> 1/3
[1] 0.33
> 5/4
[1] 1.2
```

Operações básicas com números

Operação módulo entre dois números: x %% y

```
> 3 %% 2
[1] 1
```

Operação divisão inteira entre dois números: x %/% y

```
> 10 %/% 3
[1] 3
```

Operações com valores lógicos

- **Os valores lógicos são TRUE e FALSE.**

Função	R
Negação	!
Igual	==
Diferente	!=
Maior	>
Menor	<
Maior ou igual	>=
Menor ou igual	<=
E	&
OU	|

Operações com valores lógicos

```
> x <- 5
> x < 3
[1] FALSE

> y <- 3
> x == y
[1] FALSE

> x<10 & y>=4
[1] FALSE

> x<10 | y>=4
[1] TRUE
```

Estruturas básicas do R

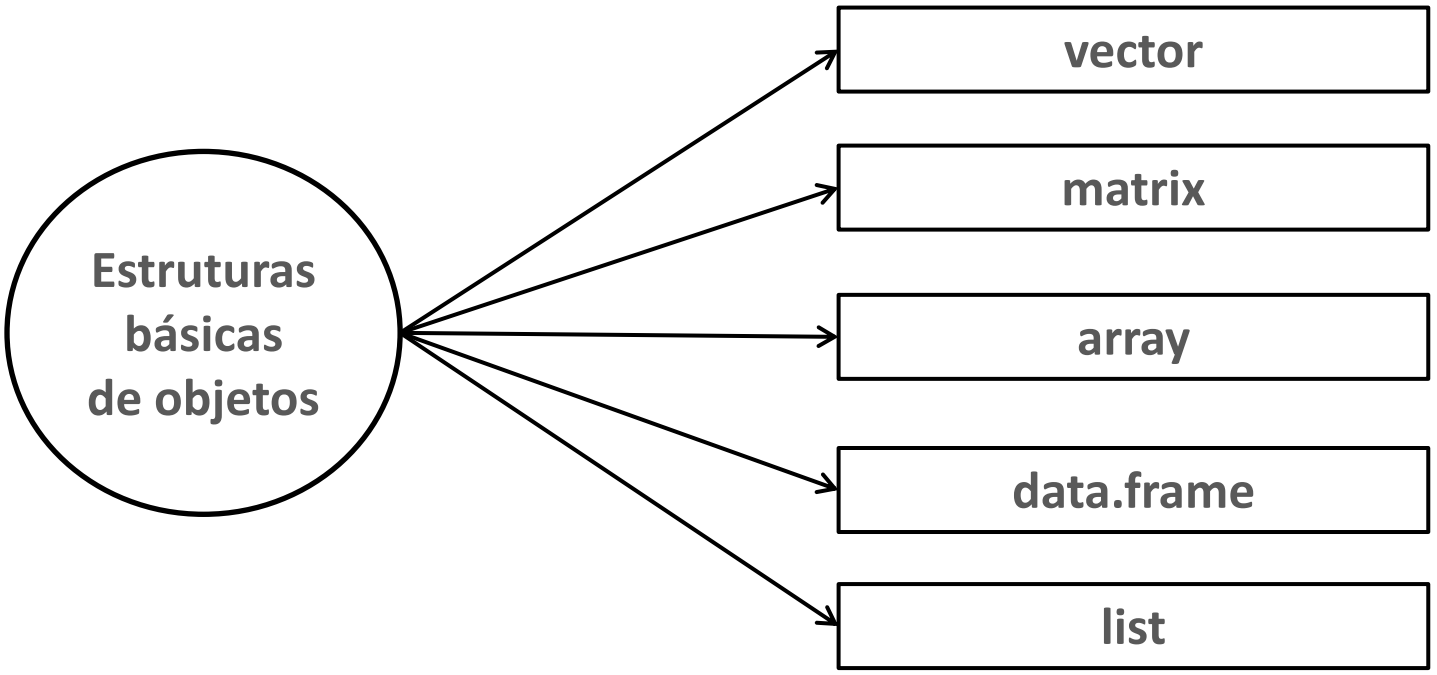
Objetos

Tudo que pode ser atribuído a uma variável.

Exemplos

- **Constantes: 3.3, "joão".**
- **Símbolos especiais: NA, TRUE.**
- **Estruturas de dados: matrizes, vetores.**
- **Resultados de operações: sqrt(2.3)+2, read.table("dados.csv ").**

Estruturas básicas do R



Vetor

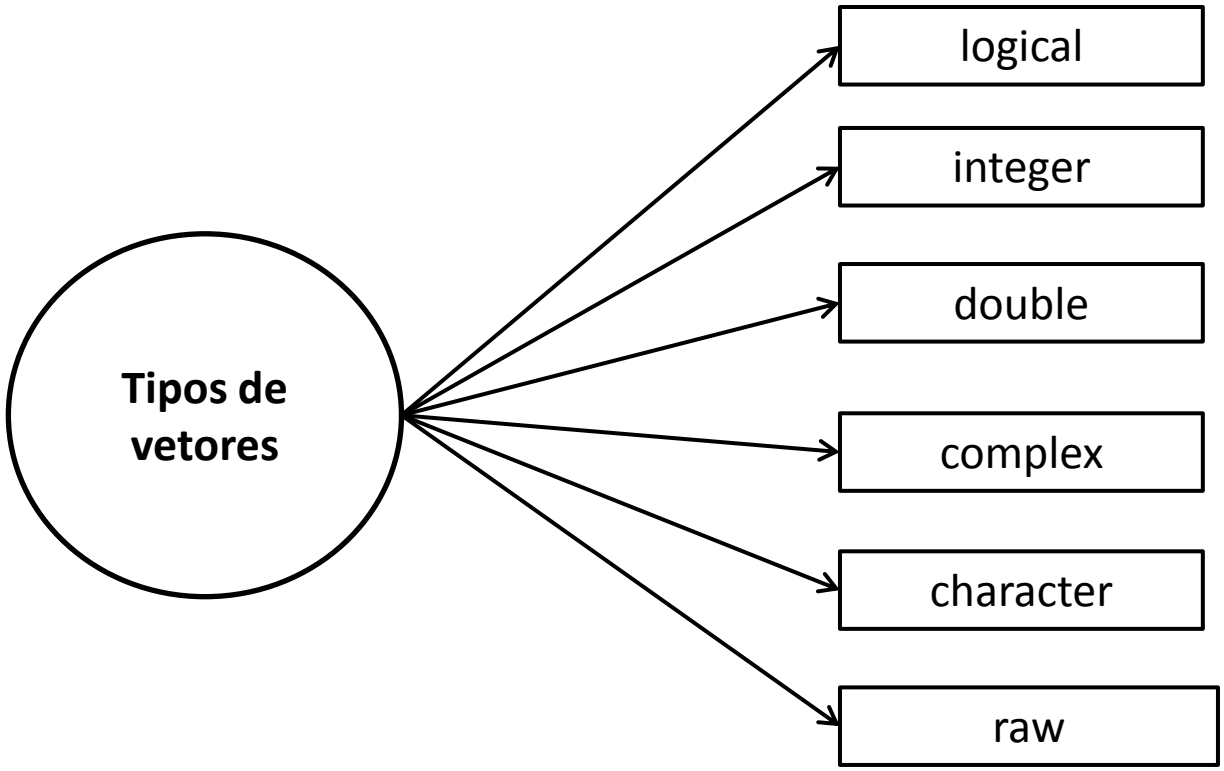
- Um vetor é uma seqüência ordenada de “células” contendo dados.
- Os componentes (dados) de um vetor só podem ser do mesmo TIPO.

Dado 1	Dado 2	Dado 3	Dado 4	Dado 5	...	Dado n
Posição	**1**	**2**	**3**	**4**	**5**	**n**

- O valor de n representa o tamanho do vetor.
- Uma variável que recebe um único valor, por exemplo um número, equivale a um vetor de tamanho 1.

```
> x <- 4
> x
[1] 4
```

Vetor


Tipos de vetores
logical
integer
double
complex
character
raw

Vetor

Função para gerar vetores: c()

```
> x <- c(1,3,5,3,2)
> x
[1] 1 3 5 3 2

> w <- c(1.33,3,0,-34,3)
> w
[1]   1.33   3.00   0.00 -34.00   3.00

> y <- c("a","b","c")
> y
[1] "a" "b" "c“

> z <- c(TRUE,FALSE,FALSE,TRUE)
> z
[1]  TRUE FALSE FALSE  TRUE
```

Vetor

```
> a <- c(1,3,4,"d")
> a
[1] "1" "3" "4" "d"
```

Função para gerar vetores com números: scan()

```
> x <- scan()
1: 11
2: 3
3: 23
4: 43
5: 54
6:
Read 5 items
> x
[1] 11  3 23 43 54
```

Vetor

Gera um vetor de seqüência de valores com incremento de uma unidade: a:b

```
> 1:10
 [1]  1  2  3  4  5  6  7  8  9 10

> 10:1
 [1] 10  9  8  7  6  5  4  3  2  1

> 1.1:10
[1] 1.1 2.1 3.1 4.1 5.1 6.1 7.1 8.1 9.1
```

Vetor

Gera seqüência de valores: seq(from = , to = , by =)

```
> seq(from = 1, to = 10, by = 0.5)
 [1]  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5  7.0  7.5  8.0
[16]  8.5  9.0  9.5 10.0
```

```
> seq(1.5,10.5,0.25)
 [1]  1.5  1.8  2.0  2.2  2.5  2.8  3.0  3.2  3.5  3.8  4.0  4.2  4.5  4.8  5.0
[16]  5.2  5.5  5.8  6.0  6.2  6.5  6.8  7.0  7.2  7.5  7.8  8.0  8.2  8.5  8.8
[31]  9.0  9.2  9.5  9.8 10.0 10.2 10.5
```

Vetor

Acessa os elementos de um vetor: []

```
> x <- c(1,4,6,0,22,10)

> x[2]
[1] 4

> x[c(1,2)]
[1] 1 4

> x[3:6]
[1]  6  0 22 10
```

Vetor

Acessa os valores de um vetor sob uma condição: []

```
> x <- c(1,4,6,0,22,10)
> x[x >=10]
[1] 22 10
```

Acessa as posições de um vetor sob uma condição: which()

```
> x <- c(1,4,6,0,22,10)
> which(x>=10)
[1] 5 6
```

Vetor

Muda o valor de um elemento específico de um vetor

```
x <- c(1,4,6,0,22,10)
x[3] <- 0
> x
[1]  1  4  0  0 22 10

> x <- c(1,4,6,0,22,10)
> x <= 5
[1]  TRUE  TRUE FALSE  TRUE FALSE FALSE
> x[x <= 5] <- 4
> x
[1]  4  4  6  4 22 10
```

Vetor

Função	R
Tamanho	length(x)
Máximo	max(x)
Mínimo	min(x)
Amplitude	range(x)
Soma dos elementos	sum(x)
Produto dos elementos	prod(x)
Média	mean(x)
Mediana	median(x)
Variância	var(x)
Desvio padrão	sd(x)
Distância Interquartílica	IQR(x)
Sumario	summary(x)

Vetor

Define um valor não disponível : NA

- NA significa NOT AVAILABLE.

```
> x <- c(2,31,0,-3,NA,2,NA)
```

- Algumas operações que envolvem vetores com NA tem como resultado NA.

```
> sum(x)
[1] NA
> mean(x,na.rm=TRUE)
[1] 6.4
```

Testa a presença de NA : is.na()

```
> is.na(x)
[1] FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE
```

Vetor

Define um valor infinito: Inf

- Inf significa INFINITE.

```
> 1/0
[1] Inf
```

Define um valor não disponível por processo computacional: NaaN

- NaaN significa NOT A NUMBER.

```
> 0/0
[1] NaN
```

Vetor

Ordena os elementos de um vetor: sort()

```
> x <- c(1,4,6,0,22,10)
> sort(x)
[1]  0  1  4  6 10 22

> sort(x,decreasing=TRUE)
[1] 22 10  6  4  1  0
```

Vetor

Junta dois vetores numéricos: c()

```
> x <- c(1,4,6,0,22,10)

> y <- c(2,4,6,4,2)

> z <- c(x,y)

> z

 [1]  1  4  6  0 22 10  2  4  6  4  2
```

Vetor

Gera um vetor numérico: numeric()

- Vetor vazio.

```
> x<-numeric()
> length(x)
[1] 0
> x[3]<-3
> x
[1] NA NA  3
```

- Vetor de zeros.

```
> numeric(25)
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Vetor

Operações com vetores

- Soma vetor e um número.

```
> x <- c(1,4,6,0,22,10)
> x+10
[1] 11 14 16 10 32 20
```

- Soma dois vetores com mesmo tamanho.

```
> x <- c(1,4,6,0,22,10)
> y <- c(2,4,6,4,2,0)
> x + y
[1]  3  8 12  4 24 10
```

Vetor

Operações com vetores

- Produto de vetor e um número.

```
> x <- c(1,4,6,0,22,10)
> 10*x
[1]  10  40  60   0 220 100
```

- Produto de dois vetores com mesmo tamanho.

```
> x <- c(1,4,6,0,22,10)
> y <- c(2,4,6,4,2,0)
> x*y
[1]  2 16 36  0 44  0
> x/y
[1]  0.5  1.0  1.0  0.0 11.0  Inf
```

Vetor

Apagando elementos de um vetor

```
> x <- c(1,4,6,0,22,10)
> x <- x[-6]
> x
[1]  1  4  6  0 22
```

```
> x <- c(1,4,6,0,22,10)
> x <- x[-c(1,6)]
> x
[1]  4  6  0 22
```

Vetor

Obtém uma amostra de valores de um vetor: sample()

- Sem reposição.

```
> sample(1:50,10)
 [1] 14 41 31  9  4  7 47 18 38 23
```

- Com reposição.

```
> sample(1:10,30,replace=TRUE)
 [1]  9  5  6  6  9  9  4 10  7  7  2  9  3  4  1  5 10 10  5  2  9  3  7  2  5  2  1  3  2  9
```

Observar que cada vez que o comando é repetido com os mesmo parâmetros o resultado é diferente.

Vetor

Gera uma tabela de frequências dos valores de um vetor: table()

```
> ss<-sample(1:10,30,replace=TRUE)

> ss
 [1]  3  9  4  3  1  8  2  7  2  9  4  8  1 10  2  9 10  7  4  2  2  1  7  9  7  1 10  4  7 10

> table(ss)
ss
 1  2  3  4  7  8  9 10
 4  5  2  4  5  2  4  4
```

Vetor

Operações de conjuntos com vetores

Função	R
União	union(x,y)
Interseção	intersection(x,y)
Diferencia simétrica	setdiff(x,y)
Igual	setequal(x,y)
Pertence	is.element(el,x)

Vetor

Operações de conjuntos com vetores

```
> x <- c(sort(sample(1:20, 9)),NA)
> y <- c(sort(sample(3:23, 7)),NA)
> x
 [1]  2  3  4  6  7  9 10 12 16 NA
> y
[1]  3  4 10 15 16 18 19 NA

> union(x,y)
 [1]  2  3  4  6  7  9 10 12 16 NA 15 18 19

> intersect(x,y)
[1]  3  4 10 16 NA
```

Vetor

Operações de conjuntos com vetores

```
> x
 [1]  2  3  4  6  7  9 10 12 16 NA
> y
[1]  3  4 10 15 16 18 19 NA

> setdiff(x, y)
[1]  2  6  7  9 12
> setdiff(y, x)
[1] 15 18 19
> setequal(x, y)
[1] FALSE
> is.element(19,x)
[1] FALSE
> is.element(10,x)
[1] TRUE
```

Vetor

Verifica se algum valor de um vetor de valores lógicos é TRUE: any()

```
> ss<-sample(-10:10,50,replace=TRUE)
> ss
 [1] -3 -8 -1 -4  7  9 -4 -5  8 -7 -4  8 10 -9  7 -6 -6 -3  2 -6  6
[22]  2 -6 -4 -9 -1  0  6  7  8  8  1 -6  0 -10  3 -7  7  9 -10 -6 -2
[43] -1 -4  7 -4  6 -8 10 -4
> any(ss == 0)
[1] TRUE
```

Vetor

Verifica se todos os valores de um vetor de valores lógicos é TRUE: all()

```
> ss<-sample(-1:10,50,replace=TRUE)
> ss
 [1]  9  3 10  6  6  0  6  3  6  9  3  3  4  9  1  8  7  3 10  4  2  6  7 10  5  0  4  8
[29]  1  3  3  4  4  9  4  3  3  6  4  0  7  8  5  0 -1  1  5  2  1 10
> all(ss > 0)
[1] FALSE
> all(ss >= -1)
[1] TRUE
```

Matriz

- **Uma matriz é um arranjo retangular de números.**

$$\begin{matrix} 5 & 2 & 5 & 4 \\ 3 & 4 & 2 & 6 \\ 9 & 3 & 2 & 1 \end{matrix}$$

- **A dimensão da matriz é (número de linhas) x (número de colunas).**

- **No exemplo, a dimensão é 3 x 4.**

Matriz

Gera uma matriz: matrix()

```
> matrix(c(1,2,3,4,5,6,7,8), nrow = 2, ncol = 4, byrow = FALSE)
> matrix(c(1,2,3,4,5,6,7,8),  nrow = 2, ncol = 4)
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

> matrix(c(1,2,3,4,5,6,7,8), nrow = 2, ncol = 4, byrow = TRUE)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
```

Matriz

Gera uma matriz: matrix()

```
> matrix(0,3,3)
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
[3,]    0    0    0

> matrix(1,5,5)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    1    1    1    1
[2,]    1    1    1    1    1
[3,]    1    1    1    1    1
[4,]    1    1    1    1    1
[5,]    1    1    1    1    1
```

Matriz

Extração de valores de uma matriz: []

```
> m <- matrix(c(1,2,3,4,5,6,7,8,9), nrow = 3, ncol = 3)

> m

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


> m[2,3]
[1] 8

> m[2,]
[1] 2 5 8

> m[,2]
[1] 4 5 6
```

Matriz

```
> m<-matrix(1:25,5,5)
> m
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25
```

```
> m[1:3,]
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
```

```
> m[,2:4]
     [,1] [,2] [,3]
[1,]    6   11   16
[2,]    7   12   17
[3,]    8   13   18
[4,]    9   14   19
[5,]   10   15   20
```

```
> m[c(1,3,5),]
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    3    8   13   18   23
[3,]    5   10   15   20   25
```

Matriz

Dimensão de uma matriz: dim()

```
> m[1:3,]
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23

> dim(m[1:3,])
[1] 3 5
```

Matriz

Diagonal de uma matriz: diag()

```
> m
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25

> diag(m)
[1]  1  7 13 19 25
```

Gera uma matriz diagonal: diag()

```
> diag(1,3)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

Matriz

Número de linhas de uma matriz: nrow()

```
> m<-matrix(1:21,3,7)
> m
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]    1    4    7   10   13   16   19
[2,]    2    5    8   11   14   17   20
[3,]    3    6    9   12   15   18   21
> nrow(m)
[1] 3
```

Número de colunas de uma matriz: nrow()

```
> m<-matrix(1:21,3,7)
> m
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]    1    4    7   10   13   16   19
[2,]    2    5    8   11   14   17   20
[3,]    3    6    9   12   15   18   21
> ncol(m)
[1] 7
```

Transposta de uma matriz: t()

```
> m
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]    1    4    7   10   13   16   19
[2,]    2    5    8   11   14   17   20
[3,]    3    6    9   12   15   18   21

> t(m)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
[5,]   13   14   15
[6,]   16   17   18
[7,]   19   20   21
```

Determinante de uma matriz: det()

```
> m<-matrix(sample(1:10,9),3,3)
> m
     [,1] [,2] [,3]
[1,]    6    1    5
[2,]    9    4   10
[3,]    2    7    8

> det(m)
[1] -5
```