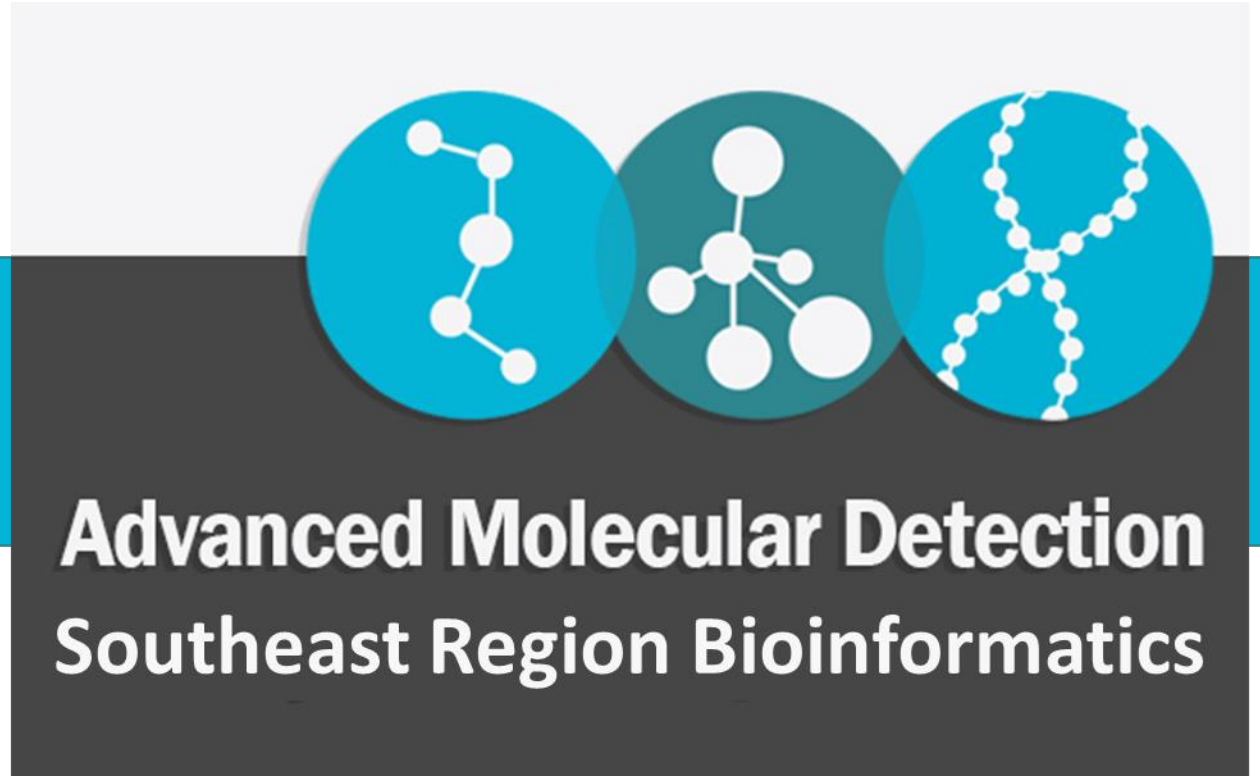



**Importing Data & Iterations in R**

06/10/2024

# Outline

Agenda

Notes

How to Import data in R?

Iterations & Loops in R

Questions

# Agenda

**June 24** – Graphics in R Part - 4
**July 8** – Tidyverse Package in R Part - 5

Future Trainings
- ONT & FL's Flisochar pipeline
- StaPH-B Toolkit Programs/Pipelines
- GISAID flagged SARS-CoV-2
- R Training Series
- Dryad pipeline
- …and more

Advanced Molecular Detection
Southeast Region Bioinformatics

# Updates

- Staffing changes at BPHL

Advanced Molecular Detection
Southeast Region Bioinformatics

# Packages

- Packages are an appropriate way to organize the work and share it with others
- Typically, a package will include code (not only R code!), documentation for the package and the functions inside, some tests to check everything works as it should, and data sets
- Packages in R are a set of R functions, compiled code, and sample data. These are stored under a directory called “library” within the R environment

# Difference Between Package & Library

- **library():** the command used to load a package, and it refers to the place where the package is contained, usually a folder on our computer
- **Package:** a collection of functions bundled conveniently. The package is an appropriate way to organize our own work and share it with others

# What is a Repository?

- A repository is a place where packages are located and stored so you can install R packages from it. Most popular repositories include:
  - CRAN - Comprehensive R Archive Network(CRAN) is the official repository maintained by the R community
  - Bioconductor - Bioconductor is a topic-specific repository, intended for open-source software for bioinformatics
  - GitHub - GitHub is the most popular repository for open-source projects

Advanced Molecular Detection
Southeast Region Bioinformatics

# How to Install Packages in R?

- Installing R Packages from CRAN:

```
install.packages("package name")
```

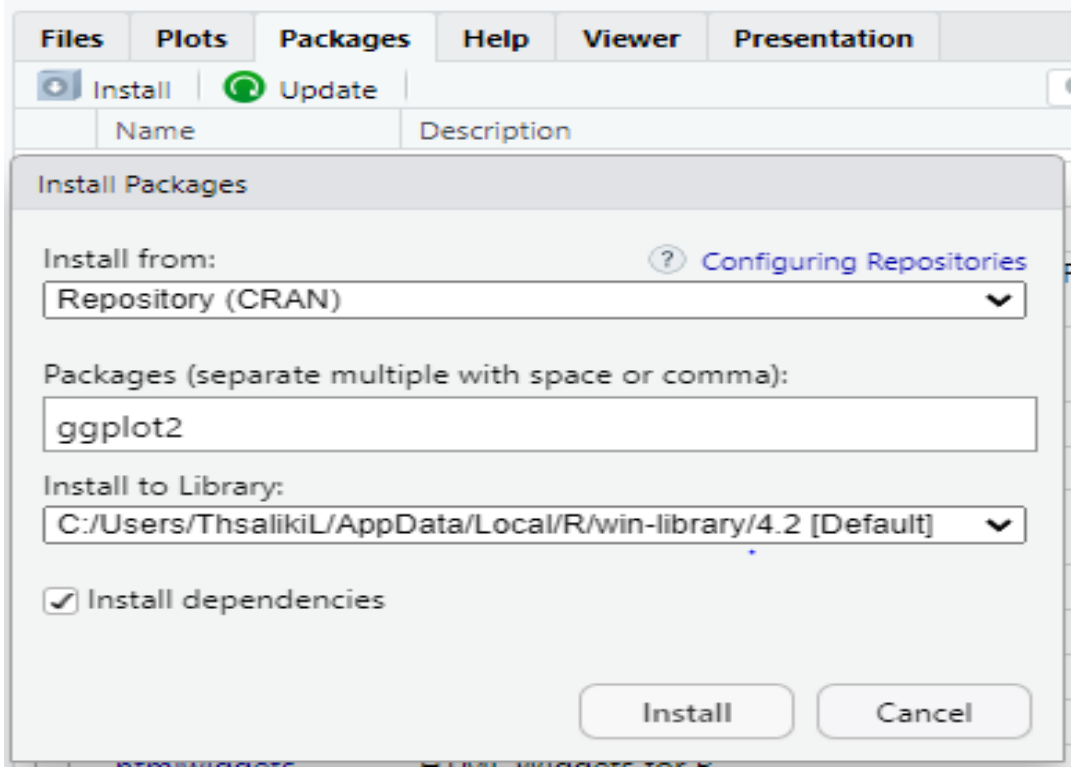
- Installing Package from CRAN is the most common and easiest way
- To install more than a package at a time, a character vector is given in the first argument of the **install.packages()** function

```
install.packages(c("tidyverse", "plotly"))
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# Install packages from package tab in R studio

- Packages can also be installed from the fourth pane in the R studio under packages tab
- When you click on the packages tab, it should display install and update options
- Click on the install option, which pops up a box as shown here, then give the name of the package you want to install and click install




Advanced Molecular Detection
Southeast Region Bioinformatics

# Installing Packages from Bioconductor

- BiocManager package is used to install and manage Bioconductor packages

```
install.packages("BiocManager")
```

- This will install some basic functions which are needed to install bioconductor packages, such as the biocLite() function. To install the core packages of Bioconductor just type it without further arguments:

```
BiocManager::install("edgeR")
```

- This will install the edgeR package and its dependencies from the Bioconductor repository

Advanced Molecular Detection
Southeast Region Bioinformatics

# How to Load Packages in R?

- Load a package using the library function

```
library(dplyr)
```

- Load a package using the require function

```
require(dplyr)
```

- Both functions attempt to load the specified package, but there is a subtle difference between the two: library() returns an error if the package is not found or cannot be loaded, whereas require() returns a warning and sets the value of the package variable to FALSE

Advanced Molecular Detection
Southeast Region Bioinformatics

# Packages from CRAN

- Project & file management

| | |
|---|---|
| here | File paths relative to R project root folder |
| rio | Import/export of many types of data |
| openxlsx | Import/export of multi-sheet Excel workbooks |

- Plots general

| | |
|---|---|
| ggplot2 | Included in tidyverse |
| cowplot | Combining plots |
| patchwork | Combining plots (alternative) |
| RColorBrewer | Color scales |
| ggnewscale | To add additional layers of color schemes |

# Packages from CRAN

- General data management

| tidyverse | Includes many packages for tidy data wrangling & presentation |
|---|---|
| dplyr | Data management |
| tidyr | Data management |
| ggplot2 | Data visualization |
| stringr | Work with strings & characters |
| forcats | Work with factors |
| lubridate | Work with dates |
| purrr | Iteration & working with lists |
| linelist | Cleaning linelists |
| naniar | Assessing missing data |

# Packages from CRAN

- Phylogenetics

| | |
|---|---|
| ggtree | Visualization & annotation of trees |
| ape | Analysis of phylogenetics & evolution |
| treeio | To visualize phylogenetic files |

- Plots – specific types

| | |
|---|---|
| gghighlight | Highlight a subset |
| ggrepel | Smart labels |
| plotly | Interactive graphics |
| gganimate | Animated graphics |

Advanced Molecular Detection
Southeast Region Bioinformatics

# Import Data into R

Data can be in different formats like txt, csv or any other delimiter separated files. Data can also be manipulated, analyzed and reported

- Import CSV file into R
- Import Data from a Text file
- Import Data from Excel
- Import Data from a delimiter file

Advanced Molecular Detection
Southeast Region Bioinformatics

# Import CSV file into R

- Import CSV file into R using **read.csv()** method

```
x <- read.csv(path, header = TRUE, sep = ",")
```

- Arguments:
    - Path – Path of the file to be imported
    - Header – By default: TRUE. Indicator of whether to import column headings
    - Sep – "," The separator for the values in each row

Advanced Molecular Detection
Southeast Region Bioinformatics

# Import Data from a Text file

- A text file can easily be imported into R using **read.table()**
- **read.table()** is used to read a file in table format. This function is easy to use and flexible

```
x <-  read.table("file_name.txt", header=TRUE/FALSE)
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# Import Data from Excel

- **read_excel()** function is used to import excel files from readxl library of the R language with the name of the file as the parameter.
- **readxl()** package can import both .xlsx and .xls files. This package is pre-installed in R-Studio

```
x <- read_excel(filename, sheet, dtype = “float32”
```

- Parameters:
    - filename:-File name to read from.
    - sheet:-Name of the sheet in Excel file.
    - dtype:-Numpy data type

Advanced Molecular Detection
Southeast Region Bioinformatics

# Import Data from a Delimiter File

- **read.delim()** function is used to read the delimited files into the list
- The file is by default separated by a tab which is represented by sep="", that separated can be a comma(, ), dollar symbol($), etc.

```
x <- read.delim("file_name.txt", sep="", header=TRUE)
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# Working with Dates

- Dates and times can be wrangled easily in R with the help of lubridate package

```
library(lubridate)
```

- Upon import of raw data, R often interprets dates as character objects, which means they cannot be used for general date operations such as making time series and calculating time intervals
- lubridate package is not part of core tidyverse as we use this only when working with dates/times

Advanced Molecular Detection
Southeast Region Bioinformatics

# Packages for dates/times

| | |
|---|---|
| lubridate | general package for handling & converting dates |
| parsedate | has functiom to “guess” messy dates |
| aweek | converts dates to weeks, and weeks to dates |
| zoo | additional date/time functions |

Advanced Molecular Detection
Southeast Region Bioinformatics

# Creating date/times

- There are 3 types of date/time data that refer to an instant in time:
    - A **date** – Tibbles print this as <date>
    - A **time** within a day – Tibbles print this as <time>
    - A **date-time** is a date plus a time – it uniquely identifies an instant in time (typically to the nearest second) – Tibbles print this as <dttm>
- To get current date or date-time we can use today() or now()

```
> library(lubridate)

Attaching package: 'lubridate'

The following objects are masked from 'package:base':

    date, intersect, setdiff, union

> today()
[1] "2023-06-27"
> now()
[1] "2023-06-27 08:25:04 EDT"
>
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# Create date/time From a String

- Parsing dates with year, month, and date components
- These functions also take unquoted numbers which helps in filtering date/time data. Ymd() is short and unambiguous
- To create a date-time, add an underscore and one or more of “h”, “m”, “s” to the name of the parsing function along with ymd
- We can also force create a date-time from a date by supplying a timezone

```
> ymd("2023-06-27")
[1] "2023-06-27"
> mdy("June 27th, 2023")
[1] "2023-06-27"
> dmy("27-06-2023")
[1] "2023-06-27"
> ymd(20230627)
[1] "2023-06-27"
> ymd_hms("2023-06-27 14:20:52")
[1] "2023-06-27 14:20:52 UTC"
> mdy_hm("06/27/2023 04:20")
[1] "2023-06-27 04:20:00 UTC"
> #force create timezone from a date by supplying a timezone
> ymd(20230627, tz = "PST")
[1] "2023-06-27 08:00:00 PST"
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# Iterations and Loops

- Iteration is a tool for reducing duplication, which helps you to do the same thing to multiple inputs, repeating the same operation on different columns, or in different datasets
- A for-loop is one of the main control-flow constructs of the R programming language
- For-loop is used to iterate over a collection of objects, such as a vector, a list, a matrix, or a dataframe, and apply the same set of operations on each item of a given data structure

# For Loop

- For loop has three components:
  - The **output** – always allocate sufficient space for the output before you start the loop. This vector function has two arguments which include type of the vector and length of the vector
  - The **sequence** – i in seq_along(df), determines what to loop over. Each run of the for loop will assign i to a different value from seq_along(df)
  - The **body** – output[[i]] <- mean(df[[i]], this is the code which does the work. This will run repeatedly each time with a different value for i

# For Loop Example

```
> # for loop
> library(tidyverse)
> df <- tibble(a=rnorm(10), b=rnorm(10), c=rnorm(10), d=rnorm(10))
> output <- vector("double", ncol(df))
> for (i in seq_along(df)) {output[[i]] <- mean(df[[i]])}
> output
[1]  0.07029384  0.09199256 -0.11446003  0.26333337
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# Conditions & If Statements

| Operator | Name | Example |
|---|---|---|
| == | Equal | x == y |
| != | Not equal | x != y |
| > | Greater than | x > y |
| < | Less than | x < y |
| >= | Greater than or equal to | x >= y |
| <= | Less than or equal to | x <= y |

# If Statement

```
> # if statement
> a <- 20
> b <- 156
> if (b > a) {print("b is greater than a")}
[1] "b is greater than a"
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# Else If Statement

```
> # else if statement
> a <- 20
> b <- 20
> if (b > a) {print("b is greater than a")} else if (a == b) {print("a and b
are equal")}
[1] "a and b are equal"
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# If Else Statement

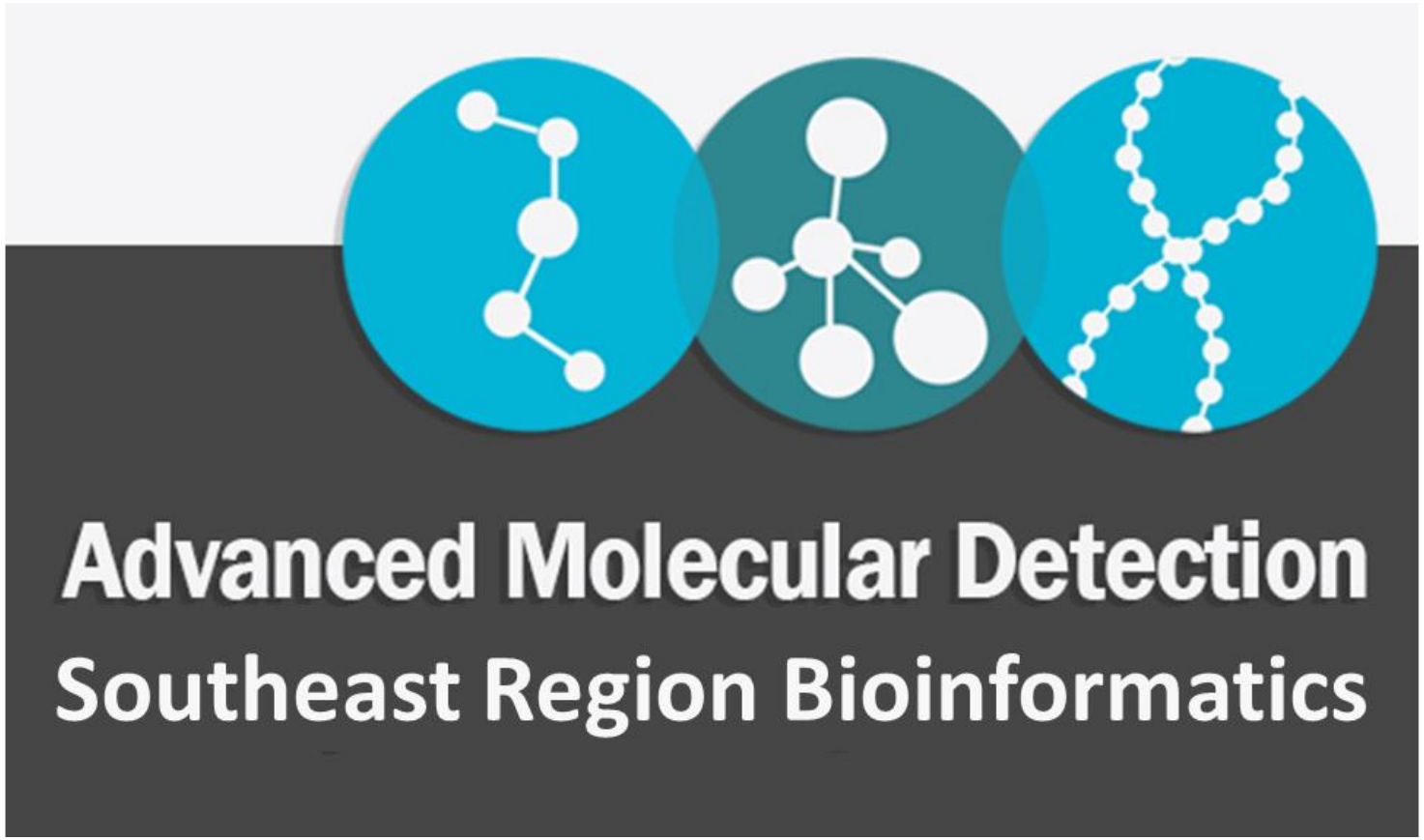
```
> # if else statement
> a <- 156
> b <- 20
> if (b > a) {print("b is greater than a")} else if (a==b) {print("a and b a
re equal")} else {print("a is greater than b")}
[1] "a is greater than b"
>
```

Advanced Molecular Detection
Southeast Region Bioinformatics

# References to Learn R

- Welcome | R for Data Science (had.co.nz)
- R-intro.pdf (r-project.org)
- R for applied epidemiology and public health | The Epidemiologist R Handbook (epirhandbook.com)

Advanced Molecular Detection
Southeast Region Bioinformatics




# Questions?

bphl-sebioinformatics@flhealth.gov

**Molly Mitchell, PhD**

Bioinformatics Supervisor

Molly.Mitchell@flhealth.gov

**Sam Marcellus, MPH**

Bioinformatician

Samantha.Marcellus@flhealth.gov