Lecture 6

Introduction to R

MCB 416A/516A
Statistical Bioinformatics and Genomic Analysis

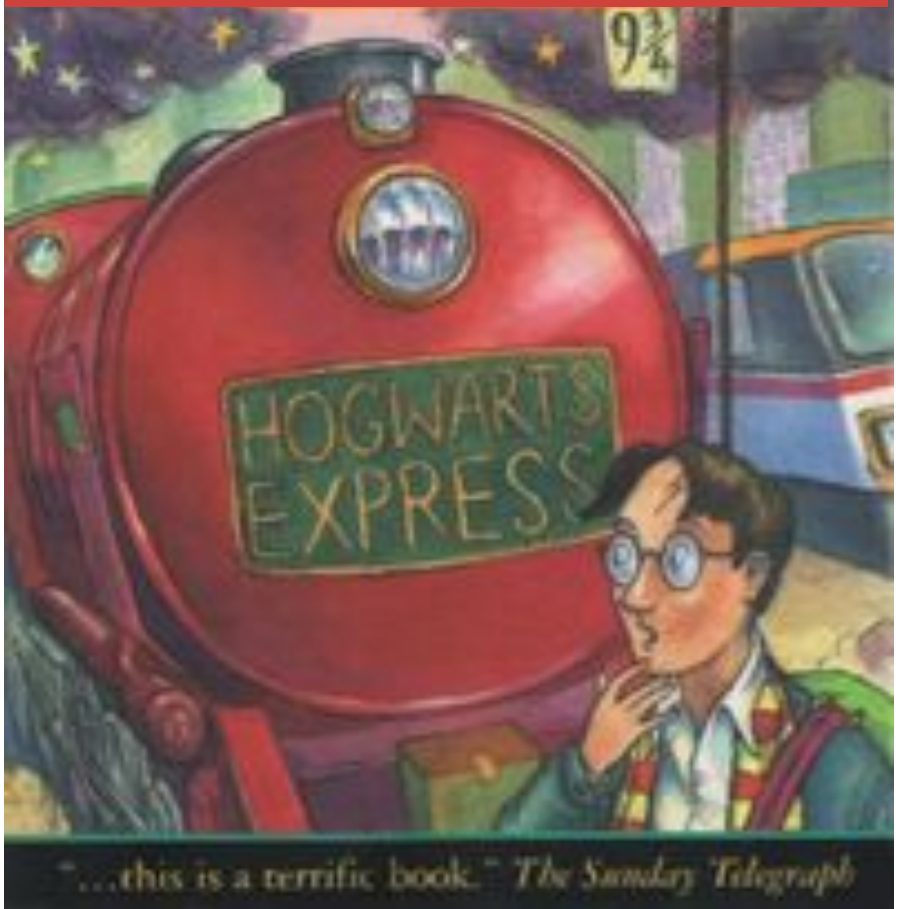
Prof. Lingling An
Univ of Arizona

- Last time we reviewed:

  - What life made of
  - DNA, RNA, protein
  - Variation and diversity
  - Bioinformatics

R programming language is a lot like magic... except instead of spells you have functions.

R, And the Rise of the Best Software Money Can't Buy



SPSS =

muggle

SPSS and SAS users are like muggles.

They are limited in their ability to change their environment. They have to rely on algorithms that have been developed for them. The way they approach a problem is constrained by how SAS/SPSS employed programmers thought to approach them. And they have to pay money to use these constraining algorithms.

=

wizard

R users are like wizards.

They can rely on functions (spells) that have been developed for them by statistical researchers, but they can also create their own. They don't have to pay for the use of them, and once experienced enough they are almost unlimited in their ability to change their environment.

History of R

- S: language for data analysis developed at Bell Labs circa 1976
- Licensed by *AT&T/Lucent* to *Insightful Corp*. Product name: *S-plus*.
- R: initially written & released as an open source software by Ross Ihaka and Robert Gentleman at U Auckland during 90s.
- Since 1997: international R-core team ~15 people & 1000s of code writers and statisticians happy to share their libraries! AWESOME!

What is it?

- R is an interpreted computer language.
  - –Most user-visible functions are written in R itself, calling upon a smaller set of internal primitives.
  - –It is possible to interface procedures written in C, C+, or FORTRAN languages for efficiency, and to write additional primitives.
  - –System commands can be called from within R
- R is used for data manipulation, statistics, and graphics. It is made up of:
  - –operators (+ - <- * %*% …) for calculations on arrays & matrices
  - –large, coherent, integrated collection of functions
  - –facilities for making unlimited types of publication quality graphics
  - –user written functions & sets of functions (packages); 10,000+ packages so far & growing.

Statistical Softwares – why R ?

- Common commerical statistical softwares: *SAS, SPSS, Stata, Statistica, Gauss, Splus*
- Costs

- R is a free version of *S+*
  - http://cran.R-project.org
- R is a *statistical language*
  - can perform any common statistical functions
  - Interactive : R studio

R: more advantages

- Fast and free.
- State of the art: Statistical researchers provide their methods as R packages. SPSS and SAS are years behind R!
- 2nd only to MATLAB for graphics.
- Active user community
- Excellent for simulation, programming, computer intensive analyses, etc.
- Forces you to *think* about your analysis.
- Interfaces with database storage software (SQL)

R-help listserve....

https://stat.ethz.ch/mailman/listinfo/r-help

Don't expect R to be like SAS/SPSS/Stata/etc…

Here's a synopsis of one person's story.

- He used SAS and, being a fan of open-source, attempted to learn R.
- He became frustrated with R and gave up.
- When he had a simple problem that he couldn't do in SAS, he quickly solved it with R.
- Then over about a month he became comfortable with R from consistent study of it. In hindsight he thinks that the initial problem was that he hadn't changed his way of thinking to match R's approach, and he wanted to master R immediately.

--Patrick Burns, UCLA Statistical Consultant

There are over 10,000+ packages

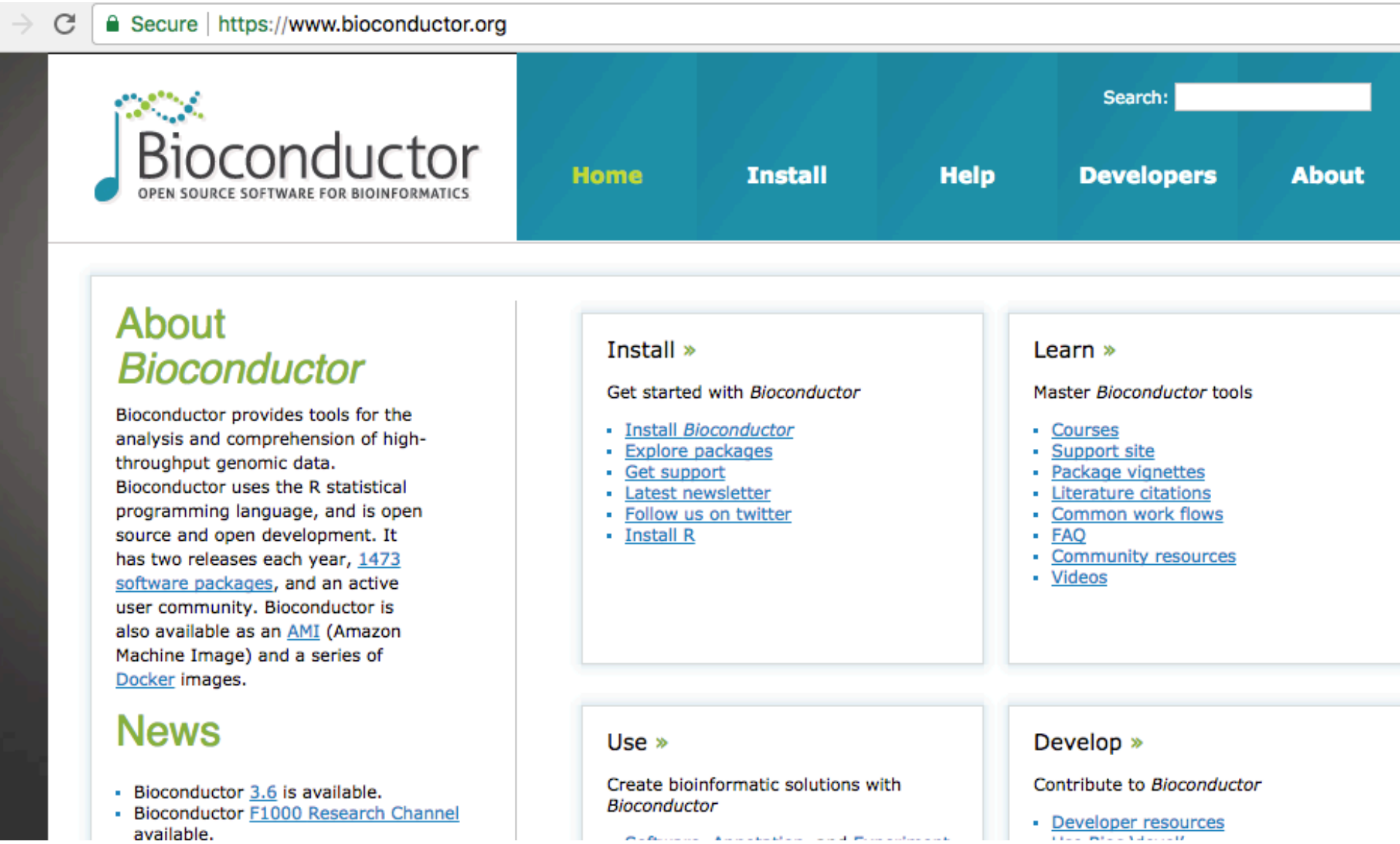
- This is an enormous advantage - new techniques available without delay, and they can be performed using the R language you already know.
- Allows you to build a customized statistical program suited to your own needs.

- Release twice a year
- Check what's new in the new version

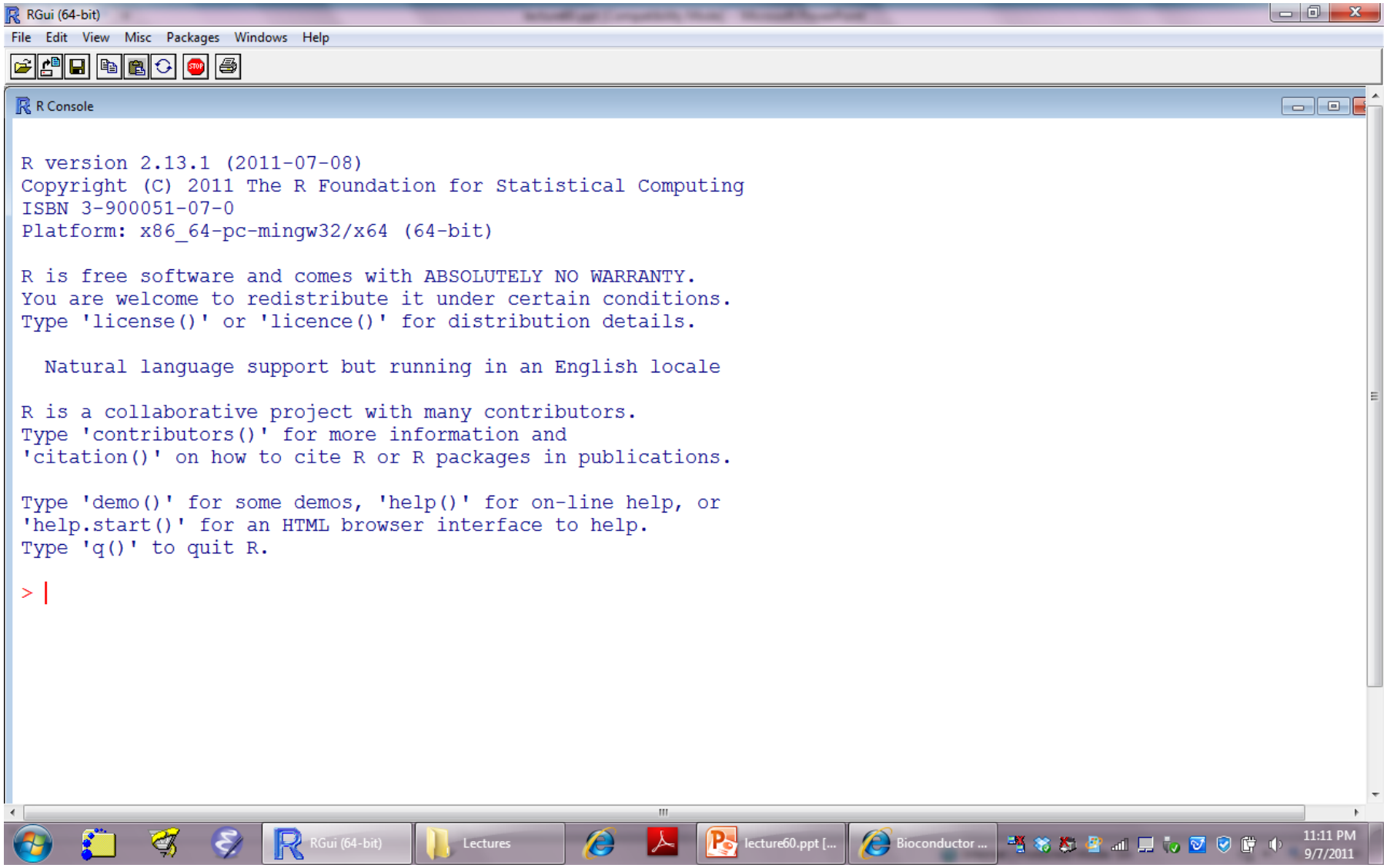
A particular R strength: Bioconductor

- Bioconductor provides tools for the analysis and comprehension of high-throughput genomic data
- Bioconductor uses the R statistical programming language, and is open source and open development.
- It has two releases each year, 1,473 software packages, and an active user community .

http://www.bioconductor.org/


Secure | https://www.bioconductor.org
Bioconductor
OPEN SOURCE SOFTWARE FOR BIOINFORMATICS
Search:
Home
Install
Help
Developers
About
About Bioconductor
Bioconductor provides tools for the analysis and comprehension of high-throughput genomic data. Bioconductor uses the R statistical programming language, and is open source and open development. It has two releases each year, 1473 software packages, and an active user community. Bioconductor is also available as an AMI (Amazon Machine Image) and a series of Docker images.
News
Bioconductor 3.6 is available.
Bioconductor F1000 Research Channel available.
Install »
Get started with Bioconductor
Install Bioconductor
Explore packages
Get support
Latest newsletter
Follow us on twitter
Install R
Learn »
Master Bioconductor tools
Courses
Support site
Package vignettes
Literature citations
Common work flows
FAQ
Community resources
Videos
Use »
Create bioinformatic solutions with Bioconductor
Develop »
Contribute to Bioconductor
Developer resources

R: get started - Screenshot



Getting Started

- How to use help in R?
  - R has a very good help system built in.
  - If you know which function you want help with simply use ?_____ with the function in the blank.
  - Ex: ?hist
  - If you don't know which function to use, then use help.search("_____")
  - Ex: help.search("histogram")

R Environments

- Prompt: >
- Current working direction: **`getwd()`**
- Change working direction: **`setwd("T:/stats")`**
- Alternative approach – click "file" then choose "change dir"

R Grammar

- object <- function(arguments)

```
reg <- lm(y ~ x)
```

or:

```
reg = lm(y ~ x)
```

- Operations

```
x == 5          x equals to 5
x != 5          x is not equal to 5
y < x           y is less than x
x > y           x is greater y
z <= 7          z is less than or equal to 7
p >= 1          p is greater than or equal to 1
is.na(x)        Is x a missing value?
A & B           A and B
A | B           A or B
!               not
```

R Grammar -2

- Case sensitivity

```
a <- 5
A <- 7
B <- a+A
```

- Name of variable must NOT contain blank

```
var a <- 5
```

- but can include a “.”

```
var.a <- 5
var.b <- 10
var.c <- var.a + var.b
```

Dataframe

Dataset	data.frame
columns	variables
rows	observations

age	insulin
50	16.5
62	10.8
60	32.3
40	19.3
48	14.2
47	11.3
57	15.5
70	15.8

Data frame: `ins`

Variables: `age, insulin`

Number of observations: 8

Data entry by c ( )

age	insulin
50	16.5
62	10.8
60	32.3
40	19.3
48	14.2
47	11.3
57	15.5
70	15.8

```
age <- c(50,62,60,40,48,47,57,70)
insulin <- c(16.5,10.8,32.3,19.3,14.2,11.3,
             15.5,15.8)
ins <- data.frame(age, insulin)
attach(ins)
ins
```

```
  age insulin
1  50    16.5
2  62    10.8
3  60    32.3
4  40    19.3
5  48    14.2
6  47    11.3
7  57    15.5
8  70    15.8
```

Read data from external file: `read.table()`

id	sex	age	bmi	hdl	ldl	tc	tg
1	Nam	57	17	5.000	2.0	4.0	1.1
2	Nu	64	18	4.380	3.0	3.5	2.1
3	Nu	60	18	3.360	3.0	4.7	0.8
4	Nam	65	18	5.920	4.0	7.7	1.1
5	Nam	47	18	6.250	2.1	5.0	2.1
6	Nu	65	18	4.150	3.0	4.2	1.5
7	Nam	76	19	0.737	3.0	5.9	2.6
8	Nam	61	19	7.170	3.0	6.1	1.5
9	Nam	59	19	6.942	3.0	5.9	5.4
10	Nu	57	19	5.000	2.0	4.0	1.9
				...			
46	Nu	52	24	3.360	2.0	3.7	1.2
47	Nam	64	24	7.170	1.0	6.1	1.9
48	Nam	45	24	7.880	4.0	6.7	3.3
49	Nu	64	25	7.360	4.6	8.1	4.0
50	Nu	62	25	7.750	4.0	6.2	2.5

```
setwd("T:/works/r")
chol <- read.table("chol.txt", header=TRUE)
```

Read data from an excel, SPSS file: `read.csv()`, `read.spss`

- **Save excel file in .csv format**
- **Use R to read the file:**

```
setwd("T:/works/r")
gh <- read.csv ("excel.csv", header=TRUE)
Mu<-read.xlsx("example.xlsx", header=TRUE)
```

- SPSS file: test.sav
- Use R to read the file via the foreign package

```
library(foreign)
setwd("T:/works/r")
testo <-read.spss("test.sav",to.data.frame=TRUE)
```

Install packages

- install.packages("Hmisc“)
  - select a CRAN mirror for use in this session
- Alternative way:
  - Check “packages”
  - Click “install packages”
  - Choose the package - Hmisc

Subsetting dataset

```
setwd("T:/works/r")
chol <- read.table("chol.txt", header=TRUE)
attach(chol)

nam <- subset(chol, sex=="Nam")
nu <- subset(chol, sex=="Nu")
old <- subset(chol, age>=60)
n60 <- subset(chol, age>=60 & sex=="Nam")
```

Merge two datasets

```
d1
 id sex tc
 1 Nam 4.0
 2  Nu 3.5
 3  Nu 4.7
 4 Nam 7.7
 5 Nam 5.0
 6  Nu 4.2
 7 Nam 5.9
 8 Nam 6.1
 9 Nam 5.9
10  Nu 4.0
```

```
d2
 id sex tg
 1 Nam 1.1
 2  Nu 2.1
 3  Nu 0.8
 4 Nam 1.1
 5 Nam 2.1
 6  Nu 1.5
 7 Nam 2.6
 8 Nam 1.5
 9 Nam 5.4
10  Nu 1.9
11  Nu 1.7
```

```
d <- merge(d1, d2, by="id", all=TRUE)
d
   id sex.x  tc sex.y  tg
1   1   Nam 4.0   Nam 1.1
2   2    Nu 3.5    Nu 2.1
3   3    Nu 4.7    Nu 0.8
4   4   Nam 7.7   Nam 1.1
5   5   Nam 5.0   Nam 2.1
6   6    Nu 4.2    Nu 1.5
7   7   Nam 5.9   Nam 2.6
8   8   Nam 6.1   Nam 1.5
9   9   Nam 5.9   Nam 5.4
10 10    Nu 4.0    Nu 1.9
11 11  <NA>  NA    Nu 1.7
```

Data coding

```
bmd <- c(-0.92,0.21,0.17,-3.21,-1.80,-2.60,
         -2.00,1.71,2.12,-2.11)
diagnosis <- bmd
diagnosis[bmd <= -2.5] <- 1
diagnosis[bmd > -2.5 & bmd <= (-1.0)] <- 2
diagnosis[bmd > -1.0] <- 3
data <- data.frame(bmd, diagnosis)
data
```

```
     bmd   diagnosis
1  -0.92           3
2   0.21           3
3   0.17           3
4  -3.21           1
5  -1.80           2
6  -2.60           1
7  -2.00           2
8   1.71           3
9   2.12           3
10 -2.11           2
```

```
diagnosis <- bmd
diagnosis <- replace(diagnosis, bmd <= -2.5, 1)
diagnosis <- replace(diagnosis, bmd > -2.5 & bmd <= (-1.0),2)
diagnosis <- replace(diagnosis, bmd > -1.0, 3)
```

Grouping data

```
library(Hmisc)
bmd <-
  c(-0.92,0.21,0.17,-3.21,-1.80,-2.60,
          -2.00,1.71,2.12,-2.11)


group <- cut2(bmd, g=2)
table(group)

group
[-3.21,-0.92) [-0.92, 2.12]
            5             5
```

R as a calculator

- Arithmetic calculations

```
> -27*12/21
[1] -15.42857

> sqrt(10)
[1] 3.162278

> log(10)
[1] 2.302585

> log10(2+3*pi)
[1] 1.057848

> exp(2.7689)
[1] 15.94109

> (25 - 5)^3
[1] 8000

> cos(pi)
[1] -1
```

```
Permulation: 3!
> prod(3:1)
[1] 6

# 10.9.8.7.6.5.4
> prod(10:4)
[1] 604800

> prod(10:4)/prod(40:36)
[1] 0.007659481

> choose(5, 2)
[1] 10

> 1/choose(5, 2)
[1] 0.1
```

R as a number generator

- Sequence – `seq(from, to, by= )`
  - Generate a variable with numbers ranging from 1 to 12:

```
> x <- (1:12)
> x
[1]  1  2  3  4  5  6  7  8  9 10 11 12

> seq(12)
[1]  1  2  3  4  5  6  7  8  9 10 11 12

> seq(4, 6, 0.25)
[1] 4.00 4.25 4.50 4.75 5.00 5.25 5.50 5.75 6.00
```

R as a number generator

- Repetition – `rep(x, times, …)`

```
> rep(10, 3)
[1] 10 10 10

> rep(c(1:4), 3)
 [1] 1 2 3 4 1 2 3 4 1 2 3 4

> rep(c(1:4), each=3)
 [1] 1 1 1 2 2 2 3 3 3 4 4 4

> rep(c(1.2, 2.7, 4.8), 5)
 [1] 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8 1.2 2.7 4.8 1.2
   2.7 4.8
```

R as a number generator

- Generating levels – `gl(n, k, length = n*k)`

```
> gl(2,4,8)
[1] 1 1 1 1 2 2 2 2
Levels: 1 2

> gl(2, 10, length=20)
[1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
Levels: 1 2
> gl(2, 10)
[1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
Levels: 1 2
> gl(2, 8, labels = c("Control", "Treat"))
 [1] Control Control Control Control Control Control Control
   Control Control Treat
[10] Treat   Treat   Treat   Treat   Treat   Treat   Treat
Levels: Control Treat
```

R as a probability calculator

Binomial probability

$$P(k \mid n, p) = C_k^n p^k (1-p)^{n-k}$$

```
dbinom(k, n, p)
```

```
> dbinom(2, 3, 0.60)
[1] 0.432
```

Poisson probability

$$P(X = k \mid \lambda) = \frac{e^{-\lambda}\lambda^k}{k!}$$

```
dpois(k, l)
```

$$P(X = 2 \mid \lambda = 1) = \frac{e^{-2}1^2}{2!} = 0.1839$$

```
> dpois(2, 1)
[1] 0.1839397
```

R as a probability calculator

Normal probability

$$P(a \le X \le b) = \int_a^b f(x)\,dx$$

`pnorm (a, mean, sd)` $= \int_{-\infty}^{a} f(x)\,dx$

$= P(X \le a \mid \text{mean, sd})$

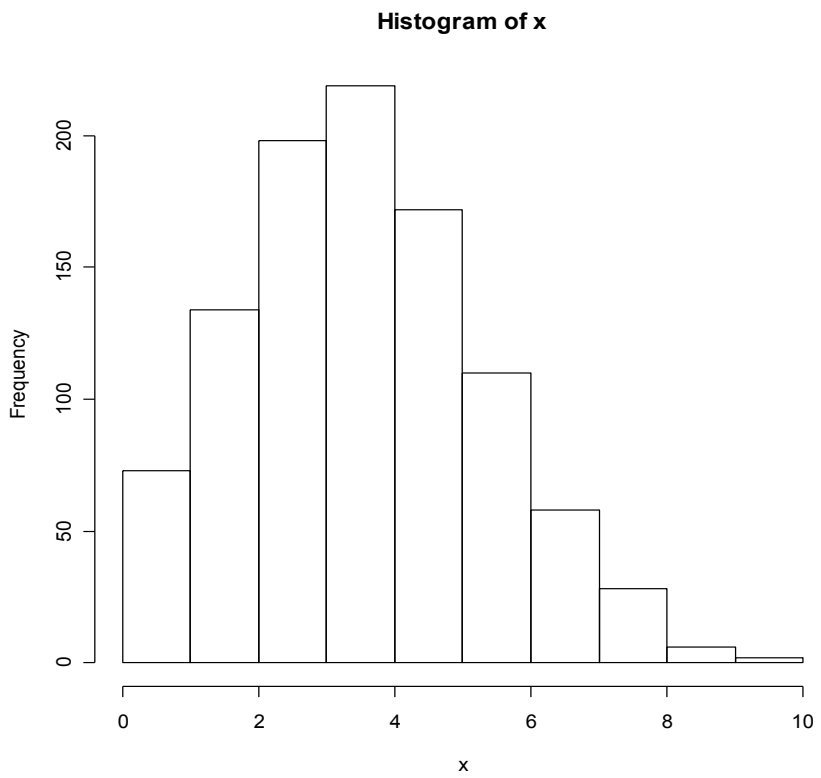
Probability of height less than or equal to 150 cm, given that the distribution has mean=156 and sd=4.6

```
> pnorm(150, 156, 4.6)
[1] 0.0960575
```

R as a simulator – Binomial distribution

- In a population, 20% have a disease, if we do 1000 studies; each study selects 20 people from the population. In each study, we observe the number of people with disease. Let this number be x. What is the distribution of 1000 values of x ?

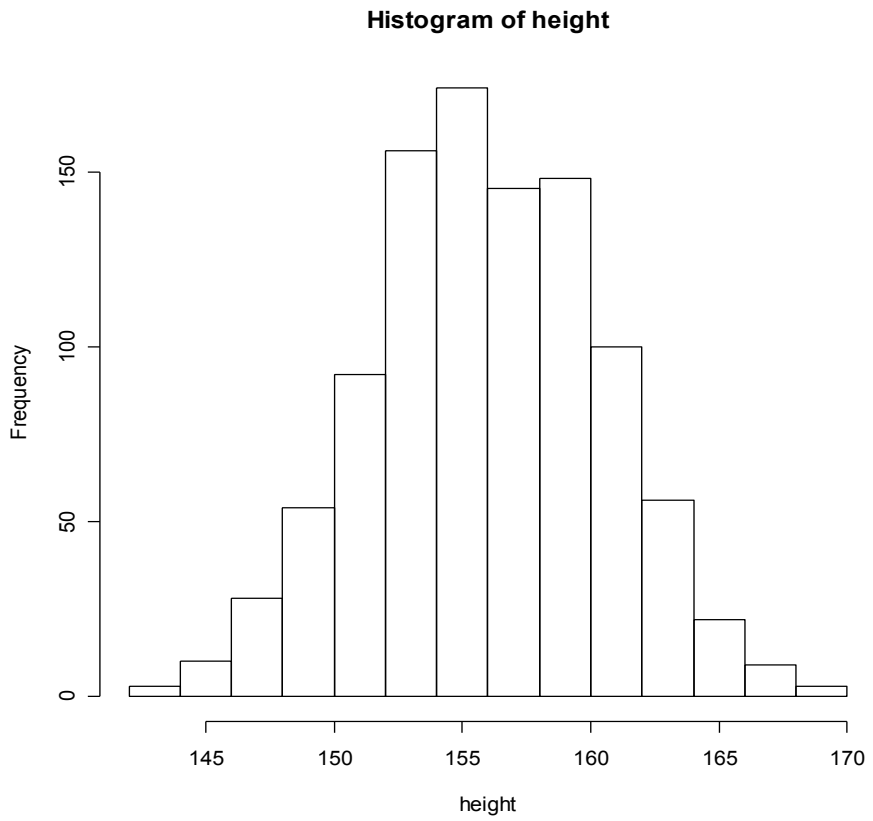
```
x <- rbinom(1000, 20, 0.20)
hist(x)
```



R as a simulator – Normal distribution

- Average height of Vietnamese women is 156 cm, with standard deviation being 4.6 cm. If we randomly take 1000 women from this population, what is the distribution of height?

```
height <- rnorm(1000, mean=156, sd=4.6)
hist(height)
```



R as a sampler

- We have 40 people (1,2,3,…,40). If we randomly select 5 people from the group, who would be selected?

```
>sample(1:40, 5)
[1] 32 26  6 18  9

>sample(1:40, 5)
[1]  5 22 35 19  4

>sample(1:40, 5)
[1] 24 26 12  6 22

>sample(1:40, 5)
[1] 22 38 11  6 18
```

`>set.seed(321)`

Sampling with Replacement

- **Sampling with replacement:** If we want to sample 10 people from a group of 50 people. However, each time we select one, we put the id back and select from the group again.

```
>sample(1:50, 10, replace=T)
 [1] 31 44  6  8 47 50 10 16 29 23
```

Summary

- R is an interactive statistical language
- Extremely flexible and powerful
- Data manipulation and coding
- Can be used as a calculator, simulator and sampler
- FREE!

- More advanced statistical analyses

Final Words of Warning

- "Using R is a bit akin to smoking. The beginning is difficult, one may get headaches and even gag the first few times. But in the long run, it becomes pleasurable and even addictive. Yet, deep down, for those willing to be honest, there is something not fully healthy in it."

  --Francois Pinard