Basic R Functionality

EPID 7500

Steve Bellan (slides altered from John Muschelli)

Common new users frustations

1. **Different versions of software**
2. **Data type problems (is that a string, number, or factor?)**
3. Working directory problems: trying to read files that R "can't find"
   - RStudio can help, and so do RStudio Projects
   - discuss in Data Input/Output lecture
4. Typos (R is **case sensitive**, `x` and `X` are different)
   - RStudio helps with "tab completion"
   - discussed throughout

Getting Started

- You should have the latest version of R installed!
- Open RStudio
- Files –> New –> R Script
- Save the blank R script in a directory of your choosing
- Add a comment header

Commenting in Scripts

Add a comment header to day1.R :`#` is the comment symbol

```
#################
# Title: Demo R Script
# Author: Steve Bellan
# Purpose: Demonstrate comments in R
###################

# nothing to the right of the symbol is evaluated by R

# this # is still a comment
### you can use many #'s as you want

# sometimes you have a really long comment,
#    like explaining what you are doing
#    for a step in analysis.
# Take it to another line
```

Explaining output on slides

In slides, a command (we'll also call them code or a code chunk) will look like this

```
print("I'm code")
```

```
[1] "I'm code"
```

And then directly after it, will be the output of the code.
So `print("I'm code")` is the code chunk and [1] "I'm code" is the output.

R as a calculator

```
2 + 2
```

```
[1] 4
```

```
2 * 4
```

```
[1] 8
```

```
2 ^ 3
```

```
[1] 8
```

Note, when you type your command, R inherently thinks you want to print the result.

For now, let's ignore the [1]. We'll return back to why R prints this in front of each object it prints later on.

R as a calculator

- The R console is a full calculator
- Try to play around with it:
    - +, -, /, * are add, subtract, divide and multiply
    - ^ or ** is power
    - parentheses – (and) – work with order of operations

R as a calculator

```
2 + (2 * 3)^2
```

```
[1] 38
```

```
(1 + 3) / 2 + 45
```

```
[1] 47
```

R as a calculator

Try evaluating the following:

- `2 + 2 * 3 / 4 -3`
- `2 * 3 / 4 * 2`
- `2^4 - 1`

- You can create variables from within the R environment and from files on your computer
- R uses "=" or "<-" to assign values to a variable name
- Variable names are case-sensitive, i.e. X and x are different

```
x = 2 # Same as: x <- 2
x
```

```
[1] 2
```

```
x * 4
```

```
[1] 8
```

```
x + 2
```

```
[1] 4
```

R variables

- The most comfortable and familiar class/data type for many of you will be `data.frame`
- You can think of these as essentially Excel spreadsheets with rows (usually subjects or observations) and columns (usually variables)

R variables

- `data.frame`s are somewhat advanced objects in R; we will start with simpler objects;
- Here we introduce "1 dimensional" classes; often referred to as 'vectors'
- Vectors can have multiple sets of observations, but each observation has to be the same class.

```
class(x)
```

```
[1] "numeric"
```

```
y = "hello world!"
print(y)
```

```
[1] "hello world!"
```

```
class(y)
```

```
[1] "character"
```

R variables

Try assigning your full name to an R variable called `name`

R variables

Try assigning your full name to an R variable called `name`

```
name = "Steve Bellan"
name
```

```
[1] "Steve Bellan"
```

The 'combine' function

The function `c()` collects/combines/joins single R objects into a vector of R objects. It is mostly used for creating vectors of numbers, character strings, and other data types.

```
x <- c(1, 4, 6, 8, 1, 1, 3, 4)
x
```

```
[1] 1 4 6 8 1 1 3 4
```

```
class(x)
```

```
[1] "numeric"
```

The 'combine' function

Try assigning your first and last name as 2 separate character strings into a single vector called `name2`

The 'combine' function

Try assigning your first and last name as 2 separate character strings into a length-2 vector called `name2`

```
name2 = c("Steve","Bellan")
name2
```

```
[1] "Steve"  "Bellan"
```

R variables

`length()`: Get or set the length of vectors (including lists) and factors, and of any other R object for which a method has been defined.

```
length(x)
```

```
[1] 8
```

```
y
```

```
[1] "hello world!"
```

```
length(y)
```

```
[1] 1
```

R variables

What do you expect for the length of the `name` variable? What about the `name2` variable?

What are the lengths of each?

R variables

What do you expect for the length of the `name` variable? What about the `name2` variable?

What are the lengths of each?

```
length(name)
```

```
[1] 1
```

```
length(name2)
```

```
[1] 2
```

R variables

You can perform functions to entire vectors of numbers very easily.

```
x + 2
```

```
[1]  3  6  8 10  3  3  5  6
```

```
x * 3
```

```
[1]  3 12 18 24  3  3  9 12
```

```
x + c(1, 2, 3, 4)
```

```
[1]  2  6  9 12  2  3  6  8
```

R variables

But things like algebra can only be performed on numbers.

Learn to read R's Errors and Warnings. They will help you debug code.

```
> name2 + 4
[1] Error in name2 + 4 : non-numeric argument
 to binary operator
```

R variables

And save these modified vectors as a new vector.

```
y = x + c(1, 2, 3, 4)
y
```

```
[1]  2  6  9 12  2  3  6  8
```

Note that the R object `y` is no longer "Hello World!" - It has effectively been overwritten by assigning new data to the variable

R variables

- You can get more attributes than just class. The function `str` gives you the structure of the object.

```
str(x)
```

```
 num [1:8] 1 4 6 8 1 1 3 4
```

```
str(y)
```

```
 num [1:8] 2 6 9 12 2 3 6 8
```

This tells you that `x` is a numeric vector and tells you the length.

Review

- Creating a new script
- Using R as a calculator
- Assigning values to variables
- Performing algebra on numeric variables

Website

EPID 7500 Course Website

John Muschelli's Course at Johns Hopkins & github repository