UNIVERSIDADE ESTADUAL DE CAMPINAS

LABORATÓRIO CENTRAL DE TECNOLOGIAS DE ALTO DESEMPENHO

Apostila:

Conceitos Básicos de R

Janeiro/2015

SUMÁRIO

1. Introdução.

- O que é o R?

De acordo com (http://www.r-project.org/about.html):
"R é um software livre composto por uma linguagem e um ambiente capazes de computar dados estatísticos e criar gráficos. R fornece uma ampla variedade de estatística (modelagem linear e não-linear, testes estatísticos clássicos, análises de séries temporais) e de gráficos, os quais são de fácil visualização. Além disso, pode ser utilizado como uma linguagem de programação simples que inclui condicionais, loops, recursos para leitura e gravação de arquivos. Para a manipulação de dados estatísticos, R possui um conjunto de operadores para cálculos em tabelas, especialmente matrizes."

Por sua vez, quando há necessidade de se realizar análises de dados de bioinformática, podemos utilizar um conjunto de pacotes específico denominado "Bioconductor", que é instalado e manipulado através do R.

Esta apostila apenas possui alguns comandos que serão utilizados no curso de RNA-seq do *Lactad*. Para maiores informações, indicamos consultar um manual de "Introdução ao R" no site (http://cran.r-project.org/doc/manuals/r-release/R-intro.pdf).

2. Tipos de dados.

- *“Numeric”*:

Nesse tipo de dados estão contidos a valores numéricos inteiros (0, 1, 2, 3, 4, 5, 6, 7, 8, 9…) e decimais (1.0 ... 1.9, ...).

- *“Complex”*:

Contém dados de números complexos (1i, 2i, 3i, 4i,...).

- “*Character”*:

São dados que se referem a caracteres, podendo conter um (“A”, ”B”, ”C”, ”D”… “a”, ”b”, ”c”, ”d”…) a vários caracteres (“ABCD”... “abcd”...).

- *“Logical”*:

São compostos por dados lógicos (TRUE, FALSE).

3. Operadores.

Os operadores básicos citados nessa sessão são: soma, subtração, divisão, multiplicação, exponencial, raiz quadrada e função logarítmica.

- *Soma* (+).

Exemplo:

```
> 1 + 2
[1] 3
```

- *Subtração* (-).

Exemplo:

```
> 9 - 2
[1] 7
```

- *Divisão* (/).

Exemplo:

```
> 9/3
[1] 3
```

- *Multiplicação* (*).

Exemplo:

```
> 2*4
[1] 8
```

- *Exponencial* (^).

Exemplo:

```
>2^2
[1] 4
```

- *Raiz quadrada* (sqrt).

Exemplo:

```
>sqrt(4)
[1] 2
```

- *Logaritmo* (log2(); log10()).

Exemplo:

```
> log2(9)
[1] 3.169925

> log10(9)
[1] 0.9542425
```

4. Variáveis e atribuição simples de valores.

Os símbolos para a atribuição de valores em R são “<-“ ou “=”.

Exemplo:

```
> x = 1 # x é o nome de uma variável. O valor 1 foi atribuído à variável
> x
[1] 1
```

5. Comando de Comparação.

Os símbolos de citados nessa sessão são usados para fazer comparação de valores: igual, menor, menor ou igual, maior, maior ou igual, “e” e “ou”.

- *Igual* (==).

Exemplo:

```
> x = 1
> y = 2
> x == y
[1] FALSE
```

- *Menor*(<).

Exemplo:

```
> x < y
[1] TRUE
```

- *Menor ou igual* (<=).

Exemplo:

```
> x <= y
[1] TRUE
```

- *Maior* (>).

Exemplo:

```
> x > y
[1] FALSE
```

- *Maior ou igual* (>=).

Exemplo:

```
> x >= y
[1] FALSE
```

- *Diferente* (!=).

Exemplo:

```
> x != y
[1]  TRUE
```

- *E* (&).

Exemplo:

```
> (x > 0) & (x <= 1)
[1] TRUE
```

- *Ou* (|).

Exemplo:

```
> (y > 0) | (y < 1)
[1] TRUE
```

6. Comandos de condicional.

Os comandos de condicional são capazes de verificar se uma condição pode ser satisfeita e, desta maneira, permitem a execução do comando seguinte a essa condição. Os comandos mais comuns em R são: se ("*if*") e senão ("*else*").

- Se (if).

Exemplo:

```
> x = 1
> if(x >= 0){
+ x
+}
[1] 1
```

- Senão (else).

Exemplo:

```
> x = 1
> y = 2
> if(x > 1){
+ x
+ } else{
+ y
+}
[1] 2
```

7. Objetos.

O R trabalha com diferentes estruturas de dados. Dentre elas estão as mais utilizadas: "*vector", "data frame", "list", "matrix"*, entre outras.

7.1. *Vector*.

O "*vector"* é uma lista de elementos, os quais devem ser do mesmo tipo de dados ("*numeric", "character", "logical"*).

- "*Numeric*"

Exemplo:

```
> x = c(1,2,3,4,5,6)
> x
[1] 1 2 3 4 5 6
> class(x)
[1] "numeric"
```

- "*Character"*

Exemplo:

```
> y = c("one","two","three", "four", "five","six")
> y
[1] "one"   "two"   "three" "four"  "five"  "six"
> class(y)
[1] "character"
```

- "*Logical*"

Exemplo:

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

```
> class(z)
[1] "logical"
```

7.2. *Data frame*.

"*Data frame*" é uma estrutura de dados que permite diferentes tipos de dados na mesma coluna, mas todas colunas devem ter com o mesmo tamanho de linhas.
Por exemplo:

```
> df = data.frame(X=c(1,"two",3),Y=c("one",2,"three"))
> df
    X     Y
1   1   one
2 two     2
3   3 three
> class(df)
[1] "data.frame"
```

7.3. Matriz (*Matrix*)

A matriz permite somente um tipo de dados por coluna e estas devem ter o mesmo número de linhas.
Por exemplo:

```
> w = cbind(x,y,z) #comando que cria uma matriz de elementos
a partir de 3 vectors
>w     x   y       z
[1,] "1" "one"   "TRUE"
[2,] "2" "two"   "FALSE"
[3,] "3" "three" "TRUE"
[4,] "4" "four"  "FALSE"
[5,] "5" "five"  "TRUE"
[6,] "6" "six"   "FALSE"
```

```
> class(w)
[1] "matrix"
```

7.4. Lista (*List*).

A lista consiste em um conjunto ordenado de objetos conhecidos como os seus componentes. Também aceita diferentes tipos de dados na mesma estrutura de dados.

Por exemplo:

```
> list_samples = list( name = "Sample01", group = "Control",
no.replicates = 3, time = c( 1, 2, 3))
> list_samples
$name
[1] "Sample01"

$group
[1] "Control"

$no.replicates
[1] 3

$time
[1] 1 2 3

> class(list_samples)
[1] "list"
```

8. Manipulação de objetos

8.1. *Vector*.

Operadores básicos funcionam sobre os "*vectors*": soma, subtração, divisão, multiplicação, exponenciação, entre outras.

Exemplo:

```
> x = c(1,2,3)
> y = c(6,7,8)
> y + x # soma
[1]  7  9 11
> y - x # subtracao
[1] 5 5 5
> y/x # divisao
[1] 6.000000 3.500000 2.666667
> y*x # multiplicacao
[1]  6 14 24
> y^x # exponenciacao
[1]   6  49 512
```

Comandos comuns para manipulação de vector: tamanho e impressão de elementos.

Exemplo:

```
> x = c(9,8,7,6,5,4,3,2,1,0)
> x
[1] 9 8 7 6 5 4 3 2 1 0
> length(x) #tamanho do vector
[1] 10
> x[4] # imprime o quarto elemento do vector
[1] 6
> x[1:3] # imprime na tela os elementos 1 a 3
```

```
[1] 9 8 7
> head(x) # imprime os 6 primeiros elementos do vector
[1] 9 8 7 6 5 4
> tail(x) # imprime os 6 últimos elementos do vector
[1] 5 4 3 2 1 0
```

8.2. *Data Frame*.

É possível atribuir nomes de linhas para os "*data.frames*", assim como verificar o tamanho e imprimir elementos desse tipo de estrutura de dados.
Exemplo:

```
> df = data.frame(X = c( 1, "two", 3, "four", "five", "six",
7, 8, 9, 0), Y = c( "one",  2, "three", 4, 5, 6, "seven", 8,
"nine", 10))
> line_name = c("A","B","C","D","E","F","G","H","I","J")
> # comando que atribui os nomes das linhas de um data.frame
> row.names(df) = line_name
> head(df)  # comando que imprime as 6 primeiras linhas do data.frame
     X     Y
A    1   one
B  two     2
C    3 three
D four     4
E five     5
F  six     6

> df["C",]    # imprime somente a linha chamada "C"
  X     Y
C 3 three

> df[2:5,] # imprime as linhas 2 a 5
```

```
     X     Y
B  two     2
C    3 three
D four     4
E five     5

> dim(df) # comando que imprime as dimensões de um data.frame
[1] 10  2
```

8.3. Lista.

A manipulação de listas é diferente de um data.frame ou de uma matriz.

Exemplo:

```
> list_samples = list( name = "Sample01", group = "Control",
no.replicates = 3, time = c(1,2,3))
> list_samples
$name
[1] "Sample01"

$group
[1] "Control"

$no.replicates
[1] 3

$time
[1] 1 2 3

### Os dados podem ser acessados pelos nomes dos campos
> list_samples$name
[1] "Sample01"
> list_samples$group
```

```
[1] "Control"
> list_samples$no.replicates
[1] 3
> list_samples$time #"time" é um objeto vector com 3 elementos
[1] 1 2 3
> list_samples$time[1] # acesso de cada campo do vector "time"
[1] 1
> list_samples$time[2]
[1] 2
> list_samples$time[3]
[1] 3

### Ou os dados podem ser vistos pelo índice dos campos
> list_samples[[1]]
[1] "Sample01"
> list_samples[[2]]
[1] "Control"
> list_samples[[3]]
[1] 3
> list_samples[[4]] ## lembrando que esse elemento é um vector com 3
elementos, que podem ser acessados separados
[1] 1 2 3
> list_samples[[4]][1]
[1] 1
> list_samples[[4]][2]
[1] 2
> list_samples[[4]][3]
[1] 3
```

9. Comandos de iteração (loops).

Os comandos de iteração mais usados no R são: "*for*" e "*while*".

Exemplo de "*for*":

```
> x = c(9,8,7,6,5,4,3,2,1,0)
> # comando que cria um vector vazio do tamanho do vector x
> a = vector(mode = "numeric",length=length(x))
> # comando de iteração FOR
> for(i in 1:length(x)){
+ a[i]=x[i]+log2(x[i])+x[i]^2
+ }
> a
 [1]  93.16993  75.00000  58.80735  44.58496  32.32193  22.00000
13.58496   7.00000
 [9]   2.00000      -Inf
```

Exemplo de "*while*":

```
> x = c(9,8,7,6,5,4,3,2,1,0)
> a = vector(mode = "numeric",length=length(x))
>
> i = 1
> # comando de iteracao WHILE
> while(i <= length(x)){
+ a[i] = x[i] + x[i]^2
+ i = i + 1
+ }
>
> a
 [1] 90 72 56 42 30 20 12  6  2  0
```

10. Importação e manipulação de arquivos.

O R é capaz de importar arquivos exportados do Excel ou arquivos formato texto, contando que estejam os dados estejam separados por marcas de tabulação ("\t") ou por vírgulas (arquivos formato csv - *comma separated values*).

Os comandos mais comuns são:

- *read.delim;*
- *read.table;*
- *read.csv* - para a leitura de arquivos de extensão csv.

Exemplo:

```
> getwd()
[1] "/Users/Teste"
> setwd("/Users/Teste/tables")
> table01 = read.table( file = "text_table01.txt", sep = "\t",
header = TRUE)
> head(table01)
   Alunos Disciplina_A Disciplina_B Disciplina_C
1 Aluno01            A         10.0            D
2 Aluno02            A          7.5            B
3 Aluno03            B          8.5            A
4 Aluno04            B          4.5            A
5 Aluno05            A          2.0            B
6 Aluno06            D          5.0            A
> row.names(table01) = table01$Alunos
> table01 = table01[,2:4]
> names(table01)
[1] "Disciplina_A" "Disciplina_B" "Disciplina_C"
> head(table01)
        Disciplina_A Disciplina_B Disciplina_C
Aluno01            A         10.0            D
```

```
Aluno02            A          7.5            B
Aluno03            B          8.5            A
Aluno04            B          4.5            A
Aluno05            A          2.0            B
Aluno06            D          5.0            A

> # Comando que cria um subconjunto com notas = A na Disciplina A
> notas_A = subset(table01,Disciplina_A = "A")
> head(notas_A)
        Disciplina_A Disciplina_B Disciplina_C
Aluno01            A         10.0            D
Aluno02            A          7.5            B
Aluno05            A          2.0            B
Aluno08            A          6.7            A
Aluno13            A          9.0            C

> class(notas_A)
[1] "data.frame"
> # Comando que calcula a média de notas da Disciplina B
> mean(table01$Disciplina_B)
[1] 7.3
> # Comando para salvar uma tabela em um arquivo formato texto
> write.table(notas_A, file="text_table01_notas_A_discA.txt",
row.names=TRUE)
```

Observação:

- "***Factor***": é um objeto vector usado para classificar os componentes de outro "vector", visualizado através do comando "levels".

Exemplo:

```
> classif_nota = factor(table01$Disciplina_A)
> classif_nota
```

```
 [1] A A B B A D C A B D C C A B C
Levels: A B C D
> levels(classif_nota)
[1] "A" "B" "C" "D"
```