16SrRNA Intermediate Bioinformatics Online Course: Int_BT

Introduction to R

Why you’re going to love R…

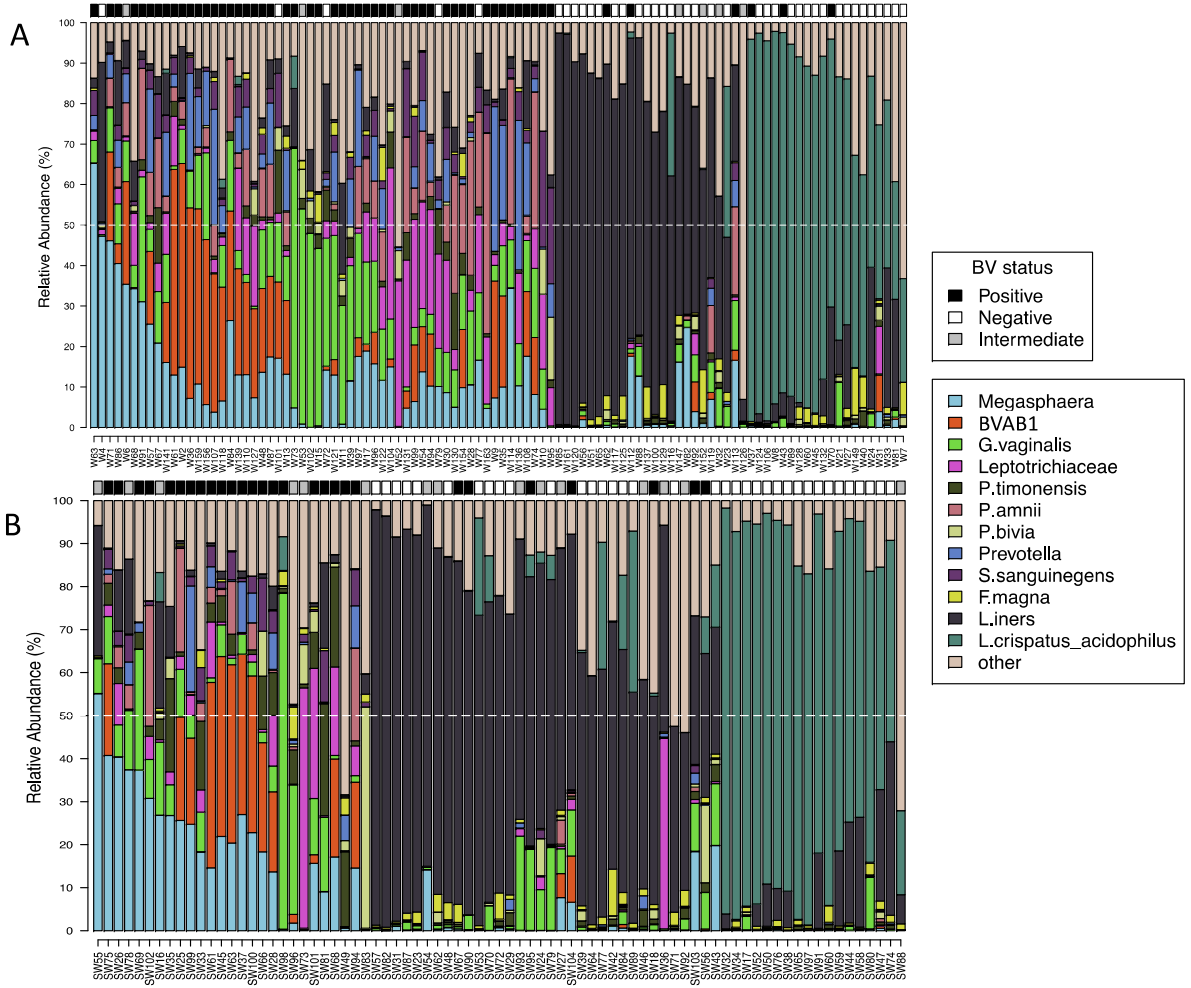
What is R?

- R is an open source programming language used for statistical analyses and graphics
- RStudio is the user-friendly interface commonly used when programming in R
  - Allows you to see your R script, console and graphics all on one screen
  - Easy package installation & updates & help
  - Reproducibility

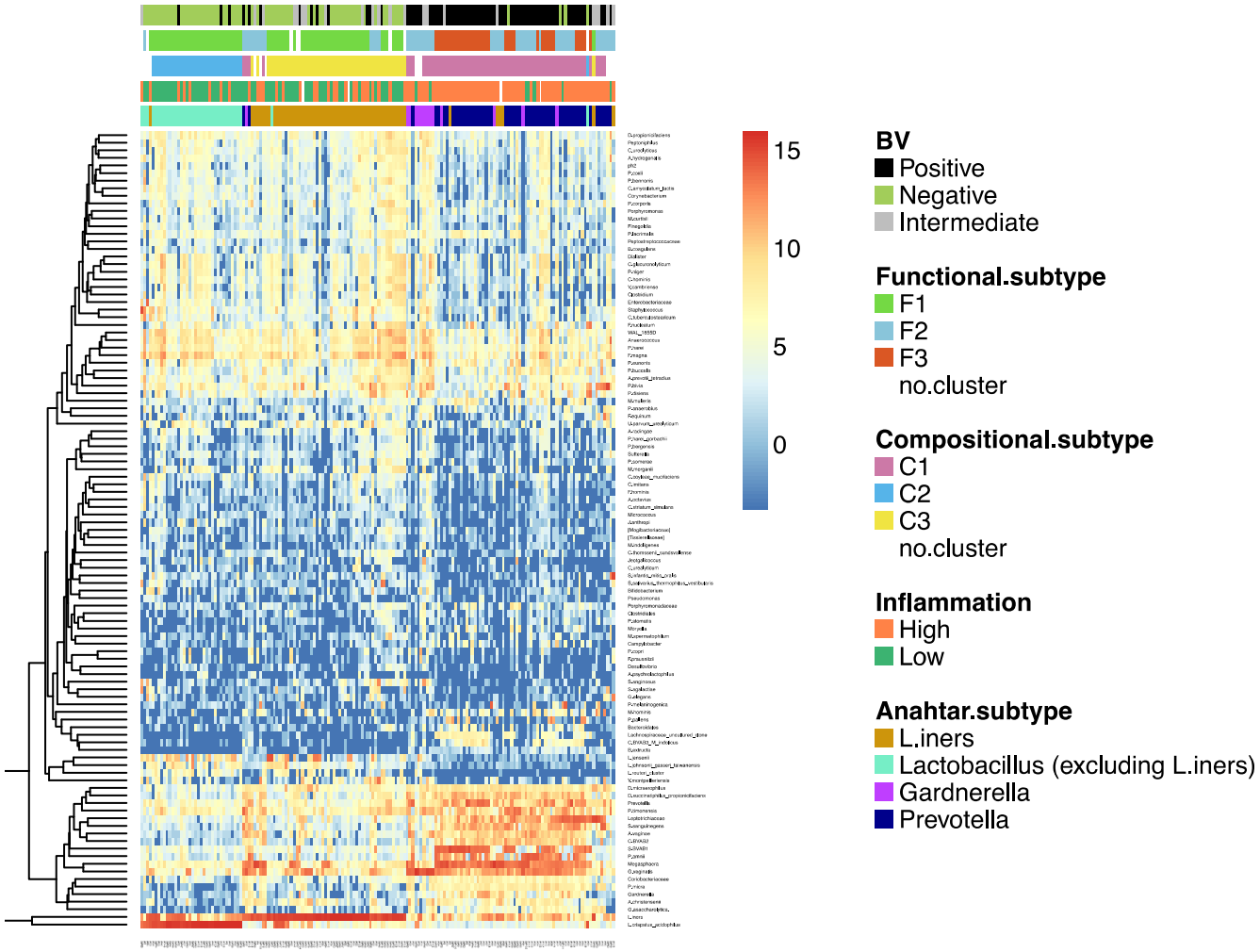
Why learn R?

- Non-programming point-and-click software can be dangerous..
  - Are you sure you know what you're doing or what it is doing?
- R = written commands which has several advantages:
  - You have to think about what you're writing and know what each command is doing
  - Your analysis is now reproducible because you've written each command in an R script file (more and more journals require this)
  - Easy collaboration – share your R scripts
- Specialized software packages that are likely only available through R
  - E.g. you're working on 16S microbiome data: custom R packages (phyloseq, metagenomeSeq…)
- Publication quality graphics, customizable (fun)

Customizable, publication-quality graphics



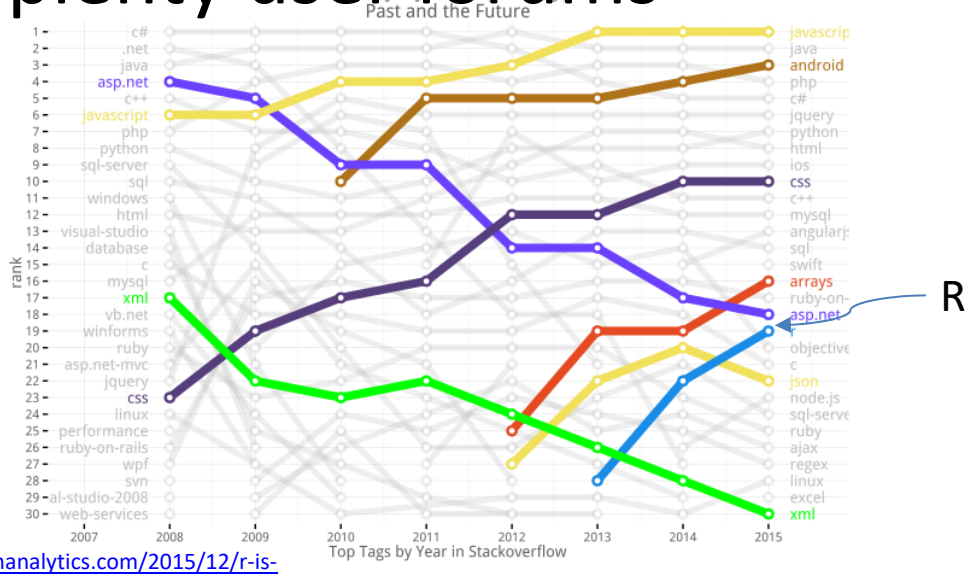
Customizable, publication-quality graphics



R help?!

- Just ask uncle Google…
  - If you get an error someone else has had it too (and posted about it)
- Large R community, plenty user forums

[stackoverlow example]



https://blog.revolutionanalytics.com/2015/12/r-is-the-fastest-growing-language-on-stackoverflow.html

Syllabus (learning objectives)

- Introduction to RStudio
- Define the following terms as they relate to R: object, assign, function, arguments
- Importing data into R as dataframes
- Subsetting, indexing of dataframes
- Best practices for writing R code

Learning Outcomes

- Comfortable navigating RStudio
- Comfortable with basic R operations and syntax
- Importing data from .csv or .txt files into R as dataframes

Introduction to R website

16SrRNA Intermediate Bioinformatics Online Course: Int_BT_2019

Using RStudio

RStudio

- RStudio is an Integrated Development Environment (IDE) for working with R
- Open-source
- https://www.rstudio.com
- RStudio course material

RStudio: useful shortcut keys

	Windows	Mac
Comment/uncomment current line/selection	Ctrl+Shift+C	Command+Shift+C
Run current line/selection	Ctrl+Enter	Command+Enter
Insert code section	Ctrl+Shift+R	Command+Shift+R
Insert assignment operator	Alt+-	Option+-
Jump to console pane	Ctrl+2	Ctrl+2
Jump to script pane	Ctrl+1	Ctrl+1

RStudio useful links

- Knowing your way around RStudio
- Writing code in RStudio
- Debugging code in RStudio

H3ABioNet

Pan African Bioinformatics Network for H3Africa

16SrRNA Intermediate Bioinformatics Online Course: Int_BT_2019

Introduction to R: objects, functions & data types/structures

Intro to R: assigning values to objects

- Assign values to objects in R
  - Assignment operator <- OR = (see here)
- Consistent styling (see here)
  - Objects cannot start with a number e.g. 2x ✗ x2 ✓
  - Case sensitive
  - Base R function names are off-limits (e.g. if else for)

Intro to R: functions and arguments

- A **function** is **pre-written code** that can be accessed by '**calling**' the function **name** and specifying it's **arguments**
  - E.g. sqrt(); round()
  - Input: 'arguments' can be anything (numbers, filenames, other objects)
    - User-specified OR default?
  - Output: 'return' can be anything (or empty!)
  - Understanding functions: ?sqrt

Intro to R: data types & structures

- R data types: **logical, integer, real**, complex, **string (or character)**
- R data structures: **vector**, list, matrix, **data frame, factor**
- Vector: one-dimensional array
  - elements **in a vector** all have the same data type
  - R type conversion: logical → numeric → character ← logical
- **Subset** and extract values from **vectors []**
  - Conditional subsetting: > < >= <= == | & !
- Analyze vectors with **missing data (NA is not the same as 'NA' or N/A)**
  - is.na() na.omit() complete.cases()

Intro to R: Factors

- Categorical data
- Predefined set of unique values named ‘levels’ e.g. ‘male’ ‘female’
- Alphabetically sorted (use function relevel() to change if necessary e.g. for plotting purposes)
- Careful with conversion of factor → numeric
- Careful when importing (stringsAsFactors=FALSE)
- Convert between strings and factors (as.factor())

R

- R course material

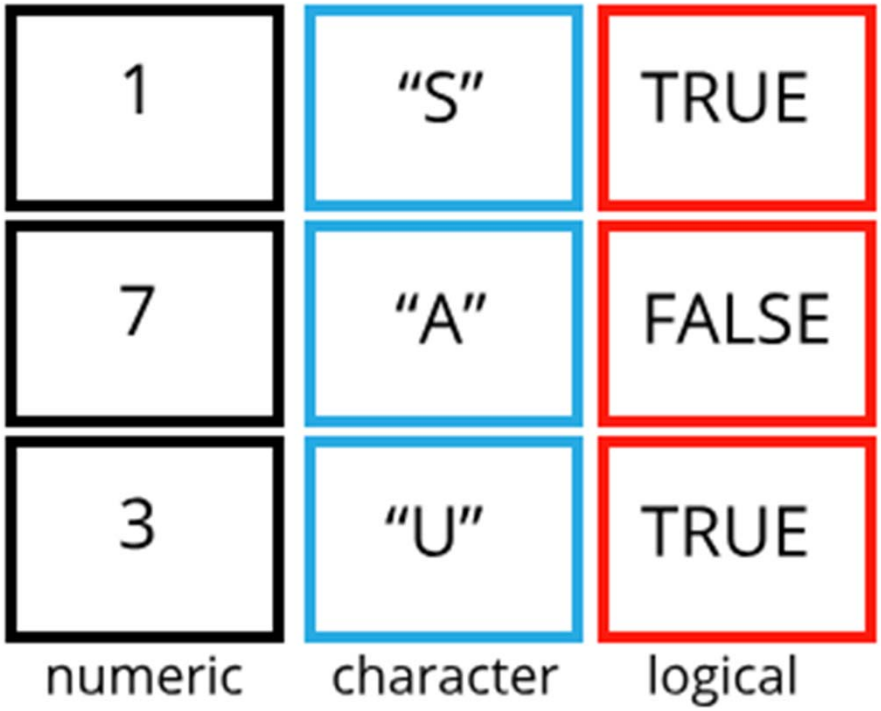
R data types useful links

- R data types and structures: additional hands on

16SrRNA Intermediate Bioinformatics Online Course: Int_BT_2019

Introduction to R: importing data, data frames

Intro to R: what is a data frame?



- A list of vectors of equal length
- Clinical data: sample = rows and variables of interest = columns
- Functions to examine data frames
  - str()
  - head()
  - dim()
  - rownames() colnames()

Working with data frames

- Load external data from a .csv file into a data frame
  - read.csv() read.table()
- Subsetting
  - [rows , columns] **‘comma-first-for-columns’ [,1]**
  - Select a range: [,1:10]
  - Exclusion: [,-1]
  - Many ways to select a column: e.g. surveys[,2] == surveys[,"month”] == surveys$month

R factors, data types, data frames

- course material

R data types useful links

- R data types and structures: additional hands on

16SrRNA Intermediate Bioinformatics Online Course: Int_BT_2019

Introduction to R: best practices for coding

Intro to R: good coding practices

- Start each program with a description of what it does.
- Then load all required packages.
- Improve reproducibility/usability by limiting 'hard-coding'
  - Keep all of the source files for a project in one directory and use relative paths to access them
  - Define all user-specific code and functions at the start of your script

Intro to R: good coding practices

R

```
input_file <- "data/data.csv"
output_file <- "data/results.csv"

# read input
input_data <- read.csv(input_file)
# get number of samples in data
sample_number <- nrow(input_data)
# generate results
results <- some_other_function(input_file, sample_number)
# write results
```

```
# check
input_data <- read.csv("data/data.csv")
# get number of samples in data
sample_number <- nrow(input_data)
# generate results
results <- some_other_function("data/data.csv", sample_number)
# write results
write.table("data/results.csv", output_file)
```

Intro to R: good coding practices

- Break code into logical sections with comments to inform script
  - Ctr/Cmd + shift + R
- Create functions for repeated code rather than copy-paste over and over – makes script long and error-prone
- Record sessionInfo()
- Get someone else to review your code
- Use version control, or Github gists!

Good coding practices – useful links

- Good enough practices in scientific computing
- The tidyverse R style guide