



Workshop: Statistics & Bioinformatics Using R

Workshop Introduction

ASMS 2017 Annual Meeting • 6/6/2017

Ryan Benz & Jeffrey Jones

Goals of the Workshop

- Present and discuss R and the R ecosystem as a useful tool for performing scientific data analysis, visualization and reporting
- Provide a high-level view of what R can do and why it's a great language for scientific data analysis
- Show a few concrete examples of how R can be used in the context of mass spectrometry (and related) data
- Discuss the data analysis needs of the community, and how R might fit in

All workshop materials can found found at:
<https://goo.gl/x247gC>

Potential Questions for Discussion

- Do you see a need for a programming language tool, like R, in your research and work?
 - Are existing tools (commercial and open source) adequate for your needs?
 - Is it just easier to collaborate with statisticians/data analysts when needed?
- What are your main data challenges?
 - Existing tools don't always do exactly what I want or need
 - Data storage, handling, access, processing is hard/challenging
 - Data analysis takes too long or is difficult to do
 - ...
- If there is a need for a tool like R and the need isn't being met, why not?
 - Students, employees aren't interested in using or learning
 - More training needed (academic & professional)
 - Learning to use the tools (e.g. learning to program) is too hard

More Thoughts for Discussion

- In the “–omics” age, all data is “big data”
 - Simply dealing with data can be a challenge (size, amount, complexity)
 - Standard tools (e.g. spreadsheets) are often inadequate
 - Data processing and analysis can take a *long* time, require *big* resources
- We are currently in a period of flux regarding “big data”
 - New tools, methods are constantly being developed – exciting!
 - New tools, methods are not always easy to use – frustrating!
 - What package/language/method/algorithm to use?
- For scientists, coding skills are now essential
 - We don’t all have to be expert software developers
 - Basic/essential programming skills will get you a very long way
- Reproducible research is an extremely important topic in all fields of science right now – coding skills can help

Introduction to R and R Studio

R is A Great Choice for Scientists

- R describes itself as

“a language & environment for statistical computing and graphics” - r-project.org

More than just the language

Data

Visualization

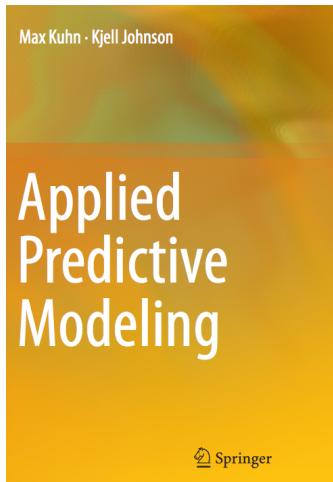
- Free software, GPLv2 license, cross platform (Win, Mac, Linux)
- Great IDE (RStudio)
- Easy package installation and management (CRAN, Bioconductor)
- Active community
- One of the top languages for data science

R Is At The Forefront of Data Science



ggplot2

<http://ggplot2.tidyverse.org/>



Applied
Predictive
Modeling

Springer



<https://shiny.rstudio.com>

<http://appliedpredictivemodeling.com>

The tidyverse



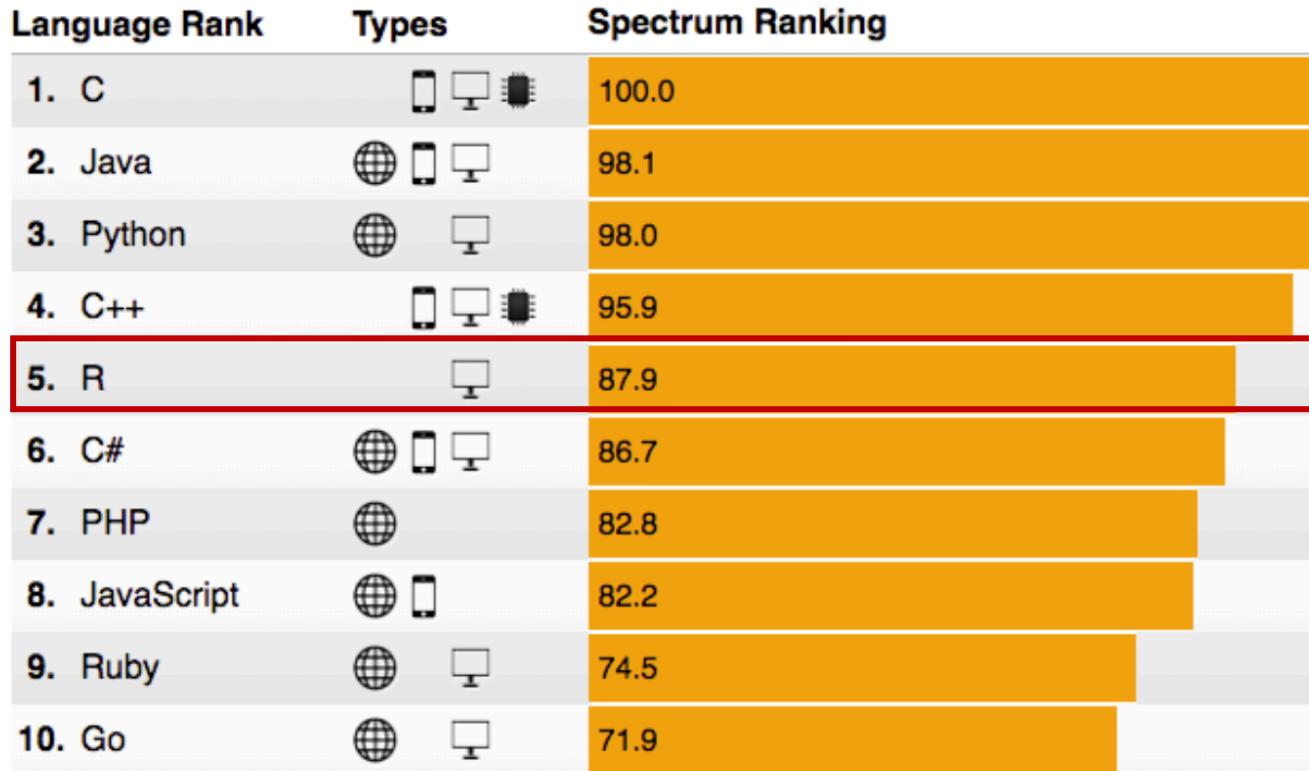
Components



<http://tidyverse.org>

R Has Become a Top Programming Language

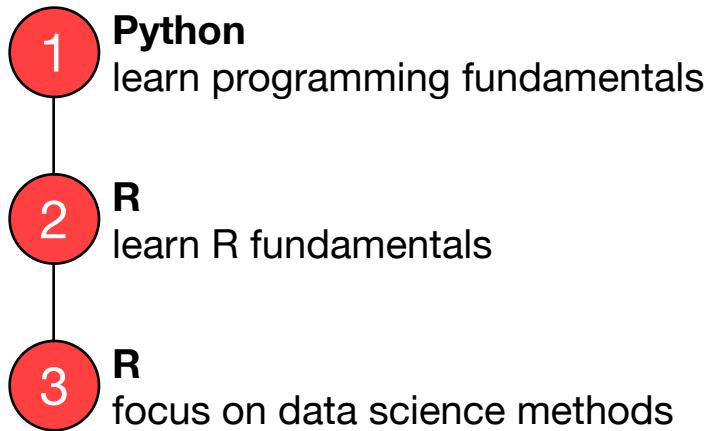
IEEE 2016 Top Programming Languages



<http://spectrum.ieee.org>

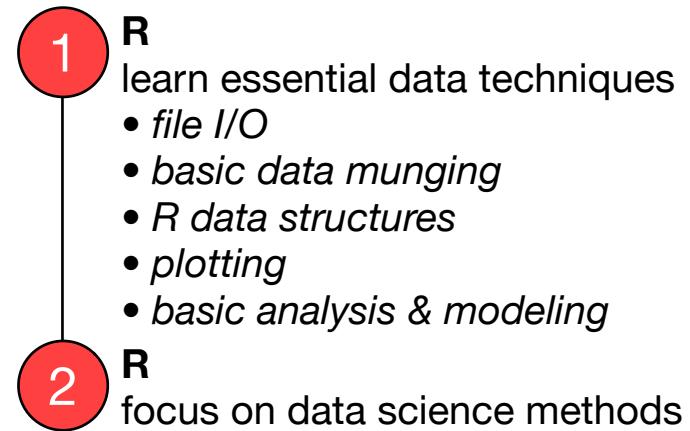
Two Different Tracks for Learning R

The Programmer



Learning is focused on forming a **strong programming base** from which to learn data science methods

The Applied Data Scientist



Learning is focused on **practical coding** specifically for data analysis

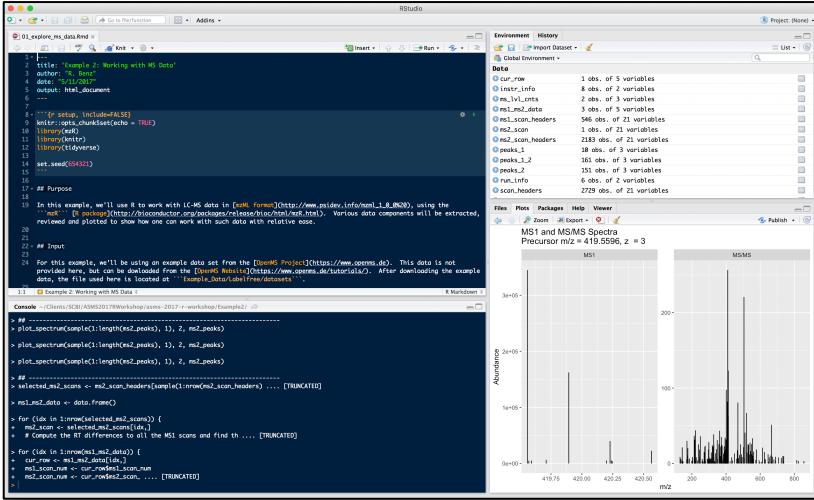
RStudio: A Top Notch IDE for Data Analysis

The screenshot displays the RStudio interface with several red callout boxes highlighting its features:

- text editor**: Located in the top-left corner, showing an R Markdown script titled "01_explore_ms_data.Rmd". The script includes code for setting up packages like mzR and tidyverse, and a detailed explanation of the task.
- env. vars, history**: Located in the top-right corner, pointing to the Environment tab in the Global Environment pane where various R objects and their dimensions are listed.
- files, plots, help**: Located in the bottom-right corner, pointing to the Plots tab which displays two mass spectra: MS1 and MS/MS. The MS1 plot shows abundance versus m/z for a precursor at m/z = 419.5596, z = 3.
- console**: Located in the bottom-left corner, showing the R Console window with a truncated list of R commands related to plotting spectra and processing data frames.

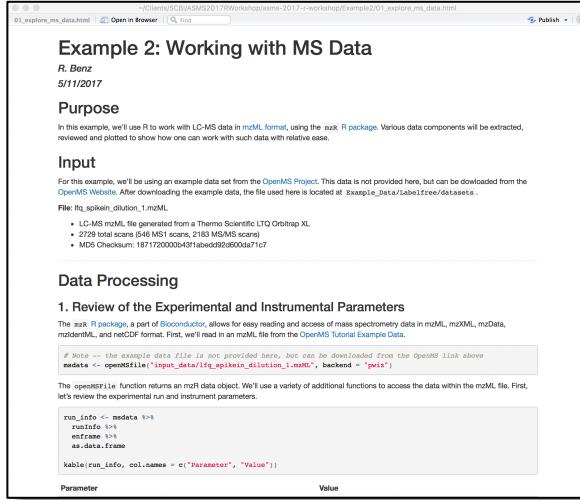
RStudio Does More Than Just Code

Data analysis workspace



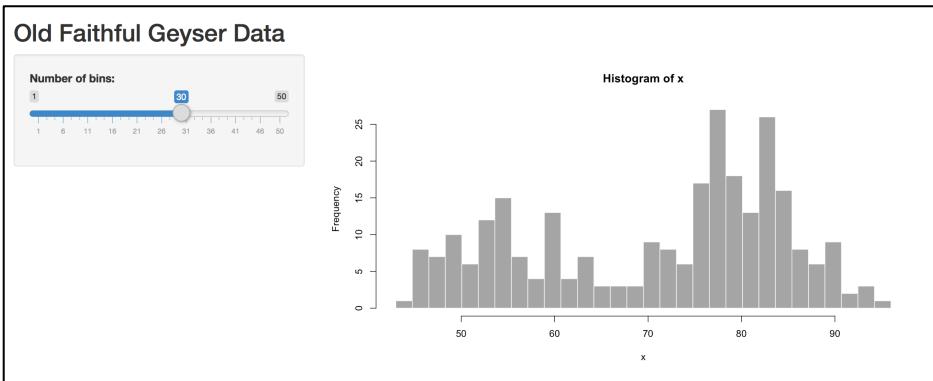
A screenshot of the RStudio interface. The code editor shows an R script named `01_exploring_ms_data.R`. The script includes comments explaining its purpose and how it will work with MS data. It uses the `askR` package to interact with the OpenMS Project's API. The plot window displays a mass spectrum with two panels: 'LC-MS' and 'MS/MS'. The x-axis is labeled 'm/z' and ranges from 400 to 800. The y-axis is labeled 'Abundance' and ranges from 0 to 3e+05. The plot shows several peaks, with a very large peak at approximately m/z 419.

Reproducible Reports



A screenshot of a web browser displaying a reproducible report titled "Example 2: Working with MS Data". The report was created by R. Benz on 5/11/2017. It has a "Purpose" section describing the use of R to work with LC-MS data in mzML format. An "Input" section shows the mzML file `lq_spirken_dilution_1.mzML` with details like 2729 total scans, 546 MZS, and a MD5 checksum. A "Data Processing" section contains R code for reading the data and extracting parameters. A "Parameter" table lists various parameters with their values.

Interactive Web Applications (Shiny)



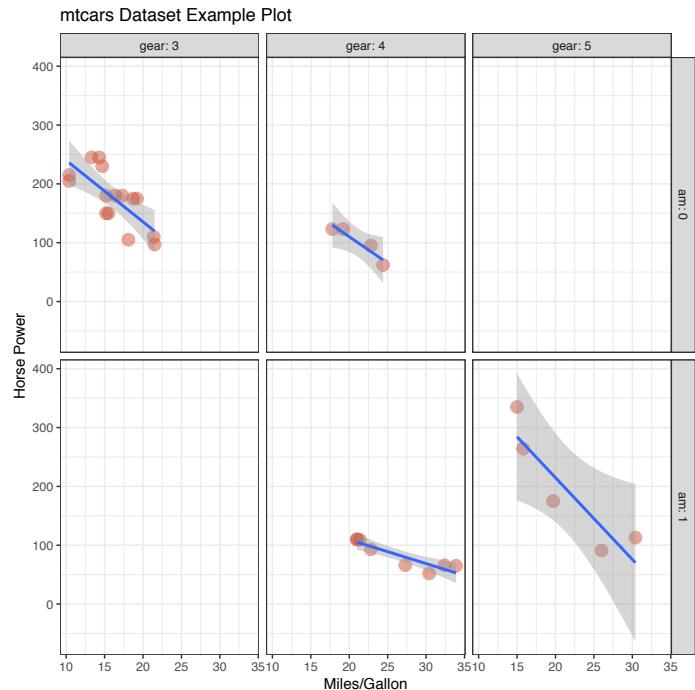
- Git/GitHub integration
- Publish to the web
- Work in the cloud with RStudio Server

A ggplot2 Example

The Code

```
# Set-up the plot & specify data
ggplot(mtcars, aes(mpg, hp)) +
  # Represent data as points
  geom_point(size = 4, color = "tomato3", alpha = 0.5) +
  # Separate out the data by transition type and # of gears
  facet_grid(am ~ gear, labeller = label_both) +
  # Add a linear model fit for each group
  geom_smooth(method = "lm") +
  # Label the x-axis
  xlab("Miles/Gallon") +
  # Label the y-axis
  ylab("Horse Power") +
  # Add a title
  ggtitle("mtcars Dataset Example Plot") +
  # Use the "black and white" theme
  theme_bw()
```

The Output



Plots are built-up layer by layer,
giving you the flexibility to create just what you

A Predictive Modeling Example

The Task

Given an image of a handwritten digit, predict what the digit actually is

The screenshot shows a competition page for "Digit Recognizer" on Kaggle. On the left, there are four handwritten digits: 9, 6, 4, 5, 4, 0, 7, 4, 0, 1, 3, 1, 3, 4, 7, 2, 7, 1, 2, 1, 1, 7, 4, 2, 3, 5, 1, 2, 4, 4. To the right of these digits is the title "Digit Recognizer". Below the title is a description: "Learn computer vision fundamentals with the famous MNIST data" and "1,640 teams · 3 years to go". At the bottom of the page, there is a navigation bar with links: "Overview" (which is underlined in blue), "Data", "Kernels", "Discussion", "Leaderboard", and "More". To the right of these links is a blue button labeled "Submit Predictions".

Data, example code, and competition leaderboards all available from Kaggle (<https://www.kaggle.com/c/digit-recognizer>)

A Predictive Modeling Example

Code

This script has been released under the [Apache 2.0](#) open source license.

[Download Code](#)

```
1 # Creates a simple random forest benchmark
2
3 library(randomForest)
4 library(readr)
5
6 set.seed(0)
7
8 numTrain <- 10000
9 numTrees <- 25
10
11 train <- read_csv("../input/train.csv")
12 test <- read_csv("../input/test.csv")
13
14 rows <- sample(1:nrow(train), numTrain)
15 labels <- as.factor(train[rows,1])
16 train <- train[rows,-1]
17
18 rf <- randomForest(train, labels, xtest=test, ntree=numTrees)
19 predictions <- data.frame(ImageId=1:nrow(test), Label=levels(labels)[rf$test$predicted])
20 head(predictions)
21
22 write_csv(predictions, "rf_benchmark.csv")
```

This model is 93.5% accurate on the test set

Most of the code is about preparing the data

Build the model and make predictions in 2 lines of code!

<https://www.kaggle.com/benhamner/random-forest-benchmark-1/code/code>

dplyr Data Pipelines

- Makes common data analysis manipulations easy and human readable
- Pipelines are built from data sets + data manipulation verbs (functions)
 - `mutate()` adds new variables that are functions of existing variables
 - `select()` picks variables based on their names.
 - `filter()` picks cases based on their values.
 - `summarise()` reduces multiple values down to a single summary.
 - `arrange()` changes the ordering of the rows.

<http://dplyr.tidyverse.org>

dplyr Data Pipelines

Consider the mtcars data set

```
> head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

... 32 total entries

dplyr Data Pipelines

Goal

Compute the **mean horsepower**
by **number of cylinders**

A simple first approach might
look like this 

```
hp_vals_4 <- c()
hp_vals_6 <- c()
hp_vals_8 <- c()
for (i in 1:nrow(mtcars)) {
  current_car <- mtcars[i,]

  cyl <- current_car$cyl
  hp <- current_car$hp

  if (cyl == 4) {
    #... accumulate values
    hp_vals_4 <- c(hp_vals_4, hp)
  } else if (cyl == 6) {
    #... accumulate values
  } #...
  #...

}
# compute summary stats
mean_hp_4 <- mean(hp_vals_4)
#...
```

dplyr Data Pipelines

The dplyr approach is MUCH simpler, easier to read and understand (and does more)

```
mtcars %>%  
  group_by(cyl) %>%  
  summarize(n = n(),  
            mean_hp = mean(hp),  
            median_hp = median(hp),  
            stdev_hp = sd(hp))
```



```
# A tibble: 3 x 5  
  cyl     n   mean_hp median_hp stdev_hp  
  <dbl> <int>    <dbl>      <dbl>    <dbl>  
1     4     11    82.63636    91.0  20.93453  
2     6     7    122.28571   110.0  24.26049  
3     8     14   209.21429   192.5  50.97689
```

dplyr Data Pipelines

Examples of other questions you can easily “ask”

Which automatic car has
the most horsepower?

```
mtcars %>%  
  filter(am == 0) %>%  
  filter(hp == max(hp))
```

Which automatic and
manual cars are the most
fuel efficient?

```
mtcars %>%  
  group_by(am) %>%  
  filter(mpg == max(mpg)) %>%  
  arrange(desc(mpg))
```

dplyr allows you to **focus on the question** not the
implementation details of how to answer it

A Shiny Example

ui.R

```
library(shiny)

shinyUI(fluidPage(
  # Application title
  titlePanel("Old Faithful Geyser Data"),

  # Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("bins",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30)
    ),
    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot")
    )
  )
))
```

Define the interface

server.R

```
library(shiny)

shinyServer(function(input, output) {
  output$distPlot <- renderPlot({
    # generate bins based on input$bins from ui.R
    x     <- faithful[, 2]
    bins <- seq(min(x), max(x), length.out = input$bins + 1)

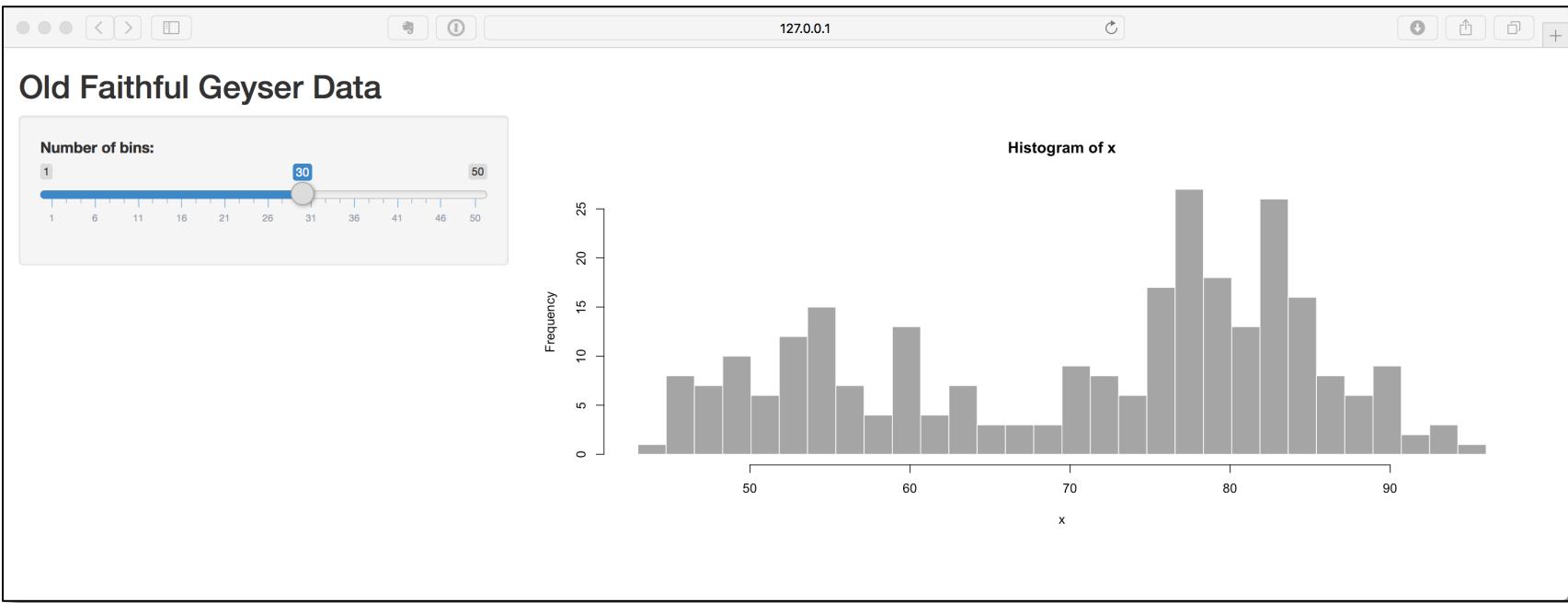
    # draw the histogram with the specified number of bins
    hist(x, breaks = bins, col = 'darkgray', border = 'white')
  })
})
```

Define the application logic, functionality

Default Shiny example code
<https://www.rstudio.com>



A Shiny Example



*A full, interactive (albeit very simple) web app created in <50 lines of code
No need to understand any web technologies*

*Default Shiny example code
<https://www.rstudio.com>*

Getting Started with R & RStudio

Follow these steps to run and work with the workshop examples

1. Download and install R

<https://www.r-project.org>

- Choose the cloud.r-project.org mirror, https is preferable
- Latest version as of 06/02/2017:
3.4.0 (2017-04-21) -- "You Stupid Darkness"

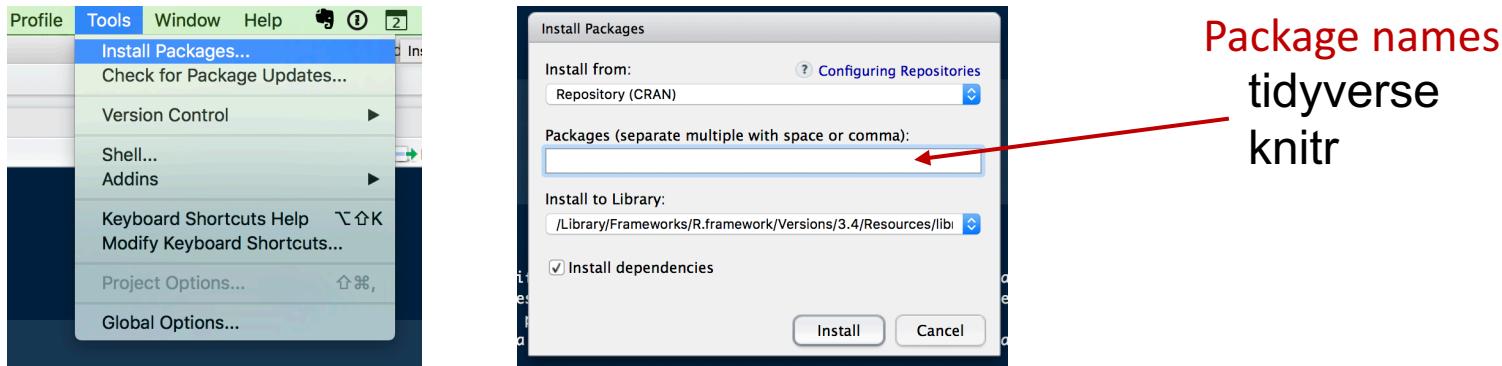
2. Download and install RStudio

<https://www.rstudio.com/products/rstudio/>

- Choose the Desktop, Open Source Edition (free)

Getting Started with R & RStudio

3. Install R packages from RStudio



4. Install mzR from Bioconductor from the R prompt

<https://bioconductor.org/packages/release/bioc/html/mzR.html>

Installation

To install this package, start R and enter:

```
## try http:// if https:// URLs are not supported
source("https://bioconductor.org/biocLite.R")
biocLite("mzR")
```

Additional Topics & Examples

- R Example #1: Working with Experimental Meta-data
- R Example #2: Workings with LC-MS Data
- Example interactive web application (Shiny)
- Discussion of key data science themes for scientific analysis:
Reproducible data analysis

Key Data Science Themes for Scientific Data Analysis

The “Reproducibility/Replication Crisis” is An Important Current Topic

Essay

Why Most Published Research Findings Are False

John P. A. Ioannidis

PLoS Medicine, Aug 2005 2(8)

Science & Environment

Most scientists 'can't replicate studies by their peers'

By Tom Feilden
Science correspondent, Today programme

© 22 February 2017 | Science & Environment

f t m Share



Scientists attempting to repeat findings reported in five landmark cancer studies confirmed only two

Science is facing a "reproducibility crisis" where more than two-thirds of researchers have tried and failed to reproduce another scientist's

Slatest YOUR NEWS COMPANION MARCH 3 2016 2:10 PM

Psychologists Call Out the Study That Called Out the Field of Psychology

By Rachel E. Gross

5.9k 373 23

Remember that study that found that most psychology studies were wrong? Yeah, that study was wrong. That's the conclusion of four researchers who recently interrogated the methods of that study, which itself interrogated the methods of 100 psychology studies to find that very few could be replicated. (Wow.) Their damning commentary will be published Friday in the journal *Science*. (The scientific body that publishes the journal sent *Slate* an early copy.)

In case you missed the hullabalo: A key feature of the scientific method is that scientific results should be reproducible—that is, if you run an experiment again, you should get the same results. If you don't, you've got a problem. And a problem is exactly what 270 scientists found last August, when they decided to try to reproduce 100 peer-reviewed journal studies in the field of social psychology. Only around 39 percent of the reproduced studies, they found, came up with similar results to the originals.

FUTURE TENSE THE CITIZEN'S GUIDE TO THE FUTURE. APRIL 15 2016 7:08 AM

FROM SLATE, NEW AMERICA, AND ASU

The Reproducibility Crisis Is Good for Science

Weak statistics are getting called out, and replication is gaining respect.

By Morya Baker

590 127 67



After reports of widespread problems in psychology and biomedicine, scientists have become increasingly anxious that many published studies do not stand up.

http://www.slate.com/blogs/the_slatest/2016/03/03/psychology_study_that_induced_the_reproducibility_crisis_was_wrong.html

<http://www.bbc.com/news/science-environment-39054778>

http://www.slate.com/articles/technology/future_tense/2016/04/the_reproducibility_crisis_is_good_for_science.html

The Reproducible Data Analysis Problem Can Be Addressed By Software Dev/Data Science Best Practices

Some examples include...

Tidy Data

- Organizing (raw) data into a well structured form ready for data analysis
- Tidy data facilitates straight-forward data analysis and sharing with others
- The steps performed to tidy data need to be well documented

Data Analysis Pipelines

- All analysis steps should be captured in a set of scripts or programs
 - Analysis steps, input, processes should be well documented
 - Anyone should be able to reproduce the complete analysis path

Literate Programming

- A programming paradigm where code and documentation live in the same “document”, human-centered
 - A natural fit for scientific reporting
 - Analysis code lives side-by-side with the results

Version Control Systems

- Central repositories for storing and tracking code changes
- Helps manage code, avoid incorrect versions and conflicts
- Facilitates open sharing of code (e.g. GitHub)

A Data Analysis Might Not Be Reproducible If...

- you had to copy and paste at some point
- you had to manually edit the data, in any way
- you had to touch a mouse to perform any part of the analysis
- an outside person can't reproduce the analysis

Some additional complex questions to consider:

- Are your platform, operating system, software versions reproducible?
- Are your inputs (e.g. data and parameters) reproducible?
- Could “future you” reproduce your own analysis?

A Few Thoughts

- Reproducibility does not imply correctness/accuracy
 - you can still do the wrong thing in a reproducible way!
 - ...but knowing EXACTLY what went wrong is an important first step to correcting the mistake
 - sharing your code with others can help you find problems you didn't even know about
- There's a continuum of complexity for doing reproducible data analysis, lots of factors to consider
- At the very least:
strive to capture all analysis steps in as set of programs/scripts!

Resources

Learning R

- Learning R and Data Science Methods
 - Coursera online classes
e.g. JHU Data Science Series
<https://www.coursera.org/specializations/jhu-data-science>
 - RStudio's curated learning resource lists
<https://www.rstudio.com/online-learning/>
- Books
 - R for Data Science: <http://r4ds.had.co.nz>
 - ggplot2: Elegant Graphics for Data Analysis, 2nd Ed.
 - Advanced R: <http://adv-r.had.co.nz>
- Practice
 - UCI Machine Learning Repo: <http://archive.ics.uci.edu/ml/>
 - Kaggle Datasets & Competitions: <https://www.kaggle.com>
 - Build a small tool that make one part of your job easier

Additional Materials

- Books
 - *Applied Predictive Modeling* (Kuhn, Johnson)
 - *The Grammar of Graphics* (Wilkinson)
- Websites
 - R Bloggers (<https://www.r-bloggers.com>)
 - Kaggle (<http://www.kaggle.com>)
 - FlowingData (<http://flowingdata.com>)
- Articles
 - Tidy Data (<http://vita.had.co.nz/papers/tidy-data.html>)
 - A Layered Grammar of Graphics
(<http://vita.had.co.nz/papers/layered-grammar.html>)
- Visualization Frameworks
 - d3.js (<https://d3js.org>)
 - Processing, p5.js (<https://processing.org>)