



From Basics to Brilliance:

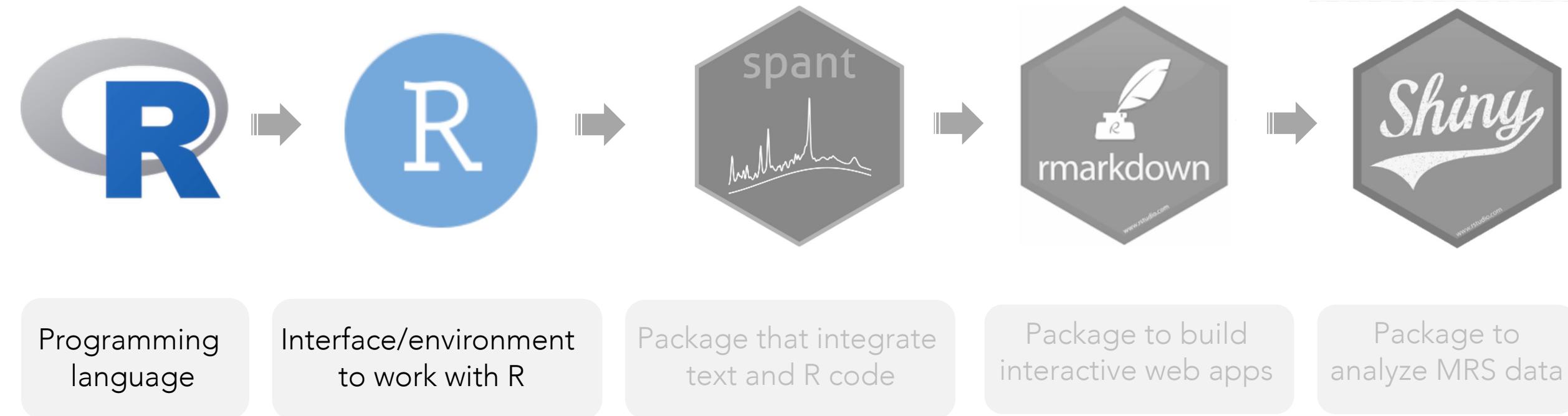
An Introduction to R-Studio for MR Spectroscopy analysis, visualization, and beyond

MRS Hackathon Toronto 2023

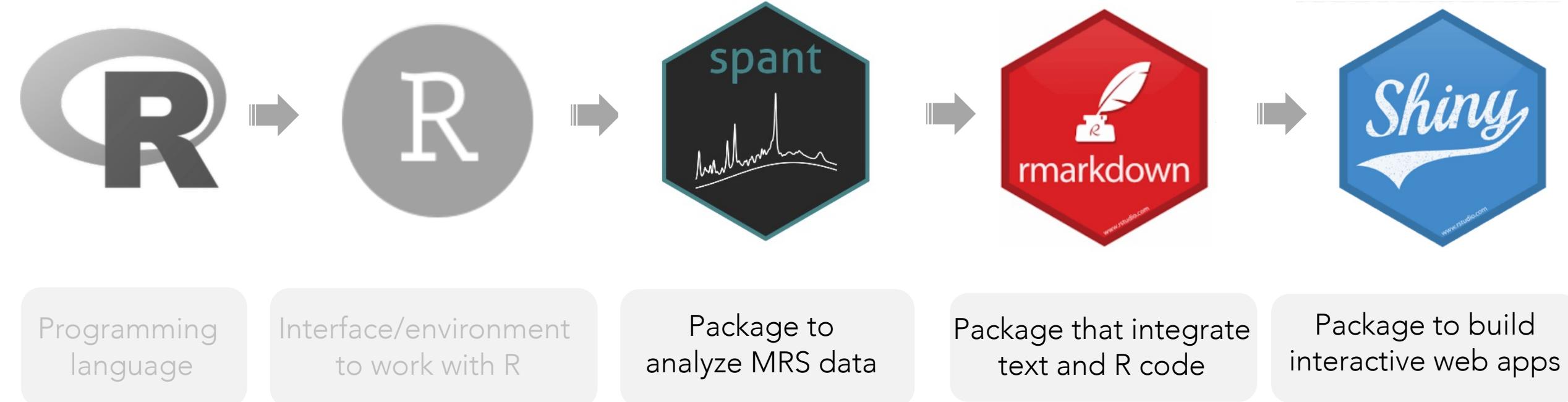
Jessica Archibald, PhD.

 @ArchibaldJes

Outline



Outline



Why R?



- Signal processing
- Numerical computing
- Simulations and modeling



- General-purpose programming
- Data analysis and ML
- Web development

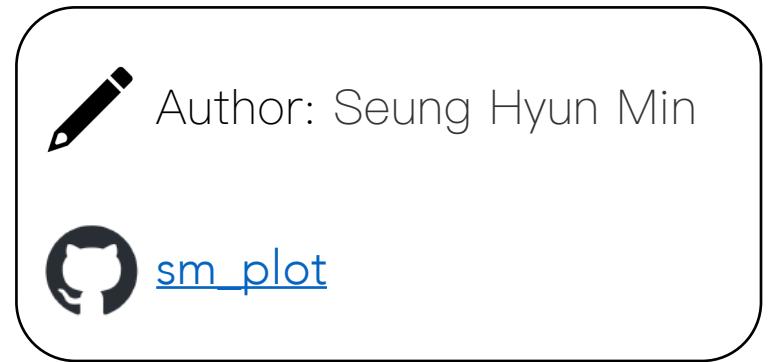
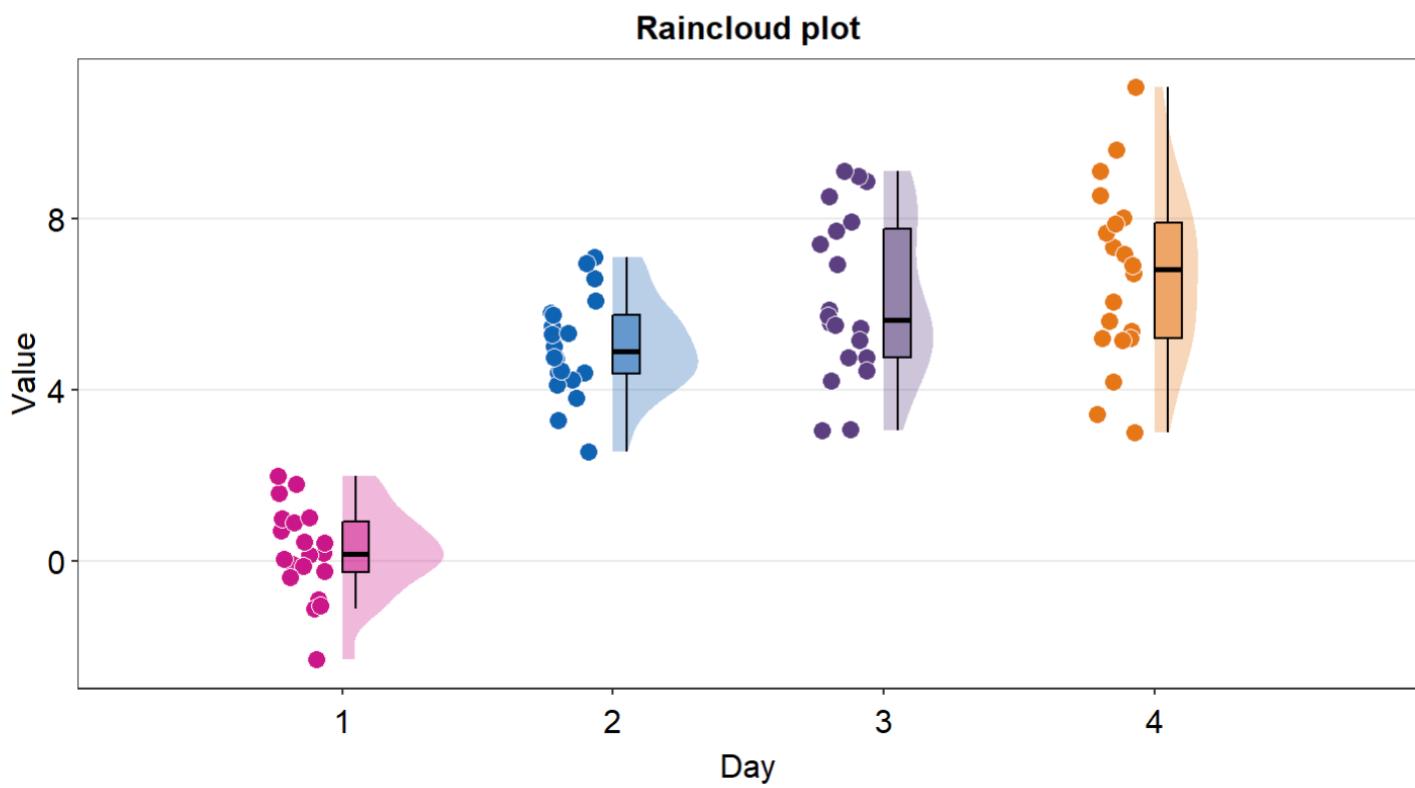


- Statistical Analysis
- Data visualization
- Data science and ML
- Reproducible research

Why R?

- Statistical and data analysis capabilities: collection of packages and libraries
- Data Visualization: high-quality customizable visualizations
- Community–driven approach
- Reproducibility and documentations → **R***Pub*s by RStudio
- Free!! & open source

User's creations



How knowing R has improved my research

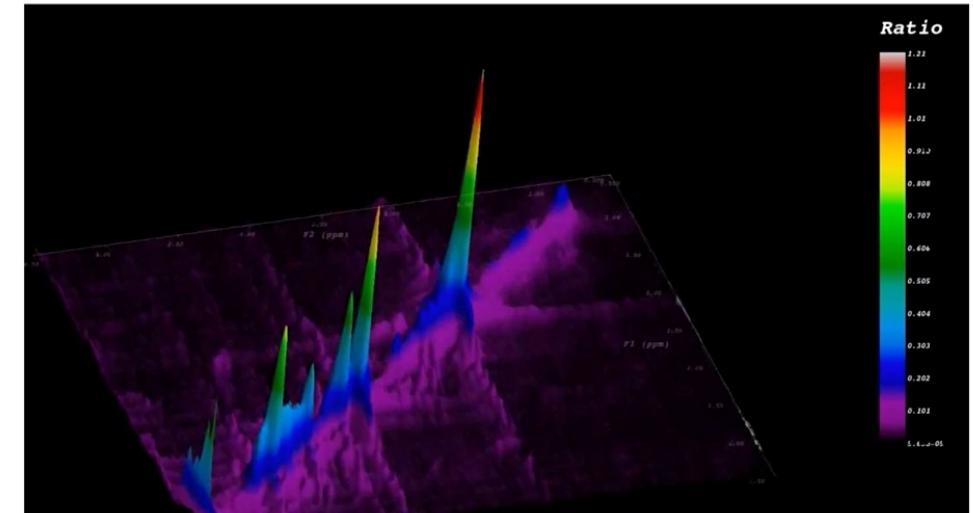
-  knowledge =  stress
- Efficacy
 1. Analysis
 2. Presentation
 3. Code access
- Collaboration

Two-Dimensional J-Resolved 1H-MRS using PRESS and Prior Knowledge Fitting (ProFit) in the Posterior Insula

A practical guide for balancing time and quality of metabolic information

JessArchibald

15/12/2021

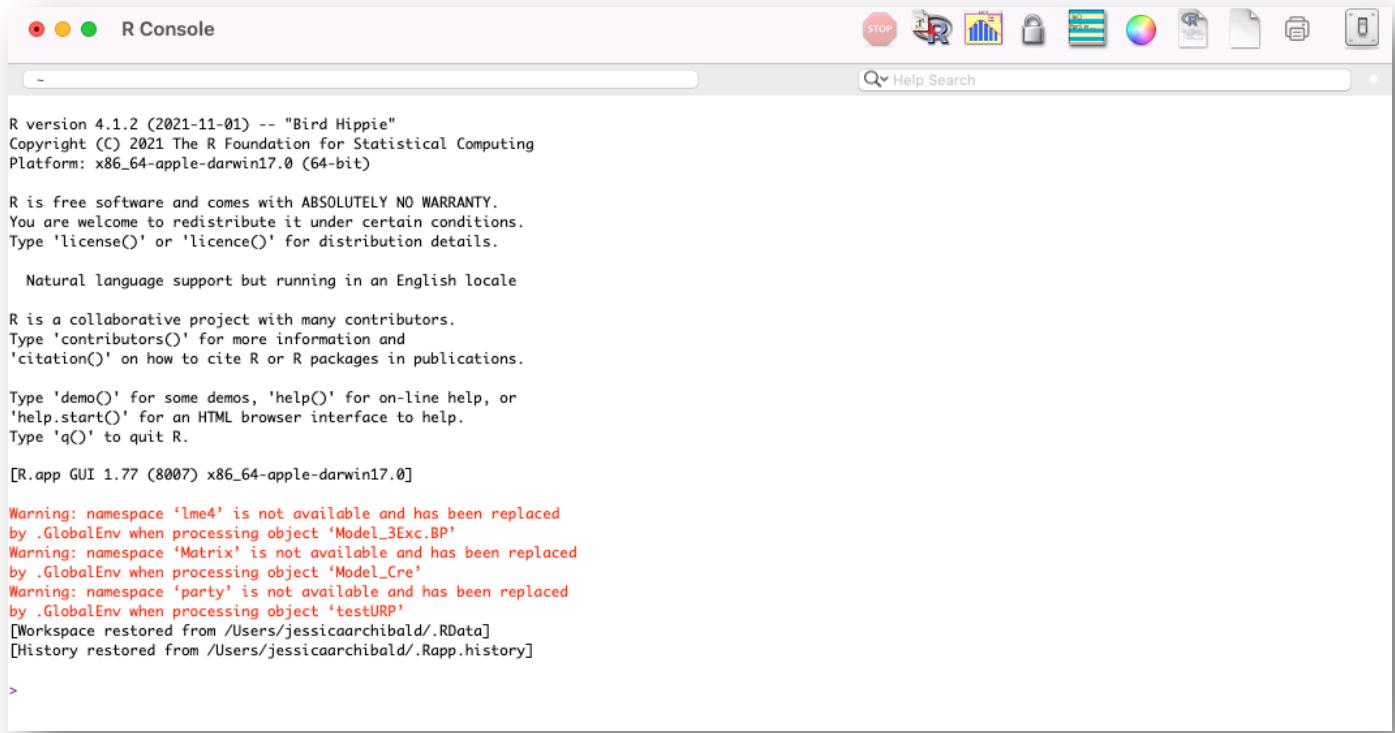


Question 1 - How much is gained when 2 blocks are combined ?

Subjects: 24.

R

- Programming language for statistical computing.
- Comprehensive toolset for exploring, visualizing, and modeling data.



```
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.

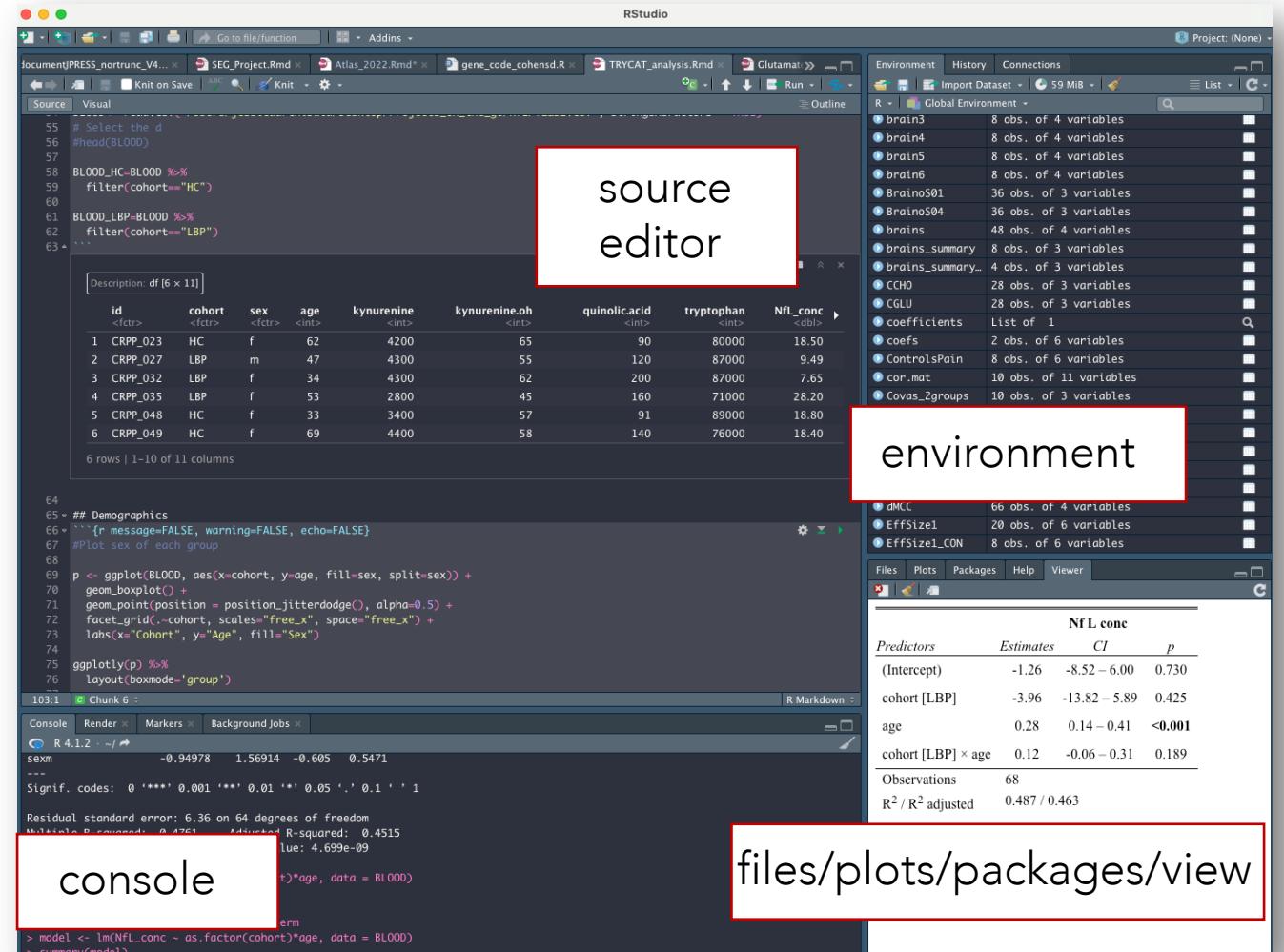
[ R.app GUI 1.77 (8007) x86_64-apple-darwin17.0]

Warning: namespace 'lme4' is not available and has been replaced
by .GlobalEnv when processing object 'Model_3Exc.BP'
Warning: namespace 'Matrix' is not available and has been replaced
by .GlobalEnv when processing object 'Model_Cre'
Warning: namespace 'party' is not available and has been replaced
by .GlobalEnv when processing object 'testURP'
[Workspace restored from /Users/jessicarchibald/.RData]
[History restored from /Users/jessicarchibald/.Rapp.history]

>
```

R Studio

- Integrated development environment
- User-friendly
- Enhance programming experience



Tutorial 1

The screenshot shows the RStudio interface. The top bar displays tabs for 'TRYCAT_analysis.Rmd', 'Glutamate-RNA_microarray-analy...', 'gene_code_cohensd.R', 'temp.R*', and 'Untitled1*'. The main area contains the following R code:

```
1 # Tutorial # 1 MRS-Hackathon
2 # Jessica Archibald Toronto 2023
3
4 # Basic functionalities:
5 # 1) How to install a package
6 # 2) How to load in data
7 # 3) Basic plot
8 # 4) Run a lmer
9
10
11
12
```

The Global Environment pane on the right lists various objects:

- Model1_N... Formal class lmer...
- Model2 Formal class lmer...
- Model3 Formal class lmer...
- Model4 Formal class lmer...
- Modelb List of 13
- Modellm List of 13
- mrs_data List of 9
- mrs_proc List of 9
- mtcars 32 obs. of 11 vari...
- N2P2_1 40 obs. of 9 varia...
- N2P2_2 40 obs. of 8 varia...
- N2P2_3 40 obs. of 8 varia...
- N2P2_4 40 obs. of 8 varia...
- NRS 9 obs. of 2 variab...
- one 20 obs. of 8 varia...
- p List of 9

The bottom panel shows the R Console with the message 'R 4.1.2 · ~/`'.



Data types & functions

- Character
- Factor
- Integer
- Numeric

```
> str(BLOOD)
'data.frame':   68 obs. of  11 variables:
 $ id           : Factor w/ 68 levels "CRPP_023","CRPP_0...
 $ cohort       : Factor w/ 2 levels "HC","LBP": 1 2 2 2 ...
 $ sex          : Factor w/ 2 levels "f","m": 1 2 1 1 1 ...
 $ age          : int  62 47 34 53 33 69 26 34 71 78 ...
 $ kynurenine   : int  4200 4300 4300 2800 3400 4400 270...
 $ kynurenine.oh: int  65 55 62 45 57 58 44 52 38 41 ...
 $ quinolic.acid: int  90 120 200 160 91 140 120 120 130...
 $ tryptophan   : int  80000 87000 87000 71000 89000 760...
 $ NFL_conc     : num  18.5 9.49 7.65 28.2 18.8 18.4 6.2...
 $ NFL_CV       : num  6.89 5.45 5.77 3.56 3.21 ...
```

as.factor()

as.character()

as.integer()

as.numeric()

Manipulate the data



rows

- Filter()
- Slice()
- Arrange()

columns

- Select()
- Rename()
- Mutate()
- Relocate()



pivoting

- Pivot_longer
- Pivot_wider

Tutorial 2

The screenshot shows the RStudio interface. The top menu bar includes 'File', 'Edit', 'Source', 'Plot', 'Console', 'File Watcher', 'Project', 'Help', and 'Addins'. The top toolbar has icons for file operations like 'New File', 'Open', 'Save', and 'Run'. The main window has tabs for 'T_analysis.Rmd', 'Glutamate-RNA_microarray-analys...', 'gene_code_cohensd.R', 'temp.R*', and 'Untitled1*'. The left pane shows the code editor with the following R script:

```
1 # Tutorial # 2 MRS-Hackathon
2 # Jessica Archibald Toronto 2023
3
4 # Basic functionalities:
5 # 1) Manipulate the data
6 # 2) Piping
7 # 3) Plotting with manipulations & piping
8
9
10
11
```

The right pane shows the 'Global Environment' list with the following objects:

- Modelb List of 13
- Modellm List of 13
- mrs_data List of 9
- mrs_proc List of 9
- mtcars 32 obs. of 11 variables
- N2P2_1 40 obs. of 9 variables
- N2P2_2 40 obs. of 8 variables
- N2P2_3 40 obs. of 8 variables
- N2P2_4 40 obs. of 8 variables
- NRS 9 obs. of 2 variables
- one 20 obs. of 8 variables
- p List of 9
- p2 List of 9

The bottom pane shows the 'Console' tab with the following output:

```
9:1 (Top Level) : R Script
Console Markers x Background Jobs x
R 4.1.2 ~/ ↵
>
```

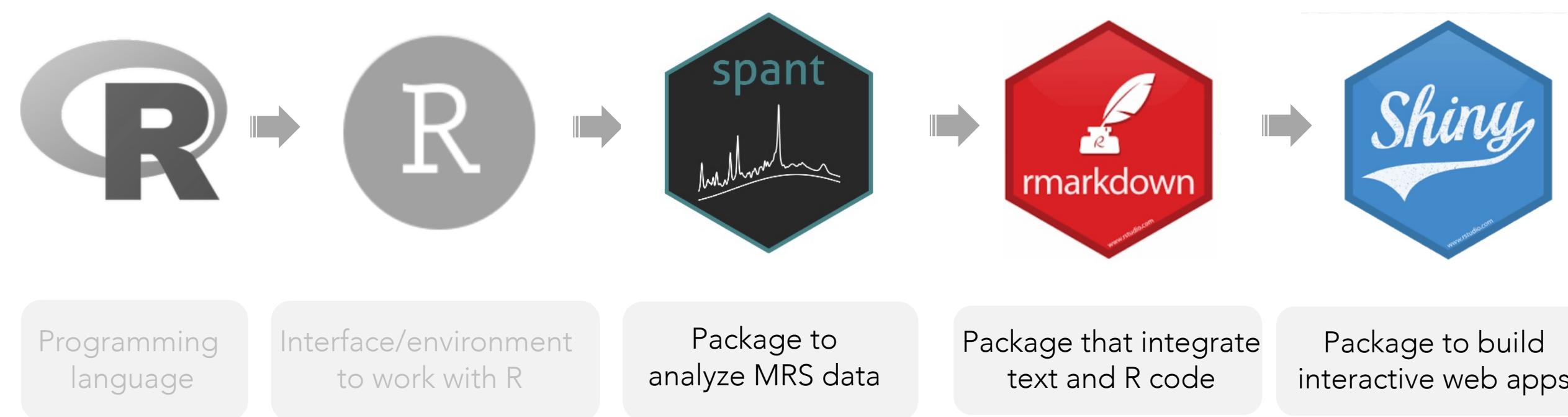


Short-cuts

- Control + L  Clear the console
- command + enter  Run code
- command + left/right  Cursor to the start or end of code

Short-cuts

- shift +  + c  Comment/uncomment code
- shift +  + p  Re-run last chunk of code
-  + d  Deletes a line of code



Spant- (SPectroscopy ANalysis Tools)



The Journal of Open Source Software

spant: An R package for magnetic resonance spectroscopy analysis

Martin Wilson¹

¹ Centre for Human Brain Health and School of Psychology, University of Birmingham, Birmingham, UK

DOI: [10.21105/joss.03646](https://doi.org/10.21105/joss.03646)

Software

- [Review](#)
- [Repository](#)
- [Archive](#)

Editor: Chris Hartgerink [✉](#)

Reviewers:

- @joaomcteixeira
- @chartgerink

Submitted: 09 August 2021
Published: 02 November 2021

License
Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

Magnetic Resonance Spectroscopy (MRS) allows the measurement of small molecules (metabolites) in the body without the use of harmful radiation. Based on the same basic principles and technology behind Magnetic Resonance Imaging (MRI), most modern MRI scanners are also capable of acquiring MRS — making the technique highly suited to a number of clinical applications ([Oz et al. \(2014\)](#)). Despite the success of MRS in the research environment, clinical translation has proven slow due to a number of technical and practical reasons, with challenges associated with reliable data processing and analysis having particular importance ([Wilson et al. \(2019\)](#)). The spant (SPectroscopy ANalysis Tools) package has been developed to: (1) provide open-source implementations of traditional and modern MRS processing and analysis techniques for routine analysis ([Near et al. \(2021\)](#)) and (2) aid the development, validation and comparison of new algorithms and analysis pipelines.

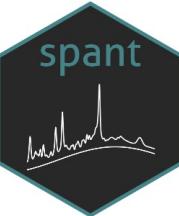
Statement of need

Traditional MRS analysis was dominated by the use of proprietary software, either supplied by scanner manufacturers or offline tools such as LCModel ([Provencher \(1993\)](#)) and jMRUI ([Naressi et al. \(2001\)](#)). In more recent years there has been a steadily increasing trend toward the use of open-source methods — with some early examples including TARQUIN ([Reynolds et al. \(2006\); Wilson et al. \(2011\)](#)) and AQSES ([Poulet et al. \(2007\)](#)). This trend is set to continue with the recent transition of LCModel to an open-source license, and an acceleration in the development of new open-source methods and packages such as Vespa ([Soher et al. \(2011\)](#)), Gannet ([Edden et al. \(2014\)](#)), FID-A ([Simpson et al. \(2017\)](#)), Osprey ([Oeltzschner et al. \(2020\)](#)), suspect ([Rowland \(2021\)](#)) and FSL-MRS ([Clarke et al. \(2021\)](#)). The availability

- Provide open-source implementations of traditional and modern MRS processing and analysis.



Spant- (SPectroscopy ANalysis Tools)



- Aid the development, validation and comparison of new algorithms and analysis pipelines.
- Particularly suited to the **interactive exploration** and **batch processing** of large and complex datasets.

R

Tutorial 3

The screenshot shows the RStudio interface. The top bar displays tabs for 'T_analysis.Rmd', 'Glutamate-RNA_microarray-analysis.R', 'gene_code_cohensd.R', 'temp.R*', and 'Untitled1*'. The bottom bar shows tabs for 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. The left pane contains the code editor with the following R script:

```
1 # Tutorial # 3 MRS-Hackathon --spant
2 # Jessica Archibald Toronto 2023
3
4 # Basic functionalities:
5 # 1) Reading and loading MRS data
6 # 2) Visualization of the data
7 # 3) Pre-process
8 # 4) Fit
9
10
11
12
13
```

The right pane is the Environment browser, listing global variables:

Variable	Type	Value
mrs_data	List of 9	
mrs_proc	List of 9	
mtcars	32 obs. of 11 variables	
N2P2_1	40 obs. of 9 variables	
N2P2_2	40 obs. of 8 variables	
N2P2_3	40 obs. of 8 variables	
N2P2_4	40 obs. of 8 variables	
NRS	9 obs. of 2 variables	
one	20 obs. of 8 variables	
p	List of 9	
p2	List of 9	
pain	10 obs. of 3 variables	
PCC	69 obs. of 4 variables	

The bottom pane is the Console, showing the following output:

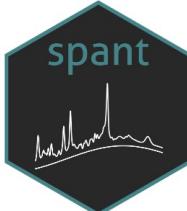
```
12:1 (Top Level) > R Script >
Console Markers Background Jobs >
R 4.1.2 ~/>
$resolution
[1] NA 2e+01 2e+01 2e+01 1e+00 NA 5e-04

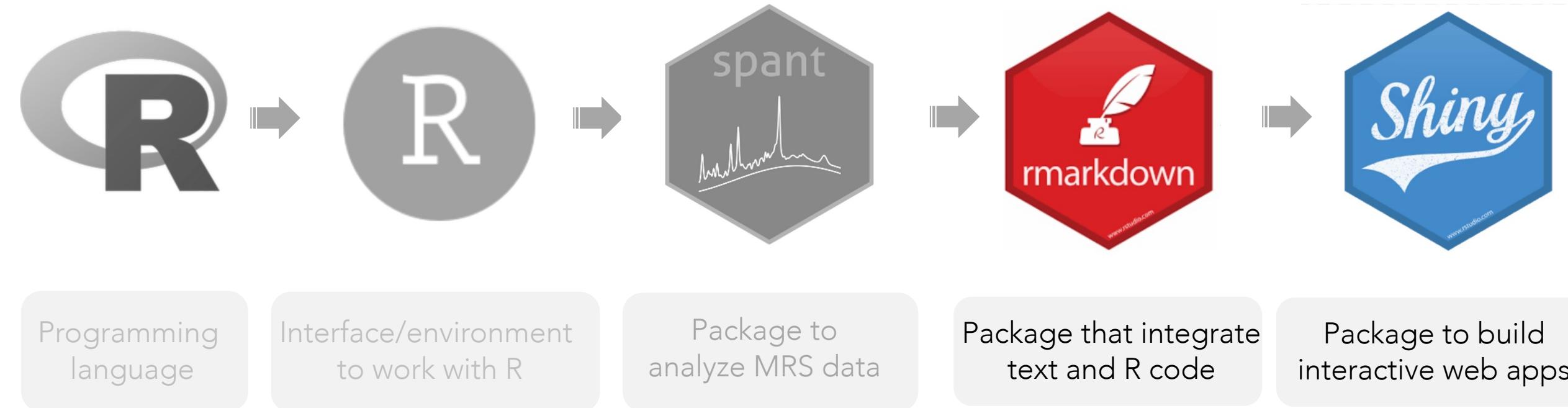
$ref
[1] 4.65

$nuc
[1] "1H"

$freq_domain
[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

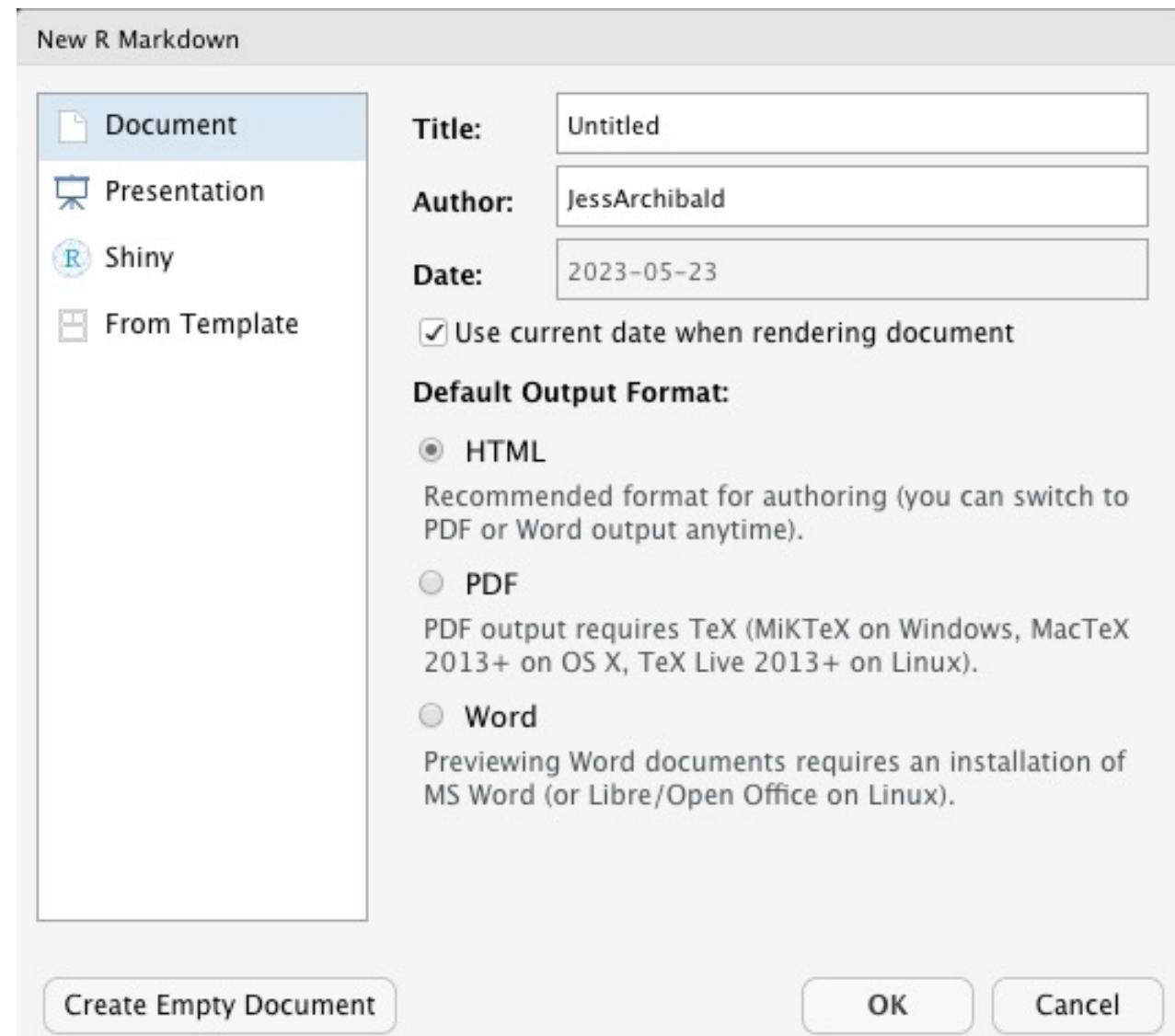
spant





R Markdown

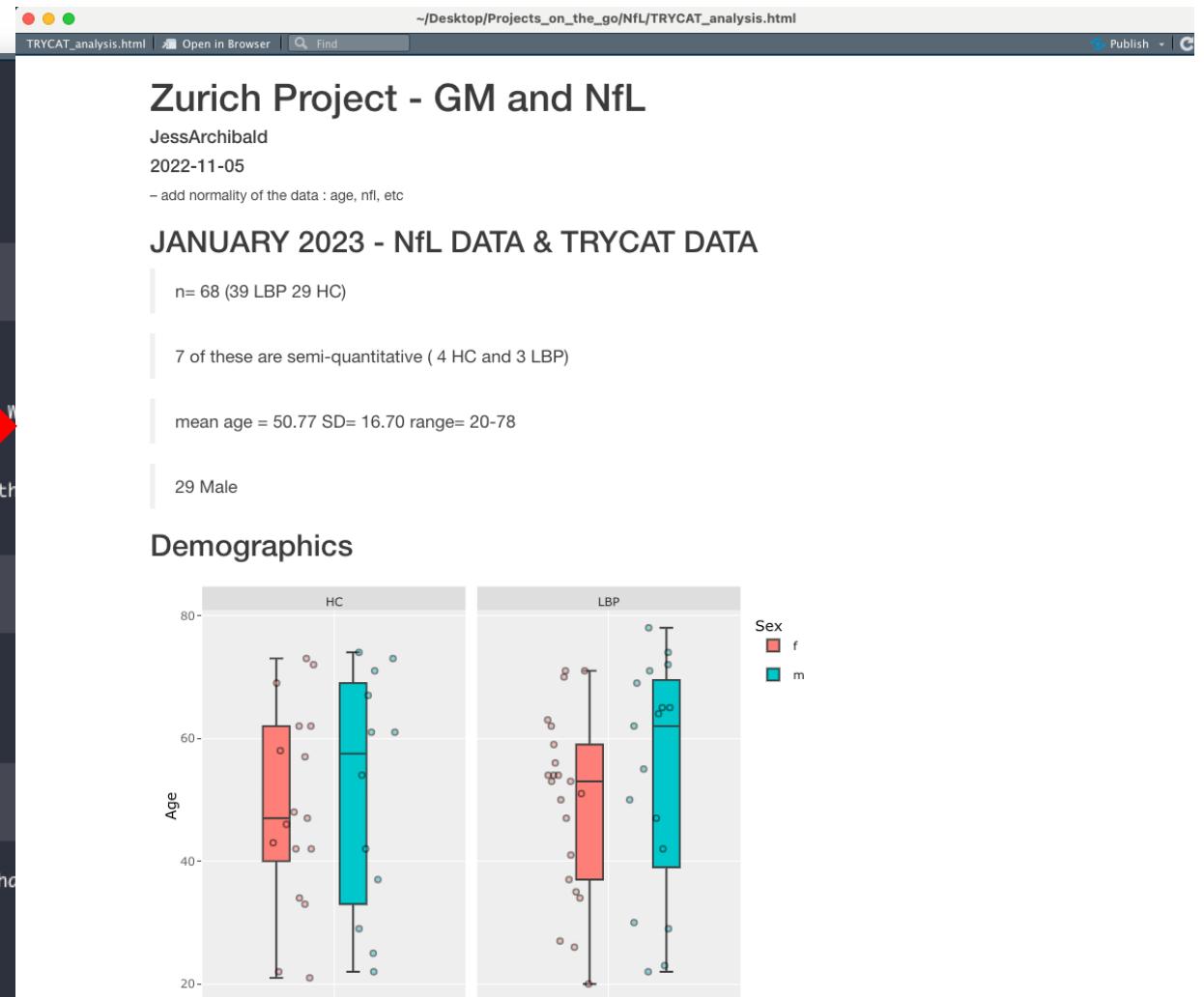
- .Rmd file
- Output formats





R Markdown

```
1  ---
2  title: "Untitled"
3  author: "JessArchibald"
4  date: ``r Sys.Date()``
5  output: html_document
6  ---
7
8  ```{r setup, include=FALSE}
9  knitr::opts_chunk$set(echo = TRUE)
10 ---
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authors to add style to their documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
15
16 When you click the **Knit** button a document will be generated that includes both content as well as the embedded R code chunks within the document. You can embed an R code chunk like this:
17
18 ```{r cars}
19 summary(cars)
20 ---
21
22 ## Including Plots
23
24 You can also embed plots, for example:
25
26 ```{r pressure, echo=FALSE}
27 plot(pressure)
28 ---
29
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that
31
```

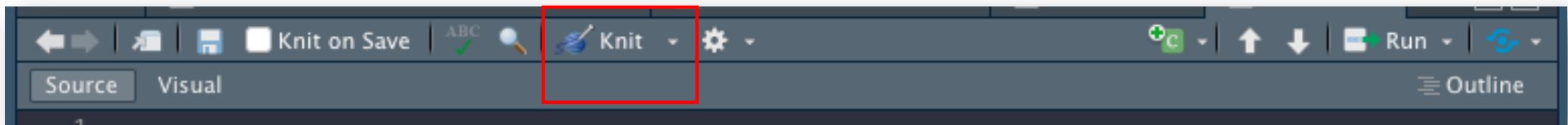




R Markdown

```
```{r cars}
summary(cars)
```

      speed          dist
Min.   : 4.0   Min.   : 2.00
1st Qu.:12.0  1st Qu.: 26.00
Median :15.0  Median : 36.00
Mean   :15.4  Mean   : 42.98
3rd Qu.:19.0  3rd Qu.: 56.00
Max.   :25.0  Max.   :120.00
```





R Markdown

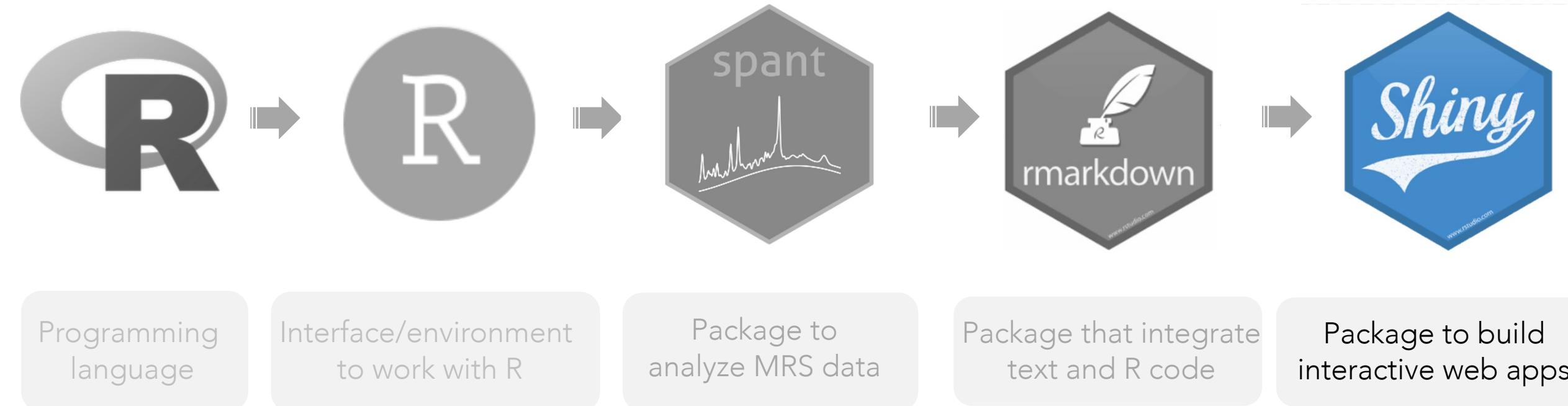
```
1 ---  
2 title: "Zurich Project - GM and NFL"  
3 author: "JessArchibald"  
4 date: '2022-11-05'  
5 output: html_document  
6 ---  
7 ```{r setup, include=FALSE}  
8 knitr::opts_chunk$set(warning = FALSE, message = FALSE)  
9 knitr::opts_knit$set(root.dir = "/Users/jessicaarchibald/Desktop/Projects_on_the_go/NFL/")  
10 ...  
11 ...  
12 ...  
13 ...  
14 library(lme4)  
15 library(lmerTest)  
16 library(effsize)  
17 library(ggpubr)  
18 library(gginference)  
19 library(tidyverse)  
20 library(rstatix)  
21 library(ggplot2)
```

```
## JANUARY 2023 - NFL DATA & TRYCAT DATA  
  
> n= 68 (39 LBP 29 HC)|  
  
> 7 of these are semi-quantitative ( 4 HC and 3 LBP)  
  
> mean age = 50.77 SD= 16.70 range= 20-78  
  
> *29 Male*  
  
- item 1  
- item 2
```

- 1 Set working directory

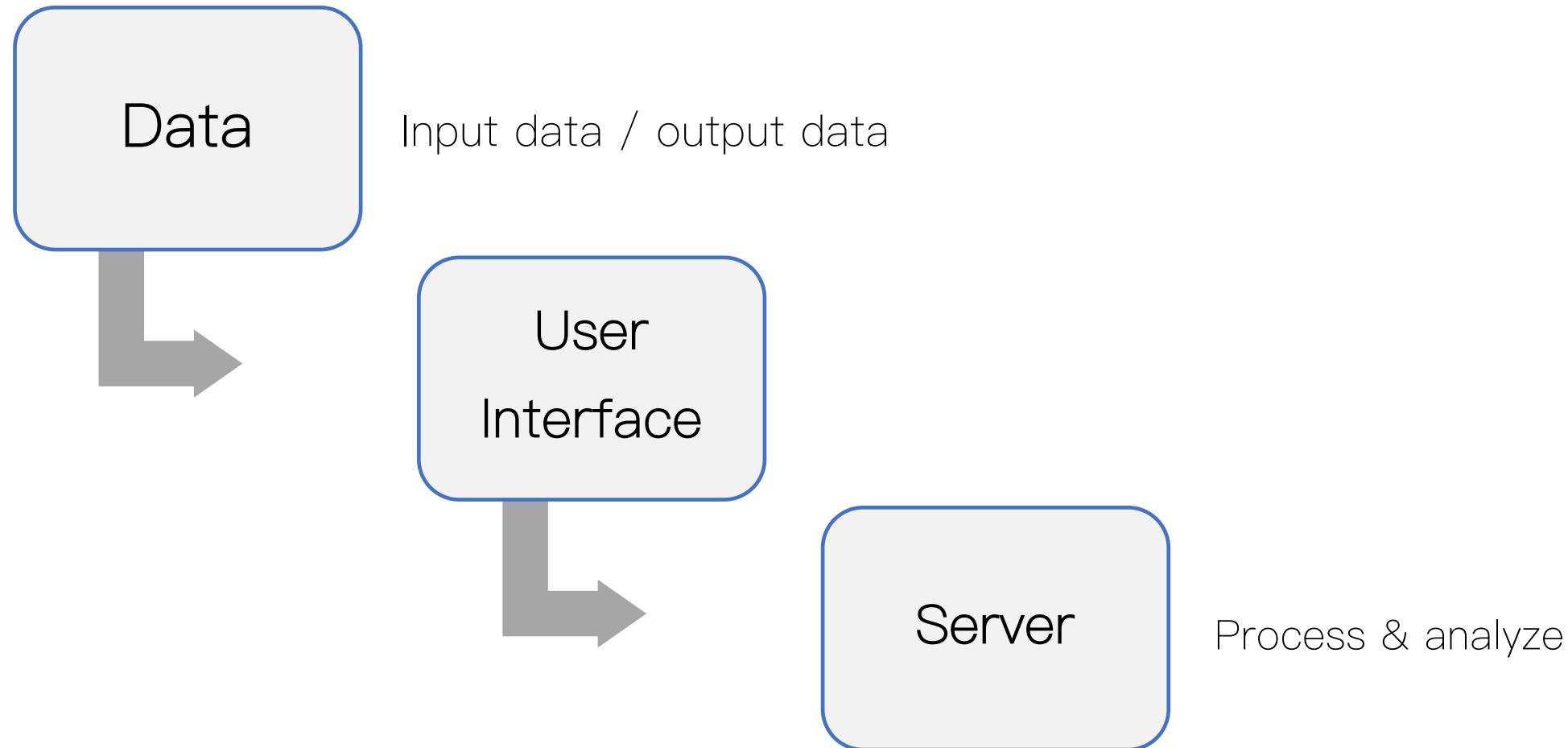
- 2 Load my libraries

- 3 Lightweight markup language



Shiny – package for web apps

Workflow





Shiny – building the app

New file

New Shiny Web Application



Application name:

Application type: Single File (app.R) Multiple File (ui.R/server.R)

Create within directory: [Browse...](#)

② Shiny Web Applications [Create](#) [Cancel](#)

R HTML R Documentation... When you click the **Knit** button a document will be generated tha

kdown document. Markdown is a simple formatting syn
R Markdown see <<http://rmarkdown.rstudio.com>>.



Shiny – building the app

A screenshot of the RStudio interface showing R code for a Shiny application. The code is divided into three main sections, each highlighted by a red box:

- Section 1 (Top Red Box):** Contains the UI definition, starting with `ui <- fluidPage(` and ending with `shinyApp(ui = ui, server = server)`.
- Section 2 (Bottom Red Box):** Contains the server logic, starting with `server <- function(input, output) {` and ending with `}`.
- Section 3 (Middle Red Box):** Contains the main panel definition, starting with `mainPanel(` and ending with `)`.

The RStudio interface shows the code in a script editor with syntax highlighting. A toolbar at the top includes a "Run App" button, which is highlighted with a red box.

```
1 #  
2 # This is a Shiny web application. You can run the application by clicking  
3 # the 'Run App' button above.  
4 #  
5 # Find out more about building applications with Shiny here:  
6 #  
7 # http://shiny.rstudio.com/  
8 #  
9  
10 library(shiny)  
11  
12 # Define UI for application that draws a histogram  
13 ui <- fluidPage(  
14  
15   # Application title  
16   titlePanel("Old Faithful Geyser Data"),  
17  
18   # Sidebar with a slider input for number of bins  
19   sidebarLayout(  
20     sidebarPanel(  
21       sliderInput("bins",  
22         "Number of bins:",  
23         min = 1,  
24         max = 50,  
25         value = 30)  
26     ),  
27  
28     # Show a plot of the generated distribution  
29     mainPanel(  
30       plotOutput("distPlot")  
31     )  
32   )  
33 )  
34  
35 # Define server logic required to draw a histogram  
36 server <- function(input, output) {  
37  
38   output$distPlot <- renderPlot({  
39     # generate bins based on input$bins from ui.R  
40     x <- faithful[, 2]  
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)  
42  
43     # draw the histogram with the specified number of bins  
44     hist(x, breaks = bins, col = 'darkgray', border = 'white',  
45           xlab = 'Waiting time to next eruption (in mins)',  
46           main = 'Histogram of waiting times')  
47   })  
48 }  
49  
50 # Run the application  
51 shinyApp(ui = ui, server = server)  
52
```

1 Run the app

2 User Interface

3 Server

Shiny – building the app

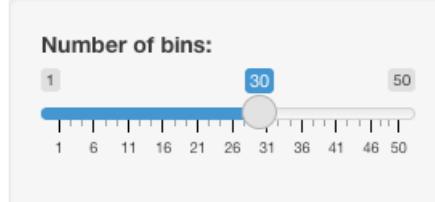
```

1 #
2 # This is a Shiny web application. You can run the application by clicking
3 # the 'Run App' button above.
4 #
5 # Find out more about building applications with Shiny here:
6 #
7 #   http://shiny.rstudio.com/
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22         "Number of bins:",
23         min = 1,
24         max = 50,
25         value = 30)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x     <- faithful[, 2]
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)
42
43     # draw the histogram with the specified number of bins
44     hist(x, breaks = bins, col = 'darkgray', border = 'white',
45           xlab = 'Waiting time to next eruption (in mins)',
46           main = 'Histogram of waiting times')
47   })
48 }
49
50 # Run the application
51 shinyApp(ui = ui, server = server)
52

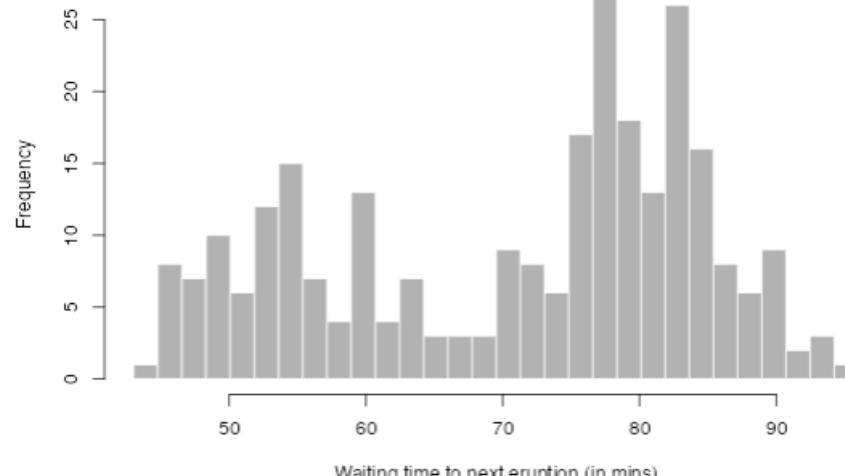
```

Old Faithful Geyser Data

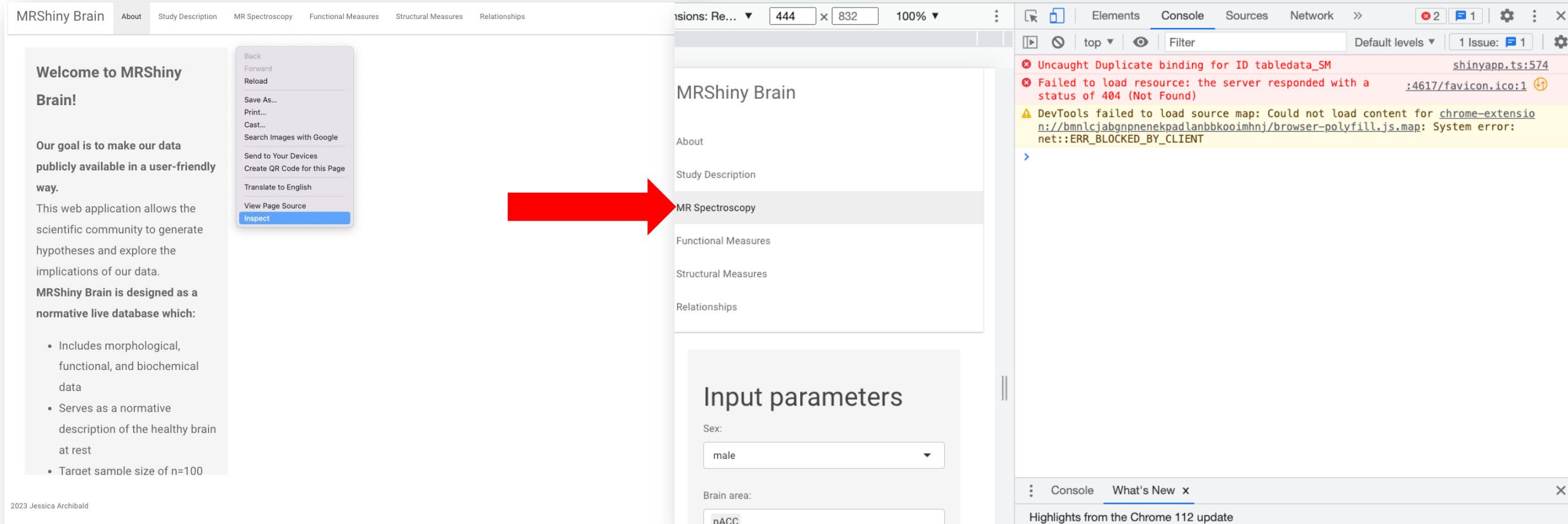
Number of bins:



Histogram of waiting times



Shiny – debugging tip



The screenshot shows a Shiny web application titled "MRShiny Brain". The application has a sidebar with links like "About", "Study Description", "MR Spectroscopy", "Functional Measures", "Structural Measures", and "Relationships". A red arrow points from the sidebar to the "Inspect" button in the browser's context menu, which is open. The main content area displays "Input parameters" with dropdown menus for "Sex" (set to "male") and "Brain area" (set to "pACC"). On the right side of the browser window, the developer tools are open, specifically the "Console" tab. It shows several error messages:

- Uncaught Duplicate binding for ID tabledata_SM
- Failed to load resource: the server responded with a status of 404 (Not Found)
- DevTools failed to load source map: Could not load content for chrome-extension://bmmlcjabgnpnenekpadlanbbkoimhnj/browser-polyfill.js.map: System error: net::ERR_BLOCKED_BY_CLIENT

At the bottom of the developer tools, there are tabs for "Console" and "What's New", with "Console" being the active tab.



Shiny—deploy

SHINY SERVER

Get your Shiny apps online

When you're ready, Posit Connect is a publishing platform for all the work your teams create in R. Share Shiny applications, R Markdown reports, interactive Python applications, dashboards, plots, APIs, and more in one convenient place.

If you prefer for us to host your Shiny applications, one of our shinyapps.io plans is sure to work for you.

[DOWNLOAD SHINY SERVER](#) [SHINYAPPS.IO CLOUD HOSTING](#)



Shiny Server

POSIT CONNECT

For data products worth sharing

Deploy everything you create in R & Python, including interactive applications (Shiny, Streamlit, Dash), documents, notebooks, and dashboards. Automate code execution so your data products are always up to date. Give stakeholders and collaborators the right access to the content they need.

[SCHEDULE A DEMO](#) [WATCH THE VIDEO](#)



Posit Connect

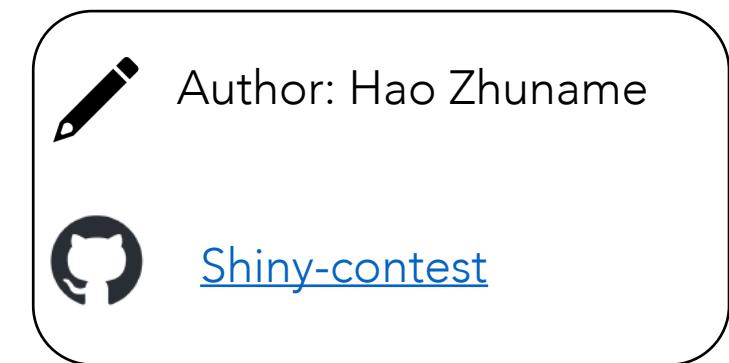
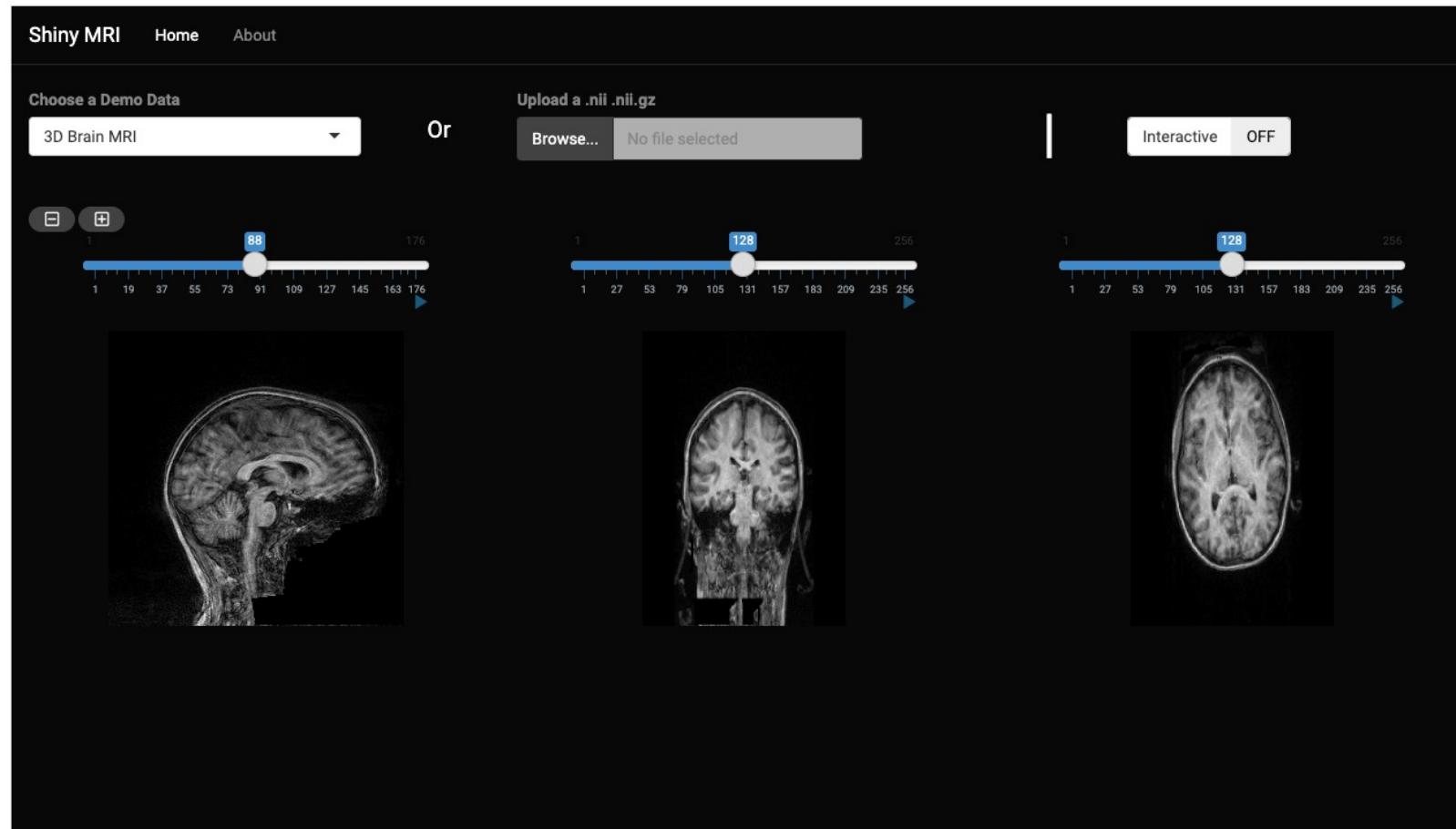
Share your Shiny Applications Online

Deploy your Shiny applications on the Web in minutes

[Sign Up](#)

Shinyapps.io

Illustrative examples



Illustrative examples

The screenshot shows the homepage of the Neurosurveillance website. The header features the Neurosurveillance logo with a stylized brain icon. The main content area includes a welcome message, four key statistics in blue boxes (5000+ Patients, 15 Countries, 20 Years, 50+ Researchers), and a section about benchmarking spontaneous functional and neurological recovery following spinal cord injury. A sidebar on the left contains links for About, User guide, EMSCI, Sygen Trial, Study details, Epidemiological features, Neurological features, Functional features, Data sources compared, Abbreviations, and Contact for collaborations.

Welcome to Neurosurveillance

5000+ Patients

15 Countries

20 Years

50+ Researchers

Benchmarking the spontaneous functional and neurological recovery following spinal cord injury

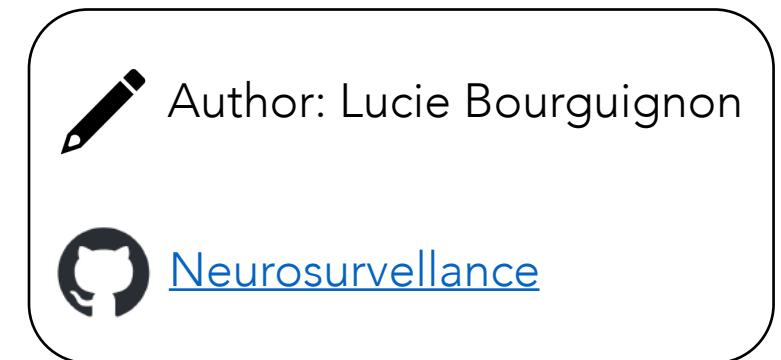
Traumatic spinal cord injury is a rare but devastating neurological disorder. It constitutes a major public health issue, burdening both patients, caregivers, as well as society as a whole. The goal of this project is to establish an international benchmark for neurological and functional recovery after spinal cord injury. Currently, Neurosurveillance leverages three decades of data from two of the largest data sources in the field facilitating the analysis of temporal trends in epidemiological landscape, providing reference values for future clinical trials and studies, and enabling monitoring of patients on a personalized level.

More information can be found here: [?](#)

What You Can Do Here:

This applet has enables visitors to directly interact with the data collected in the EMSCI study and the Sygen clinical trial. Both [EMSCI](#) and [Sygen](#) Tabs are organized as follows:

- The Study details Tab provides information the data sources, ([EMSCI study](#) and [Sygen clinical trial](#)).
- The Epidemiological feature Tabs ([EMSCI study](#) and [Sygen clinical trial](#)), Neurological features Tabs ([EMSCI study](#) and [Sygen clinical trial](#)) and Functional features Tabs ([EMSCI study](#) and [Sygen clinical trial](#)) offer an interactive interface to explore the epidemiological characteristics, as well as neurological and functional recovery after spinal cord injury,



Illustrative examples

MRShiny Brain

About

Study Team

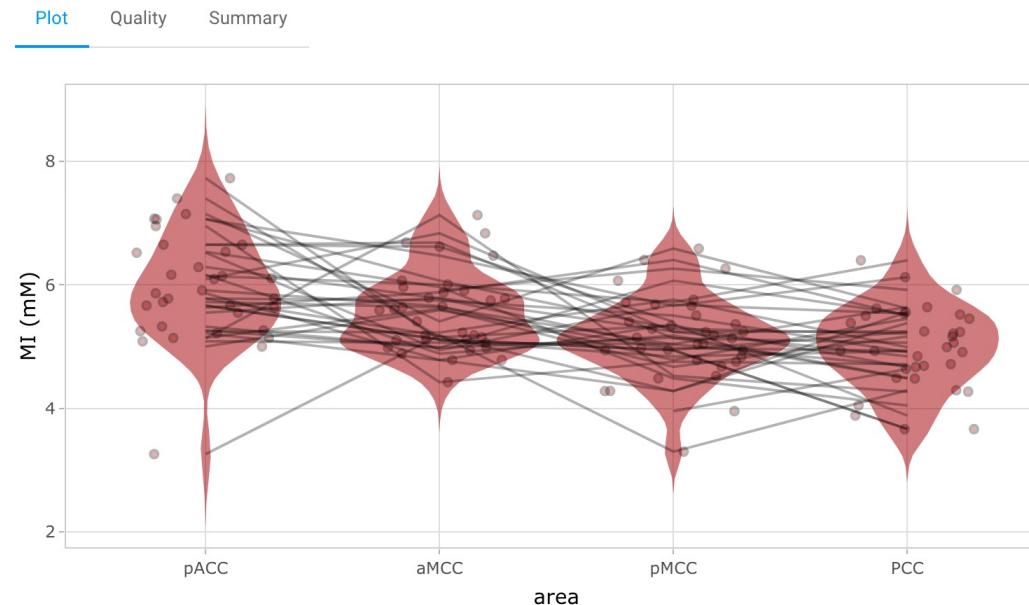
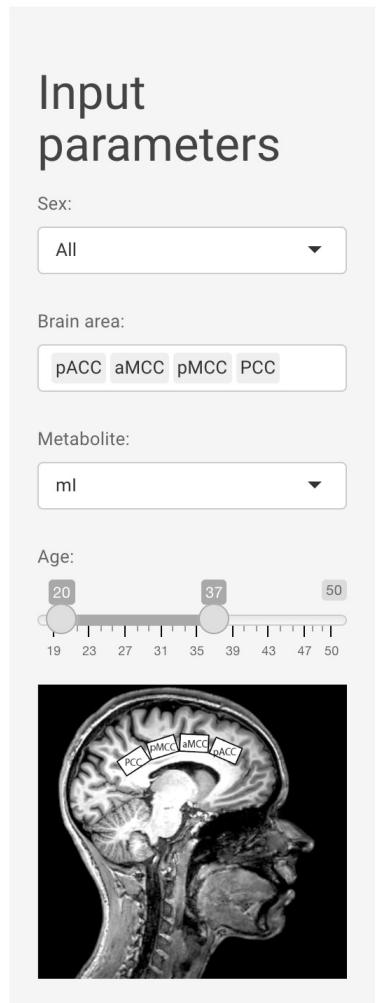
Study Description

MR Spectroscopy

Functional Measures

Structural Measures

Relationships



Author: Jessica Archibald



[MRShiny Brain](#)

Pros & Cons

Take home message

Why you should use R studio, R Markdown, Spant, and Shiny

- Enhance the efficiency, reproducibility, and interactivity of your data analysis and presentation workflow.
- Enhance quality, transparency, and accessibility.
- By using spant, you can seamlessly integrate MRS analysis into your broader R analysis pipeline.

Acknowledgements



Dr. Catherine Jutzeler



Dr. Niklaus Zölch



Lucie Bourguignon

1st MR Spectroscopy Hackathon

MRS Hackathon 2023 team

Antonia Kaiser

Georg Oeltzschnner

Candace Fleischer

Jaime Near

R via AI

```
R 4.1.2 : ~ /Documents/GitHub/openai/ 
> library(openai)
> |
```

