Intro to R for biologists

Learning R basics for the analysis and visualization of biological datasets

Brandon Ely, Doctoral Candidate
PhD program in Biology (Molecular, Cellular, Developmental)
CUNY Graduate Center

*Adapted from Dr. Weigang Qiu's "R Tutorials for biologists"

Agenda

Week 1:

- Intro to R (syntax and basics)

Week 2:

- Data transformation
- Statistical testing

Week 3:

- Data visualization

What is R and R Studio?

R is a programming language built by staticians and widely used in many areas of biology

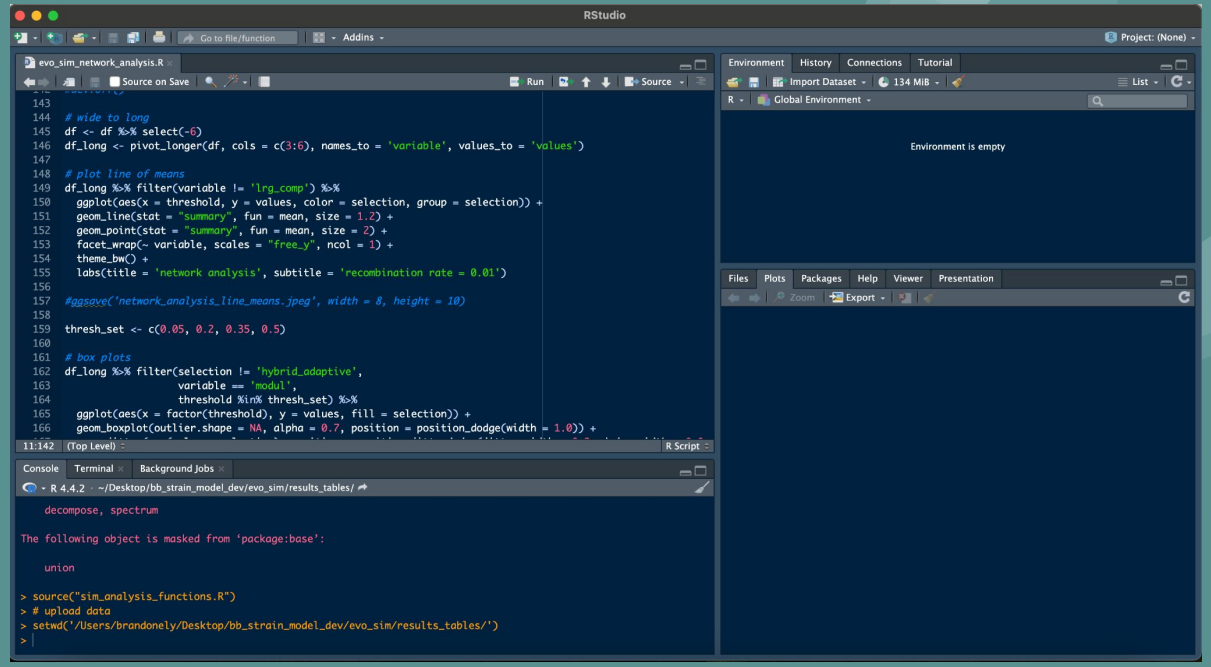
- Data transformation/computation
- Statistical analyses
- Visualization

R studio is just a user interface that makes R easier to use

Navigating R Studio

Script

Console
Terminal



Environment

Files
Plots
Packages
Help

R data types

Data types are determined by the type of value you have and dictate how it can be used and stored

Types:

- Character (text, strings)
- Numeric (real numbers; integers and decimals)
- Integer (whole numbers)
- Logical (boolean values; TRUE or FALSE)

Basic R syntax - characters

```
# print characters by using single or double quotes

'My name is Brandon'

```

```
> 'My name is Brandon'
[1] "My name is Brandon"
>
```

Basic R syntax - numeric

```r
# when printing numbers, you do not need to use quotes*
1

# you can also print the output of basic math operations
1+1
1-1
2*5
4/8
2^4
```

Console | Terminal | Background Jobs

R 4.5.1 · /cloud/project/

```
> # when printing numbers, you do not need to use quotes*
> 1
[1] 1
> # you can also print the output of basic math operations
> 1+1
[1] 2
> 1-1
[1] 0
> 2*5
[1] 10
> 4/8
[1] 0.5
> 2^4
[1] 16
>
```

R variables and data structures

Variables are objects (sometimes referred to as data containers) that store information

- You create a variable/object by defining it with `<-`
- You reference a variable calling it in your script (case sensitive!!!)

Types of R Data structures:

- Vectors (list of values of same data type)
- Lists (list of values or objects of mixed types)
- Matrices (2D dataset of columns and rows)
- Arrays (same as matrice but with more than 2 dim)
- Data Frames (data values displayed in a table format)

R variables and data structures

basic_R_use.R* | Untitled1

Run | Source

```r
# defining a variable
number <- 10
squared <- number^2
doubled <- number*2
MyName <- 'Brandon'
DOB <- '09/27'
siblings <- c('Sean', 'Carissa', 'Jenna')
MyVec <- 1:10
MyVec2 <- c(1,3,5,7,9,9)

TestScores <- data.frame(student = siblings, score = c(87, 100, 92))
```

Environment | History | Connections | Tutorial

Import Dataset | 584 MiB | List

R | Global Environment

Data	
TestScores	3 obs. of 2 variables
Values	
DOB	"09/27"
doubled	20
MyName	"Brandon"
MySiblings	chr [1:3] "Sean" "Carissa" "Jenna"
MyVec	int [1:10] 1 2 3 4 5 6 7 8 9 10
MyVec2	num [1:6] 1 3 5 7 9 9
number	10
siblings	chr [1:3] "Sean" "Carissa" "Jenna"
squared	100

	student	score
1	Sean	87
2	Carissa	100
3	Jenna	92

R Functions

Functions can perform a series of operations or tasks with given input

Inputs of functions are referred to as arguments

Syntax:
NameOfFunction(arg1, arg2,...)

Functions make code more readable, and make performing more complex operations easier to execute!

How do I know how to use the function?!?
In your script, execute: ?function OR help(function)

Let's practice together!

1. Create a character string of your name and define it as a variable in your environment called MyName
   a. Use **print** to output MyName
   b. Use **paste** to output “My name is MyName”
2. Use the **substr** function to output the 3rd and 4th letters in MyName
3. Create a vector with the names of all members of your cohort. Define this variable in your environment as “roster”
4. Check to see if any of the names in roster have consecutive letters “ic” in them using the **grepl** function
5. Use the **sample** function to randomly select 3 names from roster
6. Repeat tasks 4-5 together

Independent practice

1. Create a character vector for the 4 DNA nucleotides, store as a variable in your environment

2. Use **sample** on your vector to create a DNA sequence of length 200 and store as a variable in your environment
   *Run this line of code next to combine items in your vector to a single string. Just substitute “x” in the code with your DNA sequence name: `x <- paste(x, collapse = '')`

3. Find out if your DNA has any start codons (ATG) using **grepl**
   if output is false, repeat task 2

4. Find all locations of start codons using **str_locate_all**

5. Use **substring** to confirm coordinates are actually "ATG"

6. Calculate nucleotide % composition using **str_count** function