Explorando el Transcriptoma con Datos de Expresión Genética

Introducción a R & RStudio

Yered Pita-Juárez

6/1/2015

Caracteres

- Define una variable de caracteres usando apostrofes
- De otra manera R busca una variable definida con ese nombre

```
> m = "Manzanas"
> m
[1] "Manzanas"
> n = peras
Error: object 'peras' not found
```

Caracteres

- No podemos hacer cálculos con variables de caracteres

```
> m+2
Error in m + 2 : non-numeric argument to binary operator
```

- Pero tenemos operaciones para manipular caracteres

```
> toupper(m)
[1] "MANZANAS"
> tolower(m)
[1] "manzanas"
```

Vectores

- Se puede crear un vector con la función `c()`

```
> vec1 = c(1,4,6,8,10)
> vec1
[1] 1 4 6 8 10
```

- Los elementos de un vector se pueden acceder y modificar por su índice `[i]`

```
> vec1[5]
[1] 10
> vec1[3] = 12
> vec1
[1] 1 4 12 8 10
```

Vectores

- El comando `seq` es una forma de crear un vector a partir de una secuencia

```
> vec2 = seq(from=0, to=1, by=0.25)
> vec2
[1] 0.00 0.25 0.50 0.75 1.00
```

Matrices

- Son vectores con 2 dimensiones

```
mat=matrix(data=c(9,2,3,4,5,6),ncol=3)
> mat
[,1] [,2] [,3]
[1,] 9 3 5
[2,] 2 4 6
```

- `data` contiene las entradas de la matrix, `ncol` especifica el número de columnas y `nrow` especifica el número de hileras
- Los elementos de una matriz se acceden por su posición `[fila,columna]`

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

Matrices

- Para acceder a toda una hilera

```
> mat[2,]
[1] 2 4 6
```

- Para acceder a toda una columna

```
> mat[,1]
[1] 9 2
```

Data Frames

- Una matriz con columnas etiquetadas

```
> t = data.frame(x = c(11,12,14),y = c(19,20,21),
    z = c(10,9,7))
> t
   x  y  z
1 11 19 10
2 12 20  9
3 14 21  7
```

- Las columnas se pueden acceder por su nombre

```
> mean(t$z)
[1] 8.666667
> mean(t[["z"]])
[1] 8.666667
```

Data Frames

Ejercicio

- Crea un script para crear 3 vectores (`x1,x2,x3`) con 100 números aleatorios usando `rnorm`
- Crea un `data frame` llamado `t` con 3 columnas (`a,b,c`) que contengan respectivamente `x1,x1+x2,x1+x2+x3`
- Usa las siguientes functiones `plot(t)` y `summary(t)`
- ¿Cómo interpretarías los resultados?

Listas

- Una coleccción de objectos
- Si es lista de vectores, los vectores no tienen que ser del mismo tamaño
- Crear una lista

```
> L = list(one=1, two=c(1,2),five=seq(0, 1, length=5))
> L
$one
[1] 1
$two
[1] 1 2
$five
[1] 0.00 0.25 0.50 0.75 1.00
```

Listas

- Nombres de los componentes de la lista

```
names(L)
[1] "one" "two" "five"
```

- Operaciones con los elementos de la lista

```
> L$five + 10
[1] 10.00 10.25 10.50 10.75 11.00
```

Listas

- Las listas pueden contener differentes tipos de variables

```
Lst <- list(nombre="Jose", esposa="Maria",
            no.hijos=3,edad.hijos=c(4,7,9))
```

- Podemos acceder a los componentes de la lista usando el nombre del componente o el indice

```
> Lst$nombre
[1] "Jose"
> Lst$esposa
[1] "Maria"
> Lst$edad.hijos
[1] 4 7 9
```

```
> Lst[[1]]
[1] "Jose"
> Lst[[2]]
[1] "Maria"
> Lst[[4]]
[1] 4 7 9
```

Listas

- Si uno de los componentes tiene varios elementos, podemos accederlos usando el indice correspondiente

```
> Lst$edad.hijos[1]
[1] 4
> Lst[[4]][1]
[1] 4
```

Listas

Ejercicio

Crea una lista con tu siguientes datos

- Nombre
- Apellido
- Edad
- Escuela

NA

- Trabajando con datos reales, a veces tenemos observaciones que no están disponibles
- R representa estos datos faltantes con el valor `NA`

```
> obs=c(1,2,NA)
```

- En general, no es posible calcular estadisticas de datos incompletos

```
> mean(obs)
[1] NA
```

- Sin embargo, podemos optar por ignorar los datos que nos hacen falta

```
> mean(obs,na.rm=TRUE)
[1] 1.5
```

Combinando Dos Objectos: match, merge

- Tenemos dos `data frames` con datos acerca de unas ratas: cada uno con un ID diferente para cada rata

```
ratas <- data.frame(id = paste0("rat",1:10),
                    sexo = factor(rep(c("F","M"),each=5)),
                    peso = c(2,4,1,11,18,12,7,12,19,20),
                    longitud = c(100,105,115,130,95,150,165,180,190,175))
> ratas
      id sexo peso longitud
1   rat1    F    2      100
2   rat2    F    4      105
3   rat3    F    1      115
4   rat4    F   11      130
5   rat5    F   18       95
6   rat6    M   12      150
7   rat7    M    7      165
8   rat8    M   12      180
9   rat9    M   19      190
10 rat10    M   20      175
```

Combinando Dos Objectos: match, merge

- Tenemos dos `data frames` con datos acerca de unas ratas: cada uno con un ID diferente para cada rata

```
ratasTabla <- data.frame(id = paste0("rat",c(6,9,7,3,5,1,10,4,8,2)),
                         IDsecreto = 1:10)
> ratasTabla
      id IDsecreto
1   rat6         1
2   rat9         2
3   rat7         3
4   rat3         4
5   rat5         5
6   rat1         6
7  rat10         7
8   rat4         8
9   rat8         9
10  rat2        10
```

Combinando Dos Objectos: match, merge

- Queremos ordenar el `data frame` `ratas` usando el `IDsecreto` de `ratasTabla`.
- `match` te da por cada elemento del primer vector, el indice que corresponde al segundo vector

```
> match(ratasTabla$id, ratas$id)
 [1]  6  9  7  3  5  1 10  4  8  2
```

- El `data frame` que quieres reordenar es el segundo argumento de `match`

Combinando Dos Objectos: match, merge

- El `data frame` que quieres reordenar es el segundo argumento de `match`

```
> ratas[match(ratasTabla$id, ratas$id),]
      id sexo peso longitud
6   rat6    M   12      150
9   rat9    M   19      190
7   rat7    M    7      165
3   rat3    F    1      115
5   rat5    F   18       95
1   rat1    F    2      100
10 rat10    M   20      175
4   rat4    F   11      130
8   rat8    M   12      180
2   rat2    F    4      105
```

Combinando Dos Objectos: match, merge

- El `data frame` que quieres reordenar es el segundo argumento de `match`

```
> cbind(ratas[match(ratasTabla$id, ratas$id),],ratasTabla)
      id sexo peso longitud    id IDsecreto
6   rat6    M   12      150  rat6         1
9   rat9    M   19      190  rat9         2
7   rat7    M    7      165  rat7         3
3   rat3    F    1      115  rat3         4
5   rat5    F   18       95  rat5         5
1   rat1    F    2      100  rat1         6
10 rat10    M   20      175 rat10         7
4   rat4    F   11      130  rat4         8
8   rat8    M   12      180  rat8         9
2   rat2    F    4      105  rat2        10
```

Combinando Dos Objectos: match, merge

- Tambien podemos combinar los `data frames` usando la funcion `merge`

```
> ratasMerge <- merge(ratas, ratasTabla, by.x="id", by.y="id")
> ratasMerge[order(ratasMerge$IDsecreto),]
      id sexo peso longitud IDsecreto
7   rat6    M   12      150         1
10  rat9    M   19      190         2
8   rat7    M    7      165         3
4   rat3    F    1      115         4
6   rat5    F   18       95         5
1   rat1    F    2      100         6
2  rat10    M   20      175         7
5   rat4    F   11      130         8
9   rat8    M   12      180         9
3   rat2    F    4      105        10
```

Instalando Paquetes

Comprehensive R Archive Network (CRAN)

- Principal repositorio de paquetes para R
- 5,800 paquetes y 120,000 funciones (Junio 2014)

Bioconductor

- Repositorio de paquetes para R
- Análisis de datos genómicos

Instalando Paquetes

GitHub

- Repositorio popular para muchos proyectos open source
- Algunos paquetes de R

Instalando Paquetes de GitHub

- Necesitamos instalar primero `devtools`

```
install.packages("devtools")
```

- Vamos a instalar `dagdata` de GitHub

```
library(devtools)
install_github("genomicsclass/dagdata")
```

Instalando Paquetes de Bioconductor

- Para instalar el paquete base de Bioconductor

```
source("http://bioconductor.org/biocLite.R")
biocLite()
```

- Para instalar un paquete de Bioconductor

```
source("http://bioconductor.org/biocLite.R")
biocLite("NombreDelPaquete")
```

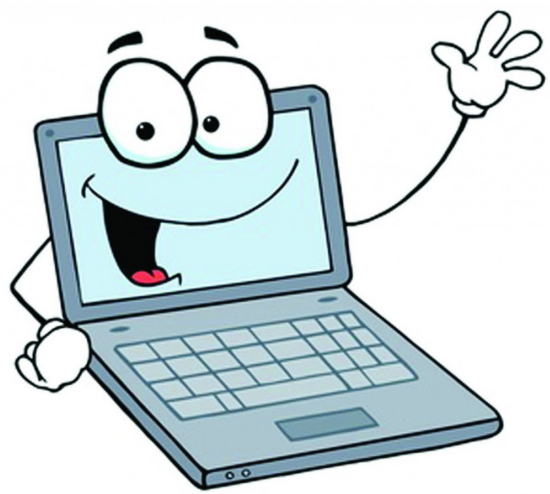
- Más información para instalar y actualizar paquetes de Bioconductor: `http://bioconductor.org/install/`

Instalando Paquetes

Ejercicio

Instala los siguientes paquetes de Bioconductor

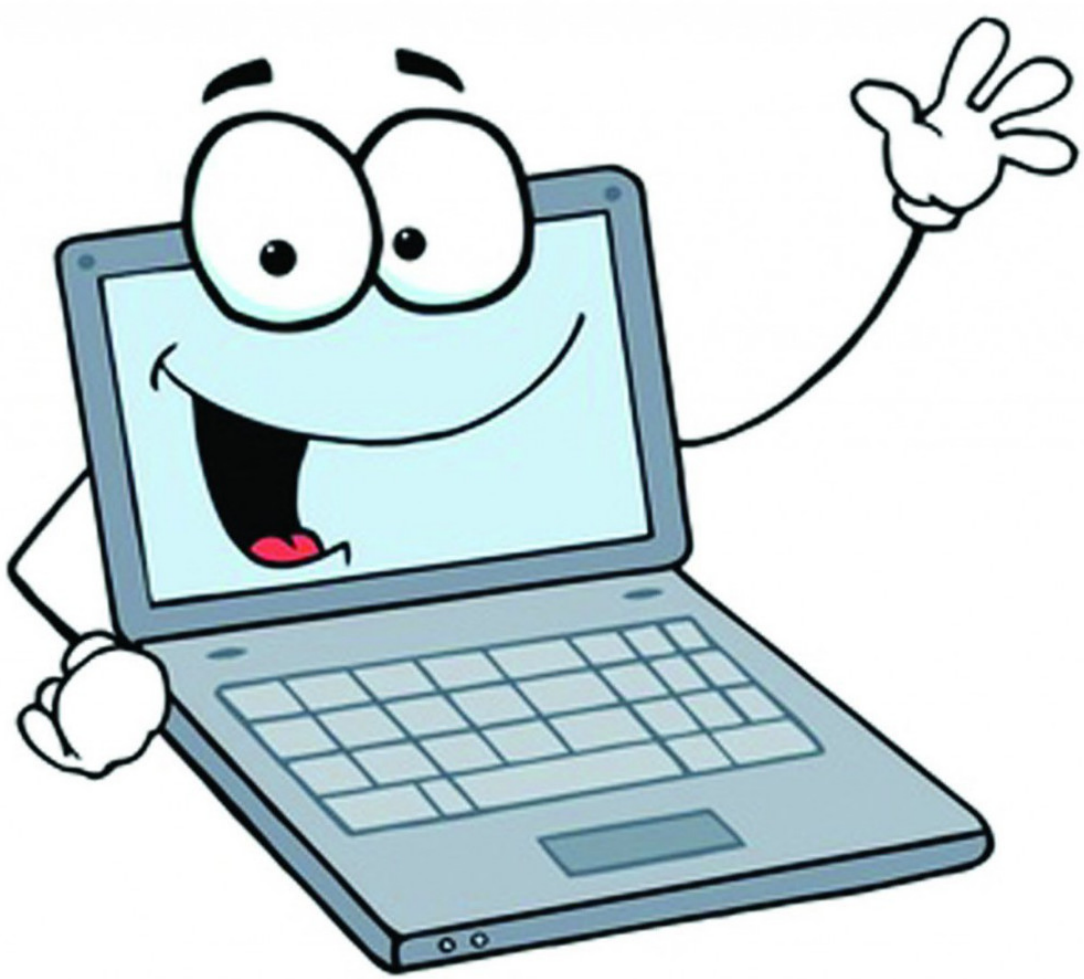
- `affy`
- `oligo`
- `GEOquery`
- `rae230a.db`
- `AnnotationDbi`
- `biomaRt`



Instalando Paquetes

Ejercicio

Carga el paquete `affy`, y accede a su vignette.



Vignettes

- Documentos de ayuda para paquetes de R
- Requeridos para cada paquete de Bioconductor
- Manuales con ejemplos
- ¿Cuáles vignettes están disponibles?

```
vignette(package="Biobase")
```

- Accediendo al pdf

```
vignette("ExpressionSetIntroduction")
```

- Navegando las vignettes

```
browseVignettes(package="Biobase")
```