

Ecological Analysis with R (Harvard Forest ***R***EU course)

Day 1

- [Getting setup]
- [Coding basics]
- [R as a second language]
- [R Community = Packages + help?]
- [Plotting]

[Getting setup]

- Install the newest version of **RStudio** from [here](#)
- You do not need RStudio to run R. We do suggest it however for the improvements it offers over the default R GUI (a more seamless integration of scripts, console and graphs) along with some other additional features.
- While RStudio offers some point-and-click GUI alternatives, we will not be using or teaching them in this class.

Set up your project space

- Create a directory on your Desktop called Rwksp
- Create the following folders:
- data (This is self explanatory)
- src (This is where your code lives)
- docs (This is where other files that aren't code or data should live)
- I also suggest creating a README file for you to put some general notes about the workshop

Say hello to the Console...

The Console is a command line where you can enter text, as if you are “talking” to the computer.

Activity: Try out some math

```
2+2
```

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

```
2-2
```

```
## [1] 0
```

```
2*2
```

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

```
2/2
```

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

```
2%*%2
```

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

Operators, Objects and Functions

Operators are characters that do things like math, logical operations, equalities, etc.

Objects (aka. variables) are names that can “store” information, like data.

Functions are special types of objects that can do things, they all have the form of **name**(*arguments*). For example:

```
help('help')
```

Data types and Object classes

There are multiple data types, but for now we just want you to be aware that there are different types: characters vs. numbers vs. logical

If you do not use “” (or ”), R will read whatever word/combination of characters you type as an object. Therefore R will read ABCD as an attempt to call an object that doesn’t exist, but "ABCD" as a character string.

Activity: What can we do with objects?

```
x <- 4  
y <- 6  
x + y
```

```
## [1] 10
```

```
z <- "Hello Harvard Forest"
print(z)
```

```
## [1] "Hello Harvard Forest"
```

We will go more in on object classes later in the workshop, but for now there is one important class you need to know and that is vectors. Vectors are objects that have multiple numbers (or strings, or logicals, etc) contained within. To create a vector do the following: `objectname <- c(1,2,3,4,5)` . `c()` is a function that stands for concatenate.

Activity: What if we want to store more than one value at a time?

```
v <- c(1,2,3,4,10)
sum(v)
```

```
## [1] 20
```

```
v2 <- c(10,15,20,30,40)
v + v2
```

```
## [1] 11 17 23 34 50
```

[R as a second language]

- Learning R (or any computer language) is similar to learning a foreign language.
- You have to learn more than just the words: you have to learn the grammar, punctuation and syntax of the language.
- Just like in a foreign language, R has many commands that are cognates of English words. This is helpful when learning, as you often can figure out what a function does from its name alone.
- Though be wary, just like in human languages there are sometimes false cognates.

Can you guess what the following functions do?

```
mean(x)
max(x)
plot(x)
Map(x)
```

About arguments...

- Most functions have default settings for most arguments
- You don't always have to state the argument name, R infers from the ordering of the arguments
- Like Las Vegas, what happens in a function, stays inside a function (most of the time...), so you don't have to worry about argument names that are inside of functions over-writing your objects outside of the function
- Use tab to try and complete the names of functions and arguments

Activity: Pick a function and explore the arguments!

1. `help()` or `?`
2. Use tab completion

Scripting

A script is a file that you can store your code for later use!

So, open up a new script file and save it to your `Rwkspace/src` directory.

Commenting can be done using `#` because R will ignore the entire line to the right of it.

Style and Annotation

Be sure to use style (i.e. formatting of your code) and annotation that will make your code usable later by others and yourself!

1. R script names should always end in `.R`
2. Try and keep names simple, i.e. 3-5 characters, lowercase
3. Put spaces around operators and after commas and before left parentheses except after function names
4. Use `<-` for assignment, rather than `=`
5. For more info on appropriate formatting, see <http://stat405.had.co.nz/r-style.html>