

# Day 1, PM Session

## R Basics



Richard Flamio Jr., Ph.D.

Madison Zimmerman

Kristina M. Ramstad, Ph.D.

University of South Carolina Aiken

# R and RStudio

## R = statistical programming



R version 4.1.2 (2021-11-01) -- "Bird Hippie"  
Copyright (C) 2021 The R Foundation for Statistical Computing  
Platform: x86\_64-apple-darwin17.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.

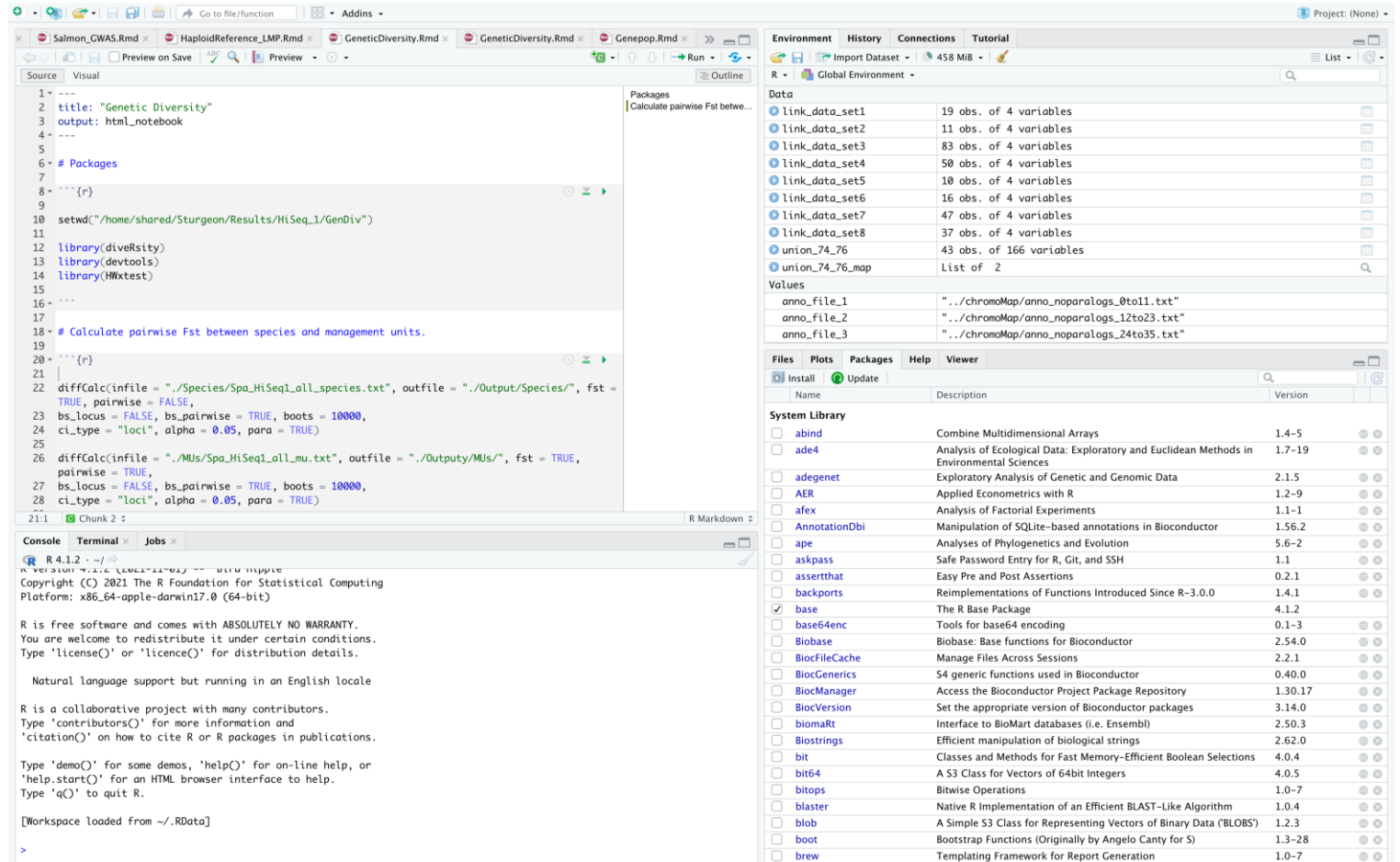
[R.app GUI 1.77 (8007) x86\_64-apple-darwin17.0]

[Workspace restored from /Users/rick/.RData]  
[History restored from /Users/rick/.Rapp.history]

>

You must download base R before you can  
download and use RStudio.

## RStudio = interface for R with extra tools



# Parts of RStudio

The screenshot shows the RStudio interface with three panels highlighted by colored boxes and corresponding text annotations:

- Editor (Blue box):** Contains R code for data import and preparation. The code includes setting the working directory, reading a table, and performing various data manipulations.
- Environment (Orange box):** Displays the Global Environment with a list of objects: Data\_in (69 obs. of 57 variables), Low (23 obs. of 57 variables), Up (23 obs. of 57 variables), UpLow (46 obs. of 57 variables), and whole (23 obs. of 57 variables). It also shows values for 'add' and 'forrownames'.
- Console (Red box):** Shows the output of the R code, including summary statistics (Min, 1st Qu., Median, Mean, 3rd Qu., Max) for various variables and a list of other objects.
- Overview over packages installed in R (Green box):** Displays a list of installed packages with their names, descriptions, and versions. The packages listed include assertthat, cli, colorspace, crayon, fansi, glue, labeling, magrittr, munsell, permute, pillar, R6, RColorBrewer, and Rcpp.

**Editor: write, edit, annotate and save your script**

**Environment: Data you have loaded into R; inspect through click**

**Consol: execute commands interactively and see output**

**Overview over packages installed in R; you can install more and activate/deactivate them with a click**

# Console

- Input, calculations, and output display
- The console is present in RStudio and base R

## Exercise

1. Type `3+4` in console, followed by Enter
2. Type `3 + 4` in console, followed by Enter
3. Type `3+4;2+3`, followed by Enter

# Console continued

- Notice how spaces do not matter in R
- Each command must be on a different line or separated by a semicolon (;)

```
> 3+3  
[1] 6  
> 3+4  
[1] 7  
> 3 + 4  
[1] 7  
>  
> 3+4;2+3  
[1] 7  
[1] 5
```

# Editor

- Unique to RStudio
- Allows you to run commands, saved in a text file, at the same time
- File > New File > R Script
- Scripts are saved as .R files
- Each line is a separate command
- Run some lines by highlighting lines of interest and clicking Run on top of the Editor
- Run whole script by CTRL+A and selecting Run

# R Markdown

- Save scripts in a format for reproducible reports
- Can save in pdf, html, etc. file extensions

```
1 ---
2 title: "Sockeye salmon GWAS"
3 author: "Richard Flaimio"
4 date: "2022-08-18"
5 output: html_document
6 ---
7
8 This script covers a workflow which covers to following: data preparation, data
9 pre-processing, imputation, and GWAS analysis.
10 # Packages
11
12 ```{r setup, include=FALSE}
13
14 # Set working directory
15
16 knitr::opts_knit$set(root.dir = "~/Salmon/Bioinformatics")
17
18 # Packages
19
20 library("data.table")
21 library("dplyr")
22 library("ggplot2")
23 library("reshape2")
24 library("rrBLUP")
25 library("qqman")
26
27 ```
```

Prelude

Non-chunk

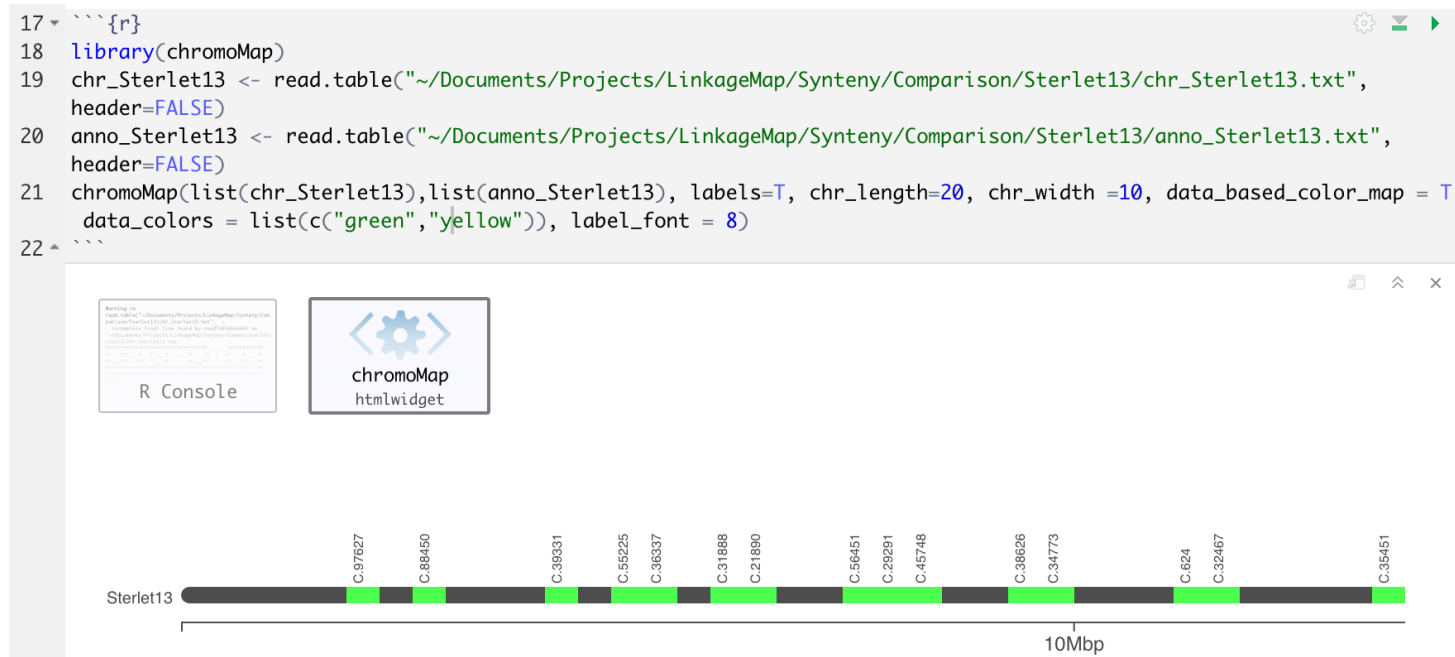
Chunk

## Outline

Packages  
Add packages to \$PATH  
Data pre-processing  
Map to reference  
Sort by coordinate (round 1)  
Mark duplicates  
Remove duplicates  
Sort by coordinate (round 2)  
Fix tags  
Add read groups  
Base quality score recalibr...  
Variant discovery  
Call variants per sample  
Consolidate GVCfs  
Joint-Call Cohort  
Hard filtering  
GWAS pre-processing  
Convert sample label names  
Covert VCF to PLINK  
Check file formats  
Filter data  
Convert plink to VCF  
Remove indels and retain o...  
Impute genotypes using be...  
Convert VCF to plink  
GWAS  
Read in files  
Extract marker calls from S...  
Calculate genomic relation...  
Read in .map file with mark...  
Create matrix for rrBLUP wi...  
Read in phenotype file with...  
and extract only phenotype...  
Genotypic filtering

# R Markdown continued

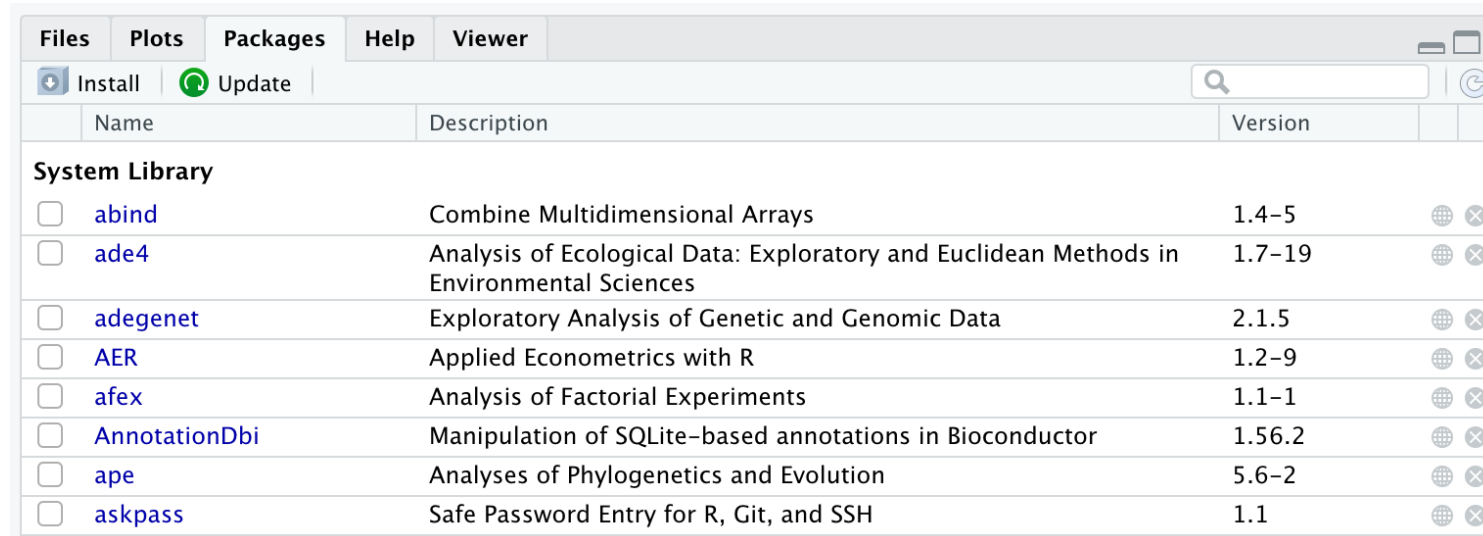
- Run a chunk by clicking the green triangle.
- To run all chunks above this chunk, press the gray triangle.
- When you run a chunk, the code will be printed beneath it.





# Lower right-hand window

- Files = open files, but usually easier to open files using script
- Plots = displays plots
- Packages = install, load, and view packages
  - Add-ons for R that extend its capabilities
  - Install package using Install → turn on package by clicking on package in list or typing `library("package")` in the console
  - Packages come with manuals that describe functions (look up online or just click the blue hyperlink in the package list to direct you to the tab Help)
- Help = documentation for packages



# File Directories

Want to load an external file? Use the working directory

Command	Description
<code>get.wd()</code>	Print working directory path
<code>set.wd()</code>	Set new working directory

# Functions

- Mathematical operators
  - + (addition)
  - - (subtraction)
  - / (division)
  - \* (multiplication)
- Most functions have form: `function(x,y,z,etc.)`
  - x,y,z are options called arguments
  - when arguments are not specified, default arguments are used

# Environment

- What is saved in your R sessions including....

# Objects

# Objects

- Most things (except graphs) in R are objects
  - E.g., datasets, functions, results, etc.
- Save output (e.g., external file) in your R session temporarily as an R object
  - `name <- function(x,y,z)`
  - Example: `data <- read.csv("turtles.csv")`
- `View()` to view R object

# Objects

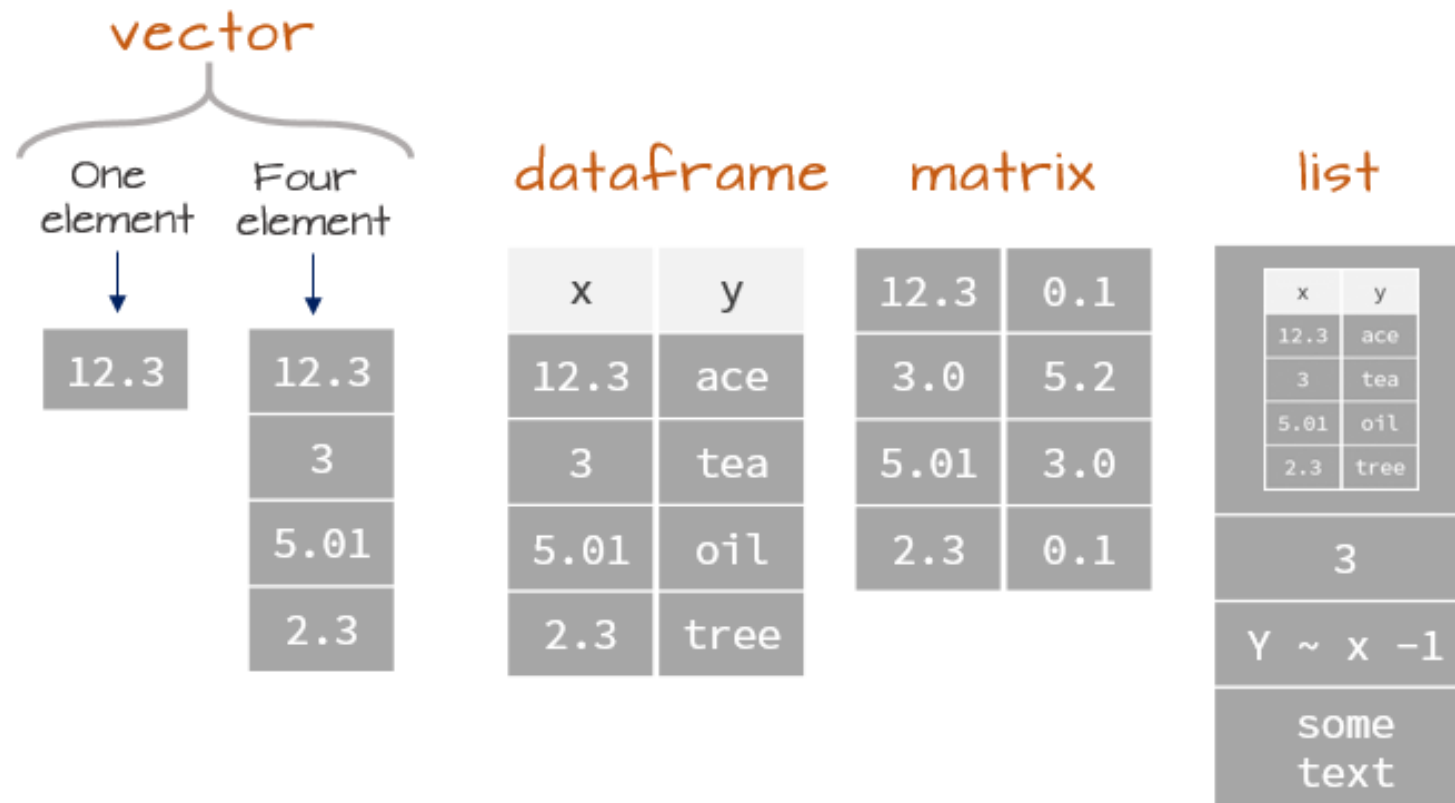
- Each R object has a mode (type) and a class
  - Type = how objects are stored, description of contents
    - Numeric, character, factor, logical
  - Class = general structure
    - Vector, matrix, list, data frame, array

Command	Description
<code>typeof(object)</code>	Determine type of object
<code>class(object)</code>	Determine class of object

# Object Types

- Numeric = values interpreted as numbers
  - 1, 1.1, 2, etc.
- Character = values interpreted as strings (combinations of letters, words, symbols, etc.) and are surrounded by quotation marks
  - “dog”, “cat”
- Factor = values interpreted as group level
  - 0, 1, 2, etc.
- Logical = values interpreted as TRUE/T or FALSE/F

# Object Classes





# Vectors

- Single dimension of values
- All values must have same type (e.g., numeric, character)

Exercise: Create two vectors and determine their lengths

```
num_vec <- c(0,1,2,3)
```

```
char_vec <- c("Yes", "No", "Maybe")
```

```
length(num_vec)
```

# Lists

- Collection of objects of different types

Exercise: Combine previous two vectors into one object

```
list1 <- list(num_vec, char_vec)
```

# Matrix (plural matrices)

- Object with two dimensions
- Must be same type

The screenshot displays the RStudio interface. The left pane shows the R console with the following code and output:

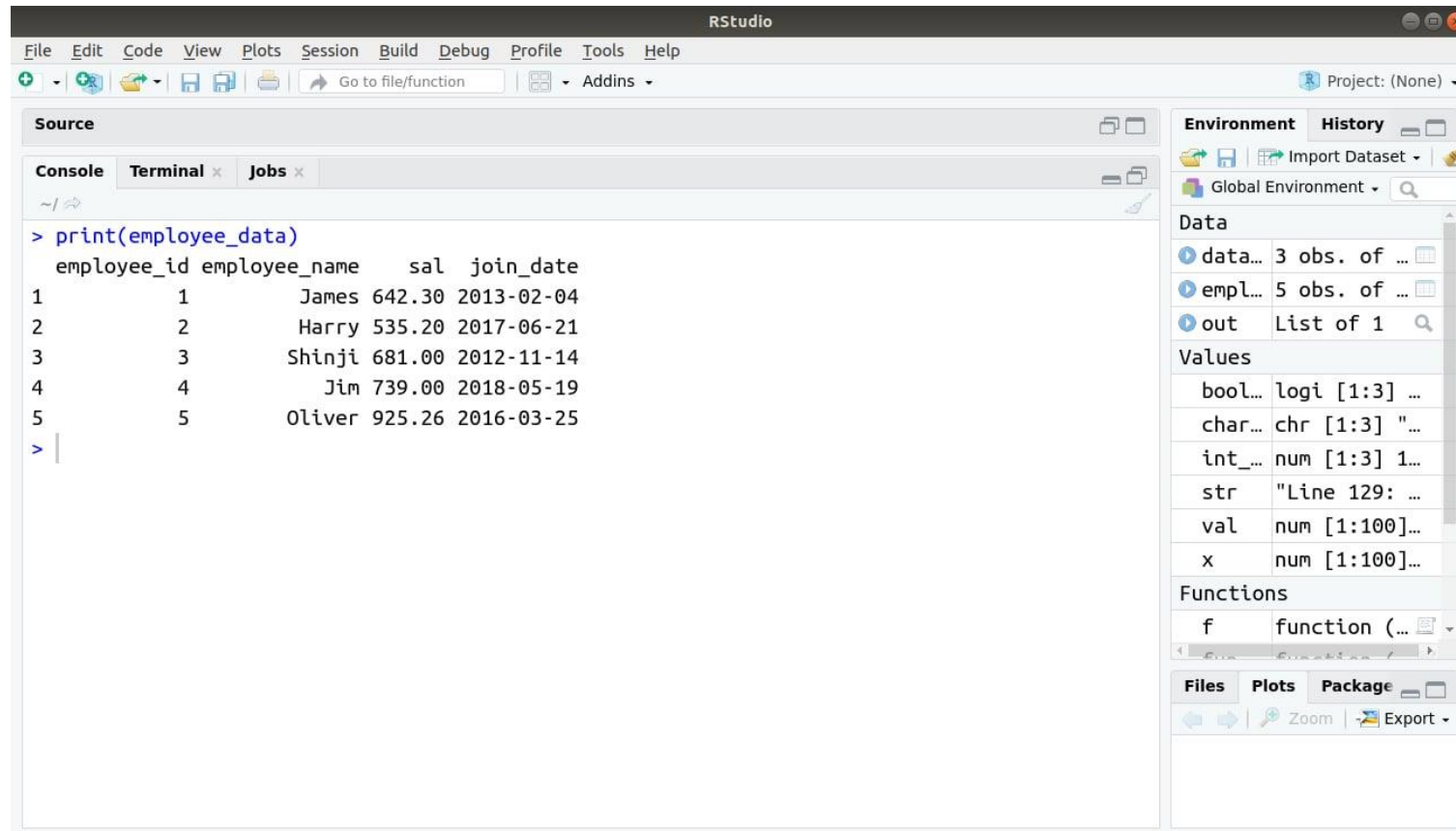
```
> vector_name <- c(1,2,4,9,8,7)
> matrix_name <- matrix(vector_name,nrow=2,ncol=3,byrow=TRUE)
> matrix_name
      [,1] [,2] [,3]
[1,]    1    2    4
[2,]    9    8    7
> |
```

The right pane shows the Environment tab with the following data:

Environment	History	Connections	Tutorial
R 4.1.2 · ~/Documents/Projects/Salmon/			
Import Dataset ▾   2.78 GiB ▾			
R ▾   Global Environment ▾			
Data			
matrix_name	num [1:2, 1:3] 1 9 2 8 4 7		
Values			
vector_name	num [1:6] 1 2 4 9 8 7		

# Data Frames

object with multiple lists, best viewed using the View() function



# Object Types

For vectors and matrices (but not lists or data frames), the type can be found using: `typeof(name)`

- double = numeric
- character

```
> typeof(vector_name)
[1] "double"
```

Exercise: Try to find the type of the number 2 and the word *dog* on your own.

# Object Types Exercise Answer

```
> typeof(2)
```

```
[1] "double"
```

```
> typeof("dog")
```

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

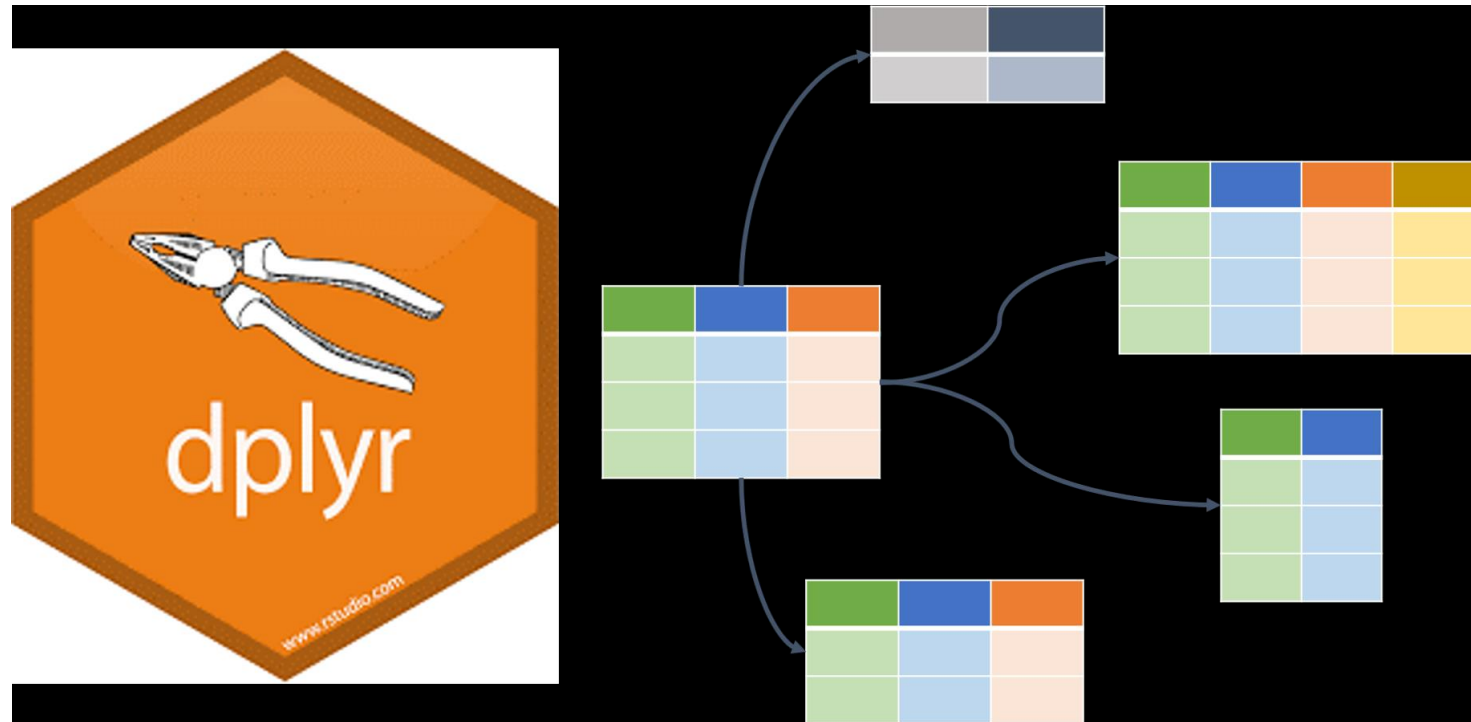
Why do you think lists or dataframes cannot be used with `typeof()`?

10 minute break



# Data Manipulation

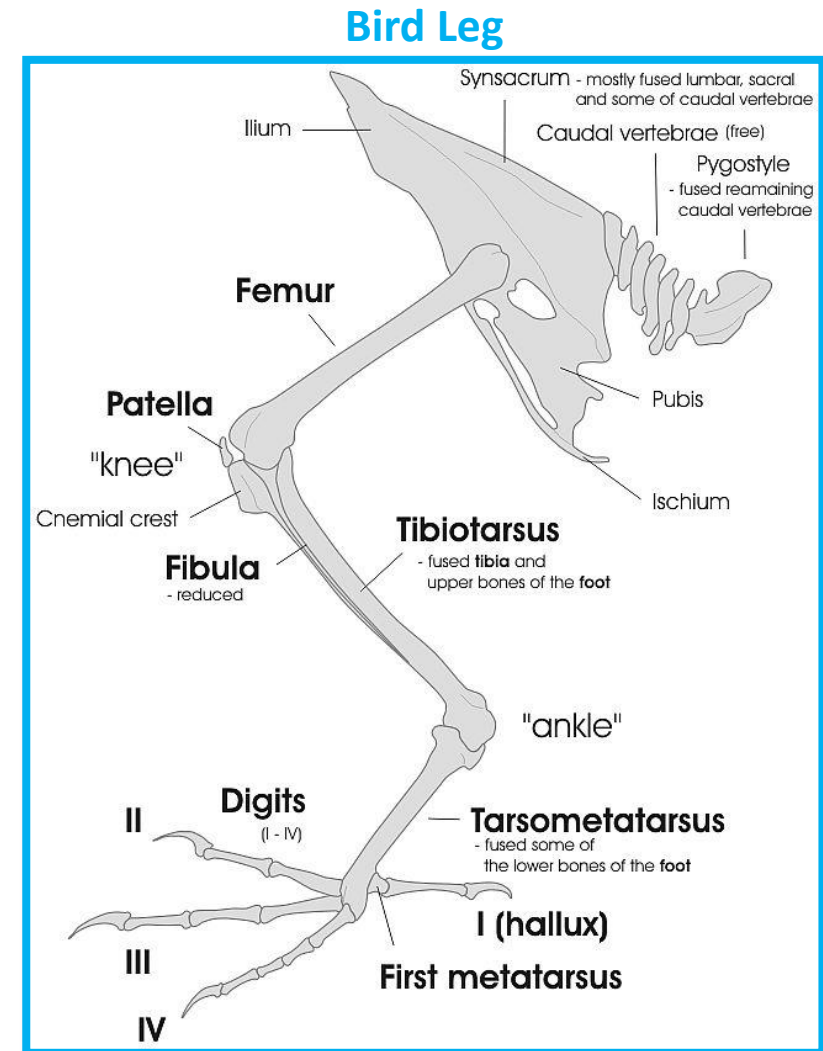
- Choosing subset of the values/observations, filtering, creating new variables, etc.
- Use the dplyr package



# Let us manipulate a data frame

## Exercise

- First, set the working directory.
  - Either through Session > Set Working Directory or the command line
- Read in the .csv file and save as R object called *test*
- View the object
- Install and load the dplyr package



# Answer

```
setwd("C:/Users/richard.Flamio/Documents/Software/Workshop")
```

```
test <- read.csv("R_dataset.csv")
```

```
View(test)  Also, notice the output if you just write test in the console.
```

```
install.packages("dplyr")
```

```
library(dplyr)
```

# Select

- Include or exclude certain variables/columns and reorder the columns
- `select(dataset, variable1, variable2)`
- Variable order matters!
- Removal of a column requires “-” in the beginning of the name

## Exercise:

Produce an R object called *test\_s* that has only information from the 1<sup>st</sup> and 3<sup>rd</sup> column of the data frame

Produce an R object called *test\_s2* that has only information from the 1<sup>st</sup> and 3<sup>rd</sup> column, but the data frame is now in the order Column 3, Column 1

# Answer

```
test_s <- select(test, Sample, Age)
```

```
test_s2 <- select(test, Age, Sample)
```

# Filter

- Reduce dataset based on criteria
- `filter(dataset, criteria1, criteria2)`
- Requires conditional operators

Operator	Description
<code>==</code>	equal to
<code>!=</code>	not equal to
<code>&gt;</code>	greater than
<code>&gt;=</code>	greater than or equal to
<code>&lt;</code>	less than
<code>&lt;=</code>	less than or equal to

- criterion takes form of column name, conditional operator, value
- For example, `Sex=="F"` is all females.

# Filter

Exercise:

Produce a dataset “filter1” that does not include female observations.

Produce a dataset “filter2” that only includes observations with femur length equal to or greater than 8.

Produce a dataset “filter3” that only includes observations of males with tibiotarsus lengths above 8.

Reminder: character type needs to be in parentheses, numeric type does not

# Answer

```
filter1 <- filter(test, Sex!="F")
```

```
filter2 <- filter(test, Femur>=8)
```

```
filter3 <- filter(test, Sex=="M"&Tibiotarsus>8)
```



# Mutate

- Create new variable to the dataset from other variables

Exercise: Produce a column called *Leg* to the data table where you are determining the leg length (femur + tibiotarsus). Call the object *mutate1*.

# Answer

```
mutate1 <- mutate(test, Leg=Femur+Tibiotarsus)
```

# Rename

- `rename(data, variable1_newname=variable1_oldname)`
- Notice the new name goes first

Exercise: Change the name of Sample to ID using an R object *rename1*

# Answer

```
rename1 <- rename(test, ID=Sample)
```

# Combining Datasets

## bind columns

x

A	B	C
a	t	1
b	u	2
c	v	3

+

y

A	B	D
a	t	3
b	u	2
d	w	1

=

A	B	C	A	B	D
a	t	1	a	t	3
b	u	2	b	u	2
c	v	3	d	w	1

CC BY RStudio

- Combines tables
- Notice this function is appending table y to table x based on order
- Make sure rows align!
- `bind_cols(table x, table y)`

# Produce a dataframe

Two step procedure

1. Create vectors.
2. Combine vectors using: `df <- data.frame(vector1, vector2)`

OR

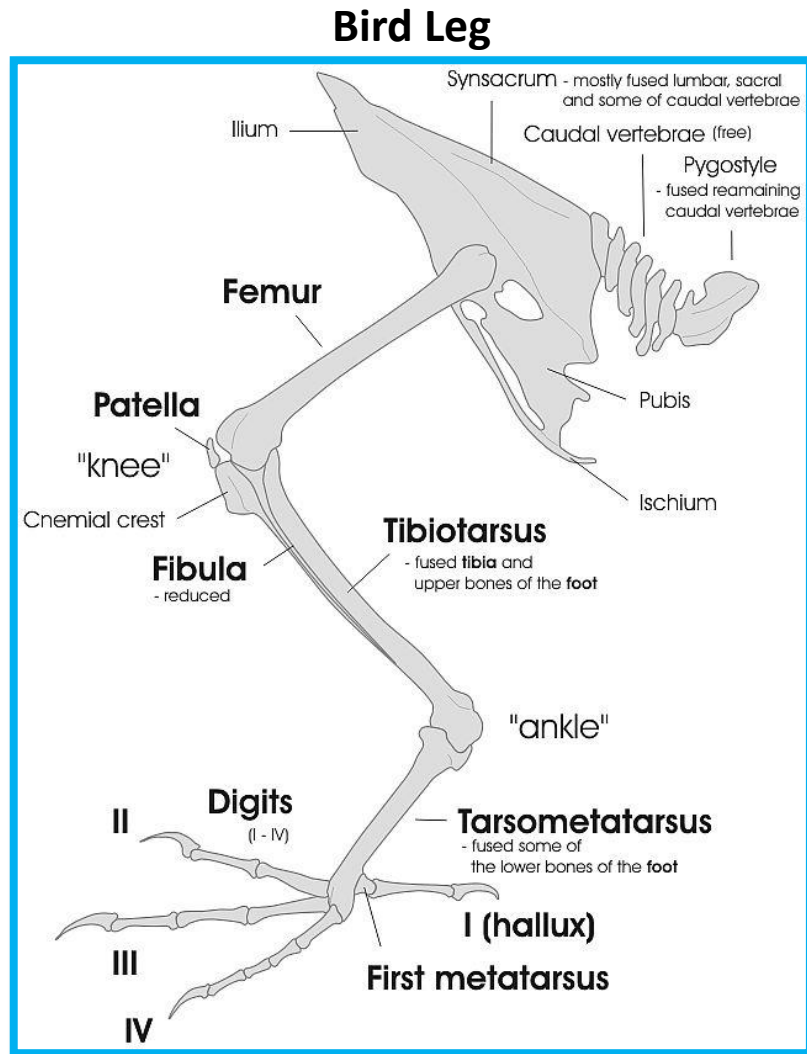
One step procedure

```
df <- data.frame(vector1=c("value1", "value2", ...), vector2=c("value1",  
"value2", ...))
```

Remember: the use of quotations differs between numeric and character type

# Exercise

Create an R object called *Tarsometatarsus\_data* with the following data:



Sample	Tarsometatarsus
AIK001	6.3
AIK002	4.9
AIK003	3.4
AIK004	5.8
AIK005	3.1
AIK006	3.2
AIK007	5.0
AIK008	6.8
AIK009	7.4
AIK010	10.6

# Answer

In coding, there are usually multiple ways to get the same result!

- Option 1: create individual vectors and combine to produce dataframe.
- Option 2: produce dataframe with vectors inside function.
- Option 3: select column 1 from *test* dataframe and combine it with a vector of tarsometatarsus data (can use `bind(cols)`)



# My Answer

```
test3 <- select(test, Sample)
```

```
Tarsometatarsus <- c(6.3, 4.9, 3.4, 5.8, 3.1, 3.2, 5.0, 6.8, 7.4, 10.6)
```

```
df <- data.frame(test3, Tarsometatarsus)
```

# Combining Datasets

- Use mutating joins if you want to combine two tables by matching rows with a column present in both datasets
- I highly recommend using this

a		b	
x1	x2	x1	x3
A	1	A	T
B	2	B	F
C	3	D	T

## Mutating Joins

x1	x2	x3
A	1	T
B	2	F
C	3	NA

**dplyr::left\_join(a, b, by = "x1")**

Join matching rows from b to a.

x1	x3	x2
A	T	1
B	F	2
D	T	NA

**dplyr::right\_join(a, b, by = "x1")**

Join matching rows from a to b.

x1	x2	x3
A	1	T
B	2	F

**dplyr::inner\_join(a, b, by = "x1")**

Join data. Retain only rows in both sets.

x1	x2	x3
A	1	T
B	2	F
C	3	NA
D	NA	T

**dplyr::full\_join(a, b, by = "x1")**

Join data. Retain all values, all rows.

# Exercise

- Combine the tarsometarsus data with the *test* data in one object called *all\_data*.
- Produce a new variable *TotalLength* in the *all\_data* object (call the new object *all\_data2*) that is the summation of femur + tibiotarsus + tarsometatarsus length
- Produce a new R object called *simple\_data* that contains the following variables: Sample, Sex, Age, TotalLength in that order
- View *simple\_data*

# My Answer

```
all_data <- left_join(test,df,by="Sample")
```

*where test = test data and df is the tarsometatarsus data*

```
all_data2 <- mutate(all_data,  
TotalLength=Femur+Tibiotarsus+Tarsometatarsus)
```

```
simple_data <- select(all_data2, Sample, Sex, Age, TotalLength)
```

```
View(simple_data)
```

# Combining functions with a pipe operator

- The pipe operator (`%>%`) allows you to combine dplyr functions in the same line.
- `object_name <- data %>% function(argument1) %>% function(argument 2)`

## Exercise:

Produce an R object *female\_data* from *test* that contains only female data and excludes the sex column.

# Answer

```
female_data <- test %>% filter(Sex=="F") %>% select(-Sex)
```

# References

- [grunwaldlab.github](https://github.com/grunwaldlab)
- [https://bookdown.org/kdonovan125/ibis\\_data\\_analysis\\_r4/documenting-your-results-with-r-markdown.html](https://bookdown.org/kdonovan125/ibis_data_analysis_r4/documenting-your-results-with-r-markdown.html)