R basics

Elie Gurarie

Biol 709 - Lecture 1b
January 2, 2018

January 2, 2018

What is R?

- R is a language and environment for statistical computing, graphics and much more
- It is a (**open source**) GNU project which is similar to the S language and environment developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers.
- R can be considered as a different implementation of S with more flexibility and power gained from contributions by other users.

What is R?

- R is a language and environment for statistical computing, graphics and much more
- It is a (**open source**) GNU project which is similar to the S language and environment developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers.
- R can be considered as a different implementation of S with more flexibility and power gained from contributions by other users.

What is R?

- R is a language and environment for statistical computing, graphics and much more
- It is a (**open source**) GNU project which is similar to the S language and environment developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers.
- R can be considered as a different implementation of S with more flexibility and power gained from contributions by other users.

What is R?

- an effective **data handling** and storage facility
- a large, coherent, integrated collection of tools for **data analysis**
- **graphical facilities** for data analysis and display either on-screen or on hardcopy
- a **well-developed**, simple and effective programming language including traditional statements such as conditionals, loops, and user-defined functions.

What is R?

- an effective **data handling** and storage facility
- a large, coherent, integrated collection of tools for **data analysis**
- **graphical facilities** for data analysis and display either on-screen or on hardcopy
- a **well-developed**, simple and effective programming language including traditional statements such as conditionals, loops, and user-defined functions.

What is R?

- an effective **data handling** and storage facility
- a large, coherent, integrated collection of tools for **data analysis**
- **graphical facilities** for data analysis and display either on-screen or on hardcopy
- a **well-developed**, simple and effective programming language including traditional statements such as conditionals, loops, and user-defined functions.

What is R?

- an effective **data handling** and storage facility
- a large, coherent, integrated collection of tools for **data analysis**
- **graphical facilities** for data analysis and display either on-screen or on hardcopy
- a **well-developed**, simple and effective programming language including traditional statements such as conditionals, loops, and user-defined functions.

Some historical background

R is the most recent and full-featured implementation of the S language.

- Original S - AT & T Bell Labs
- S-PLUS (S plus a GUI)
  - Statistical Sciences, Inc.
  - Mathsoft, Inc.
  - Insightful, Inc.
  - Tibco, Inc.
- R - The R Project for Statistical Computing

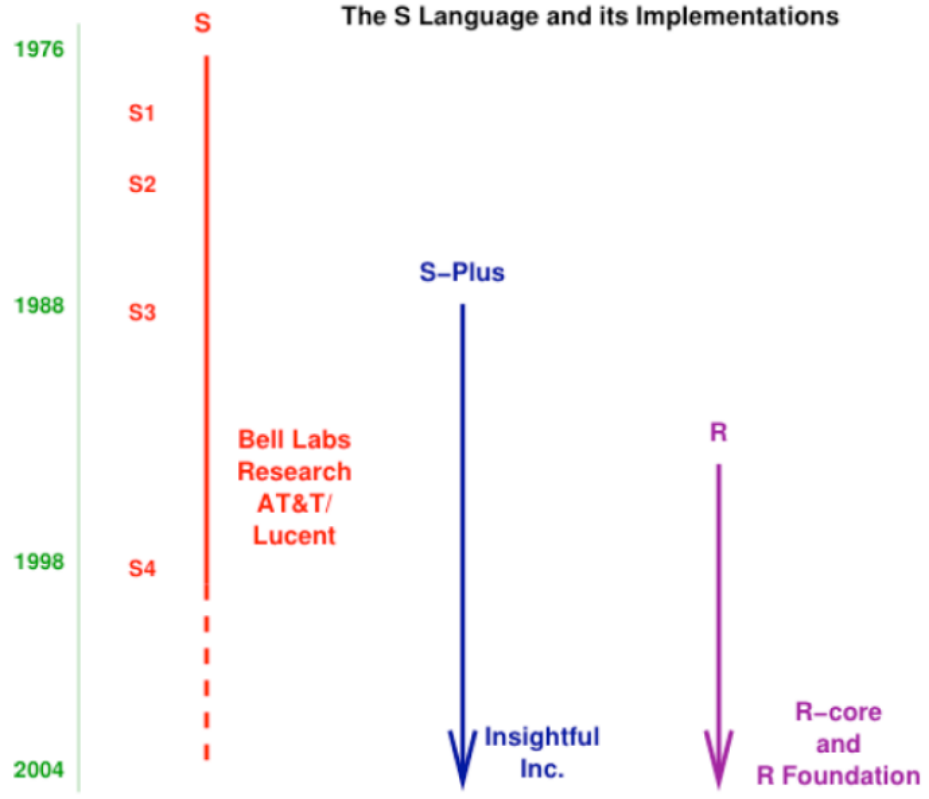
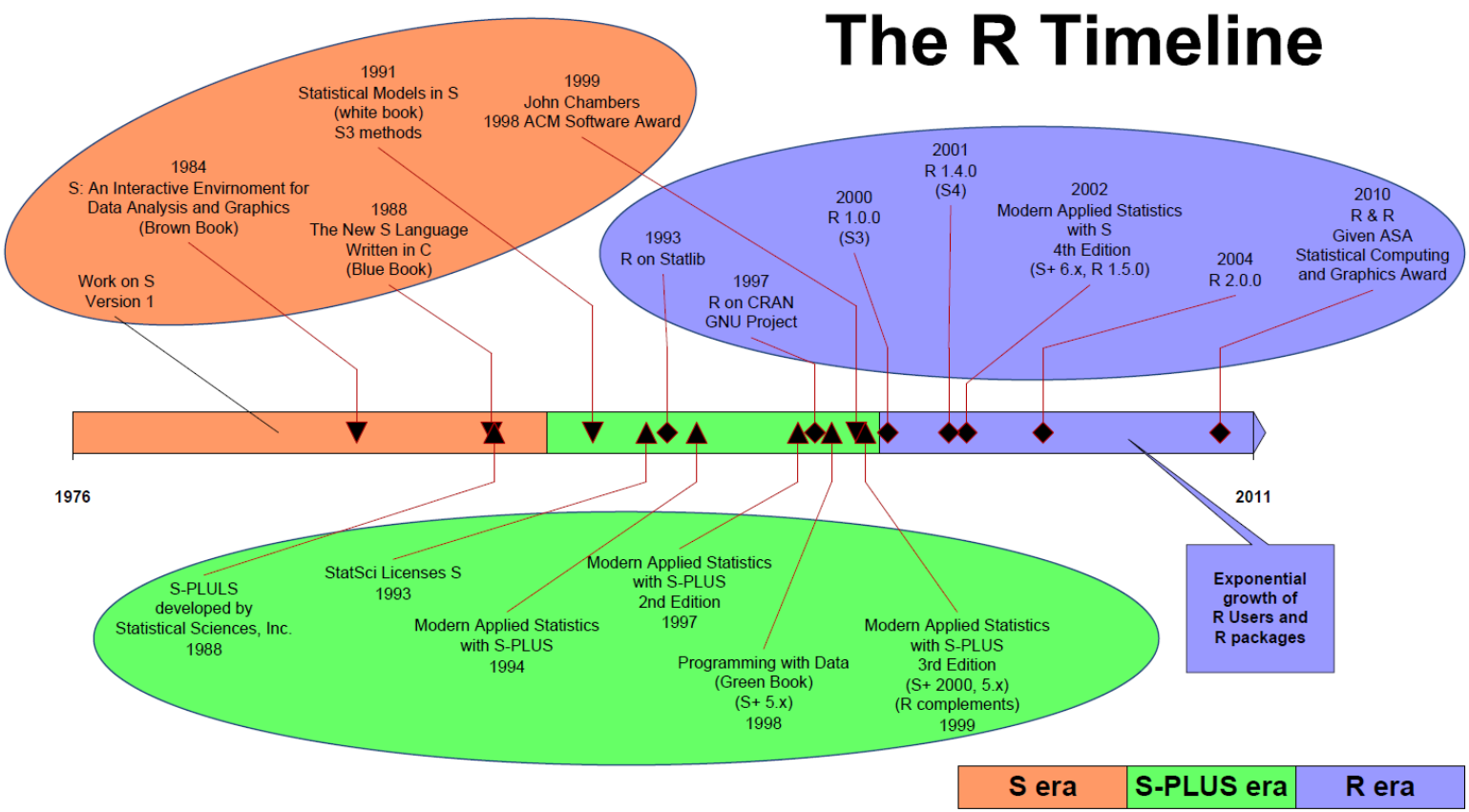


Figure from *The History of S and R, John Chambers, 2006*

Some historical background

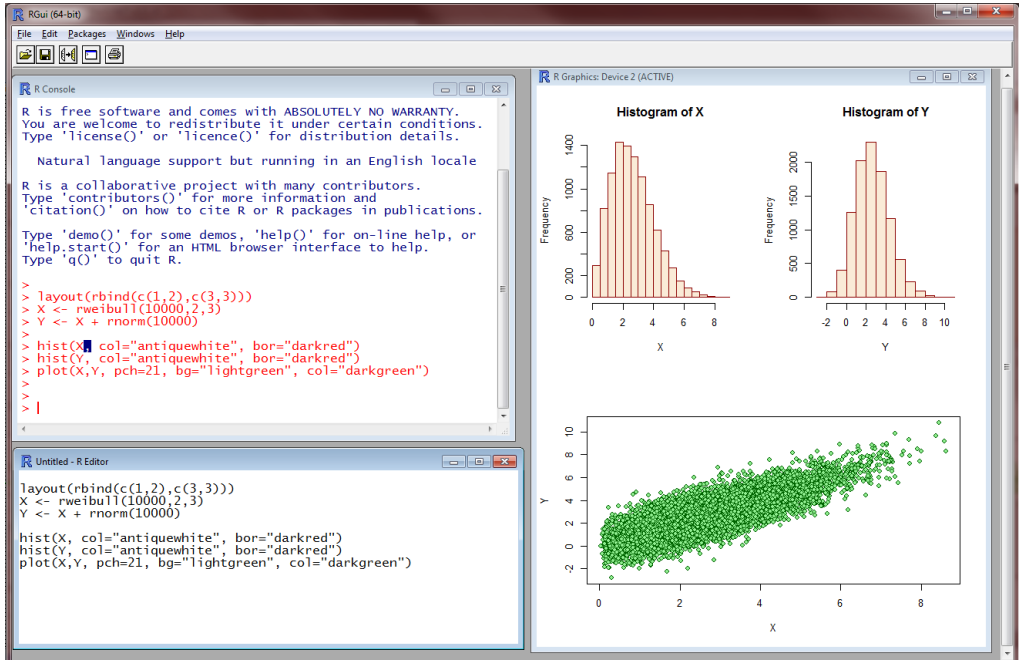


1

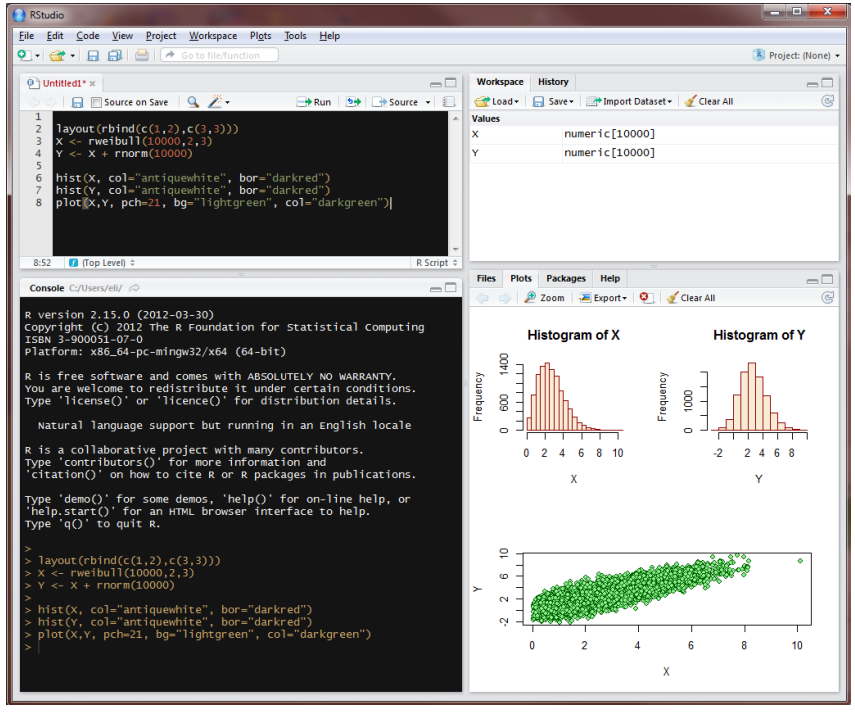
[1] slide thanks to G. Yollin

Running R

R GUI
(Graphical user interface)



RStudio IDE
(Integrated development environment)



Running scripts in R

Run this code in R:

```
par(mar=c(0,0,0,0))
for(i in 1:400)
{
  cols <- rainbow(i, alpha=1:i/i)
  Z <- complex(mod=sqrt(1:i), arg=1:i + i/20)
  plot(Z, col=cols, pch=19, cex=sqrt(i:1), asp=1)
}
```

Do this in four ways:

1. Carefully copy the code by hand into the R-console.
2. Copy it and paste it directly into the R-console.
3. Open a script window in the Rgui, paste it into the window, highlight everything (`Ctrl-A`), and hit `Ctrl-R`
4. Paste into Rstudio, highlight, and hit `Ctrl-Enter`

Running scripts in R

Run this code in R:

```
# Rainbow bubbles!
par(mar=c(0,0,0,0))
for(i in 1:400)
{
  cols <- rainbow(i, alpha=1:i/i)
  Z <- complex(mod=sqrt(1:i), arg=1:i + i/20)
  plot(Z, col=cols, pch=19, cex=sqrt(i:1), asp=1)
}
```

- Note that the # denotes a comment.

```
> 1+2 # This is a comment that we are adding 1 + 2!
```

- A good rule for coding: Use more comments than you think is necessary!

R Basics

R can be a calculator. **R code: Calculator**

```
# addition, substraction, divisi
3+4
```

```
## [1] 7
```

```
6*7
```

```
## [1] 42
```

```
13/17
```

```
## [1] 0.7647059
```

```
5^17
```

```
## [1] 762939453125
```

```
sqrt(16)
```

```
## [1] 4
```

Note that the input comes after the caret (>) and the output comes after a '[1]', because in all of these cases, there is on only one element to the output.

Functions

R uses *functions*:

- Functions can be defined to take zero or more arguments
- Functions typically (but not always) return a value
- Functions are called by name with arguments enclosed in parentheses, even if the function has no arguments the parentheses are required

R code: Calling Functions

```
sin(pi/2)

## [1] 1

print("Hello, world")

## [1] "Hello, world"

abs(-8)

## [1] 8

cos(2*sqrt(2))

## [1] -0.9513631

date()

## [1] "Tue Jan 02 11:56:52 2018"
```

Functions

Help files:

- Every function comes with a "help" page accessed via "?"
  - try: `?sin`, `?log`, `?abs`
- The help page contains instructions for input and output, some examples, links to the help files of related functions.

R code: Calling Functions

```
sin(pi/2)

## [1] 1

print("Hello, world")

## [1] "Hello, world"

abs(-8)

## [1] 8

cos(2*sqrt(2))

## [1] -0.9513631

date()

## [1] "Tue Jan 02 11:56:52 2018"
```

The big question is: ...

How do you know a function exists?

- Short answer: You don't!
- Long answer: You learn about in (in this class, or books), you get your hands on as much code as you can, you search on-line, or you create your own.

Assignment operators

An extremely important feature of R (and all programming languages) is the ability to assign a value to an arbitrarily named variable.

This is done in one of three ways:

- Assignment operator: '<-' (or '->')
  - This is the MOST COMMON METHOD.
- Assignment function: `assign()`
- A simple equal sign: '='
  - Note: In some contexts, this can be ambiguous

R code: Storing values in variables

```
# mass of electron
m <- 9.1e-31
m

## [1] 9.1e-31

# speed of light
assign("c", 299792458)
c

## [1] 299792458

# energy
E = m*c^2
E

## [1] 8.178672e-14
```

Vectors

Vectors are lists of objects of a certain type. They are most commonly formed using the `c()` function.

R code: Making vectors

```
constants <- c(3.1416,2.7183,1.4142,1.6180); constants

## [1] 3.1416 2.7183 1.4142 1.6180

my.labels <- c("pi","euler","sqrt2","golden"); my.labels

## [1] "pi"     "euler"  "sqrt2"  "golden"
```

The last example involves a type of variable called a *character*. Note that they MUST BE in quotes, otherwise R thinks it is the name of an object in memory. You can explore the "type" of variable you have stored in memory using the `is()` function:

R code: Identifying variable types

```
is(constants)

## [1] "numeric" "vector"

is(my.labels)
```

Vectors

There are often shortcuts to making useful vectors. For example:

```
# 1. a sequence of numbers
 c(1,2,3,4,5,6,7,8,9,10)

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

 1:10

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

 seq(1,10)

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

```
# 2. Even Numbers
c(2,4,6,8,10,12)

## [1]  2  4  6  8 10 12

seq(2,12,2)

## [1]  2  4  6  8 10 12

1:6*2

## [1]  2  4  6  8 10 12
```

```
# 3. letters
c("A","B","C","D","E")

## [1] "A" "B" "C" "D" "E"

LETTERS[1:5]

## [1] "A" "B" "C" "D" "E"
```

Note, the difference between `[]` as opposed to the `()`.

More functions we will use:

In the lab we learn more about generating and subsampling objects like vectors and matrices. Here is a list of some of the functions we will be using:

R lab 1: Functions

```
c()
seq()
rep()
plot()
points()
lines()
curve()
matrix()
which()
```