

# STAT588/BIOL588: Genomic Data Science

## Lecture 2: Introduction to R

Dr. Yen-Yi Ho (hoyen@stat.sc.edu)

# R Topics Outline

- ▶ Get Started
- ▶ R as a calculator
- ▶ Vectors
- ▶ Matrices, Arrays, Factors, List, Data Frame
- ▶ Import/Export Data
- ▶ R Graphics
- ▶ Random number generating
- ▶ Writing R function
- ▶ for loops
- ▶ rep, seq, which, match

# Get Started

- Installation
  - ▶ google R → The R project for Statistical Computing
  - ▶ R64 bits (large datasets) vs. R32 bits
- ▶ Getting help with R
  - ▶ At the command prompt, type, for example `?read.table` or `help(read.table)`
  - ▶ At the command prompt, type, for example, `help.search("read")` or `apropos("read")`.
  - ▶ Within R, use the menu bar: **Help: R help**.
  - ▶ Quitting R. `q()`
  - ▶ How to save work space

## R Resources

John Verzani's SimpleR notes  
R Reference Card  
CRAN (Document/Manuals)

Note: To run some of the example in John Verzani's notes, run:

```
> install.packages("UsingR")  
> library(UsingR)
```

## R as a calculator

```
> 3 + 2
```

```
[1]5
```

```
> 7/2
```

```
[1]3.5
```

```
> 3 * 5
```

```
[1]15
```

```
> 2 ^ 3
```

```
[1]8
```

```
> 7%% 3 ## answer 1, modulo reduction
```

```
> log(1 : 4)
```

```
> log2(1 : 4)
```

```
> log(1 : 4, base = 3)
```

```
> exp(1)
```

```
[1]2.718282
```

```
> abs(-3)
```

```
> sqrt(3)
```

```
> sin(0.5)
```

# Vectors

Vectors contain elements of just 1 type, either numeric, logical, or character.

- ▶ Accessing elements in a vector: `[ ]`. (Very important)
- ▶ Use `c` to create a vector

(0)	(1)	(2)	(3)	(4)
-----	-----	-----	-----	-----

```
> x<- 3
> x # print x
>x<-4
>x
### Creating simple vectors
> x<- c(1,3.5,-28.4,10) #numerical vector
> y<-c("cat","dog","mouse","monkey") #character
> z<-c(TRUE,TRUE,TRUE,FALSE,FALSE) #logical vector
> x<-1:10
> seq(1, 10, by=1)
> seq(3,9, by=3)
> rep(2,10)
> log(seq(1,2,by=0.1))
>x <- c(1,5,10,NA,15)
> sum(x)
> sum(x,na.rm=TRUE)
> prod(x,na.rm=TRUE)
> mean(x,na.rm=TRUE)
```

## Accessing Elements in a Vector

```
> y <- c(8,32,15,-7, 2,19)
> length(y)
> y[3:5] ##position in vector as positive integer
> y[-c(1,5,6)] ## exclude: use negative integers
> y < 15
> y[y < 15]
> which(y == 32)
> x <- 1:10
> match(y, x)
> colors <- c("red", "blue", "pink", "yellow")
> which(colors == "yellow")
> x <- c(1,5,10,NA,15)
> which(is.na(x))
> which(!is.na(x))
> rep(c(1,2,3), 10)
> rep(c(1,2,3), each=10)
> rep(c(1,2,3), c(10, 9, 8))
```



&, |

```
>x<-c(T, T, F, F)
>y<-c(T, F, T, F)
>mat<-cbind(x,y)
>mat
>and<- x & y
>or<-x | y
>and
>or
```

## Factors

Factors: vectors with levels. Handy for regression modeling.

Example:

```
> colors <- c(1, 1, 2, 3)
> colors <-
factor(colors, label=c("red","green","blue"))
> table(colors)
colors
red green blue
2      1      1
```

# Matrices

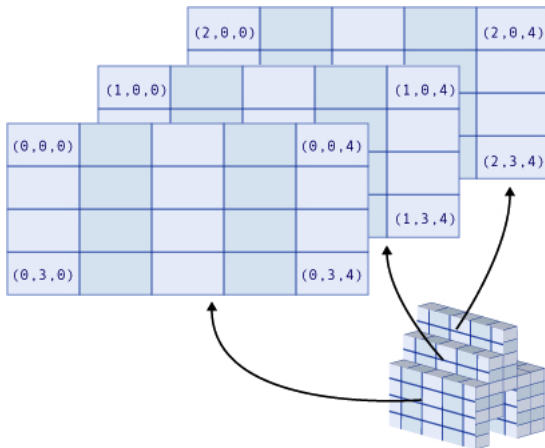
- ▶ Dimension: Row by column.
- ▶ Accessing elements in a matrix: [row, column].
- ▶ create a matrix

(0,0)				(0,4)
(1,0)				
(3,0)				(3,4)

```
> help(cbind)
> y <- c(8,32,15,-7, 2,19)
> x <- 1:6
> mat<- cbind(x,y)
> help(rbind)
> dim(mat) ## check dimension
> ncol(mat) ## the number of columns of a matrix
> nrow(mat) ## the number of rows of a matrix
> mat[2,3] # the value in the 2nd row and the 3rd column
> mat[1:3,] ## the first three row of mat
> mat[,2] ## the 2nd column of mat
> mat[-1,] ## exclude the first row
> newmat<-matrix(1:9, nrow=3) ## create new matrix
> newmat
> m<-matrix(1:9, nrow=3, byrow=T) ## fill row first
> colnames(m) <- c("a", "b", "c") ## label column name
> rownames(m) <- c("r1", "r2", "r3")
> vect<-as.vector(newmat)
```

# Arrays

- ▶ Dimension: Row by column by height.



```
> myarray<-array(1:64, dim=c(4,4,4))  
> myarray  
> myarray[1,2,3]
```

## Data Frames

Data Frame: like matrices, but each column can be a different data type.

```
>str(mydata)
```

```
'data.frame': 10 obs. of 3 variables:
```

```
$ y : num 24.2 26.6 23.9 23.6 23.6 ...
```

```
$ x1: num 3.02 2.43 3.35 3.86 3.7 É
```

```
$ x2: Factor w/ 2 levels "F","M": 2 2 2 2 2 2 1 1 1 1
```

# Lists

List: a bag contains different things (vectors, matrices, data frames,  $\dot{E}$ )

- ▶ Accessing components in a list: `[[ ]]`.
- ▶ Accessing to elements within components.



## Lists

```
> x <- list(one=c(18:36),two=c("AK","AL","AZ"),
           three=c(T,T,F,T),four=matrix(1:12,3,4))
> str(x)
List of 4
 $ one :  int [1:19] 18 19 20 21 22 23 24 25 26 27  É
 $ two :  chr [1:3] "AK" "AL" "AZ"
 $ three:  logi [1:4] TRUE TRUE FALSE TRUE
 $ four :  int [1:3, 1:4] 1 2 3 4 5 6 7 8 9 10 ...
> x[[1]]
> x$one
> y <- unlist(x)
```

## For loops

```
for(i in 1:100){  
  d <- Sys.time()  
  print(paste("Now is", d, sep=" "))  
  print(i)  
}
```

## Next Lecture

- ▶ R
  - ▶ Get Started
  - ▶ R as a calculator
  - ▶ Vectors
  - ▶ Matrices, Arrays, Factors, List, Data Frame
  - ▶ Import/Export Data
  - ▶ for loops
  - ▶ R Graphics
  - ▶ Random number generating
  - ▶ Writing R function
  - ▶ rep, seq, which, match