1 - Intro to R

Good Habits and Basic Introduction

Outline

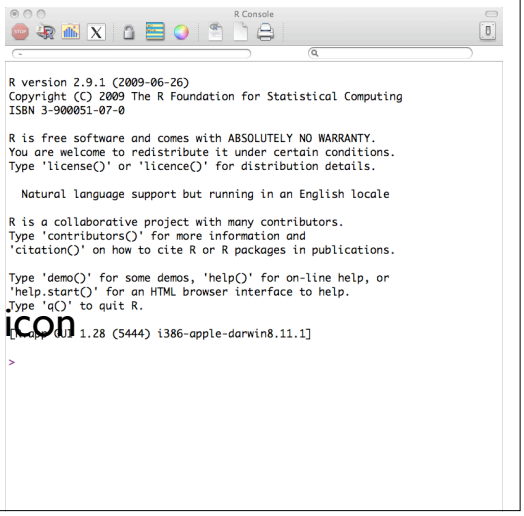
- Getting Started
- Good Habits
- Overgrown calculator
- Basic functions
- Getting help
- Doing statistics
- More...

What do you know already?

- Excel? JMP? Numbers?
- A programming language? CS course?
- R? SAS? SPSS?
- Have you used a text editor?

Install R

- On your own machine:
  - Go to http://www.r-project.org/
  - From CRAN, pick download site (ISU might be good)
  - Download from base:
    - R-2.11.1-win32.exe
    - Run the installation script
- On a lab machine:
  - Start R by double-clicking the icon



Setting up

- Open the R script in an editor, eg notepad, wordpad, emacs, ...
- Edit lines
- Cut and paste lines of code into the R interpreter window
- Or:
  - Windows: Ctrl-r
  - Mac: Apple-return
  - Linux Using Rkward: Shift-F7; Using Emacs w/ ESS: Ctrl-c n

Good Habits

- Avoid the interpreter.
- Keep a record of your work by using scripts.
- Makes coding easier
- Store your data and your scripts in a convenient location.
- Save often

- Working Directory

Navigating the R interpreter window

- Up/down arrow keys to retrieve previous lines
- Left/right arrow keys to move cursor along line
- Mouse click to set cursor position
- Delete to remove and re-type parts of command

Learning a language

- Grammar / Syntax
- Vocabulary
- "Thinking in that language"
- There are a lot of commands in R. Don't expect to memorize all of them.

Overgrown Calculator

- Basic mathematical operators:
  - +, -, *, /
- Basic Mathematical functions
  - exp, log, sin, cos
- Storing variables for later use using the assignment operator
- `a <- 32`
- Working with vectors . . .

Variables

- Variable names can't start with a number
- R is case-sensitive
- Some common letters are used internally by R and should be avoided as variable names `(c, q, t, C, D, F, T, I)`
- Try to keep names short but descriptive.
- There are reserved words that R won't let you us for variable names. A few examples:
  - `for, in, while, if, else, repeat, break`
- R will let you use the name of a predefined function. So try to not over write those!

Basics

- Basic algebra is the same
- Use `2*x` not 2x, `2^p` instead of 2^p
- Applying a function is similar
- Making a variable, use `<-` instead of =
- Everything in R is a vector
- Index a vector using [ ]

Examples

• $x = 2 / 3$	`x <- 2 / 3`
• $\sqrt{x}$	`sqrt(x)`
• $a = 2(x + 3)^2$	`a <- 2 *(x + 3)^2`
• $y = (1\ 2\ 3\ 5)^T$	`y <- c(1, 2, 3, 5)`
• y_1	`y[1]`
• $\sum y$	`sum(y)`
• 2y	`2*y`
• get the date	`date()`

...

Functions

- Typical format:
  - `foo(x, y = 1:length(x), ...)`
  - Some parameters have defaults set for you
  - `...` is special. Passes along extra parameters to functions used inside the function.

Getting Help

- `help.start()`
- `help(command)`
- `?command`
- `help.search("command")`
- `apropos()`
- Google!

Getting Out

- `q()`

Your Turn

- x = 3
- y = 5
- Square root of $(x^2 + y^5)$
- $\sin(e^{(2(pi)(x+y)/(y-x))})$
- Find the roots of $3t^2 - 2t - 17$

R Reference Card

- Download the R Reference Card from http://cran.r-project.org/doc/contrib/Short-refcard.pdf
- Open/Print so that you can glance at it while working

Vectors

- As mentioned before almost everything in R is a vector.
- Multiple ways to make a vector
- `c(1,2,3)`
- `a:b` creates a vector (a, a+1, ..., b-1, b)
- Common operations and functions work elementwise on vectors and return a new vector.
- `rep() and seq()` ...

Your Turn

- $x = (4\ 1\ 3\ 9)^T$
- $y = (1\ 2\ 3\ 5)^T$ (from examples, on previous slide)
- $d = \sqrt{\sum(x - y)^2}$
- $2(y_1 + x_2)$
- $z = (1, ..., 100)^T$
- $(\sum z_i)^2 - \sum(z_i^2)$
- pattern = $(1, 7, 7, 13, 13, 13, 19, 19, 19, 19, 25, 25, 25, 25, 25, 31, 31, 31, 31, 31, 31)^T$ (don't use `c()` for this)

Basic Statistical Functions

- Using the basic functions we've learned it wouldn't be hard to compute basic statistics.
- `x <- 1:100`
- `n <- length(x)`
- `xbar <- sum(x)/n`
- `s <- sqrt(sum((x-xbar)^2)/(n-1))`
- ... But we don't have to

Distributions

- R has a lot of distributions built in.
- We can typically obtain:
  - Density value
  - CDF value
  - Inverse CDF value
  - Random deviate
- Normal, Chi-square, F, T, Cauchy, Poisson, Binomial, Negative Binomial, Gamma, ..., lots more
- `library(help = stats)`

Your Turn

- Find the mean of 10, 1000, and 10000 random normal observations from N(0,1)
- Generate 1,000,000 random observations from N(0,1), square them, and find the .95 quantile of the observed data
- What is the .95 quantile from a Chi-square distribution with 1 df?
- x = 100 random normal observations from N(5,36)
- Calculate a 95% confidence interval for mu using x as your data [xbar ± $(t_{.95,99})(s)$]

Booleans

- R has support for logical values
- `TRUE, FALSE, T, F`
- Can result from a comparison
  - `<`
  - `>`
  - `<=`
  - `>=`
  - `==`
  - `!=`

Logical Operators

- `&` (AND)
- `|` (OR)
- Slightly different from:
- `&&` (different AND)
- `||` (different OR)
- ?"&"

Indexing

- Accessing just a part of a vector/matrix/dataframe.
- Multiple ways to index
  - `x[2]`
  - `x[c(1,3,7)]`
  - `x[c(T,F)]`
  - `x[x>10]`
  - `x[-1]`

Your Turn

- Using `pat <- seq(2,103,by = 3)`
- x = elements at even indices in pat
- y = elements in pat greater than mean(pat)
- z = even elements in pat
- prime = All primes between 1 and 100 (setdiff might be of interest)

Load Data

- ```
  >library(ggplot2)
  >data()
  >help(tips)
  ```

- Did the data import work?

Examining Objects

- `x`
- `head(x)`
- `summary(x)`
- `str(x)`
- `dim(x)`

Try these commands out for yourself!
  ```

Examine Object

- First few values of an object
  `head(tips)`

```
> head(tips)
  obs totbill  tip sex smoker day  time size
1   1   16.99 1.01   F     No Sun Night    2
2   2   10.34 1.66   M     No Sun Night    3
3   3   21.01 3.50   M     No Sun Night    3
4   4   23.68 3.31   M     No Sun Night    2
5   5   24.59 3.61   F     No Sun Night    4
6   6   25.29 4.71   M     No Sun Night    4
```


Examine Object

- Structure of an object
  `str(tips)`

```
> str(tips)
'data.frame':   244 obs. of  8 variables:
 $ obs    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ totbill: num  17 10.3 21 23.7 24.6 ...
 $ tip    : num  1.01 1.66 3.5 3.31 3.61 4.71 2 3.12 1.96 3.23 ...
 $ sex    : Factor w/ 2 levels "F","M": 1 2 2 2 1 2 2 2 2 2 ...
 $ smoker : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ day    : Factor w/ 4 levels "Fri","Sat","Sun",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ time   : Factor w/ 2 levels "Day","Night": 2 2 2 2 2 2 2 2 2 2 ...
 $ size   : int  2 3 3 2 4 4 2 4 2 2 ...
```

Examine Object

- Dimension of an object
  `dim(tips)`

```
> dim(tips)
[1] 244   8
```

the tips data set has 244 rows (tables served) and 8 columns (variables recorded)

Examine Object

- Dimension of an object
  `summary(tips)`

```
> summary(tips)
      obs            totbill           tip         sex     smoker
 Min.   :  1.00   Min.   : 3.07   Min.   : 1.000   F: 87   No :151
 1st Qu.: 61.75   1st Qu.:13.35   1st Qu.: 2.000   M:157   Yes: 93
 Median :122.50   Median :17.80   Median : 2.900
 Mean   :122.50   Mean   :19.79   Mean   : 2.998
 3rd Qu.:183.25   3rd Qu.:24.13   3rd Qu.: 3.562
 Max.   :244.00   Max.   :50.81   Max.   :10.000
  day        time          size
 Fri:19   Day  : 68   Min.   :1.000
 Sat:87   Night:176   1st Qu.:2.000
 Sun:76               Median :2.000
 Thu:62               Mean   :2.570
                      3rd Qu.:3.000
                      Max.   :6.000
```

Extracting parts

- `x$variable`
- `x[, "variable"]`
- `x[rows, columns]  # rows, columns are # indices`
  - `x[1:5, 2:3]`
  - `x[c(1,5,6), c("sex","tip")]`
- `x$variable[rows]`

Your Turn

- Calculate basic summary stats for the variables.
- Create a variable for Tipping Rate
- Find the mean bill amount for each gender
- Are there any unusual points?
- Explore the data. Can you find any interesting trends?
- How many people in this data tipped tipped greater than 20% of their bill?