STAT 302
Statistical Software and Its Applications
Introduction to R

Yen-Chi Chen

Department of Statistics, University of Washington

Spring 2017

Statistical Software

- There are many, many statistical packages, see http://en.wikipedia.org/wiki/List_of_statistical_packages
- We start out with R and follow it by introducing SAS
- SAS is favored by corporations, because it is backed by a corporation.
- There is no corporation behind R, it is supported by a worldwide consortium of developers and users.
- R and SAS $\Longrightarrow$ marketability to prospective employers.
- The yearly SAS license is expensive, R is free.

Introduction to R

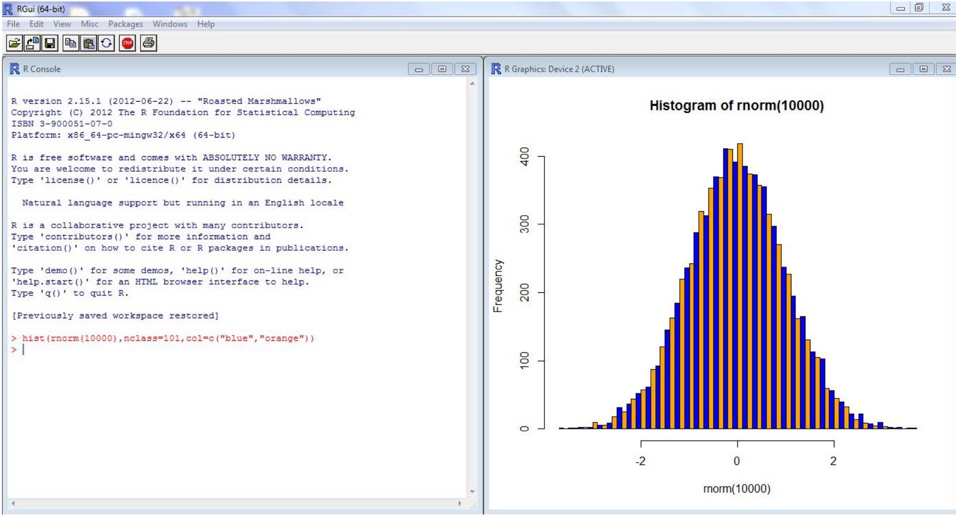
- I assume you installed R and RStudio on your computers.
- What is R?
- R is a free software environment for statistical computing and graphics.
- It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS.
- It is a gigantic calculator and a programming tool.
- Flexible data manipulation tool.
- It produces gaphics in many formats, even animation.
- Many statistical analysis tools in its basic installation.
- Today, R has more than 9000 available packages.
- It can be extended via C, C++, Fortran (for speed).

Interface to R

- On Windows it comes with an RGui interface. It opens up when you double click the large blue R on your desktop.
- On a Mac you have an application called R. When you run it, it opens a console window, similar to the above Rgui, but different.
- On Linux you type the command R in a terminal window. This turns the terminal into a command console.
- You can work with any plain text editor to store your commands.
- Cut and paste your commands from the editor into the command console.
- We will use RStudio as common interface for all common computers.
- It has a built in editor and many other utilities.

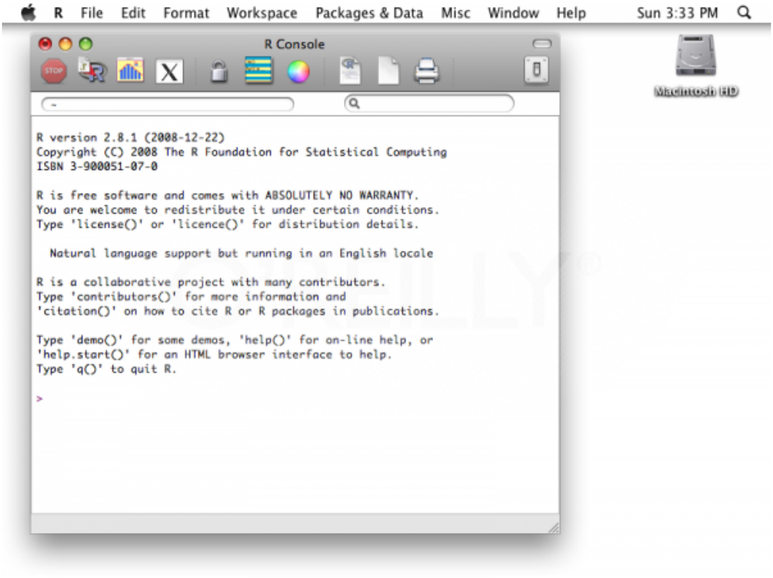
The RGui Interface to R

Windows only, different menu options, depending on active panel (here R Console)



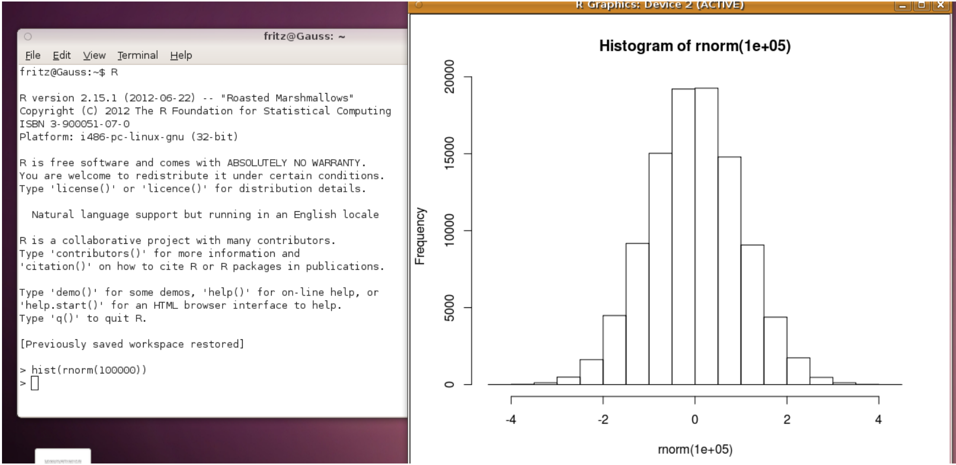
```
R version 2.15.1 (2012-06-22) -- "Roasted Marshmallows"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-pc-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> hist(rnorm(10000),nclass=101,col=c("blue","orange"))
> 
```

The Mac R Console



```
R version 2.8.1 (2008-12-22)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

The Linux R Console and Graphics Window

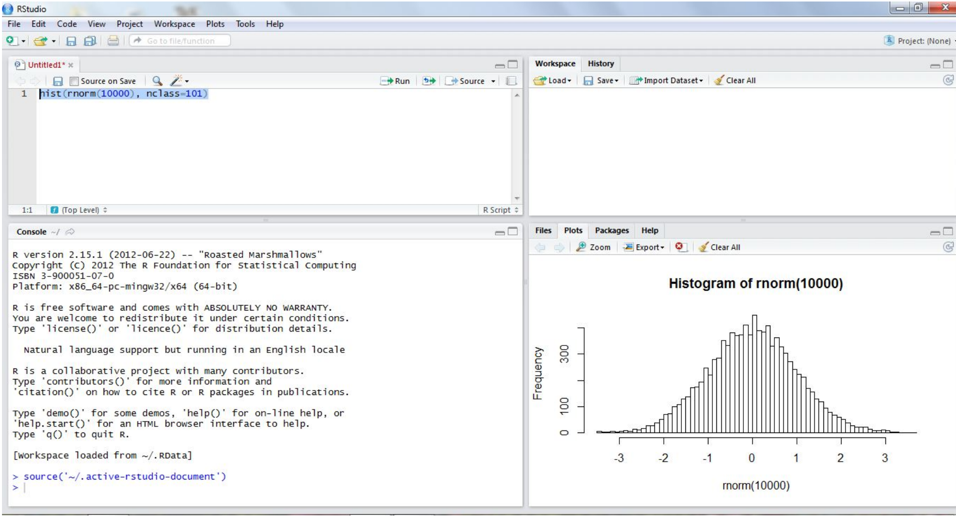


fritz@Gauss: ~
File Edit View Terminal Help
R Graphics: Device 2 (ACTIVE)
Histogram of rnorm(1e+05)
Frequency
0
5000
10000
15000
20000
-4
-2
0
2
4
rnorm(1e+05)

```
fritz@Gauss:~$ R

R version 2.15.1 (2012-06-22) -- "Roasted Marshmallows"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i486-pc-linux-gnu (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> hist(rnorm(100000))
> 
```

The RStudio Interface to R

same for Windows/Mac/Linux



```
hist(rnorm(10000), nclass=101)
```

```
R version 2.15.1 (2012-06-22) -- "Roasted Marshmallows"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-pc-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> source('~/.active-rstudio-document')
>
```

Creating Project Workspaces

- Create separate workspaces for different projects.
- Avoids clutter/confusion among all saved workspace objects.
- Within RStudio File $\Longrightarrow$ New Project
- $\Longrightarrow$ New Directory $\Longrightarrow$ Empty Project
- Enter a directory name, say Lab1.
- $\Longrightarrow$ Browse to a location where to place that working directory
- Push Create Project button.
- This switches you to a new instance of RStudio in Lab1.
- At the command prompt > type in `x <- 1:5`. Then type `x`.
- Don't type `x < - 1:5`. But try anyway and decipher result.
- After exiting that RStudio session (using `q()`) you can reopen that R session by double clicking the blue R icon in Lab1.
- Note that the `x` object is still part of your workspace if you choose 'Save workspace image'.

R as Calculator

Using RStudio

- If the Edit window is not open, then ⟹ File ⟹ New File ⟹ choose R Script or use shortcut `Ctrl+Shift+N` (Windows) or `Shift + Command + N` (Mac).
- Note the use of unexecuted comments preceded by `#`.
- In the Edit or R Script pane of RStudio's upper left enter

```
sqrt(3^2+4^2)   # or sqrt(3**2+4**2)
qnorm(.975) # normal 97.5 percentile, .975 quantile
pnorm(qnorm(.975)) # left tail prob. of qnorm(.975)
```

highlight these lines, click Run in the Edit pane, upper right.

- This executes the highlighted commands in the Console below.
- You can enter the same commands also directly after the > prompt in the Console pane.

The Difference of Source and Run

Using RStudio

- Use Run and Source buttons on these lines in Edit pane

```
exp(1) # Euler's constant
print(exp(1))
```

- Run will echo and execute the cursor line or the highlighted lines and show the results in the Console pane.
- Source will run the whole script in the Console pane, but to show results it needs the `print()` wrapper around commands that would show results at the command prompt >.
- The advantage of using the Edit pane and of running/sourcing commands is that you can try out a sequence of commands and then edit/change them for repeat tries.

Using Functions to Build up Scripts
Using RStudio

- Source the following script in the Edit pane

```
myfun <- function(x){
exp(x) # exponential function
}
```

- Note that the Workspace pane on the upper right now shows the `myfun` object, identified as function object, in addition to the integer object `x`.
- In the Console execute the command `myfun(1)`.
- Functions can have more than one argument.

```
 myfun2 <- function(x,y,z){x+y^2+z^3}
```

Try to run `myfun2(1,2,3)`.

Vector Functions

Using RStudio

- Previously used `myfun` on single number arguments. Now try

```
> myfun(x)
[1]   2.718282   7.389056  20.085537  54.598150 148.413159
```

- Evaluation of `exp(x)` is vectorized over all components of `x`.
- Vectorize computations whenever possible, avoid loops.

R Reference Materials

- Look under Manuals in http://cran.r-project.org/ and you find An Introduction to R
- and further down, under contributed documentation, many guides in many languages.
- See also http://en.wikipedia.org/wiki/R_%28programming_language%29
- The recommended introductory text is R for Dummies by de Vries and Meys.

Learning R

- R is learned by reading and doing. Experiment!
- The math expressions should be obvious.
- Many functions/commands will become second nature, because you use them a lot
- Of course, you will make mistakes, and learn from them.
- Experience = recognizing a mistake when you make it again.
- Many ways to ask R for help, e.g., `help.start()`, opens web help interface
- `?mean` or `help(mean)` opens Help pane in RStudio.
- Similarly `?"+"` and `?"if"`, note quotes.
- `??plotting` and `??"regression model"` searches for topics containing these phrases.
- `apropos("sor")` or `apropos("sort")`, good for finding relevant commands, note difference in result.

Math Expressions Should Be Obvious

- The first three of these expressions

```
-2^.5     -2**.5     -(2^.5)     (-2)^.5
```

will give same result. The last produces `NaN`, not a number.

- In EXCEL the first produces an error (also in C).
  It is interpreted just as the 4-th expression above.
- Fortan77 same as R, i.e., correct mathematical convention.
- When in doubt, use ( ) to enforce proper order of evaluation.
- `factorial(5)` produces $5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$
- `choose(8,4)` produces $\binom{8}{4} = 70$.
- `sqrt(2)` gives $\sqrt{2} = 1.414214$.
- Instead of statistical tables see R under ?Distributions.

Some Lab Exercises

Use R to compute the following

- $1 + 2(3 + 4)$
- $\log(4^3 + 3^{2+1})$
- $\sqrt{(4 + 3)(2 + 1)}$
- $\left(\frac{1+2}{3+4}\right)^2$

Use LaTeX to typeset the above 4 expressions.

R Even Handles the Infinite and π

- R's expression for ∞ is `Inf`.
- When it makes sense, R does proper arithmetic involving `Inf`
- e.g., `1/Inf` and `1/0` return `0` and `Inf`, respectively.
- When it does not make sense it returns `NaN`, Not a Number
- e.g., `0*Inf` and `Inf-Inf` and `0/0` return `NaN`.
- π is represented by `pi`, thus avoid using `pi` as object name.
- However, `pi <- 3` is legal. Then each use of `pi` means `3`, until you `rm(pi)` (you remove the object `pi` from workspace)
- Read about the infamous Indiana Pi Bill of 1897.
- No corresponding expression for Euler's number *e*, use `exp(1)` instead.
- Note that R use `NA` to represent missing value of an entry.

Work Space House Keeping Tools

- `ls()` or `objects()` lists objects in your work space
- `ls(pattern="un")` lists objects with names containing `un`.
- `getwd()` gives path of active working directory.
- `rm(x,y,z)` removes objects `x,y,z` from work space. There is no undo, but you can decline `save` when you exit R.
- `rm(list=ls())` cleans out all objects from your work space.
- `rm(list=ls(pattern="un"))` cleans out all objects containing `un` in their names.
- `save(x,y,z,file="objects.rda")` saves objects `x,y,z` to the file "objects.rda" in your working directory.
- `load("objects.rda")` loads the objects in file "objects.rda".
- `save.image("ws.rda")` saves the whole work space to "ws.rda".
- `load("ws.rda")` loads that whole work space in again.
- The above `.rda` files are not readable in a normal text editor.

Exporting/Importing R Objects via dput and dget

Any object in an R workspace can be exported to the working directory via the `dput` command

```
> x <- c(1:3,2:1)
> dput(x,"xobject")
# when editing "xobject" you see
c(1L, 2L, 3L, 2L, 1L)
# That is R’s way of storing integers,
# as opposed to numerics
> xx <- c(1,3,2)
> dput(xx,"xxobject")
# view "xxobject" in an editor and see
c(1, 3, 2)
# treated as a numeric
> rm(x,xx)
> x <- dget("xobject") # imports x back again
> xx <- dget("xxobject") # imports xx back again
```

Exporting/Importing Exercise

Enter the following instructions in R:

1. `x <- 3:7`
2. `y <- log(x)`
3. `dput(y, "yobject")`
4. `y <- 5`
5. `y`
6. `y <- dget("yobject")`
7. `y`

What will happen if we enter `x <- dget("yobject")`?

More Lab Exercises

Use R to compute the following

- $\log_{10}(20)$
- $\sin(180)$
- $\sin(\pi)$
- $5^{\cos(\pi^2)}$
- $\frac{22 \cdot 23 \cdots 25}{5 \cdot 4 \cdots 1}$
- $\log_5(15 \cdot 14 \cdots 11)$

Use LaTeX to typeset the above expressions.