

Short Course: Day 2

Data analysis, modeling and reporting using R, RStudio and RMarkdown

International symposium on current trends in modeling and
software development in data science and Statistics

Cape Town, South Africa

20.02.2024

Rudradev Sengupta
Ziv Shkedy

Johnson&Johnson



vliruos
SHARING MINDS, CHANGING LIVES

Overview

- 1** Introduction & Motivation

- 2** R Software: Basic Introduction

- 3** R Studio Interface

- 4** Installing and Loading R Packages

- 5** Projects in R and developing R packages

- 6** Case Study and analysis with R

- 7** RMarkdown

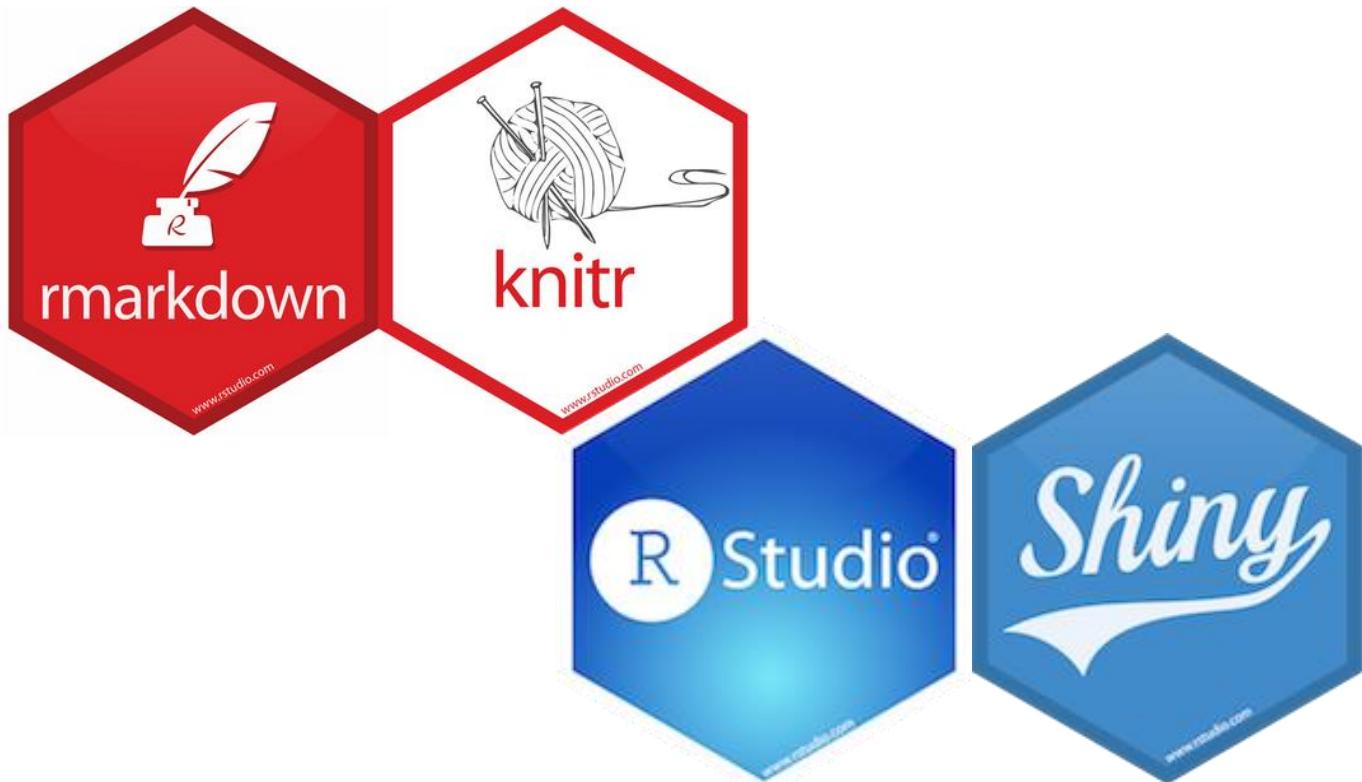
- 8** **Advanced RMarkdown**

- 9** Dashboard for Covid-19 using RMarkdown

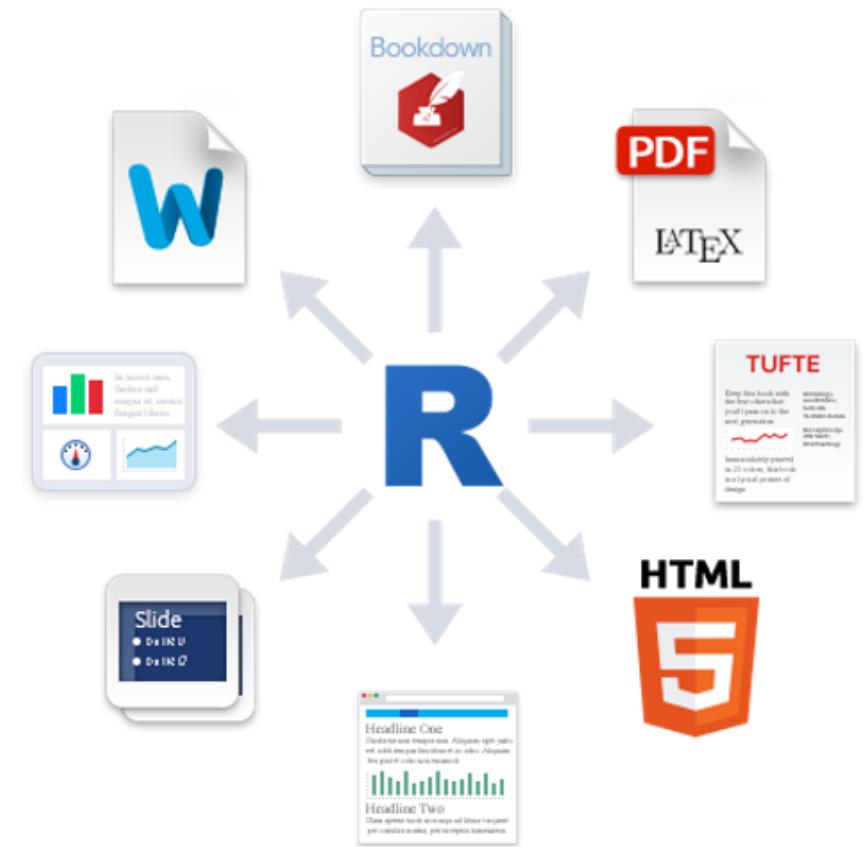
RMarkdown Refresher

Reporting in R

R Markdown & Shiny



- Automated, reproducible and dynamic reporting!
- How to share/publish your visualisations?



<https://rmarkdown.rstudio.com/gallery.html>

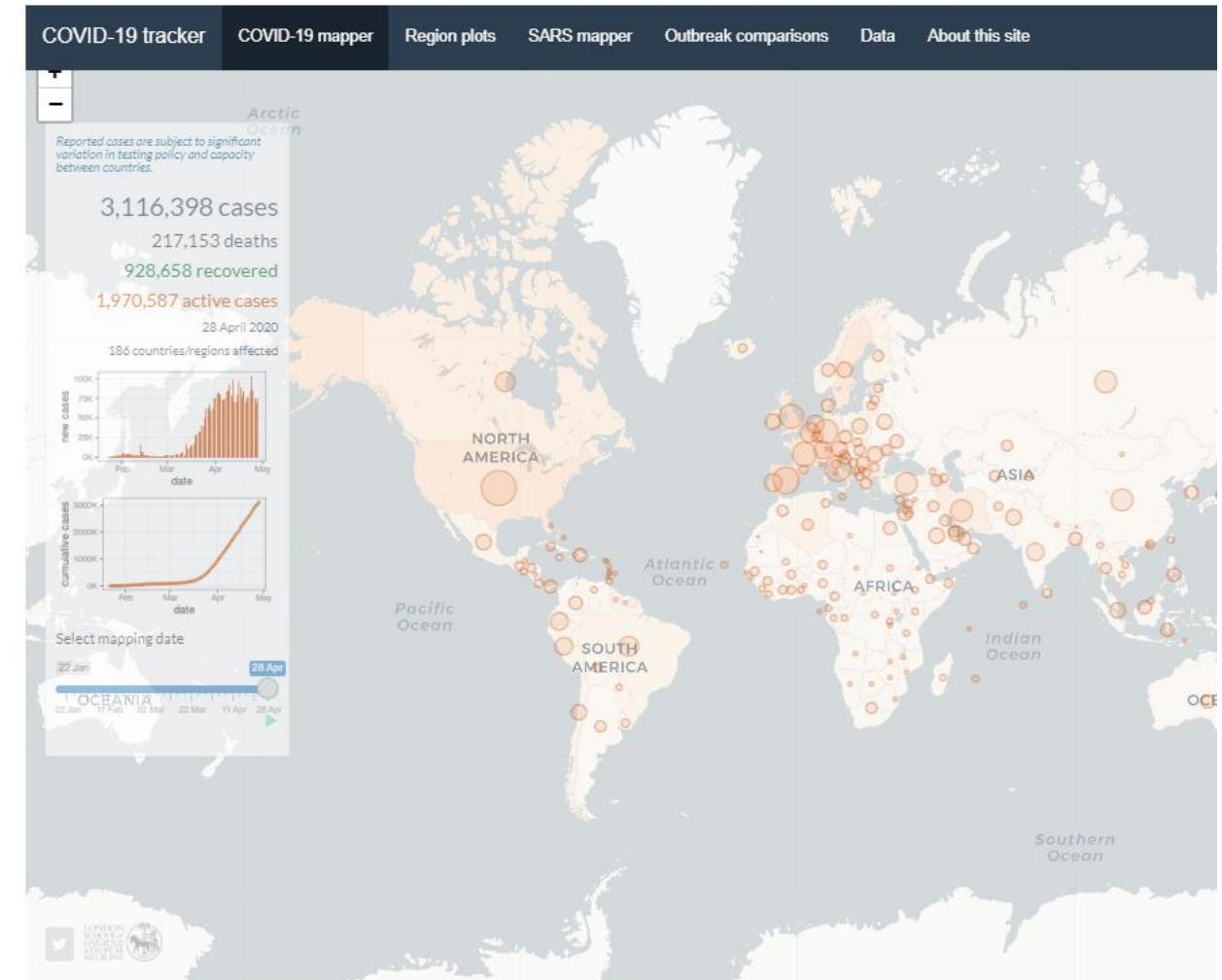
Reporting in R

- Text + Statistical Code
- Flexdashboards, Reports, Presentations, Books, Websites, Journal Templates...
- same report, different data / same report, different analysis parameters
- Static/dynamic/interactive
- Allow full analysis workflow:
Upload -> Analyse -> Visualise -> Report/Download

Reporting in R

Shiny

Fully interactive and on-the-fly computations!



Example: RMarkdown

1 Data

2 Exercises: ggplot

Exercise 1

Exercise 2

3 Exercises: plotly

4 Exercises: ganimate

5 Bonus Exercise

Exercise 1

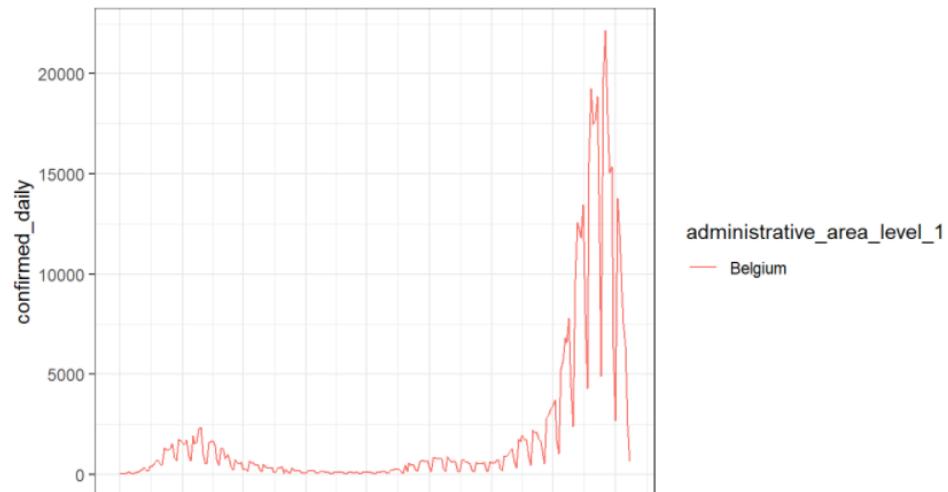
Use `data1` to visualize the daily confirmed cases in Belgium over time with a `colored` line. Also make sure all months (with year) appear on the x-axis and give the graph a title.

- **Data:** `data1`
- **Variables of interest:** `date` , `confirmed_daily` , `administrative_area_level_1`
- **Functions/tips:** `geom_line` , `scale_x_date` , `geom_smooth`

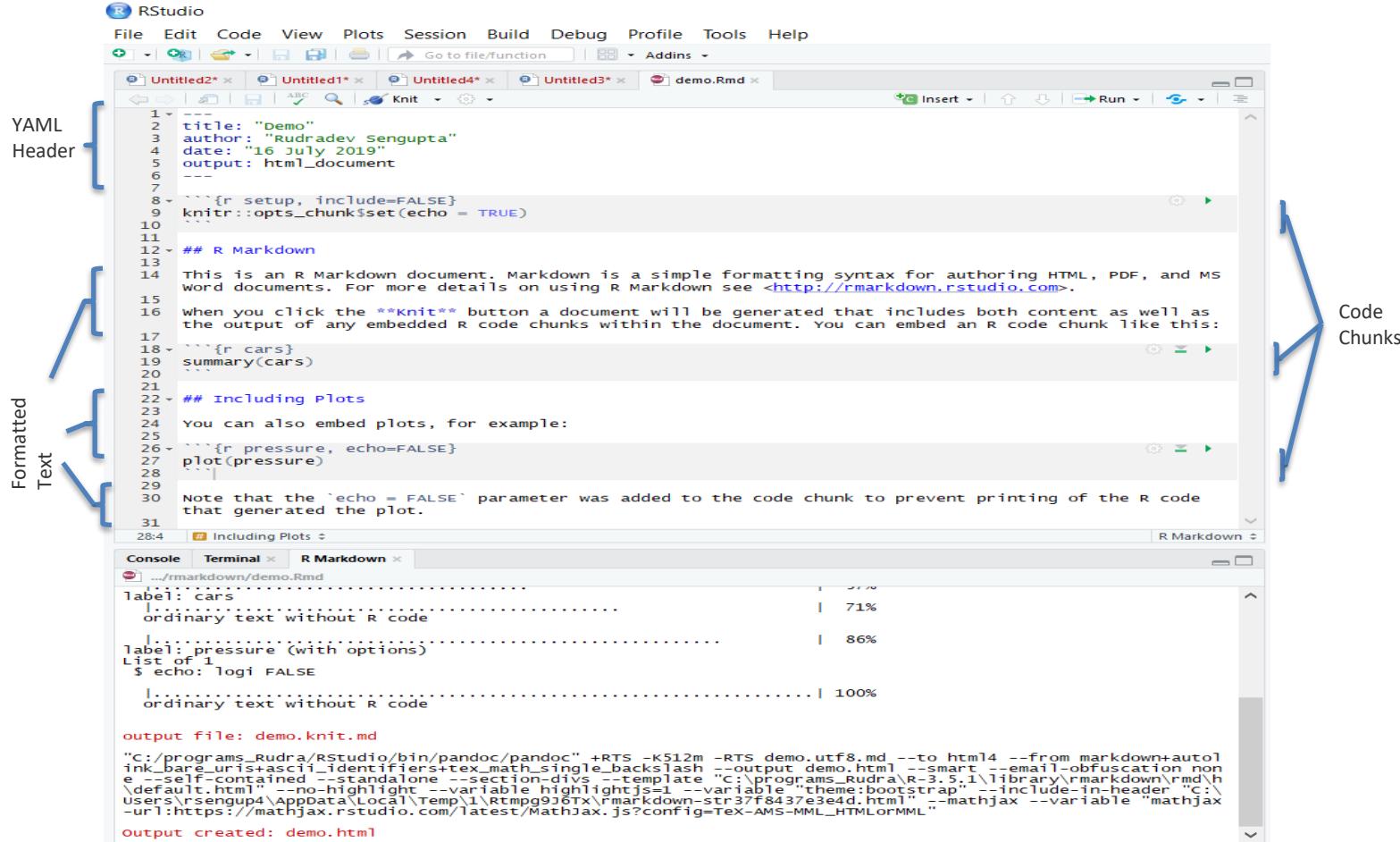
Enter your solution here

```
ggplot(data1,aes(x=date,y=confirmed_daily,col=administrative_area_level_1)) +  
  geom_line() +  
  scale_x_date(date_breaks="1 month", date_labels = format("%B %Y"))+  
  theme_bw() + labs(title="Daily confirmed cases in Belgium") +  
  theme(axis.text.x = element_text(angle=60,hjust=1))
```

Daily confirmed cases in Belgium



RMarkdown: Refresher



- There are principally three sections to an RMarkdown document:
 - YAML header surrounded by `---`
 - Code chunks surrounded by `````
 - Free text mixed with simple text formatting like `#heading` and `_italics_`
 - Also possible to include inline code to make it more dynamic:
 - ``r pressure$temperature[5] * pressure$pressure[5]``

Static Reporting vs Shiny

Static File

- rmarkdown,
bookdown,
flexdashboard,...
- Reporting
 - Html, pdf,...
- Htmlwidgets
 - Interactive Plots,
Tables,...

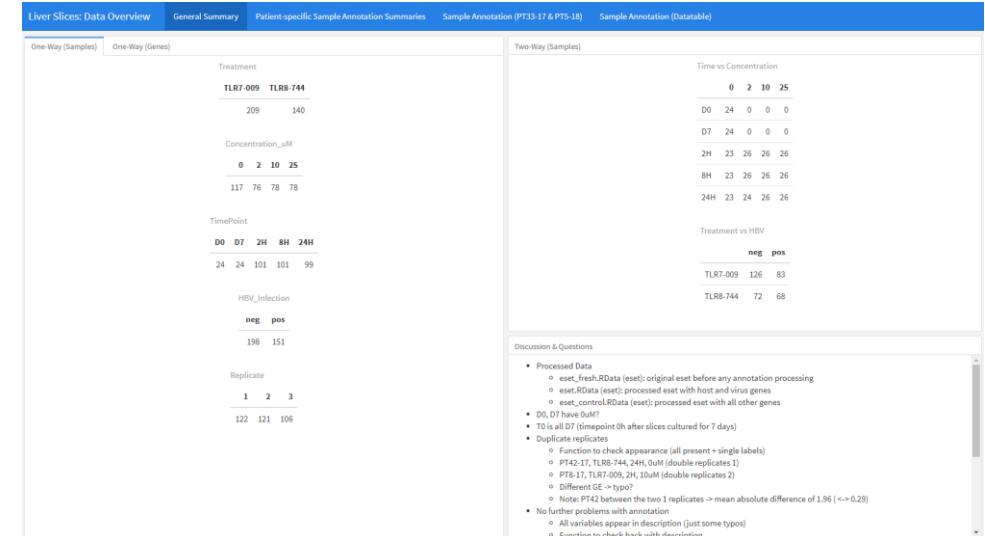
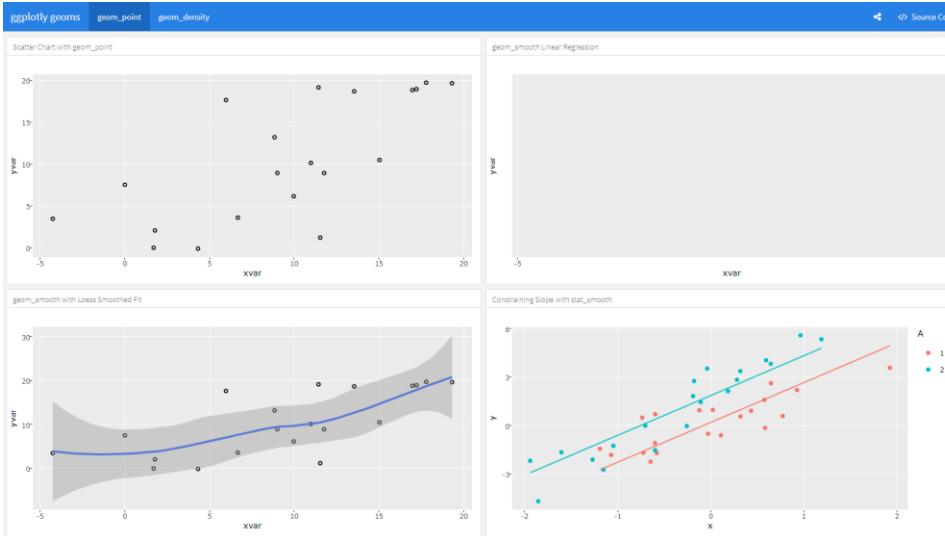
Server-Hosted

- shiny,
shinydashboard
- Upload/Download
- Full Pipeline/Analysis
- Generate Reports
- Fully Interactive
Visualisation

Example: Flexdashboard



Example: Flexdashboard



Note:

For all examples, you can click the source code button on the top right to see the .Rmd file.

Advanced RMarkdown

Customize YAML Header

```
1  ---
2  title: "Analysis"
3  author: "Rudradev Sengupta"
4  date: "November 2020"
5  output:
6    html_document:
7      theme: cerulean
8      code_folding: hide
9      toc: true
10     toc_depth: 2
11     toc_float: true
12     collapsed: true
13     number_sections: true
14 params:
15   # Filter region by upper/lower-bound.
16   region:
17     label: "Region to be Analyzed"
18     value: [25,200]
19     input: slider
20     min: 0
21     max: 500
22     step: 1
23     round: 1
24     sep: ''
25   thresh:
26     label: "Quantile Threshold"
27     value: 0.05
28     input: numeric
29     step: 0.05
30   spiralPointSize:
31     label: "Dot size for the spiral plot"
32     value: 3
33     input: numeric
34     step: 1
35   data_location:
36     label: "Input Data Location"
37     value: "default path"
38   ref_location:
39     label: "Reference Data Location"
40     value: "default path"
41 knit: (function(input, ...) {
42   rmarkdown::render(
43     input,
44     output_dir = "output path"
45   })
46 ---
```

- Code folding option
- Specify parameters to be used in the code later
 - Loation of input files, output files

Floating Table of Contents

R demo - RStudio

File Edit Code View Plots Session Build Deb

hello.R template.yaml tryhello.Rmd

ABC Knit

```
1 ---  
2 title: "tryhello"  
3 author: "Rudradev Sengupta"  
4 date: "11/6/2020"  
5 output:  
6   html_document:  
7     toc: TRUE  
8     toc_float: TRUE  
9     toc_depth: 2  
10    code_folding: hide  
11 ---
```

Floating table of contents

Code Folding

R demo - RStudio

File Edit Code View Plots Session Build Deb

hello.R template.yaml tryhello.Rmd

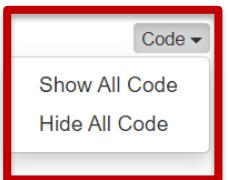
ABC Knit Hide

```
1 ---  
2 title: "tryhello"  
3 author: "Rudradev Sengupta"  
4 date: "11/6/2020"  
5 output:  
6   html_document:  
7     toc: TRUE  
8     toc_float: TRUE  
9     toc_depth: 2  
10    code_folding: hide  
11 ---
```

tryhello

Rudradev Sengupta

11/6/2020



R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

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
```

Hide

Including Plots

You can also embed plots, for example:

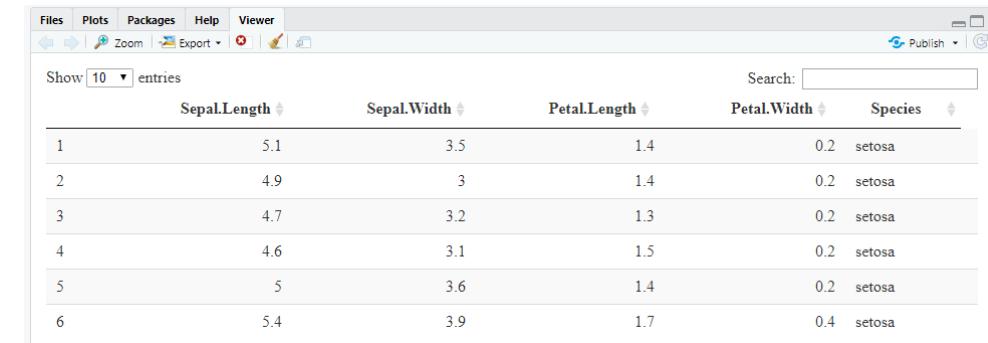
Code

Include Tables in RMarkdown

```
knitr::kable(head(iris))
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

```
DT::datatable(head(iris))
```



The screenshot shows the RStudio interface with the 'Viewer' tab selected. A data table is displayed with the following columns: Sepal.Length, Sepal.Width, Petal.Length, Petal.Width, and Species. The table includes a header row and six data rows. The 'Sepal.Length' column has values 5.1, 4.9, 4.7, 4.6, 5.0, and 5.4. The 'Sepal.Width' column has values 3.5, 3.0, 3.2, 3.1, 3.6, and 3.9. The 'Petal.Length' column has values 1.4, 1.4, 1.3, 1.5, 1.4, and 1.7. The 'Petal.Width' column has values 0.2, 0.2, 0.2, 0.2, 0.2, and 0.4. The 'Species' column has values setosa, setosa, setosa, setosa, setosa, and setosa. There are also buttons for 'Show 10 entries' and 'Search'.

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

- kable takes a data.frame as input, and outputs the table into a markdown table, which will get rendered into the appropriate output format
- datatable adds interactivity (in html) – options to search, sort, ...

Tables in RMarkdown

- Adding captions to tables:
 - knitr::kable(head(iris), caption = "A subset of the IRIS dataset.")
 - DT::datatable(head(iris), caption = "A subset of the IRIS dataset.")
- Check the functions for other available options
- Refer to a table within Rmarkdown:
 - Table \@ref(tab:tab1)
- More details:
<https://bookdown.org/yihui/bookdown/tables.html#tables>

```
```{r tab1, echo = TRUE}

knitr::kable(
 head(mtcars[, 1:8], 10), booktabs = TRUE,
 caption = 'A table of the first 10 rows of the mtcars data.')
...```

```

# Include Plots/Images in RMarkdown

- `ggplot(iris, aes(x = Sepal.Length, y = Petal.Length)) + geom_point(aes(color = Species))` or `include_graphics(img_path)`
- Add captions
- Use `plotly` to make it interactive (in html)

```
```{r fig1, echo = TRUE, fig.cap = "Scatterplot of Sepal.Length vs Petal.Length"}  
data("iris")  
ggplot(iris, aes(x = Sepal.Length, y = Petal.Length)) + geom_point(aes(color = species))  
```
```

# Control Figure Dimensions

- ````{r fig1, echo = TRUE, fig.cap = "Scatterplot of Sepal.Length vs Petal.Length", **fig.height**=2}  
  data("iris")  
  ggplot(iris, aes(x = Sepal.Length, y = Petal.Length)) + geom\_point(aes(color = Species))  
```
- ````{r fig2, echo = TRUE, fig.cap = "Scatterplot of Sepal.Length vs Petal.Length", **fig.width**=2}
 data("iris")
 ggplot(iris, aes(x = Sepal.Length, y = Petal.Length)) + geom_point(aes(color = Species))
```
- ````{r fig3, echo = TRUE, fig.cap = "Scatterplot of Sepal.Length vs Petal.Length", **fig.dim**=c(2,2)}  
  data("iris")  
  ggplot(iris, aes(x = Sepal.Length, y = Petal.Length)) + geom\_point(aes(color = Species))  
```

- Refer to a figure within Rmarkdown:
 - Figure \@ref(fig:fig1)

Creating Tabs

```
## Sales Report {.tabset}
```

```
### By Product
```

```
(tab content)
```

```
### By Region
```

```
(tab content)
```

Creating Tabs

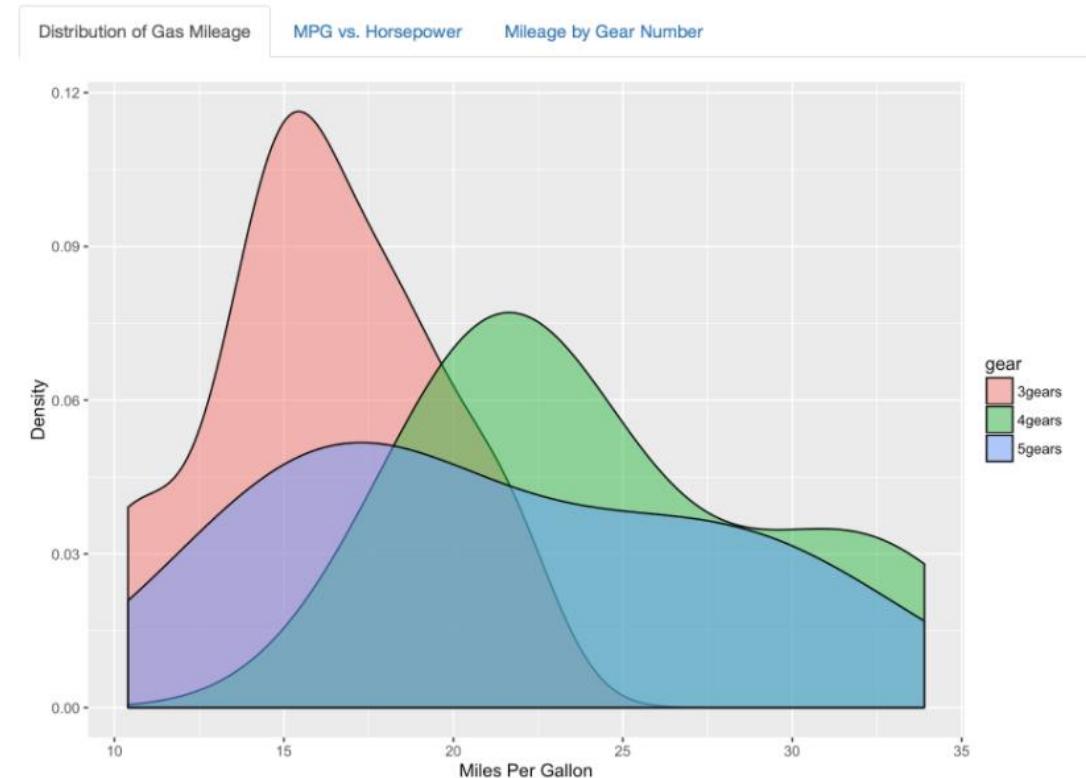
```
## Sales Report {.tabset}
```

```
### By Product
```

```
(tab content)
```

```
### By Region
```

```
(tab content)
```



Overview

- 1** Introduction & Motivation

- 2** R Software: Basic Introduction

- 3** R Studio Interface

- 4** Installing and Loading R Packages

- 5** Projects in R and developing R packages

- 6** Case Study and analysis with R

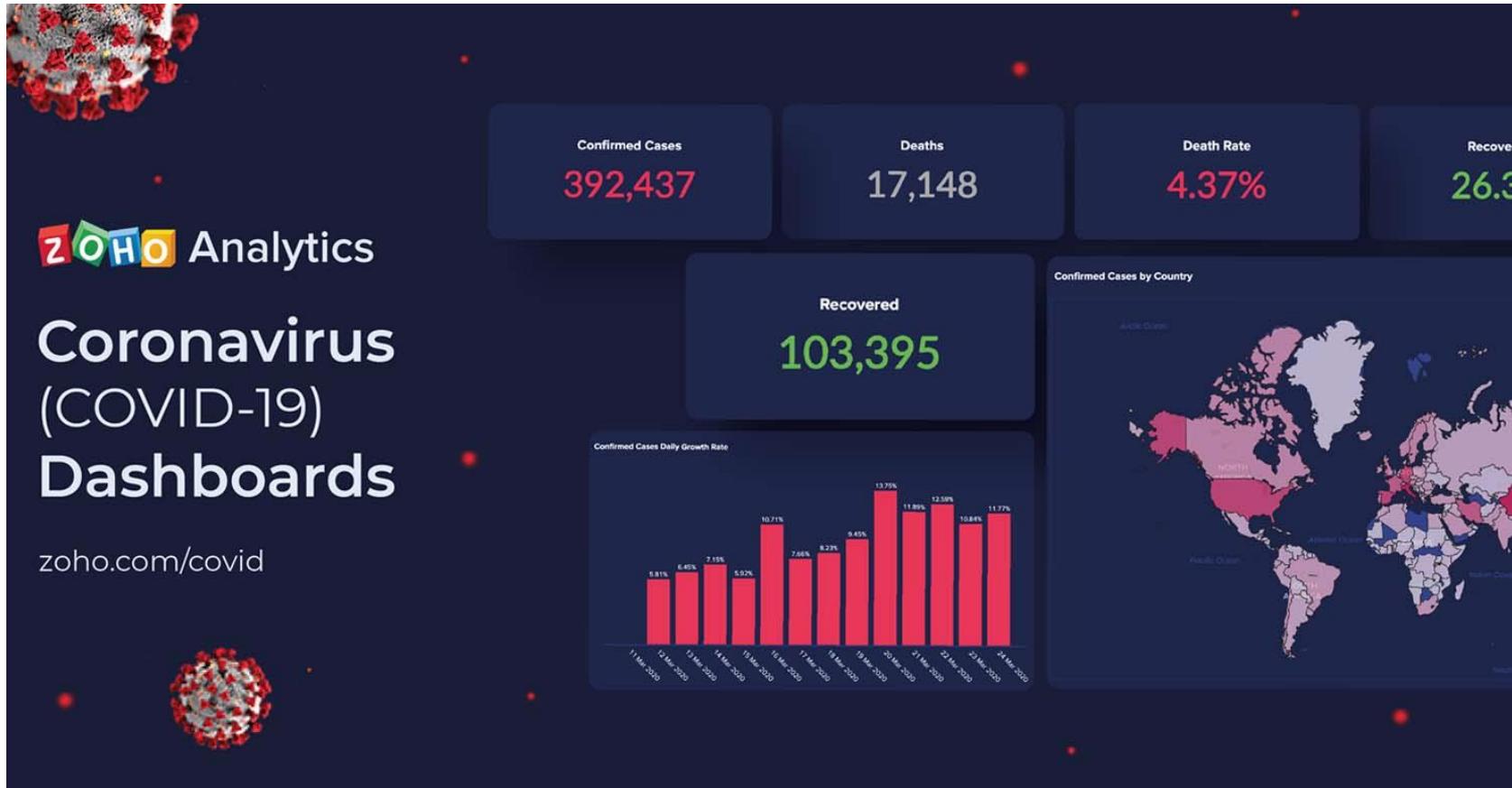
- 7** RMarkdown

- 8** Advanced RMarkdown

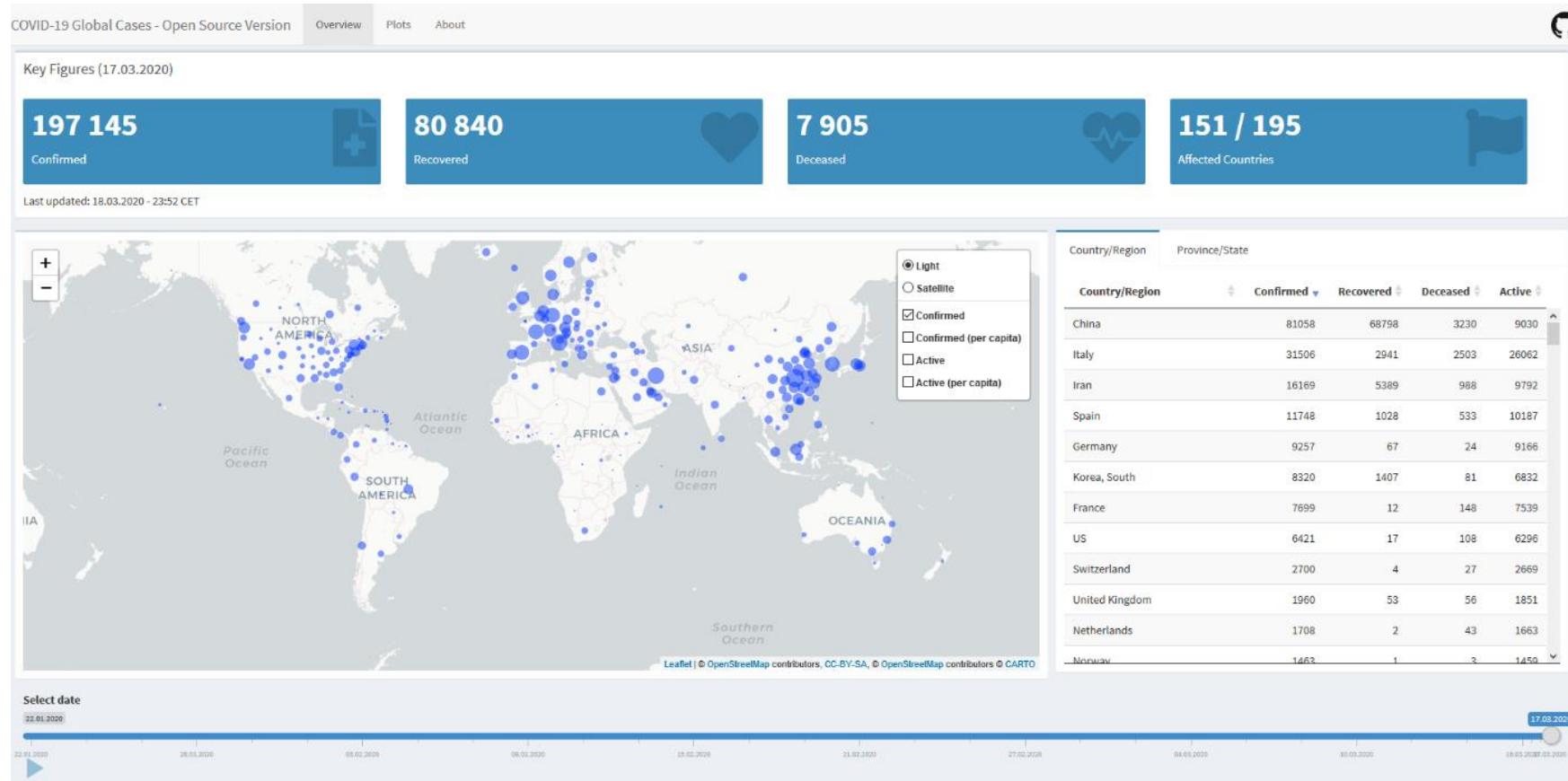
- 9** Dashboard for Covid-19 using RMarkdown

Flexdashboard

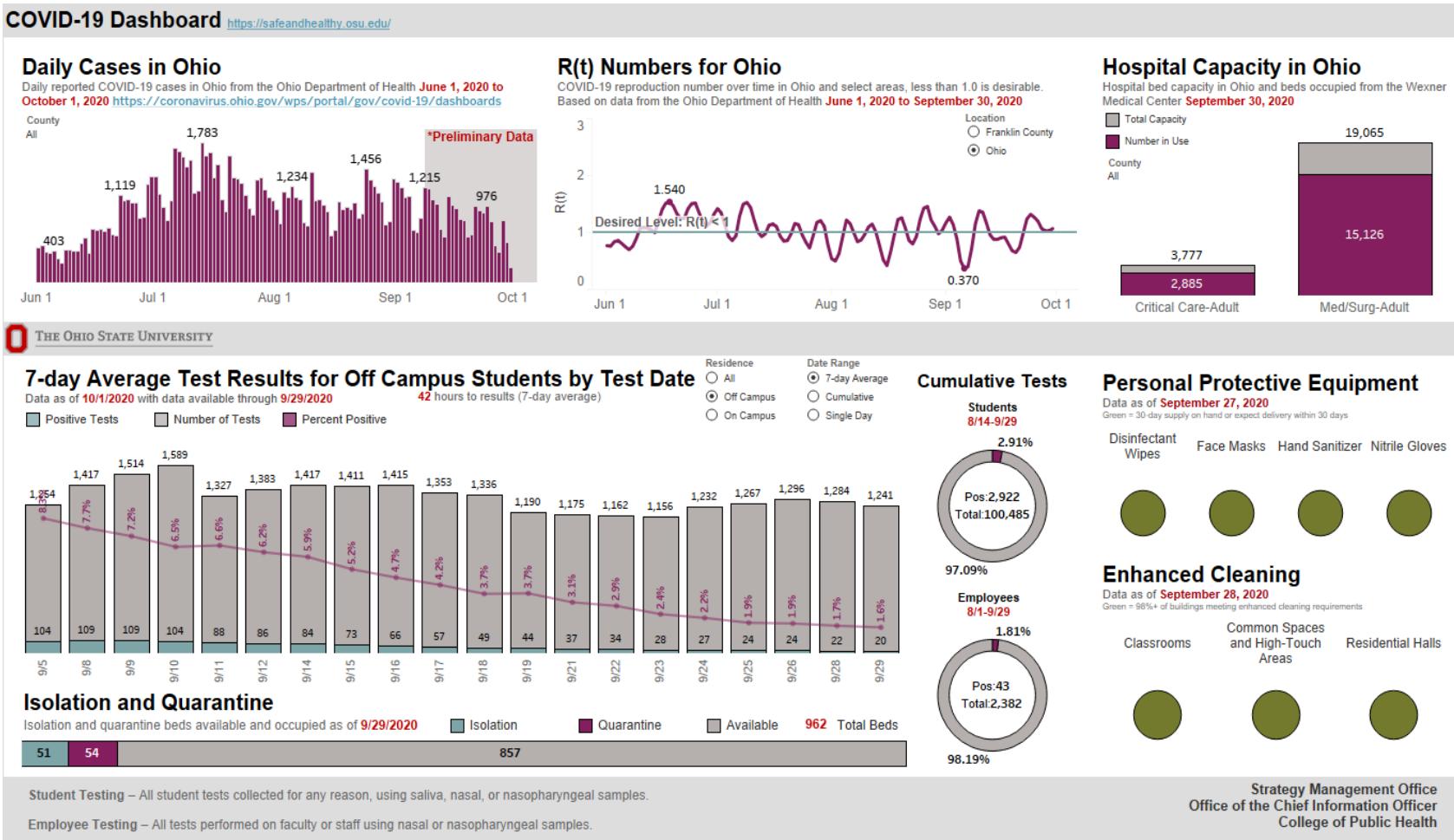
What is a Dashboard ?



What is a Dashboard ?



What is a Dashboard ?



What is a Dashboard ?



Why a Dashboard ?

- Easier online communication of the modelling results and current outbreak situation for:
 - the general public.
 - policy makers.
- Very useful:
 - for understanding the real picture.
 - against fake news.

Why a Dashboard using R ?

- Free software to do statistical analysis.
- Results can be reproduced and validated by anybody.
- Publicly available datasources
 - Governments can sometimes manipulate the data.
 - Delay in official data publication.

What is Flexdashboard?

- Publish data report/visualisations as a **dashboard**
- Combine:
 - Text Annotation, R code, R graphics (base, ggplot2,...), Tables (knitr::kable()), Value Boxes, Gauges...
 - Storyboard: Design, Methodology, Exploration, Results, Discussion,...

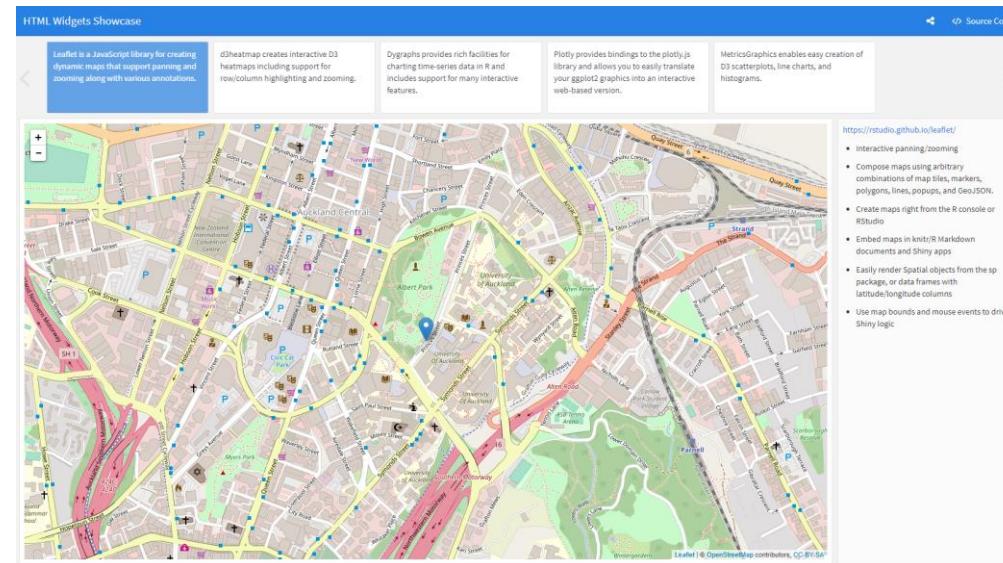
Flexdashboard	Shiny
<ul style="list-style-type: none">- Static<ul style="list-style-type: none">- No on-the-fly calculations- Interactivity: htmlwidgets- <i>Deployment</i><ul style="list-style-type: none">- Html file- R Markdown Syntax- Optional: include Shiny- Change size: window, mobile,...	<ul style="list-style-type: none">- Fully Interactive<ul style="list-style-type: none">- On-the-fly calculations- Interactivity: widgets, htmlwidgets,...- <i>Deployment</i><ul style="list-style-type: none">- Server- Shiny Syntax

Why Flexdashboard?

- Visually appealing alternative to standard Rmarkdown
- Quick learning curve due if you are familiar to Rmarkdown
- Can provide a better format to visually tell a story and/or explore results
- Extremely flexible layout system
- Static html file
- Shiny (server-hosted) integration also possible

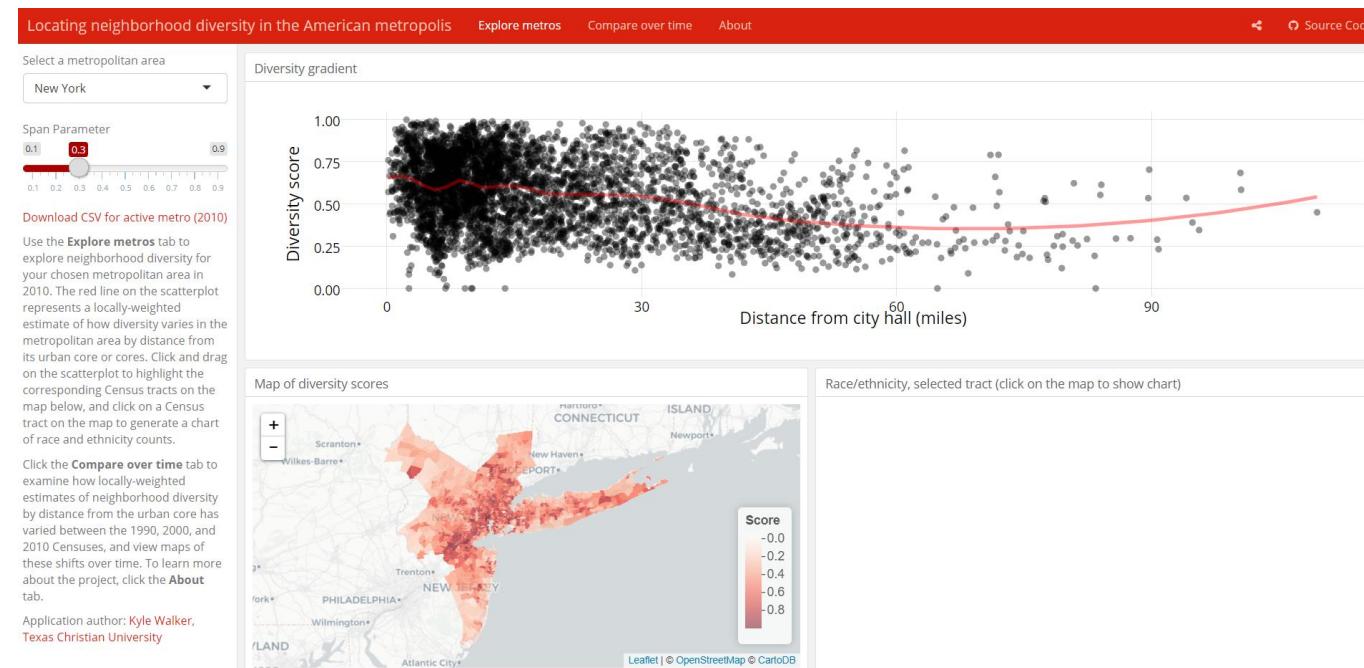
Flexdashboard: htmlwidgets

- **htmlwidgets for R**
 - Employ JavaScript visualisation libraries in R
 - Can be used in **Static HTML**
 - => Can be used in *flexdashboard*, *R Markdown*,...
 - Leaflet, Plotly, d3heatmap, DataTables, d3scatter, dygraphs,...



Flexdashboard: Shiny

- Integrate Shiny in a flexdashboard
 - Interactive
 - Run on a server/cloud



Flexdashboard

- Installation

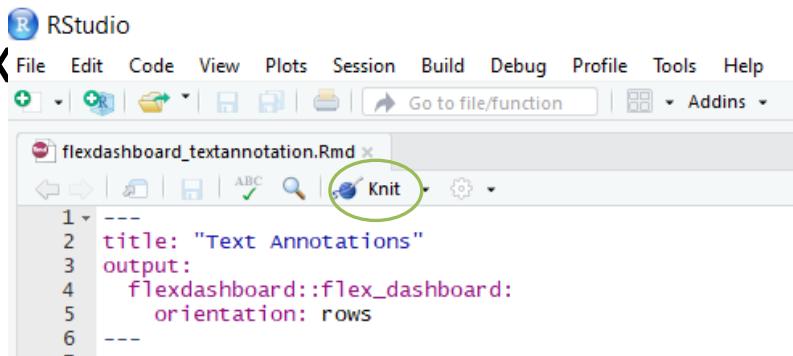
```
install.packages("flexdashboard")
```

- Syntax

- Rmarkdown = Flexdashboard
(i.e. Text, Plots, Tables, R chunks, Htmlwidgets,...)

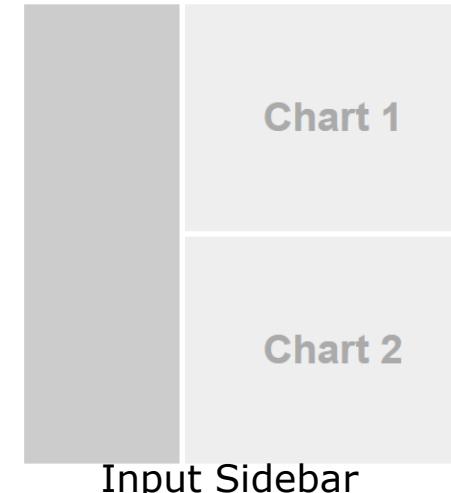
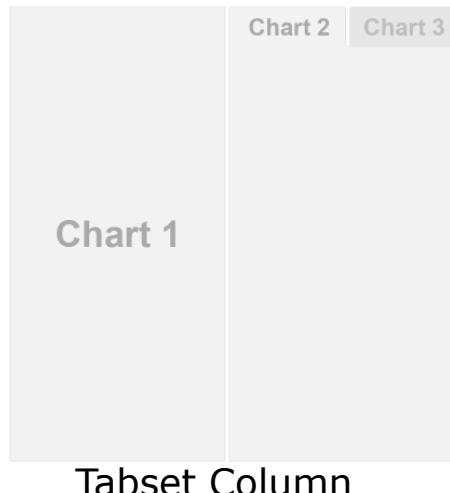
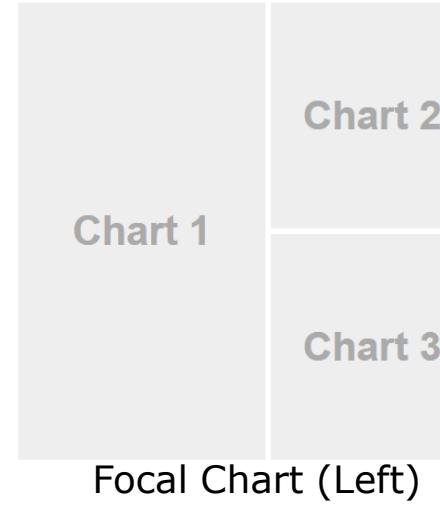
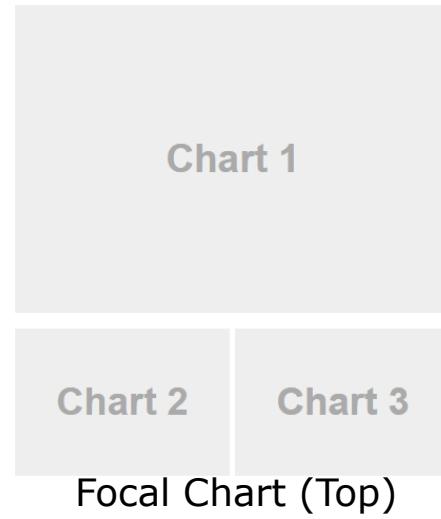
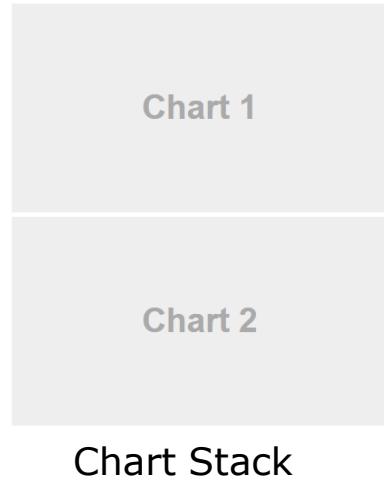
- Simple layout syntax

- Compile to html



```
rmarkdown::draft("dashboard.Rmd", template = "flex_dashboard", package = "flexdashboard")
```

Flexdashboard: Layout Overview



- And more...**
- Multiple Pages
 - Storyboard
 - Mobile Specific
 - Scrolling Layout
 - ...

Flexdashboard: Starting Point

```
---
```

```
title: "Arranging Data for Clear Communication"
output:
  flexdashboard::flex_dashboard:
    storyboard: true
    orientation: columns
    vertical_layout: fill
    source_code: embed
---
```

Flexdashboard: Layout with Fill/Scroll

```
1 ---  
2 title: "Single Column (Fill)"  
3 output:  
4   flexdashboard::flex_dashboard;  
5   vertical_layout: fill  
6 ---  
7  
8 ### Chart 1  
9  
10 ````{r}  
11 ...  
12 ...  
13  
14 ### Chart 2  
15  
16 ````{r}  
17 ...  
18 ...  
19  
20  
21  
22  
23  
24  
25  
26
```



```
1 ---  
2 title: "Single Column (Scrolling)"  
3 output:  
4   flexdashboard::flex_dashboard;  
5   vertical_layout: scroll  
6 ---  
7  
8 ### Chart 1  
9  
10 ````{r}  
11 ...  
12 ...  
13  
14 ### Chart 2  
15  
16 ````{r}  
17 ...  
18 ...  
19  
20 ### Chart 3  
21  
22 ````{r}  
23 ...  
24 ...  
25  
26  
27  
28  
29
```

Allow scrolling of the full page when more height is needed for multiple charts



Flexdashboard: Layout by Row/Column

```
1 ---  
2 title: "Column Orientation"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5 Column  
6 -----  
7  
8  
9 ### Chart 1  
10 `r`  
11 ...  
12  
13  
14 Column  
15 -----  
16  
17 ### Chart 2  
18 `r`  
19 ...  
20  
21  
22 ### Chart 3  
23 `r`  
24 ...  
25  
26
```

Chart 1

Chart 2

Chart 3

```
1 ---  
2 title: "Row Orientation"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     orientation: rows  
6 ---  
7  
8 Row  
9 -----  
10  
11 ### Chart 1  
12 `r`  
13 ...  
14  
15 Row  
16 -----  
17  
18 ### Chart 2  
19 `r`  
20 ...  
21  
22  
23  
24 ### Chart 3  
25 `r`  
26 ...  
27  
28
```

Chart 1

Chart 2

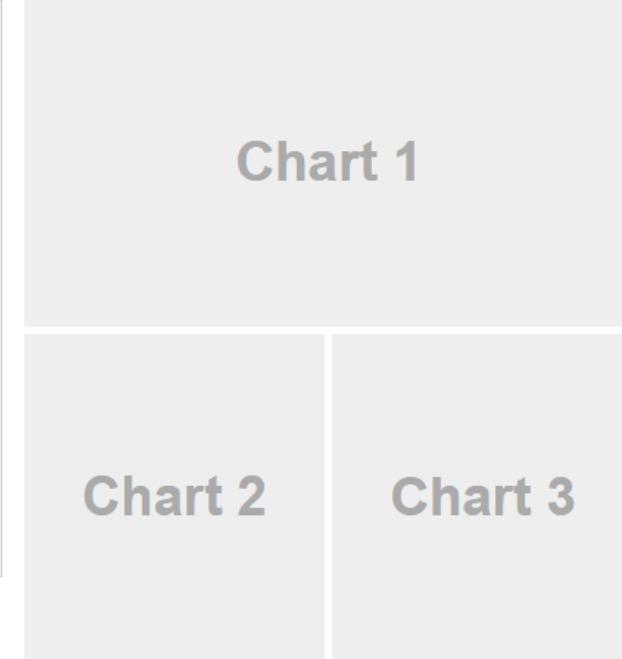
Chart 3

Flexdashboard: Layout by Row/Column

```
1 ---  
2 title: "Column Orientation"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5 Column  
6 -----  
7  
8  
9 ### Chart 1  
10 ````{r}  
11 ...  
12  
13 Column  
14 -----  
15  
16  
17 ### Chart 2  
18 ````{r}  
19 ...  
20  
21  
22 ### Chart 3  
23 ````{r}  
24 ...  
25  
26
```



```
1 ---  
2 title: "Row Orientation"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     orientation: rows  
6 ---  
7  
8 Row  
9 -----  
10  
11 ### Chart 1  
12 ````{r}  
13 ...  
14  
15 Row  
16 -----  
17  
18  
19 ### Chart 2  
20 ````{r}  
21 ...  
22  
23  
24 ### Chart 3  
25 ````{r}  
26 ...  
27  
28
```



Flexdashboard: Layout Tabssets

The image displays a flexdashboard interface with three tabs at the top: "Bicluster Heatmap", "Parallel Coordinates", and "Data for Selected Cluster". Below the tabs are three chart components: "Chart 1", "Chart 2", and "Chart 3".

The left panel shows the R code for the "Bicluster Heatmap" tab, which includes:

```
1 ---  
2 title: "Tabset Column"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5  
6 Column  
7 -----  
8  
9 ### Chart 1  
10  
11 `r`  
12 ...  
13  
14 Column( .tabset )  
15 -----  
16  
17 ### Chart 2  
18  
19 `r`  
20 ...  
21  
22 ### Chart 3  
23  
24 `r`  
25 ...  
26
```

The middle panel shows the R code for the "Parallel Coordinates" tab, which includes:

```
1 ---  
2 title: "Tabset Row"  
3 output:  
4 flexdashboard::flex_dashboard:  
5 orientation: rows  
6 ---  
7  
8 Row  
9 -----  
10  
11 ### Chart 1  
12  
13 `r`  
14 ...  
15  
16 Row( .tabset .tabset-fade )  
17 -----  
18  
19 ### Chart 2  
20  
21 `r`  
22 ...  
23  
24 ### Chart 3  
25  
26 `r`  
27 ...  
28
```

The right panel shows the R code for the "Data for Selected Cluster" tab, which includes:

```
1 ---  
2 title: "Chart 1"  
3 output:  
4 flexdashboard::flex_dashboard:  
5 orientation: rows  
6 ---  
7  
8 Row  
9 -----  
10  
11 ### Chart 1  
12  
13 `r`  
14 ...  
15  
16 Row( .tabset .tabset-fade )  
17 -----  
18  
19 ### Chart 2  
20  
21 `r`  
22 ...  
23  
24 ### Chart 3  
25  
26 `r`  
27 ...  
28
```

A red oval highlights the ".tabset" and ".tabset-fade" code blocks in both the middle and right panels.

Fade: fade-in/fade-out effect when switching tabs

Flexdashboard: Layout Sizing

```
1 ---  
2 title: "Chart Sizing"  
3 output:  
4 flexdashboard::flex_dashboard:  
5 orientation: rows  
6 ---  
7  
8 Row {data-height=650}  
9 -----  
10  
11 ### Chart 1  
12  
13 `r`  
14  
15  
16 Row {data-height=350}  
17 -----  
18  
19 ### Chart 2  
20  
21 `r`  
22  
23  
24 ### Chart 3  
25  
26 `r`  
27  
28
```

Chart 1

Chart 2

Chart 3

```
1 ---  
2 title: "Custom Chart Height"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5  
6 ### Chart 1 {data-height=600}  
7  
8 `r`  
9  
10  
11 ### Chart 2 {data-height=200}  
12  
13 `r`  
14  
15  
16 ### Chart 3 {data-height=200}  
17  
18 `r`  
19  
20  
21  
22  
23
```

Chart 1

Chart 2

Chart 3

Width of charts determined by browser
(unless overwritten by `data-width`)

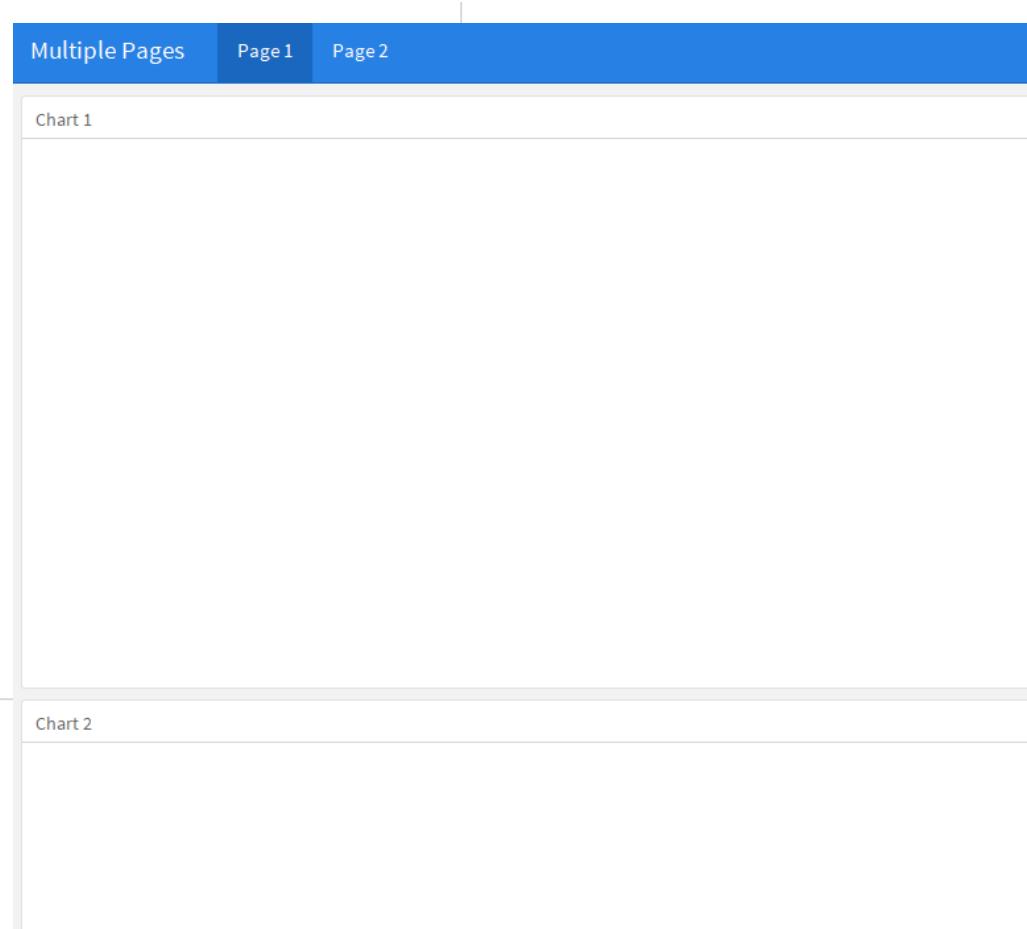
Chart Padding (default 8 pixels)

```
1 ### Chart 1 {no-padding}  
2  
3 `r`  
4  
5  
6 ### Chart 2 {data-padding=10}  
7  
8 `r`  
9
```

Flexdashboard: Layout Multiple Pages

```
1 ---
2 title: "Multiple Pages"
3 output: flexdashboard::flex_dashboard
4 ---
5
6 Page 1
7 =====
8
9 ### Chart 1
10 `r
11 ...
12
13
14 ### Chart 2
15
16 `r
17 ...
18
19 Page 2
20 =====
21
22 ### Chart 1
23
24 `r
25 ...
26
27 ### Chart 2
28
29 `r
30 ...
31
```

```
1 ---
2 title: "Page Orientation"
3 output: flexdashboard::flex_dashboard
4 ---
5
6 Page 1
7 =====
8
9
10 Page 2 {data-orientation=rows}
11 =====
12
```



Flexdashboard: Layout Multiple Pages

```
1 ---  
2 title: "Page Navigation Menus"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5  
6 Page 1 {data-navmenu="Menu A"}  
7 ======  
8  
9  
10 Page 2 {data-navmenu="Menu A"}  
11 ======  
12  
13  
14 Page 3 {data-navmenu="Menu B"}  
15 ======  
16  
17  
18 Page 4 {data-navmenu="Menu B"}  
19 ======  
20
```

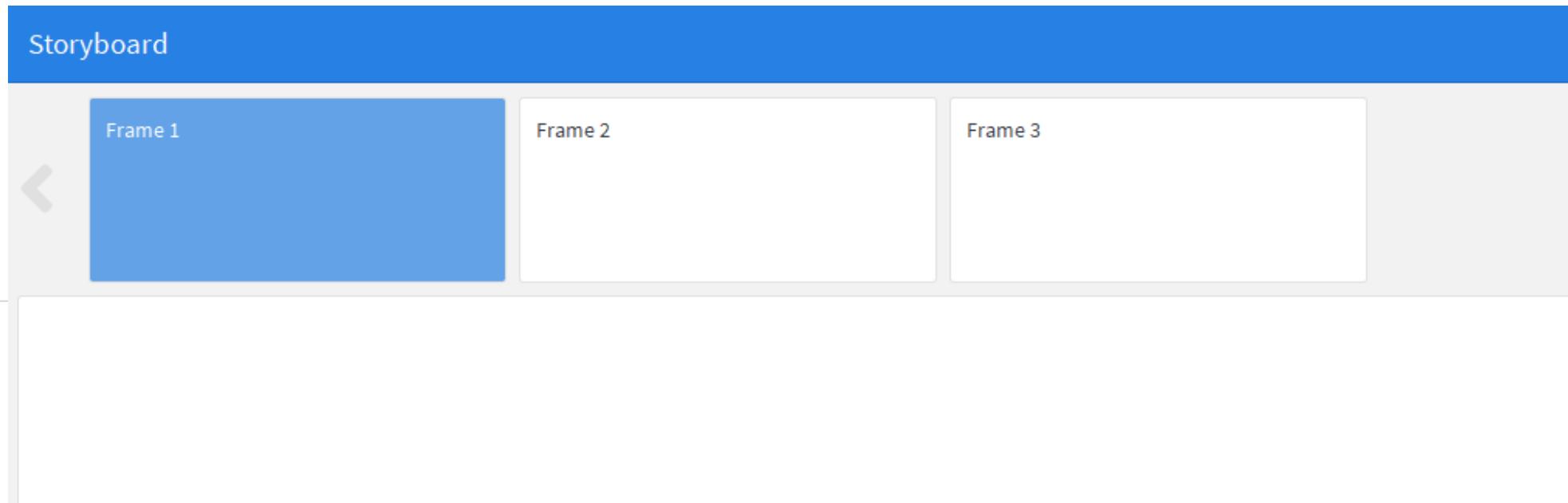
The screenshot shows a flexdashboard interface. At the top, there is a header bar with the title "Page Navigation Menus" and two dropdown menus labeled "Menu A" and "Menu B". Below the header, there are four pages listed: "page1, menu A", "Page 1", and "Page 2". "Page 1" is highlighted with a blue background, indicating it is the active page. The other pages are shown in white boxes.

Extra for Multiple Pages

- Page Links
 - Links to specific pages
- Hiding Pages
 - Using page links instead
- Page Icons

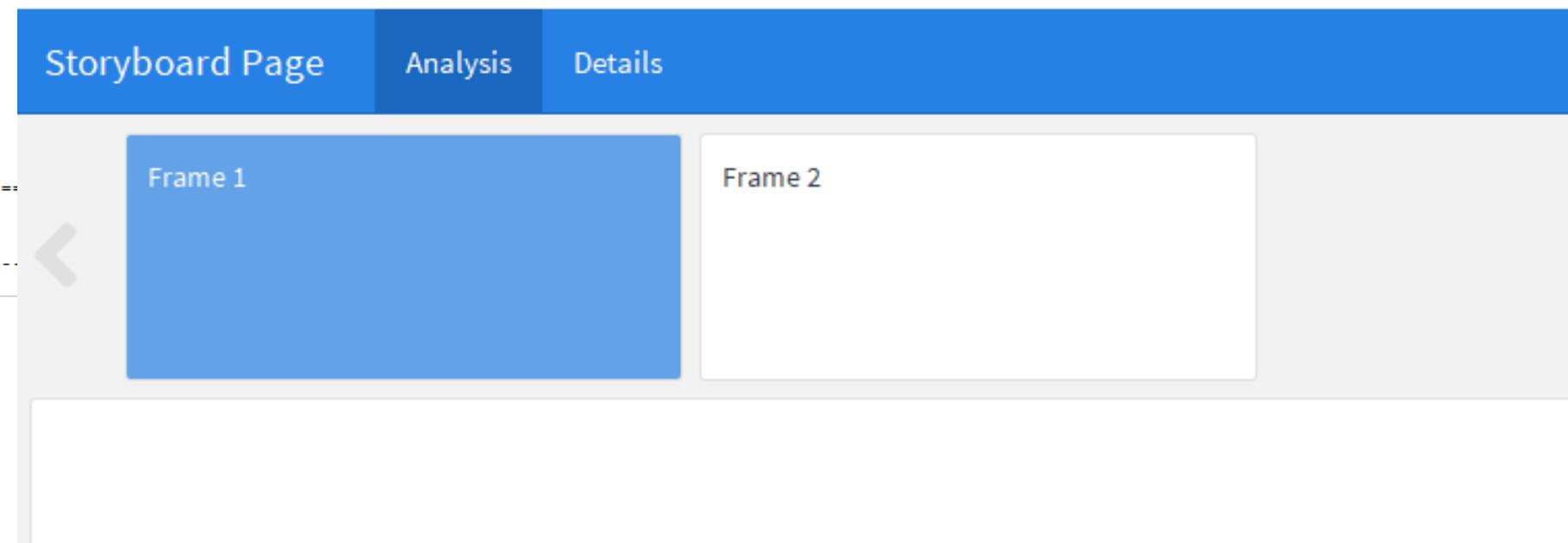
Flexdashboard: Layout Storyboards

```
1 ---  
2 title: "Storyboard"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     storyboard: true  
6 ---  
7  
8 ### Frame 1  
9  
10 ````{r}  
11 ````  
12  
13 ### Frame 2  
14  
15 ````{r}  
16 ````  
17  
18 ### Frame 3  
19  
20 ````{r}  
21 ````  
22
```



Flexdashboard: Layout Storyboards

```
1 ---  
2 title: "Storyboard Page"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5  
6 Analysis {.storybook}  
7 =====  
8  
9 ### Frame 1  
10  
11 `~~{r}  
12 ...  
13  
14 ### Frame 2  
15  
16 `~~{r}  
17 ...  
18  
19 Details  
20 =====  
21  
22 Column  
23 -----  
24
```



Flexdashboard: Layout Storyboards

```
1 ---  
2 title: "Storyboard Commentary"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     storyboard: true  
6 ---  
7  
8 ### Frame 1  
9  
10 `r`  
11 ***  
12 ***  
13 Some commentary about Frame 1.  
14  
15 Some commentary about Frame 1.  
16  
17 ### Frame 2 {data-commentary-width=400}  
18  
19 `r`  
20 ***  
21 ***  
22 Some commentary about Frame 2.  
23  
24 Some commentary about Frame 2.  
25
```

Storyboard Commentary

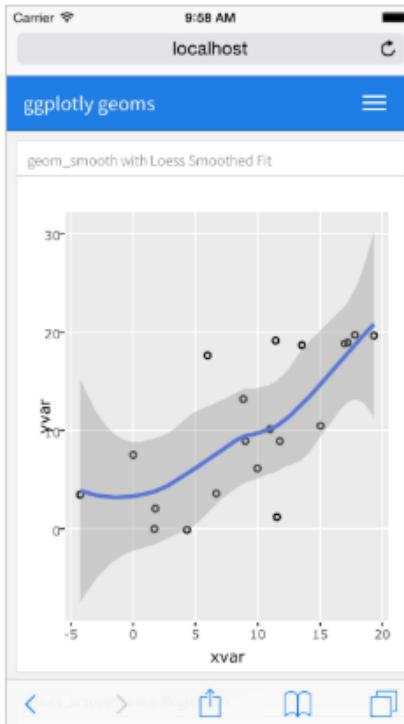
Frame 1

Frame 2

Some commentary about Frame 1.

Flexdashboard: Mobile Layout

- R graphics rendered twice, natural and mobile-optimized size
- Portrait is used (control with `fig_mobile`)



Exclude components

```
1 ---  
2 title: "Exclude on Mobile"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5  
6 ### Chart 1  
7  
8 ```{r}  
9 plot(cars)  
10 ...  
11  
12 ### Chart 2 {.no-mobile}  
13  
14 ```{r}  
15 plot(summary)  
16 ...  
17
```

Mobile-specific

```
1 ---  
2 title: "Mobile Specific Component"  
3 output: flexdashboard::flex_dashboard  
4 ---  
5  
6 ### Chart 1  
7  
8 ```{r}  
9 plot(cars)  
10 ...  
11  
12 ### Chart 1 {.mobile}  
13  
14 ```{r}  
15 plot(cars)  
16 ...  
17
```

Flexdashboard: Layout Components

Value Boxes

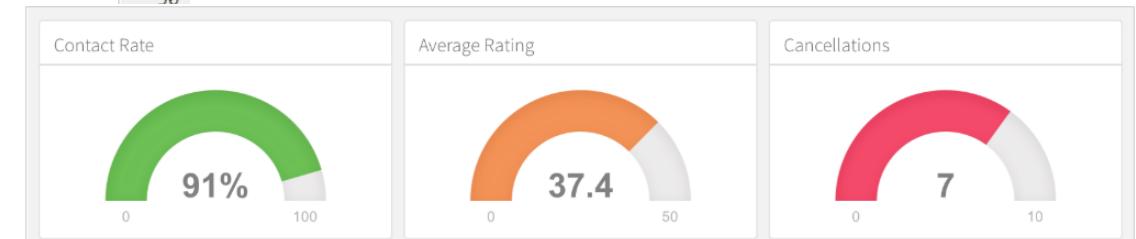
```
1 Row
2 -----
3
4 ### Articles per Day
5
6 `~~{r}
7 articles <- computeArticles()
8 valueBox(articles, icon = "fa-pencil")
9 `~~
10
11 ### Comments per Day
12
13 `~~{r}
14 comments <- computeComments()
15 valueBox(comments, icon = "fa-comments")
16 `~~
17
18 ### Spam per Day
19
20 `~~{r}
21 spam <- computeSpam()
22 valueBox(spam,
23   icon = "fa-trash",
24   color = ifelse(spam > 10, "warning", "primary"))
25 `~~
26
```

```
1 ### Linked Value Box
2
3 `~~{r}
4 valueBox(42, icon = "fa-pencil", href="#details")
5 `~~
6
```



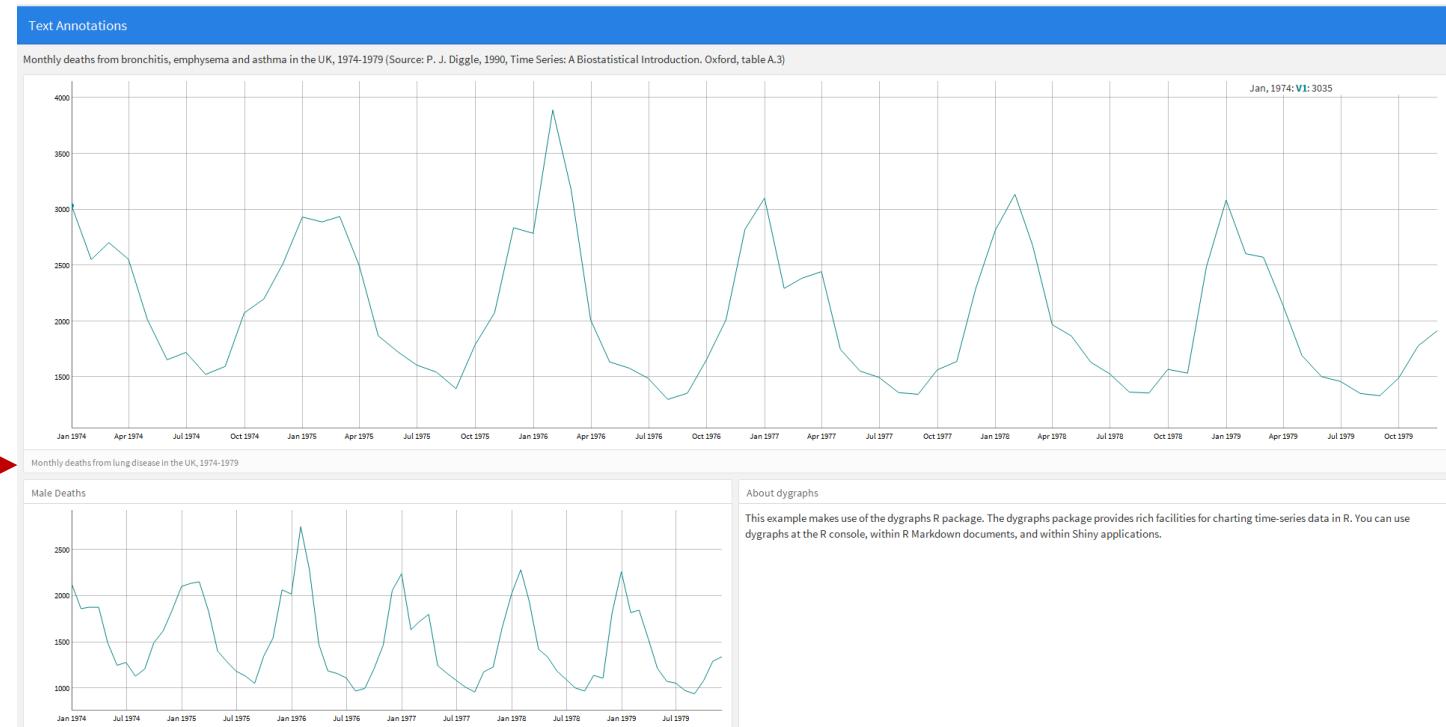
Gauges

```
1 Row
2 -----
3
4 ### Contact Rate
5
6 `~~{r}
7 rate <- computeContactRate()
8 gauge(rate, min = 0, max = 100, symbol = '%', gaugeSectors(
9   success = c(80, 100), warning = c(40, 79), danger = c(0, 39)
10 ))
11 `~~
12
13 ### Average Rating
14
15 `~~{r}
16 rating <- computeAverageRating()
17 gauge(rating, min = 0, max = 50, gaugeSectors(
18   success = c(41, 50), warning = c(21, 40), danger = c(0, 20)
19 ))
20 `~~
21
22 ### Cancellations
23
24 `~~{r}
25 cancellations <- computeCancellations()
26 gauge(cancellations, min = 0, max = 10, gaugeSectors(
27   success = c(0, 2), warning = c(3, 6), danger = c(7, 10)
28 ))
29 `~~
30
```



Flexdashboard: Layout Text Annotation

```
1 ---  
2 title: "Text Annotations"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     orientation: rows  
6 ---  
7  
8 Monthly deaths from bronchitis, emphysema and asthma in the  
9 UK, 1974-1979 (Source: P. J. Diggle, 1990, Time Series: A  
10 Biostatistical Introduction. Oxford, table A.3)  
11  
12 `~{r setup, include=FALSE}  
13 library(dygraphs)  
14`~  
15 Row {data-height=600}  
16 -----  
17  
18 ### All Lung Deaths {.no-title}  
19  
20 `~{r}  
21 dygraph(ldeaths)  
22`~  
23 <-- Monthly deaths from lung disease in the UK, 1974-1979  
24 Row {data-height=400}  
25 -----  
26  
27  
28 ### Male Deaths  
29  
30 `~{r}  
31 dygraph(mdeaths)  
32`~  
33  
34 ### About dygraphs  
35  
36 This example makes use of the dygraphs R package. The dygraphs  
37 package provides rich facilities for charting time-series data  
38 in R. You can use dygraphs at the R console, within R Markdown  
39 documents, and within Shiny applications.  
40
```

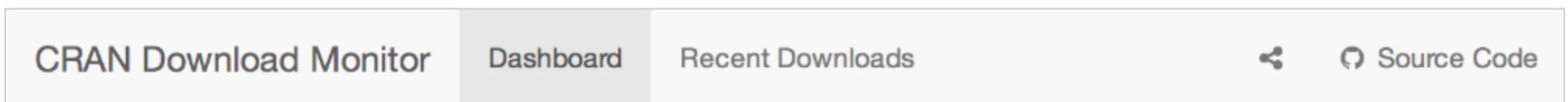


Flexdashboard: Layout Themes

- Themes:
 - default, cosmo, **bootstrap**, cerulean, journal, flatly, readable, spacelab, united, lumen, paper, sandstone, simplex, yeti
 - <https://bootswatch.com/>

```
1 ---  
2 title: "Themes"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     theme: bootstrap  
6 ---  
7 ---
```

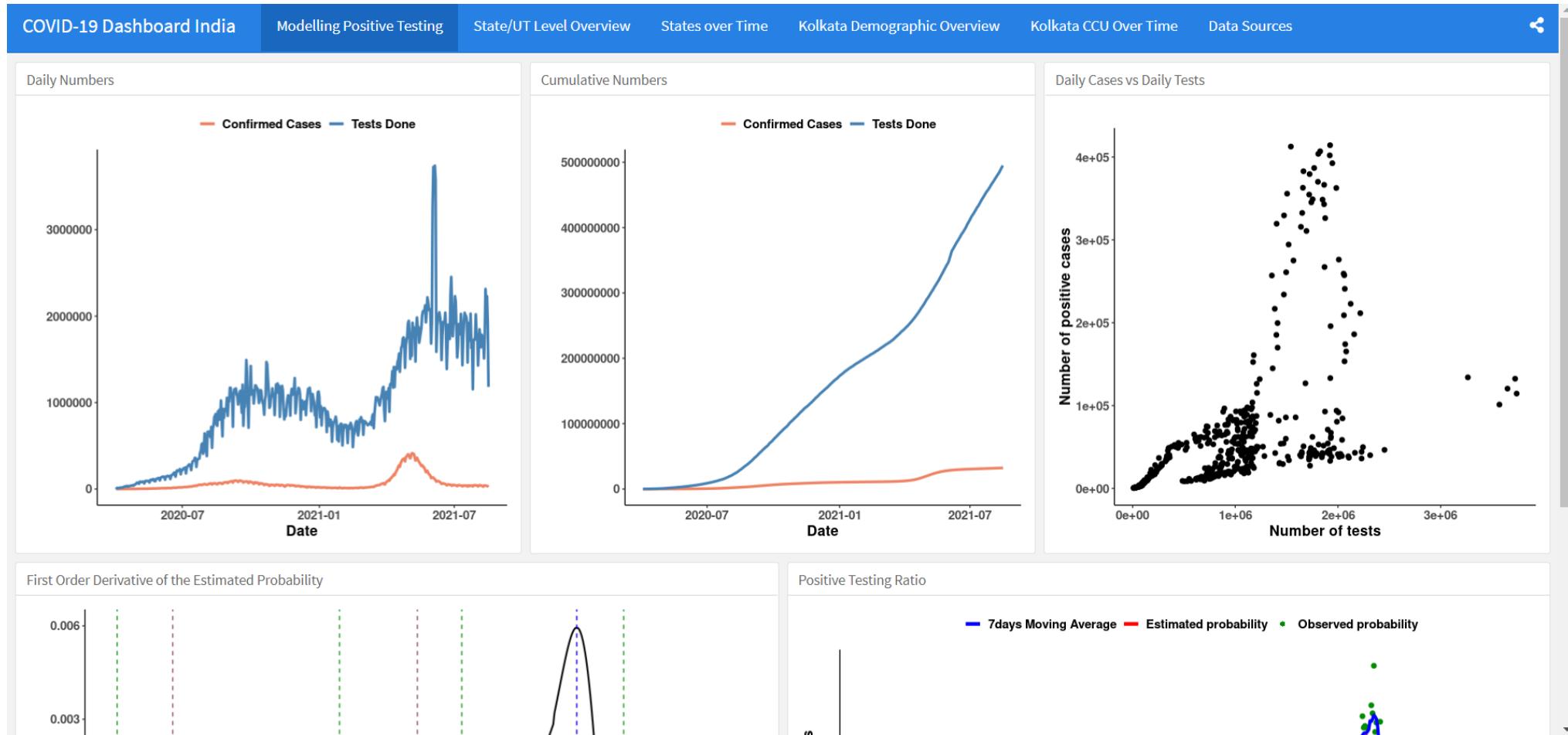
- Extra:
 - Navigation Bar
 - Source Code (Embed/Link)
 - Social links



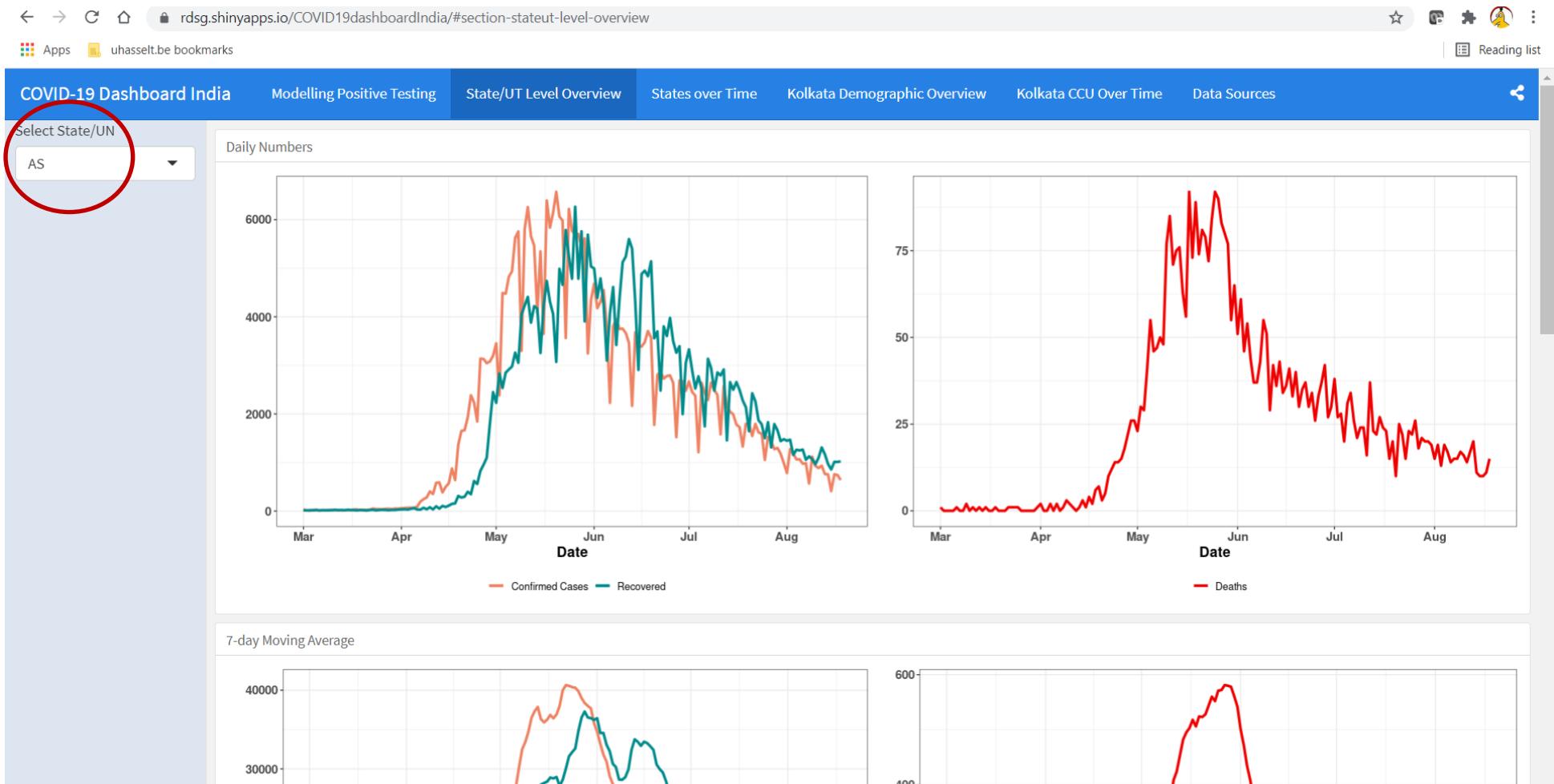
COVID-19 Dashboard: India

- Interactive and automated dashboard entirely made in R
- Information communicated in the dashboard:
 - Number of daily cases (overall and by state): local transmission.
 - Positive testing rate.
 - Number of cases by state over time.
 - Availability of CCU beds.
 - Availability of hospitals without CCU beds.
- All information from online datasets.

COVID-19 Dashboard: India



COVID-19 Dashboard: India

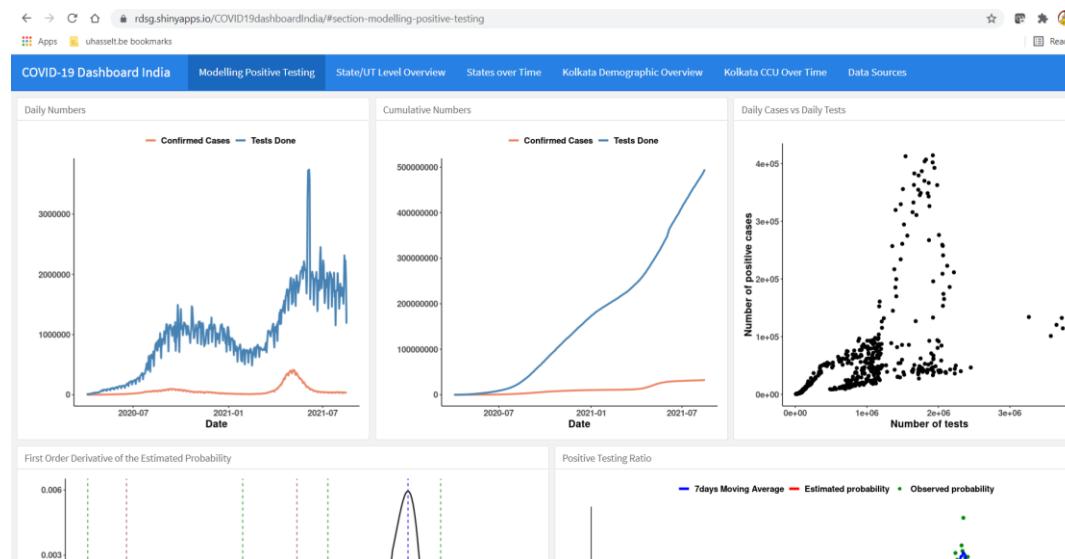


How to Create the Dashboard in R ?

- R studio.
- R markdown.
- flexdashboard/shinydashboard.
- Graphical displays: ggplot2, plotly,
- Tables: DT, data.table

RMarkdown and Dashboard

Dashboard



Rmd file

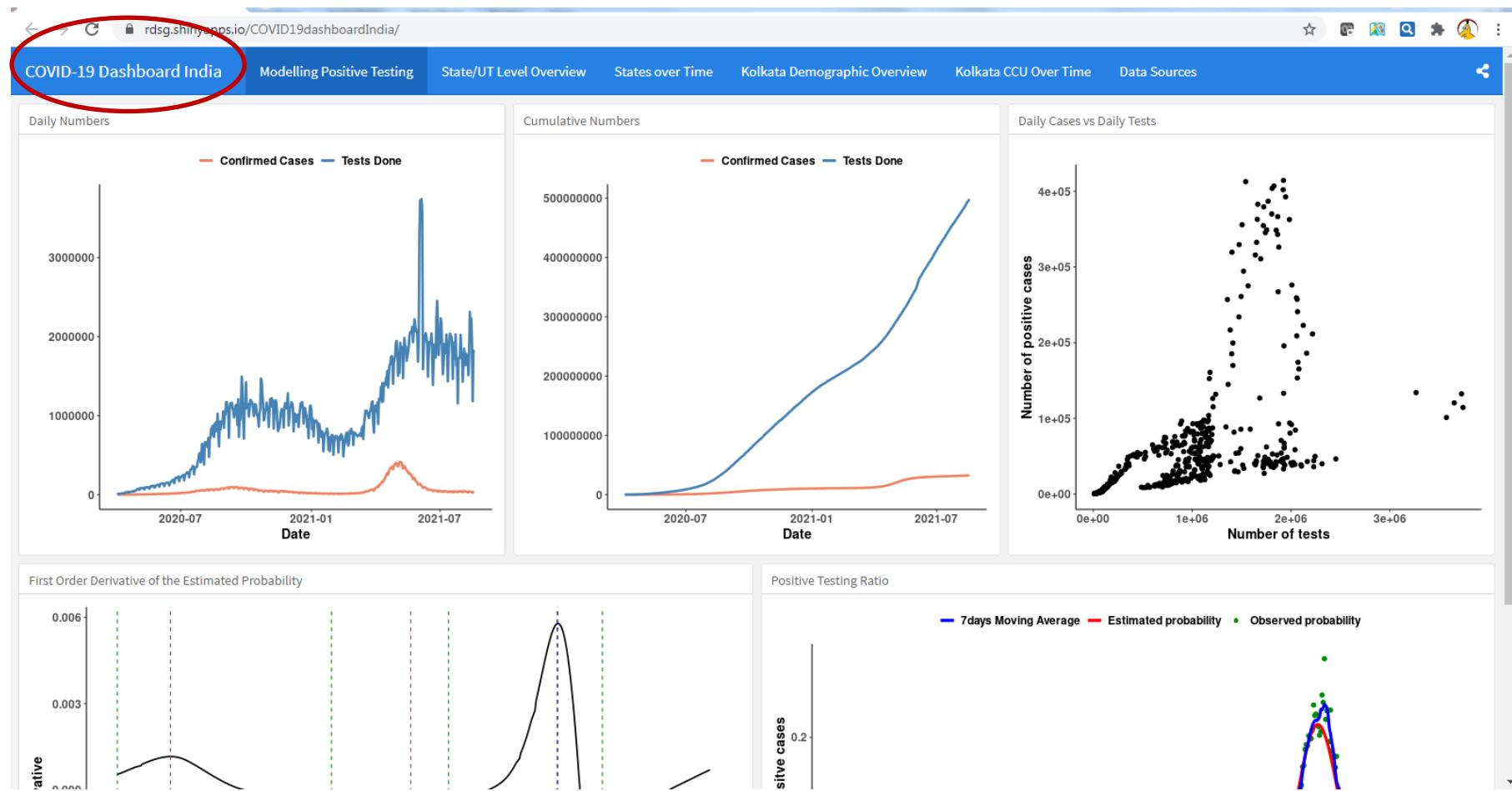
The Rmd file contains the following code:

```
25 # Modelling Positive Testing
26 library(ggplot2)
27 library(ggridges)
28 library(ggforce)
29 library(tidyverse)
30 library(gridExtra)
31 library(ggpubr)
32 library(covr)
33 library(ggpmisc)
34
35 x <- covid19
36 country_names <- unique(x$administrative_area_level_1)
37
38 # Only number of COVID-19 tests and cases in India from 01/04/2020-31/05/2021
39 enddate <- Sys.Date() - 4
40
41 COVID-19_Dashboard_Inc <-
```

The RStudio interface shows the environment pane with various packages loaded, and the global environment pane listing objects like `dat`, `dat1`, `dat2`, `datmat`, `datest`, `idd`, `out`, `outp02`, `outp04`, and `outp05`.

- We focus on:
 - Dashboard in R.
 - How to produce the dashboard for India in R.

COVID-19 Dashboard for India: Main Page



Rmd File

The screenshot shows the RStudio interface with the following components:

- File Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Project Bar:** Project: (None).
- Left Panel:** A code editor window containing an Rmd file. The code includes R Markdown syntax and R code for reading data from Excel files and loading RDA files. A red callout box highlights the title of the dashboard in the code.

```
1 ---  
2 title: "COVID-19 Dashboard India"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     orientation: rows  
6     social: menu  
7     vertical_layout: scroll  
8     runtime: shiny  
9 ---  
10 ````{r global, include=FALSE}  
11 # load data in 'global' chunk so it can be shared by all users of the dashboard  
12 library(openxlsx)  
13  
14 bed_dat <- read.xlsx("data/timeseries_beds_rudra.xlsx")  
15 bed_dat$Dates <- as.POSIXct(as.numeric(bed_dat$Dates)*86400),origin = "1899-12-30", tz="UTC")  
16  
17 load("data/koldata24052021.rda")  
18 kol_dat_timeseries2 <- kol_dat_timeseries  
19 kol_dat_timeseries2$Dates <- as.POSIXct(kol_dat_timeseries2$Dates)  
20  
21  
22  
23  
24
```
- Console:** Displays the R version information and the workspace loaded message.
- Environment:** Shows the global environment with various objects listed, such as dat, dat1, data1, datmat, dattest, idd, out, outp02, outp04, and scandat1.
- Packages:** Shows the system library with packages like abind, additivityTests, ash, askpass, assertthat, asympow, av, backports, base64enc, BH, biclust, BiocManager, BiocStyle, BiocVersion, bitops, and BMA.

Definition of the dashboard

Rmd File

The screenshot shows the RStudio interface with an Rmd file open. The code editor contains the following YAML header and R code:

```
1 ---  
2 title: "COVID-19 Dashboard India"  
3 output:  
4   flexdashboard::flex_dashboard:  
5     orientation: rows  
6     social: menu  
7     vertical_layout: scroll  
8     runtime: shiny  
9 ---  
10   
11 ```{r global, include=FALSE}  
12 # load data in 'global' chunk so it can be shared by all users of the dashboard  
13 library(openxlsx)  
14  
15 bed_dat <- read.xlsx("data/timeseries_beds_rudra.xlsx")  
16 bed_dat$Dates <- as.POSIXct(as.numeric(bed_dat$Dates)*86400),origin = "1899-12-30", tz="UTC")  
17  
18 load("data/koldata24052021.rda")  
19 kol_dat_timeseries2 <- kol_dat_timeseries  
20 kol_dat_timeseries2$Dates <- as.POSIXct(kol_dat_timeseries2$Dates)  
21  
22 ```  
23  
24
```

The R console output shows the R version and license information.

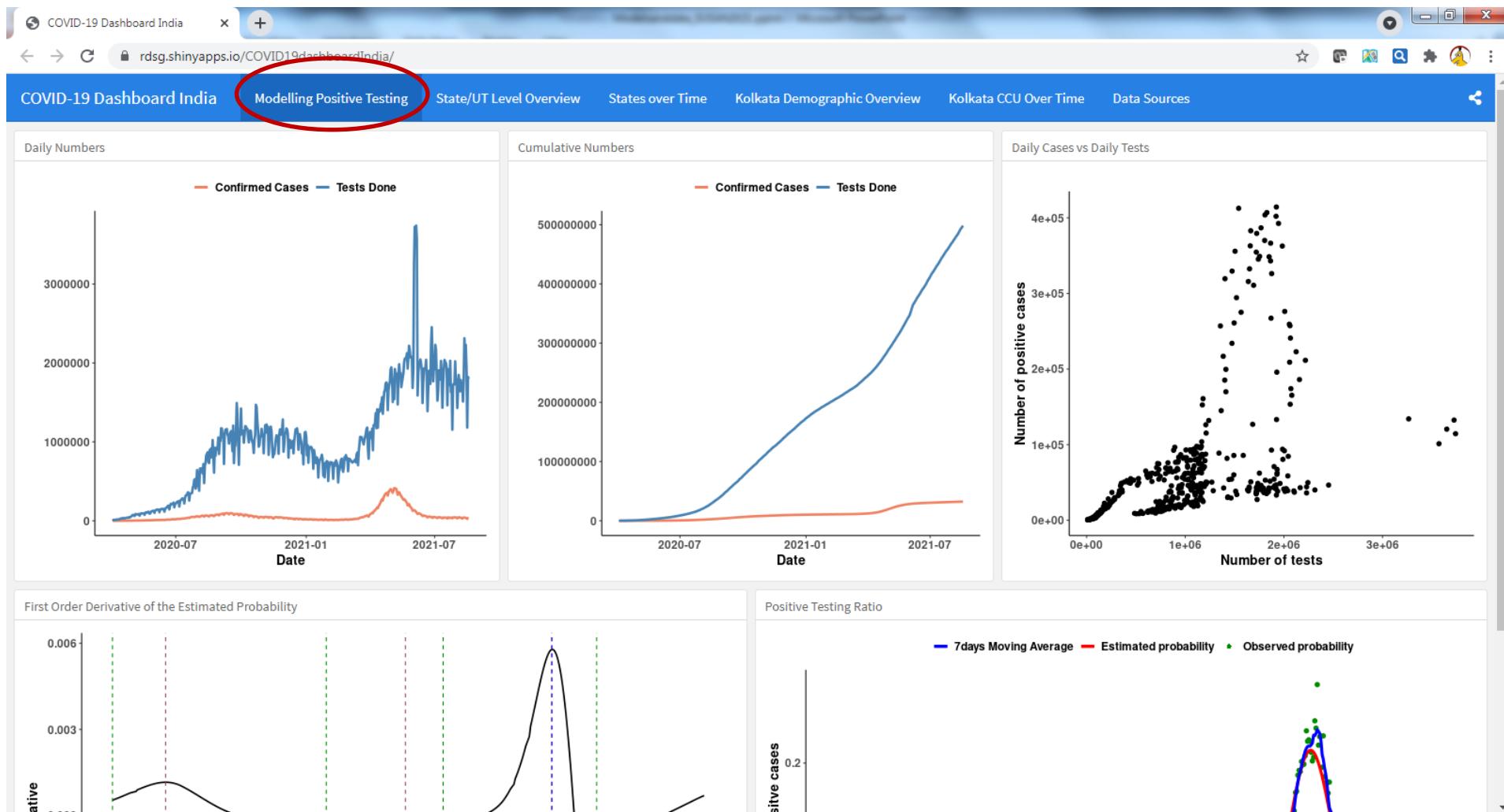
A red box highlights the YAML header with the text "Remember YAML Header??". A red arrow points from this box to the "Definition of the dashboard" box.

A red box highlights the "Definition of the dashboard" text.

The right panel shows the Global Environment and System Library.

Name	Description	Version
abind	Combine Multidimensional Arrays	1.4-5
additivityTests	Additivity Tests in the Two Way Anova with Single Sub-class Numbers	1.1-4
ash	David Scott's ASH Routines	1.0-15
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
asypow	Calculate Power Utilizing Asymptotic Likelihood Ratio Methods	2015.6.25
av	Working with Audio and Video in R	0.5.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.1.4
base64enc	Tools for base64 encoding	0.1-3
BH	Boost C++ Header Files	1.69.0-1
biclust	BiCluster Algorithms	2.0.1
BiocManager	Access the Bioconductor Project Package Repository	1.30.4
BiocStyle	Standard styles for vignettes and other Bioconductor documents	2.12.0
BiocVersion	Set the appropriate version of Bioconductor packages	3.9.0
bitops	Bitwise Operations	1.0-6
BMA	Bayesian Model Averaging	3.18.14

First Tab: Modelling Positive Testing



Rmd File

The screenshot shows the RStudio interface with the following components:

- Code Editor:** Displays an R Markdown script. A red circle highlights line 26: `# Modelling Positive Testing`. A red curly brace on the left side of the editor area groups lines 30 through 40, which are library imports: `library(ggplot2)`, `library(reshape2)`, `library(gam)`, `library(mgcv)`, `library(visreg)`, `library(tidyverse)`, `library(gridExtra)`, `library(mvtnorm)`, `library(ggpubr)`, `library(covid19)`, and `library(ggpmisc)`.
- Console:** Shows the R version information and the workspace message: [workspace loaded from `~/.RData`].
- Environment:** Shows the global environment with various objects like `dat`, `dat1`, `data1`, `datmat`, `dattest`, `idd`, `out`, `outp02`, `outp04`, and `scramdat1`.
- Packages:** Shows the system library with installed packages and their details.

A red curly brace on the right side of the editor area groups lines 30 through 40, which are library imports.

```
26 # Modelling Positive Testing
27
28 ```{r global2, include=FALSE}
29
30 library(ggplot2)
31 library(reshape2)
32 library(gam)
33 library(mgcv)
34 library(visreg)
35 library(tidyverse)
36 library(gridExtra)
37 library(mvtnorm)
38 library(ggpubr)
39 library(covid19)
40 library(ggpmisc)
41
42
43
44 x <- covid19()
45 country_names <- unique(x$administrative_area_level_1)
46
47 ## Daily number of COVID-19 tests and cases in India from 01/04/2020-31/05/2021
48 endDate <- Sys.Date() - 4
1:1 COVID-19 Dashboard India
```

Name	Description	Version
abind	Combine Multidimensional Arrays	1.4-5
additivityTests	Additivity Tests in the Two Way Anova with Single Sub-class Numbers	1.1-4
ash	David Scott's ASH Routines	1.0-15
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
asypow	Calculate Power Utilizing Asymptotic Likelihood Ratio Methods	2015.6.25
av	Working with Audio and Video in R	0.5.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.1.4
base64enc	Tools for base64 encoding	0.1-3
BH	Boost C++ Header Files	1.69.0-1
biclust	BiCluster Algorithms	2.0.1
BiocManager	Access the Bioconductor Project Package Repository	1.30.4
BiocStyle	Standard styles for vignettes and other Bioconductor documents	2.12.0
BiocVersion	Set the appropriate version of Bioconductor packages	3.9.0
bitops	Bitwise Operations	1.0-6
BMA	Bayesian Model Averaging	3.18.14

Rmd File

The screenshot shows the RStudio interface with several windows open:

- Code Editor:** Shows an Rmd file containing code for creating a COVID-19 dashboard for India. The code includes data loading, filtering, and calculating daily tests and cases.
- Environment:** Shows the global environment with objects like `dat`, `dat1`, and `data1`.
- Plots:** Shows a list of available packages in the system library.
- Console:** Displays the R startup message and workspace details.

A red callout box points from the text "New variables and calculations" to the code editor area.

New variables and calculations

```
41
42
43
44 x <- covid19()
45 country_names <- unique(x$administrative_area_level_1)
46
47 ## Daily number of COVID-19 tests and cases in India from 01/04/2020-31/05/2021
48 enddate <- Sys.Date() - 4
49 country1 <- "India"
50 Ind_daily <- covid19(country=country1,start="2020-03-31", end=enddate)
51 Ind_daily <- Ind_daily[,c("administrative_area_level_1", "date", "tests", "confirmed")]
52
53 Ind_daily1 <- Ind_daily[2:nrow(Ind_daily),]
54
55 Ind_daily1$dailytests <- diff(Ind_daily$tests)[1:(nrow(Ind_daily)-1)]
56 Ind_daily1$dailycases <- diff(Ind_daily$confirmed)[1:(nrow(Ind_daily)-1)]
57
58 colnames(Ind_daily1) <- c("country", "date", "tests", "cases", "dailytests", "dailycases")
59
60 Ind_daily1$positive <- Ind_daily1$dailycases / Ind_daily1$dailytests
61
62 Ind_daily1 <- na.omit(Ind_daily1)
63
64 ...
```

R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for statistical computing
Platform: x86_64-w64-mingw32/x64 (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.

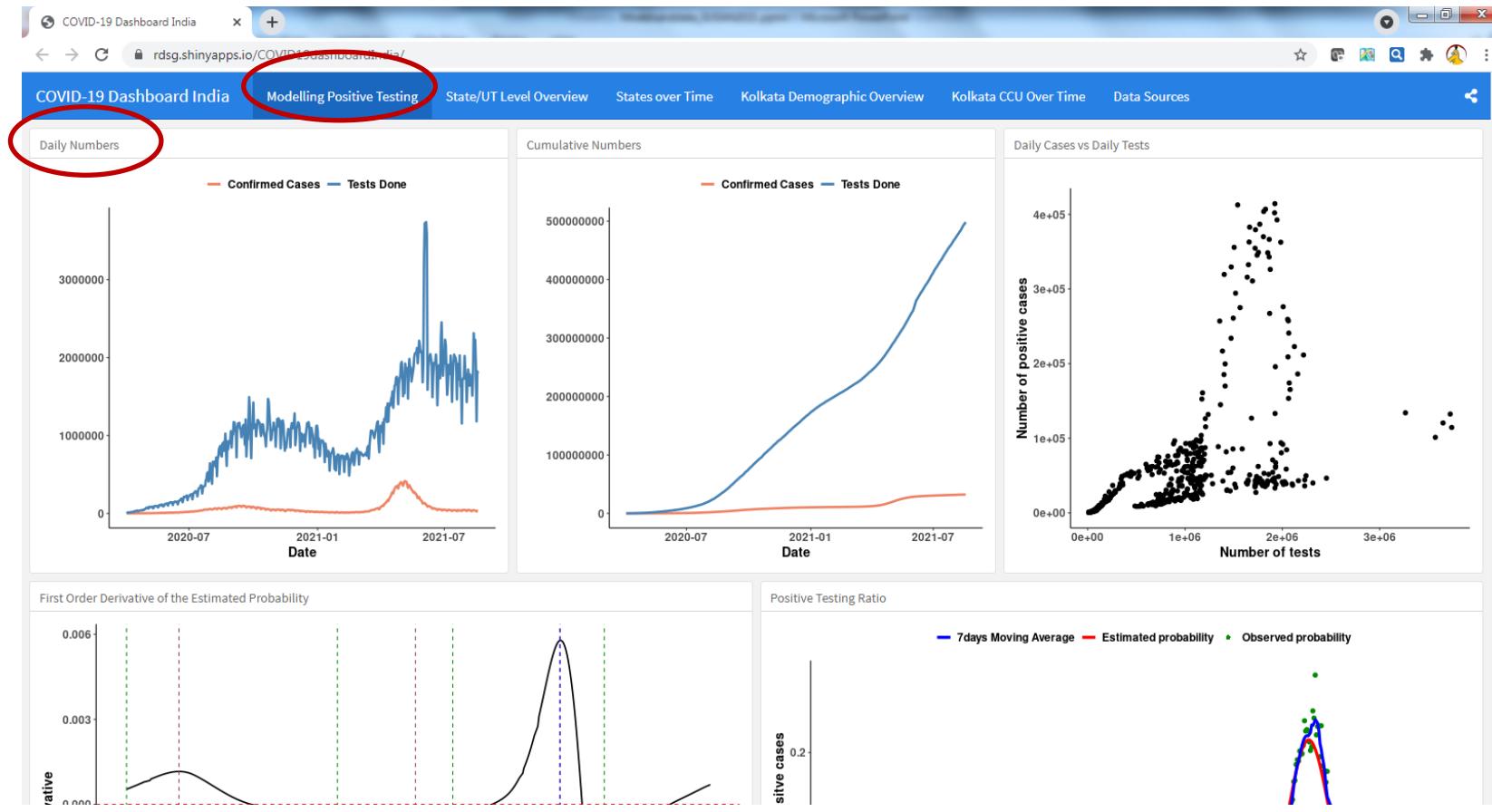
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

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.

[workspace loaded from ~/.RData]

> |

First Box: Daily Numbers



Rmd file

The screenshot shows the RStudio interface with an Rmd file open. The code editor displays the following R code:

```
69
70 ## Daily Numbers
71
72 ```{r}
73
74 colorsdt <- c("dailycases"="salmon2", "dailytests" = "steelblue")
75
76 renderPlot({
77
78   ggplot(ind_daily1) + geom_line(aes(x=as.Date(date, "%y-%m-%d"), y=dailycases, color= "dailycases"),size=1.25) +
79     geom_line(aes(x=as.Date(date, "%y-%m-%d"), y=dailytests, color= "dailytests"),size=1.25) +
80     labs(x = "Date",y = " ") +
81     scale_color_manual(" ", values = colorsdt,
82       labels=c("dailytests"="Tests Done","dailycases" = "Confirmed Cases"),
83       guide = guide_legend(direction="horizontal")) +
84     # scale_x_date(date_breaks = "2 month", date_labels = "%b %Y") +
85     scale_y_continuous(labels = function(x) format(x, scientific = FALSE)) + theme_classic() +
86     theme(legend.title = element_text(size=12),
87       legend.position = "top",
88       legend.text = element_text(size=12,face="bold"),
89       axis.text.x = element_text(size=12,face="bold"),
90       axis.text.y = element_text(size=12,face="bold"),
91       axis.title = element_text(face="bold",size=14))
92
93
94
95
96
97
98
99
100
101
```

A red circle highlights the line `## Daily Numbers`. A red bracket on the left side of the code editor covers the entire block of code from line 70 to line 92. A red box labeled "The plots in the box" is positioned over the plot area in the Rmd preview pane.

The R console output shows the standard R startup message and workspace information:

```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (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.

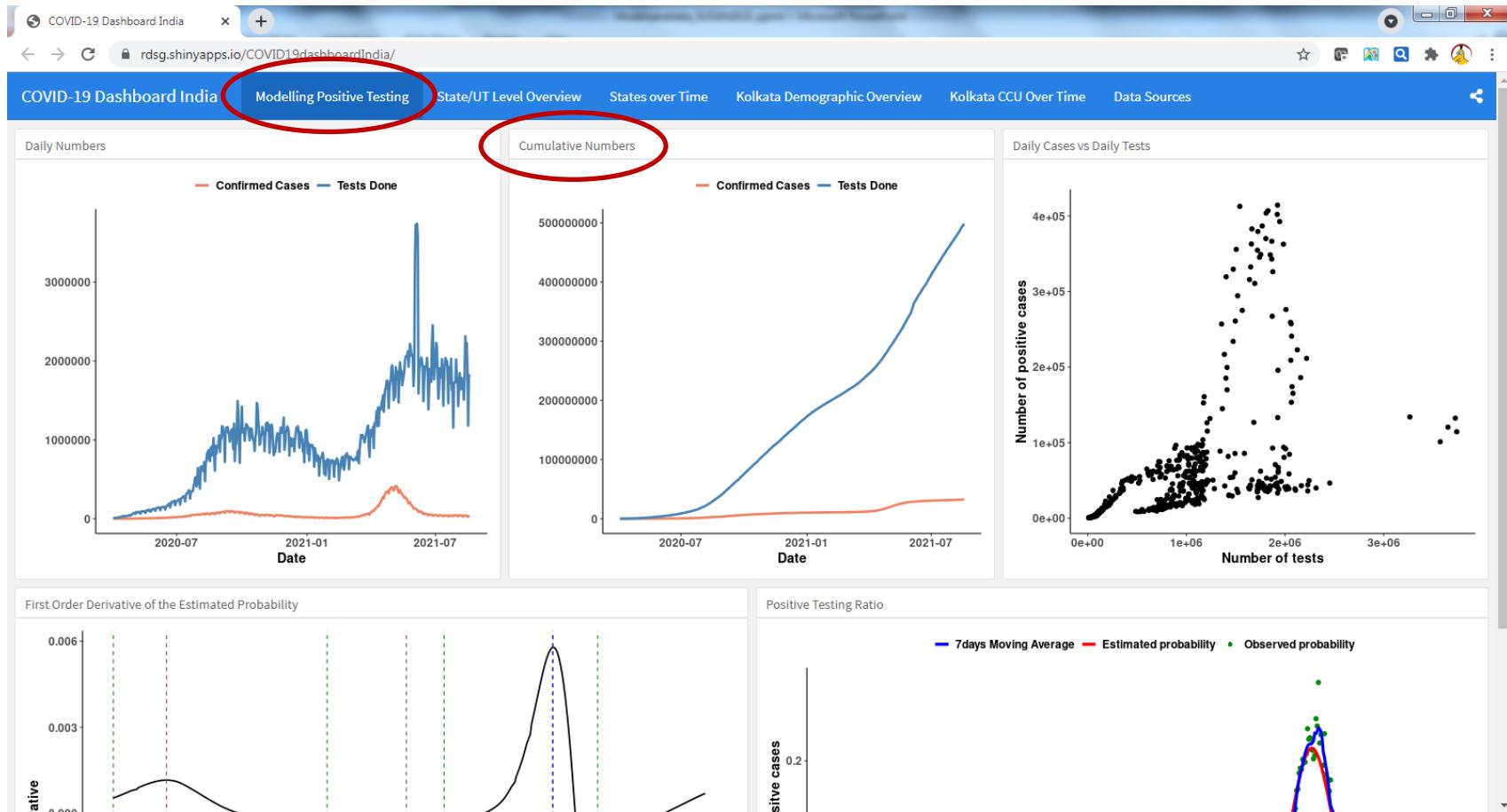
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

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.

[workspace loaded from ~/.RData]
> |
```

The Environment pane on the right lists various objects in the global environment, and the Packages pane shows the installed system library packages.

Second Box: Cumulative Numbers



Rmd File

The screenshot shows the RStudio interface with several tabs open at the top: "13_01_2021_V2(Q2).UH.R...", "Exam_Part2_19_01_2021_Final(Q1B)...", "Rudradev_dashboard.Rmd", "Exam_Part2_19_01_2021_Final(Q1A)...", and "Exam_P...". The main pane displays R code for creating a plot. A red circle highlights the line "### Cumulative Numbers". The code uses ggplot to create a line chart with two series: "cases" (salmon2) and "tests" (steelblue). The x-axis represents dates, and the y-axis represents cumulative numbers. The plot includes a legend, axis titles, and a horizontal guide. The right pane shows the "Environment" tab with various global variables listed, and the "Packages" tab showing the installed system library.

```
13_01_2021_V2(Q2).UH.R... Exam_Part2_19_01_2021_Final(Q1B)... Rudradev_dashboard.Rmd Exam_Part2_19_01_2021_Final(Q1A)... Exam_P...
97
98
99 ## Cumulative Numbers
100
101
102 ```{r}
103
104 colorsC <- c("cases"="salmon2", "tests" ="steelblue")
105
106 renderPlot({
107
108   ggplot(Ind_daily1) + geom_line(aes(x=as.Date(date, "%y-%m-%d"), y=cases, color= "cases"),size=1.25) +
109     geom_line(aes(x=as.Date(date, "%y-%m-%d"), y=tests, color= "tests"),size=1.25) +
110     labs(x = "Date",y = " ") +
111     scale_color_manual(" ", values = colorsC,
112                         labels=c("tests"="Tests Done","cases" = "Confirmed Cases"),
113                         guide = guide_legend(direction="horizontal")) +
114     scale_y_continuous(labels = function(x) format(x, scientific = FALSE)) + theme_classic() +
115     theme(legend.title = element_text(size=12),
116           legend.position = "top",
117           legend.text = element_text(size=12,face="bold"),
118           axis.text.x = element_text(size=12,face="bold"),
119           axis.text.y = element_text(size=12,face="bold"),
120           axis.title = element_text(face="bold",size=14))
121 }
122 COVID-19 Dashboard India
```

Console ~ /

```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for statistical computing
Platform: x86_64-w64-mingw32/x64 (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.

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.

[workspace loaded from ~/.RData]
> |
```

Environment

Name	Description	Version
dat	16999 obs. of 12 variables	
dat1	21 obs. of 2 variables	
data1	12 obs. of 2 variables	
datamat	num [1:100, 1:100] 17.4 17.4 17.4 17.4 17.4 ...	
datatest	num [1:10, 1:10] 24.7 24.4 24.4 21.3 20.8 ...	
idd	int [1:30, 1:30] 1 2 3 4 5 6 7 8 9 10 ...	
out	40001 obs. of 4 variables	
outp02	40001 obs. of 4 variables	
outp04	40001 obs. of 4 variables	
scandat1	16000 obs. of 12 variables	

Packages

Name	Description	Version
abind	Combine Multidimensional Arrays	1.4-5
additivityTests	Additivity Tests in the Two Way Anova with Single Sub-class Numbers	1.1-4
ash	David Scott's ASH Routines	1.0-15
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
asypow	Calculate Power Utilizing Asymptotic Likelihood Ratio Methods	2015.6.25
av	Working with Audio and Video in R	0.5.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.1.4
base64enc	Tools for base64 encoding	0.1-3
BH	Boost C++ Header Files	1.69.0-1
biclust	BiCluster Algorithms	2.0.1
BiocManager	Access the Bioconductor Project Package Repository	1.30.4
BiocStyle	Standard styles for vignettes and other Bioconductor documents	2.12.0
BiocVersion	Set the appropriate version of Bioconductor packages	3.9.0
bitops	Bitwise Operations	1.0-6
BMA	Bayesian Model Averaging	3.18.14

The R function `renderPlot()`

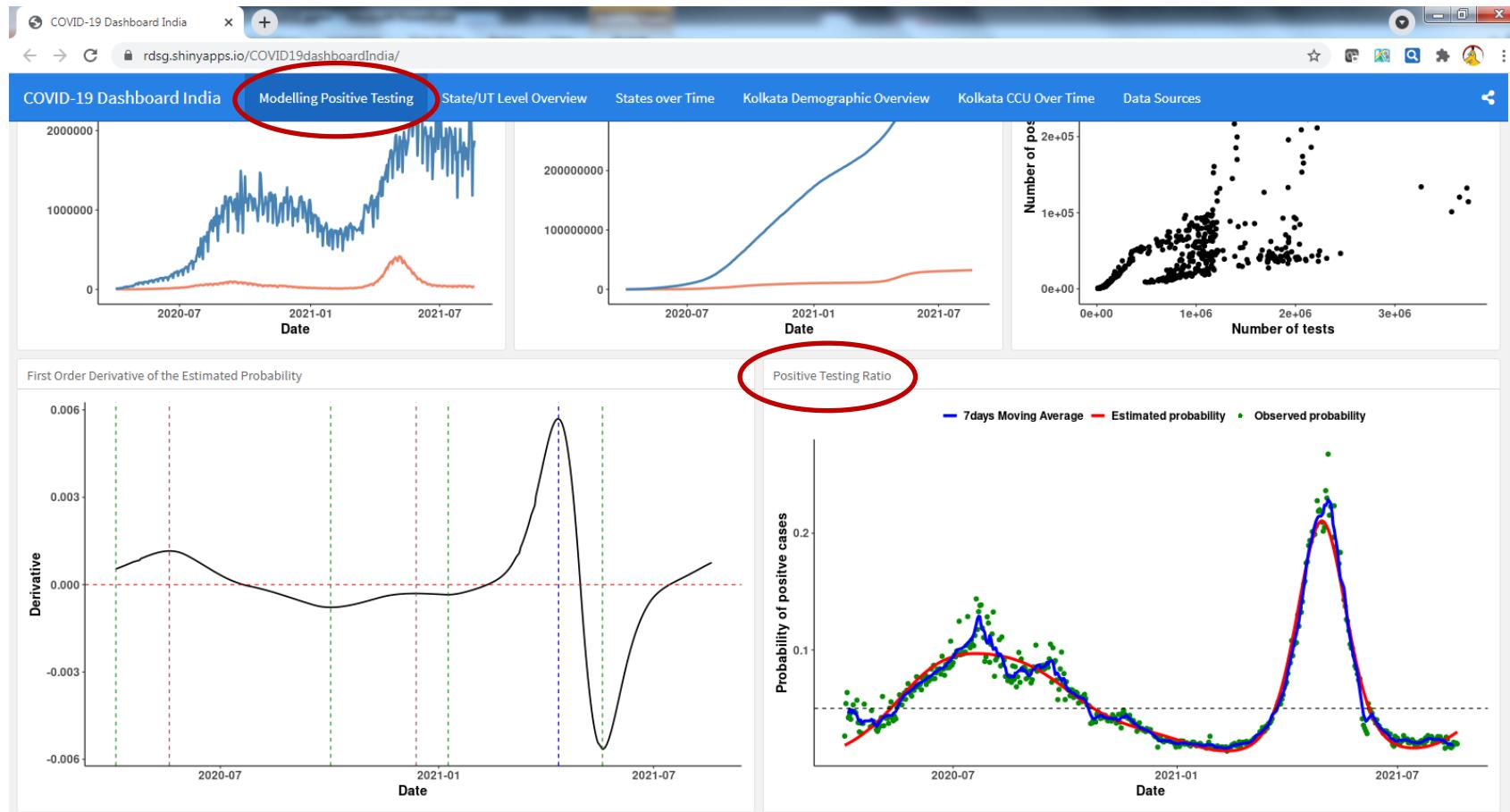
- In each box, the graphical output is defined using the function `renderPlot()`.
- Plots are produced using the `ggplot2` package.
- Flexibility to integrate figures produced using `plotly`, `leaflet`,

renderPlot()

The screenshot shows the RStudio interface with the following components:

- Code Editor:** Displays R code in a script named "Exam_Part2_19_01_2021_Final(Q1B).Rmd". The code uses the `renderPlot` function to create a dual-axis line chart. The chart plots "Tests Done" (blue line) and "Confirmed Cases" (red line) over time. The chart includes a legend at the top, bold axis labels, and a horizontal legend.
- Console:** Shows the R environment information and the R startup message.
- Environment:** Shows the global environment with various objects like `dat`, `dat1`, and `out`.
- Plots:** Shows the rendered dual-axis line chart.
- Packages:** Shows the system library with installed packages such as abind, additivityTests, ash, askpass, assertthat, aspow, av, backports, base64enc, BH, biclust, BiocManager, BiocStyle, BiocVersion, bitops, and BMA.

Last Box: Positive Testing Ratio



Rmd File

The screenshot shows the RStudio interface with an Rmd file open. The code editor contains the following R code:

```
220 axis.text.y = element_text(size=12,face="bold"),
221 axis.title = element_text(face="bold",size=14)
222 })
223 ``
224 ``
225 
226 
227 228 - ## Positive Testing Ratio
229 ``
230 
231 232 ~`{r }
233 234 Ind3_Ind <- Ind_daily1 %>% dplyr::mutate(cases7ma = zoo::rollmean(positive, k = 7, fill = NA))
235 236 jj31_Ind = data.frame(date = Ind_daily1$date,
236 "Observed probability" = Ind_daily1$positive,
236 "Estimated probability" = cases7ma,
236 "7days Moving Average" = Ind3_Ind$cases7ma,
236 check.names = FALSE
237 
238 239 renderPlot([
240 241 renderPlot([
242 
243 ````
```

A red oval highlights the line **228 - ## Positive Testing Ratio**. A red rectangle highlights the line **239 renderPlot([**. A red callout box with the text **For the R code of the plot: click on the function renderplot()** is positioned over the highlighted area.

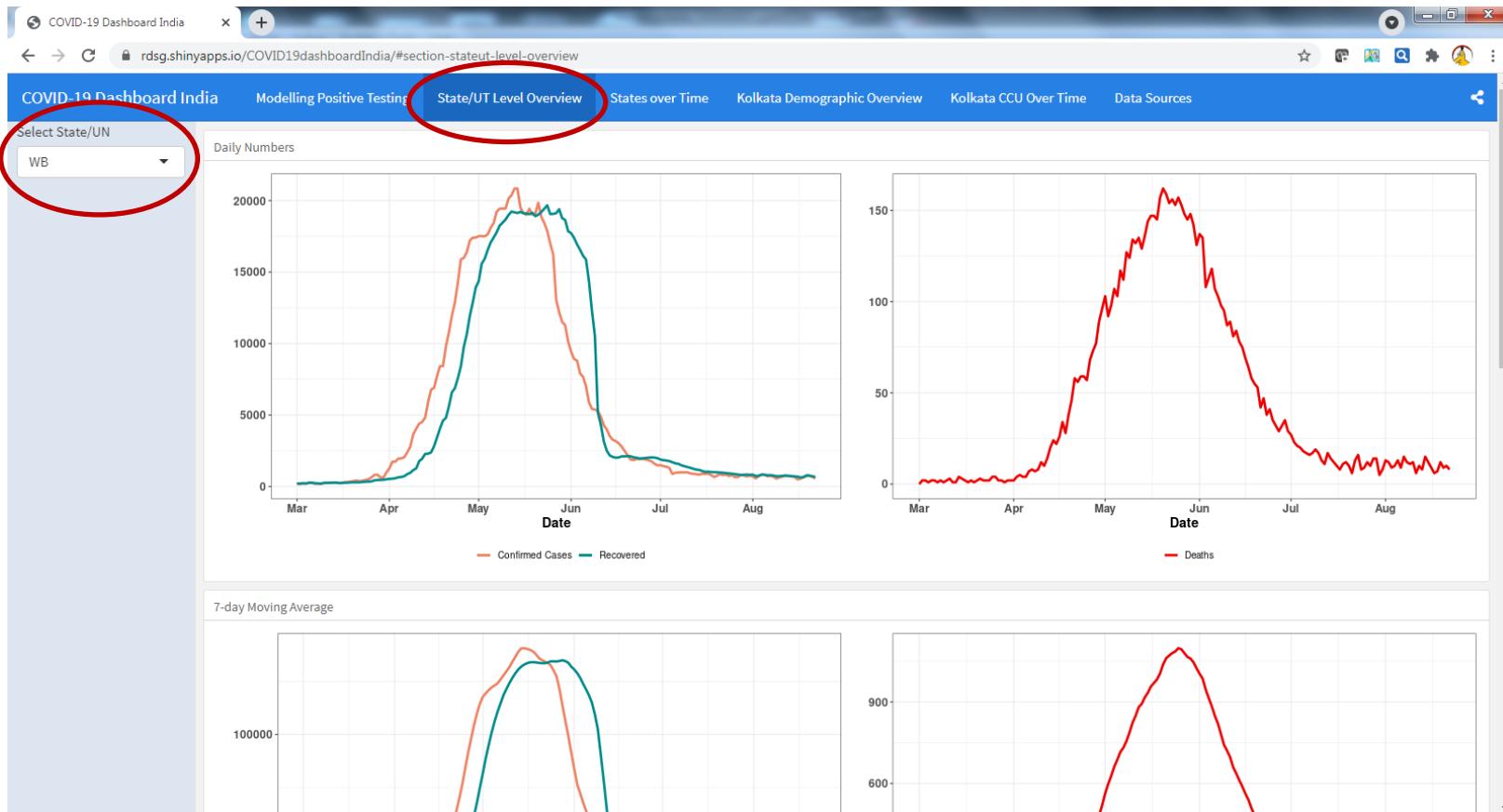
The RStudio environment pane shows the following data objects:

- dat: 16999 obs. of 12 variables
- dat1: 21 obs. of 2 variables
- data1: 12 obs. of 2 variables
- datmat: num [1:100, 1:100] 17.4 17.4 17.4 17.4 ...
- dattest: num [1:10, 1:10] 24.7 24.4 24.4 21.3 20.8 ...
- idd: int [1:30, 1:30] 1 2 3 4 5 6 7 8 9 10 ...
- out: 40001 obs. of 4 variables
- outp02: 40001 obs. of 4 variables
- outp04: 40001 obs. of 4 variables
- scandata1: 16999 obs. of 12 variables

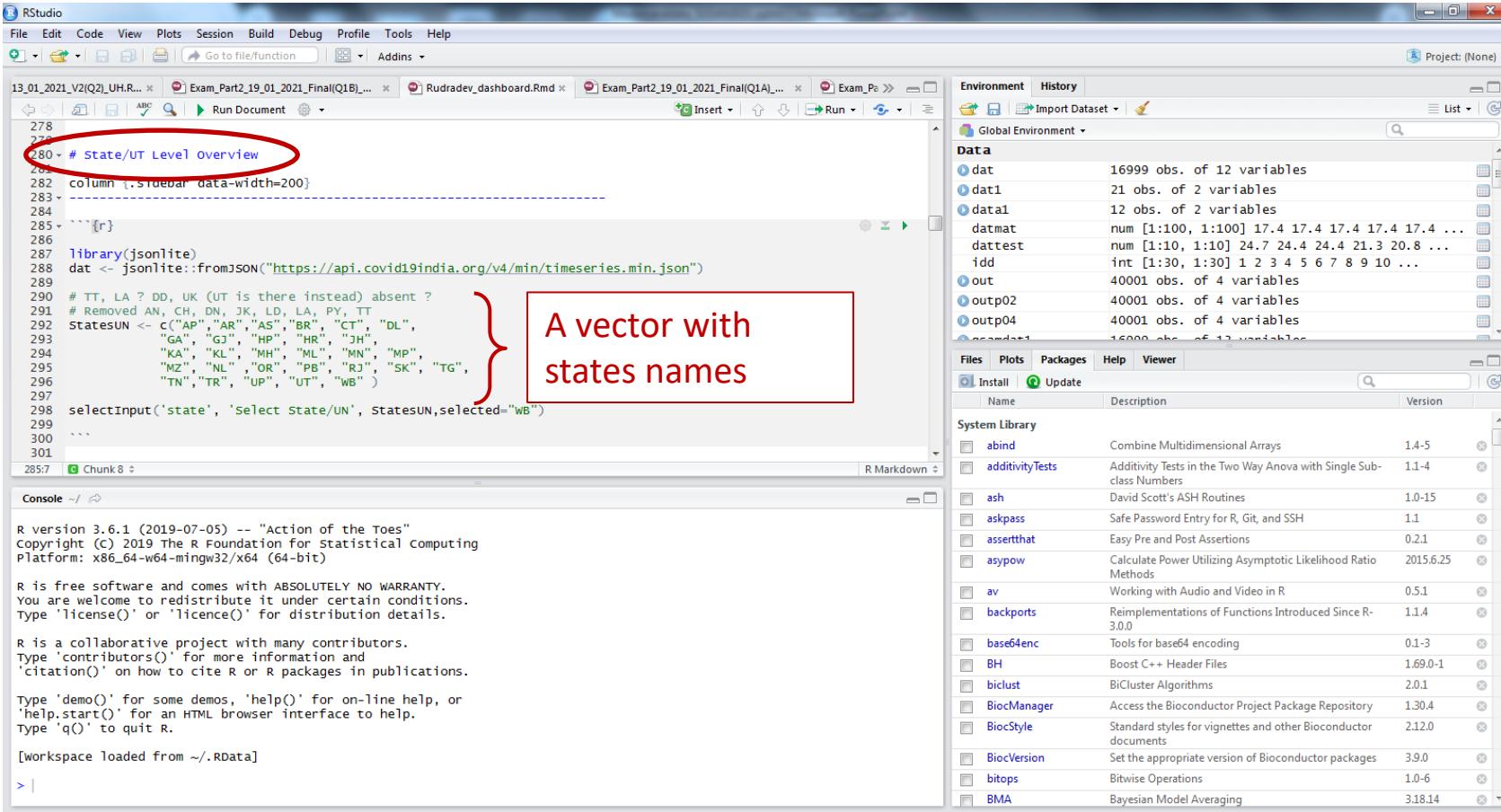
The RStudio help pane shows a list of packages:

- Combine Multidimensional Arrays 1.4-5
- Additivity Tests in the Two Way Anova with Single Sub-class Numbers 1.1-4
- ash David Scott's ASH Routines 1.0-15
- askpass Safe Password Entry for R, Git, and SSH 1.1
- assertthat Easy Pre and Post Assertions 0.2.1
- asypow Calculate Power Utilizing Asymptotic Likelihood Ratio Methods 2015.6.25
- av Working with Audio and Video in R 0.5.1
- backports Reimplementations of Functions Introduced Since R-3.0.0 1.1.4
- base64enc Tools for base64 encoding 0.1-3
- BH Boost C++ Header Files 1.69.0-1
- biclust BiCluster Algorithms 2.0.1
- BiocManager Access the Bioconductor Project Package Repository 1.30.4
- BiocStyle Standard styles for vignettes and other Bioconductor documents 2.12.0
- BiocVersion Set the appropriate version of Bioconductor packages 3.9.0
- bitops Bitwise Operations 1.0-6
- BMA Bayesian Model Averaging 3.18.14

Second page: State/UT Level overview



Rmd File



The screenshot shows an RStudio interface with several tabs open at the top: "13_01_2021_V2(Q2)_UH.R... x", "Exam_Part2_19_01_2021_Final(Q1B)... x", "Rudradev_dashboard.Rmd x", "Exam_Part2_19_01_2021_Final(Q1A)... x", and "Exam_Pa... x". The main pane displays R code:

```
278  
279  
280 # state/UT Level overview  
281 column (.sticobar data-width=200)  
282  
283  
284  
285 ````{r}  
286  
287 library(jsonlite)  
288 dat <- jsonlite::fromJSON("https://api.covid19india.org/v4/min/timeseries.min.json")  
289  
290 # TT, LA ? DD, UK (UT is there instead) absent ?  
291 # Removed AN, CH, DN, JK, LD, LA, PY, TT  
292 StatesUN <- c("AP", "AR", "AS", "BR", "CT", "DL",  
293 "GA", "GJ", "HP", "HR", "JH",  
294 "KA", "KL", "MH", "ML", "MN", "MP",  
295 "M2", "NL", "OR", "PB", "RJ", "SK", "TG",  
296 "TN", "TR", "UP", "UT", "WB")  
297  
298 selectInput('state', 'Select State/UN', StatesUN, selected="WB")  
299  
300  
301
```

A red callout box with the text "A vector with states names" points to the line of code where the vector is defined.

The right sidebar shows the "Environment" tab with a list of objects:

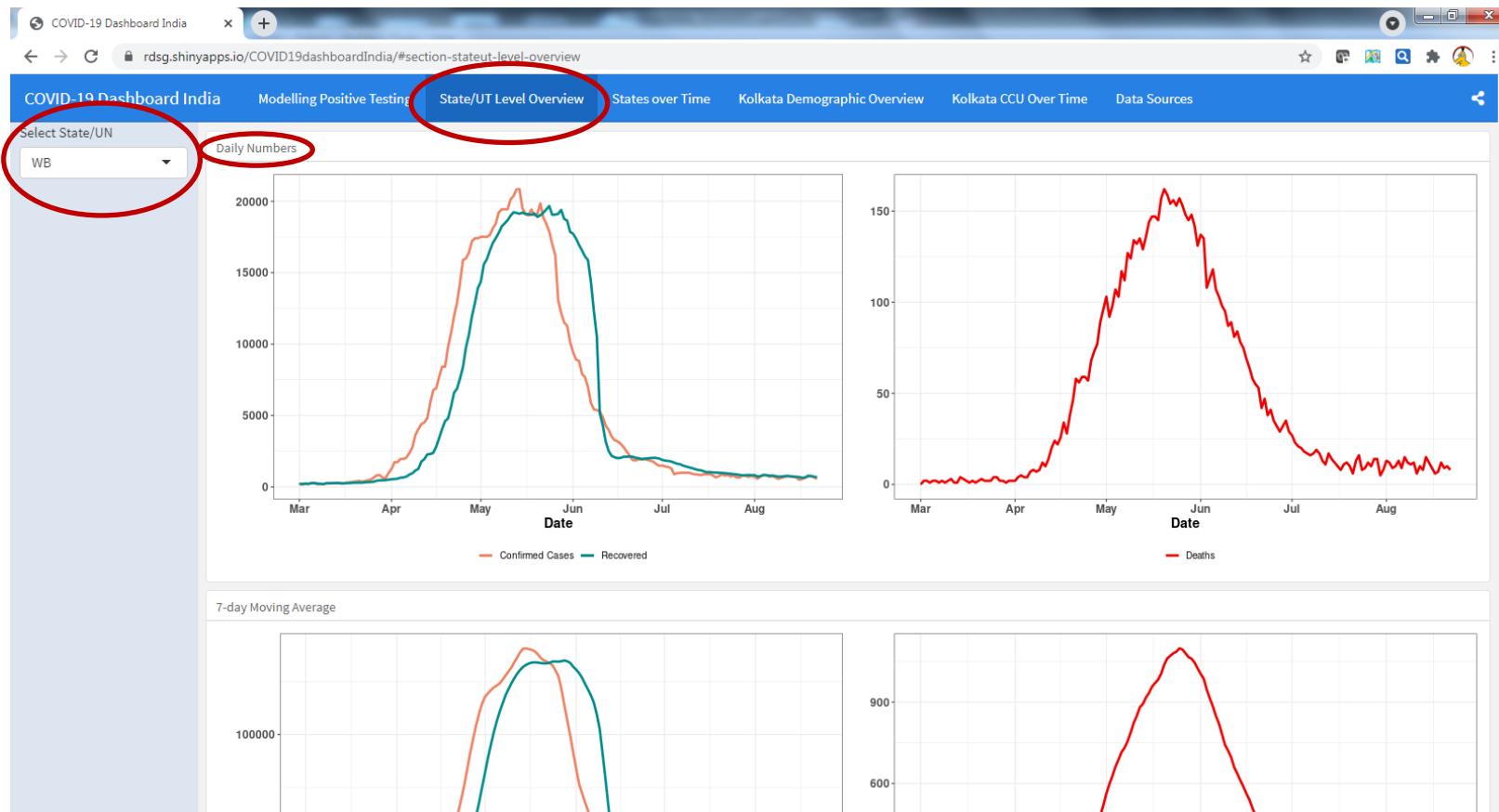
Name	Description
dat	16999 obs. of 12 variables
dat1	21 obs. of 2 variables
data1	12 obs. of 2 variables
datmat	num [1:100, 1:100] 17.4 17.4 17.4 17.4 17.4 ...
datetest	num [1:10, 1:10] 24.7 24.4 24.4 21.3 20.8 ...
idd	int [1:30, 1:30] 1 2 3 4 5 6 7 8 9 10 ...
out	40001 obs. of 4 variables
outp02	40001 obs. of 4 variables
outp04	40001 obs. of 4 variables
scandat1	16999 obs. of 12 variables

The "System Library" section lists various R packages:

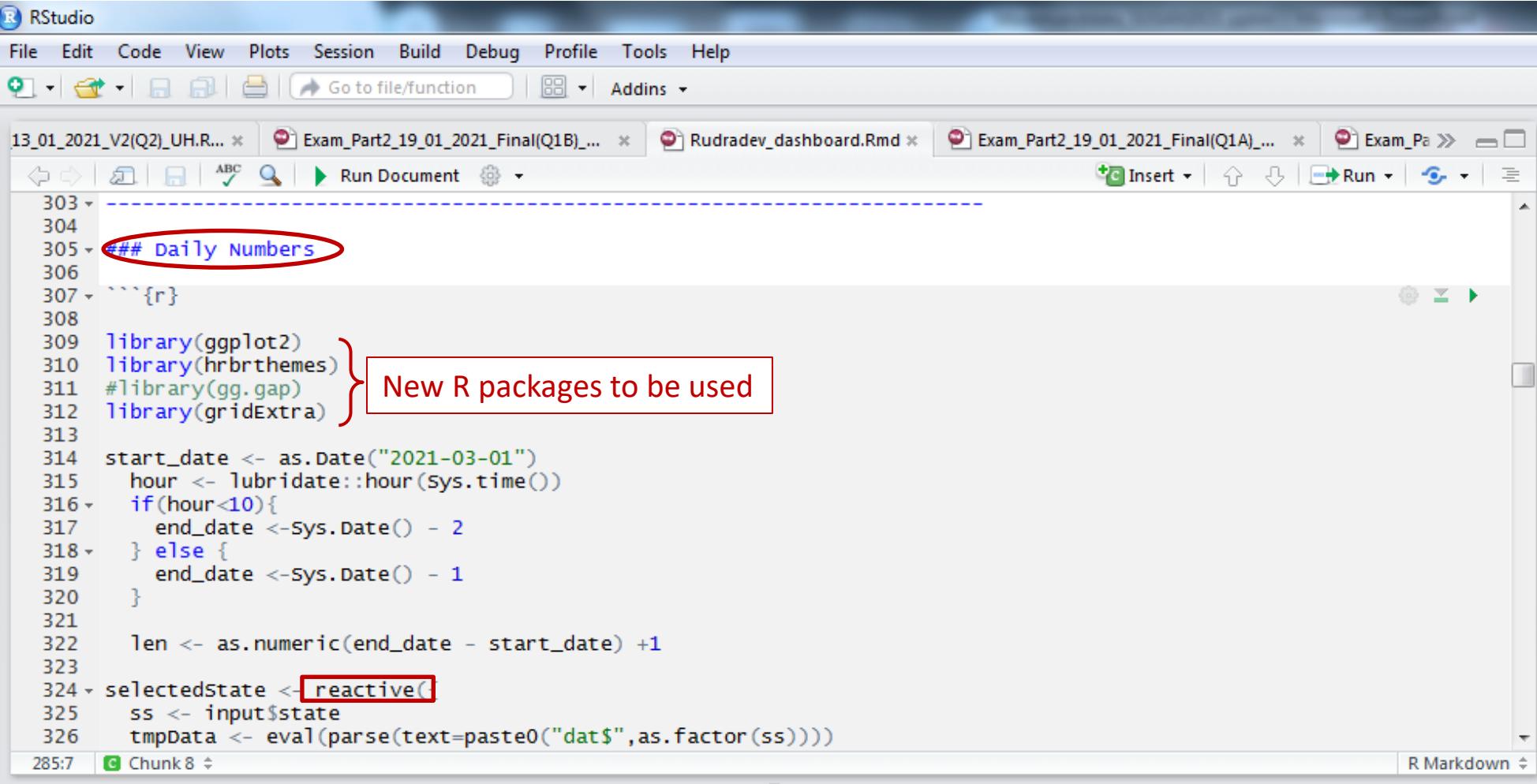
Name	Description	Version
abind	Combine Multidimensional Arrays	1.4-5
additivityTests	Additivity Tests in the Two Way Anova with Single Sub-class Numbers	1.1-4
ash	David Scott's ASH Routines	1.0-15
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
asympow	Calculate Power Utilizing Asymptotic Likelihood Ratio Methods	2015.6.25
av	Working with Audio and Video in R	0.5.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.1.4
base64enc	Tools for base64 encoding	0.1-3
BH	Boost C++ Header Files	1.69.0-1
biclust	BiCluster Algorithms	2.0.1
BiocManager	Access the Bioconductor Project Package Repository	1.30.4
BiocStyle	Standard styles for vignettes and other Bioconductor documents	2.12.0
BiocVersion	Set the appropriate version of Bioconductor packages	3.9.0
bitops	Bitwise Operations	1.0-6
BMA	Bayesian Model Averaging	3.18.14

The "Console" pane shows the R startup message and the command "R workspace loaded from ~/.RData".

First Box: Daily Numbers



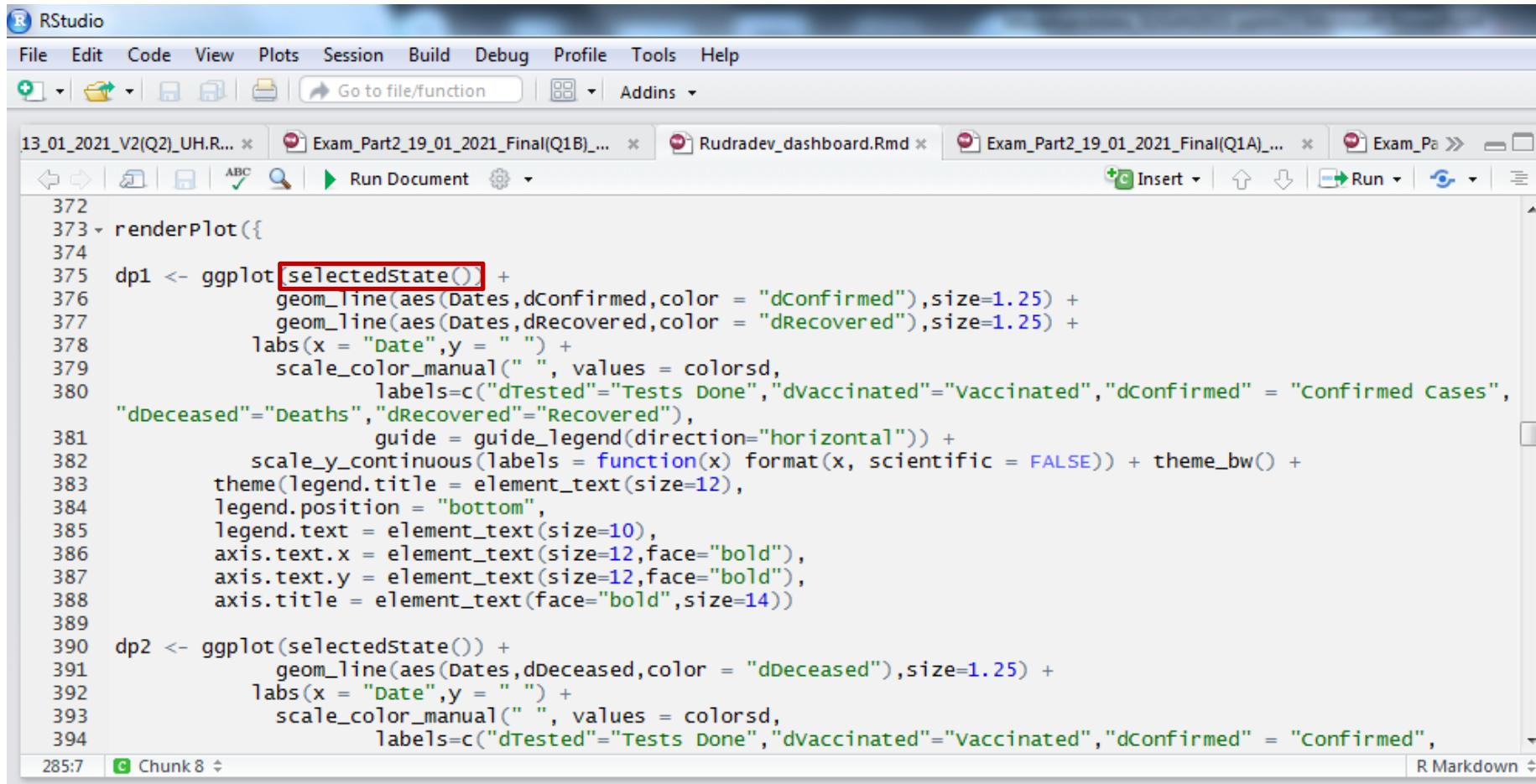
Rmd File



The screenshot shows the RStudio interface with several tabs open at the top: "13_01_2021_V2(Q2)_UH.R...", "Exam_Part2_19_01_2021_Final(Q1B)....", "Rudradev_dashboard.Rmd", "Exam_Part2_19_01_2021_Final(Q1A)....", and "Exam_Pa...". The main editor area displays R code. A red oval highlights the line "### Daily Numbers". A red callout box labeled "New R packages to be used" points to the lines "library(ggplot2)", "library(hrbrthemes)", "#library(gg.gap)", and "library(gridExtra)". Another red box highlights the word "reactive" in the line "selectedstate <- reactive()". The status bar at the bottom shows "285:7" and "Chunk 8".

```
303
304
305 ### Daily Numbers
306
307 ``{r}
308
309 library(ggplot2)
310 library(hrbrthemes)
311 #library(gg.gap)
312 library(gridExtra)
313
314 start_date <- as.Date("2021-03-01")
315 hour <- lubridate::hour(sys.time())
316 if(hour<10){
317   end_date <- Sys.Date() - 2
318 } else {
319   end_date <- Sys.Date() - 1
320 }
321
322 len <- as.numeric(end_date - start_date) +1
323
324 selectedstate <- reactive()
325 ss <- input$state
326 tmpData <- eval(parse(text=paste0("dat$",as.factor(ss))))
```

Rmd File



The screenshot shows the RStudio interface with several tabs open at the top: "13_01_2021_V2(Q2)_UH.R...", "Exam_Part2_19_01_2021_Final(Q1B)...", "Rudradev_dashboard.Rmd", "Exam_Part2_19_01_2021_Final(Q1A)...", and "Exam_Pa...". The main editor area contains R code for generating two plots, dp1 and dp2, using ggplot. The code uses the selectedState() function and various ggplot2 functions like geom_line, labs, scale_color_manual, and theme_bw. The "selectedState()" function is highlighted with a red box. The code is numbered from 372 to 394.

```
372
373 -> renderPlot({
374
375   dp1 <- ggplot(selectedState()) +
376     geom_line(aes(Dates,dConfirmed,color = "dConfirmed"),size=1.25) +
377     geom_line(aes(Dates,dRecovered,color = "dRecovered"),size=1.25) +
378     labs(x = "Date",y = " ") +
379     scale_color_manual("", values = colorsd,
380       labels=c("dTested"="Tests Done","dVaccinated"="Vaccinated","dConfirmed" = "Confirmed Cases",
381       "dDeceased"="Deaths","dRecovered"="Recovered"),
382       guide = guide_legend(direction="horizontal")) +
383       scale_y_continuous(labels = function(x) format(x, scientific = FALSE)) + theme_bw() +
384       theme(legend.title = element_text(size=12),
385         legend.position = "bottom",
386         legend.text = element_text(size=10),
387         axis.text.x = element_text(size=12,face="bold"),
388         axis.text.y = element_text(size=12,face="bold"),
389         axis.title = element_text(face="bold",size=14))
390
391   dp2 <- ggplot(selectedState()) +
392     geom_line(aes(Dates,dDeceased,color = "dDeceased"),size=1.25) +
393     labs(x = "Date",y = " ") +
394     scale_color_manual("", values = colorsd,
395       labels=c("dTested"="Tests Done","dVaccinated"="Vaccinated","dConfirmed" = "Confirmed",
```

Shiny: Mini Introduction

Shiny Environment



What?

- R package by Rstudio: `install.packages("shiny")`
- Web application framework to turn your R analysis in a fully interactive app

Why?

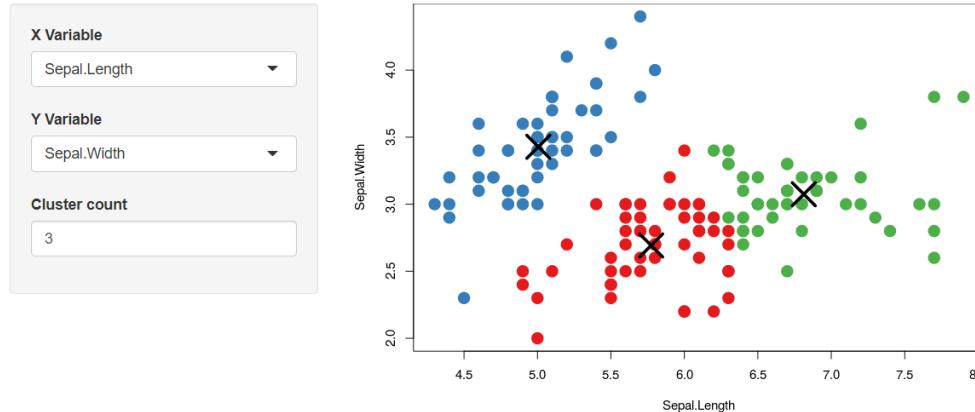
- Fully interactive environment that can be used by non-R users, customizable widgets
- Provide analysis from start to finish: Upload your data, Tweak the analysis parameters, View plots/tables/results, Download a generated report
- ...
- It's pretty cool

Shiny vs Shinydashboard

Shiny

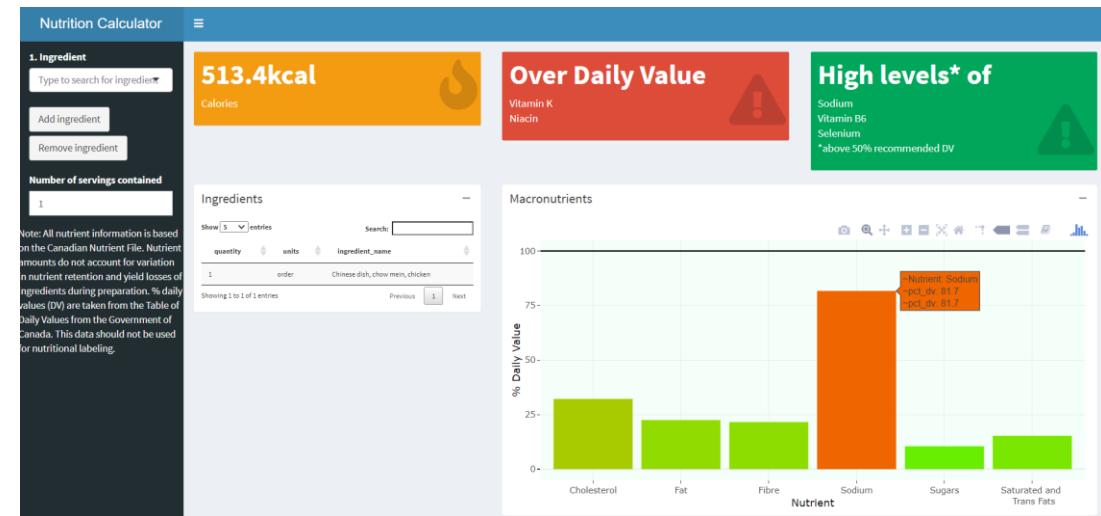
<https://shiny.rstudio.com/>

Iris k-means clustering



ShinyDashboard

<https://rstudio.github.io/shinydashboard>



- Full flexibility for designing your app
- Many extensions
- More effort to create something modern- and good-looking

- Dashboard 'template' for Shiny
- Lose flexibility
- Higher simplicity and easier to create a good-looking app with boxes, aligned plots,...
- Built-in flexibility for a log-in feature using Rstudio Pro

Shiny: The Basics

Shiny
Contains two .R
scripts

ui.R

- Controls user interface
- Designs the layout of the app
 - Where do the plots go?
 - Where do the widgets (buttons, sliders, etc.) go?
- Determines how to app *looks*.

server.R

- Contains R code that runs in the background as the user interacts with the app
 - Code that recognises what the user inputs in the app
 - Code to run an analysis
 - Code to create a plot
 - ...
- Determines how the app *works*.

Shiny: The Basics

Shiny (Alternative Structure)

Contains 1 script



app.R

- Contains both **ui** and **server** parts
- *App.R*
 - `ui <- fluidPage(`

...

)

```
server <- function(input, output){
```

...

}

```
shinyApp(ui = ui, server = server)
```

Notes:

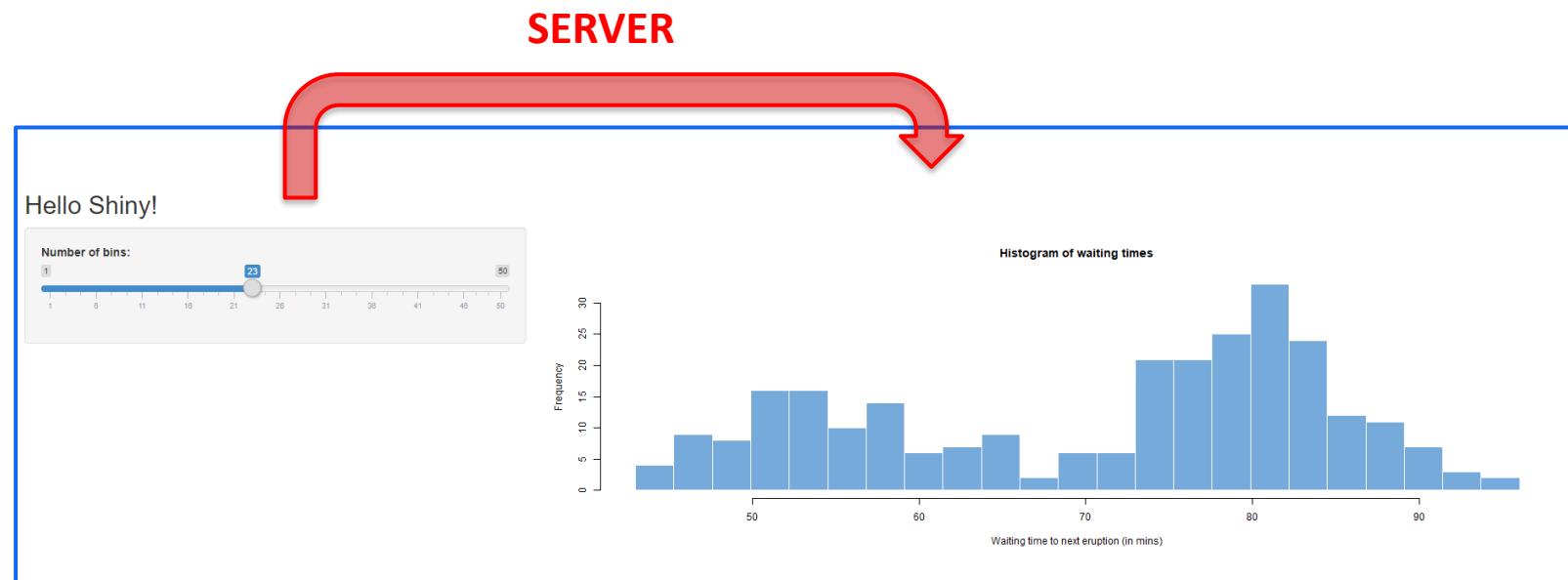
- Keep the exact names of **ui.R**, **server.R**, **app.R**. **Do not changes these!**

Shiny: Basic Example

Hello Shiny Example

- Open Shiny in R/Rstudio
- Run:

```
> library(shiny)  
> runExample("01_hello")
```



- Similar example separated over 2 files:

<https://shiny.rstudio.com/gallery/example-01-hello.html>

- When to separate?
 - Larger apps
 - Keeping your files more structured

User Interface (UI)

Shiny: ui.R

- Essentially a **HTML file**
 - Could write this script completely in HTML code
 - **shiny** provides R functions that *wrap* this html code
- Html tags are used to build UI
 - `names(tags)` : shows all available tags
- Most commonly used tags (like **h1**, **p**, **a**, etc.) have wrapper functions, no need to prefix their names with **tags\$**.

```
# Define UI for app that draws a histogram ----  
ui <- fluidPage(
```

Function Name	Equivalent HTML tag	Provides utility to create
h1	<h1>	Sub-heading of type 1
h2	<h2>	Sub-heading of type 2
h3	<h3>	Sub-heading of type 3
h4	<h4>	Sub-heading of type 4
h5	<h5>	Sub-heading of type 5
h6	<h6>	Sub-heading of type 6
br	 	breaks the line
em		italicizes the text
strong		bolds the text
code	<code>	content written in coded style
p	<p>	paragraph
a	<a>	hyperlink
img		image
div	<div>	A new line of different styles.
span		Text in same line but of different styles.

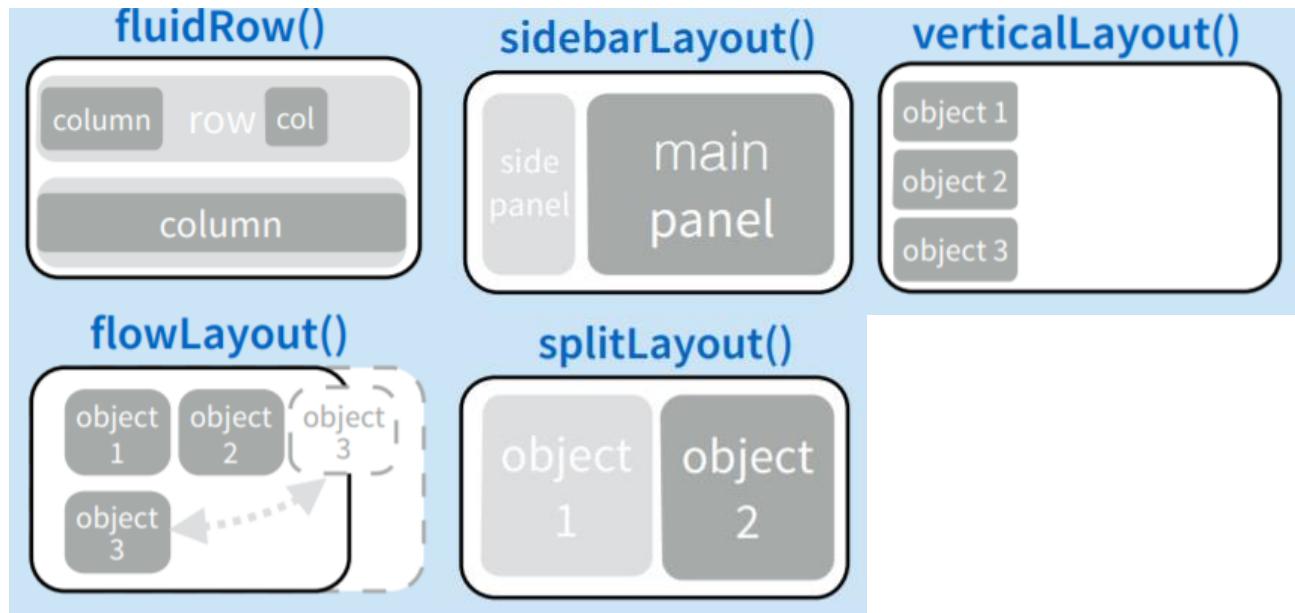
```
##  
# Main panel for displaying outputs ----  
mainPanel(
```

```
    # Output: Histogram ----  
    plotOutput(outputId = "distPlot")
```

```
)  
)  
)
```

Shiny: UI Layout

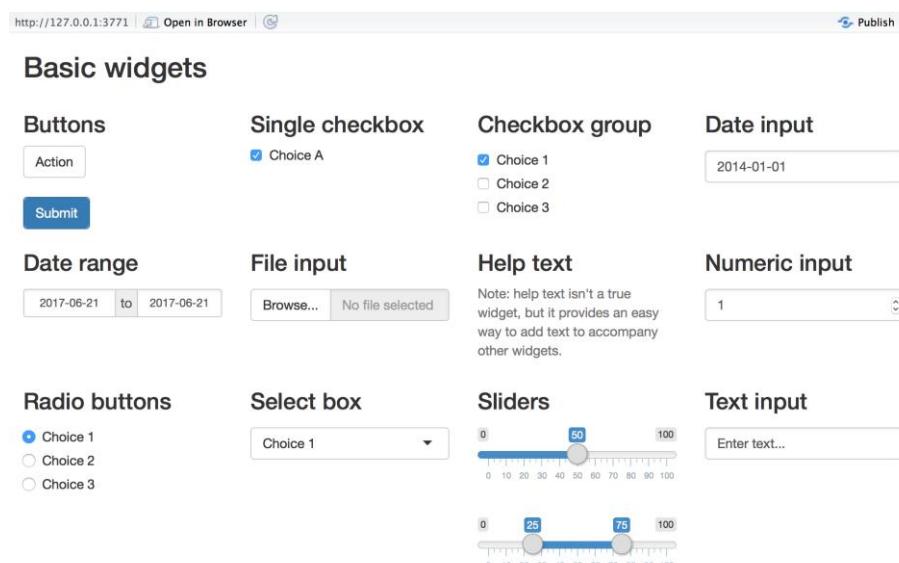
- Layout functions are used to organize panels and elements into an existing layout:
 - `sidebarLayout()` creates a layout with a sidebar and main area -one of the most commonly used besides `fluidRow()`, `flowLayout()`, etc.



```
# Define UI for app that draws a histogram ----  
ui <- fluidPage(  
  
  # App title ----  
  titlePanel("Hello Shiny!"),  
  
  # Sidebar Layout with input and output definitions ----  
  sidebarLayout(  
    # Sidebar panel for inputs ----  
    sidebarPanel(  
  
      # Input: Slider for the number of bins ----  
      sliderInput(inputId = "bins",  
                  label = "Number of bins:",  
                  min = 1,  
                  max = 50,  
                  value = 30)  
  
    ),  
  
    # Main panel for displaying outputs ----  
    mainPanel(  
  
      # Output: Histogram ----  
      plotOutput(outputId = "distPlot")  
  
    )  
  )
```

Shiny: UI Inputs/Widgets

- Widgets are web elements that users can interact with
 - Provide a way for the users to send messages to the Shiny app.
 - Collect values from the user
 - First 2 arguments: **inputID** (name) and **label**



```
# Define UI for app that draws a histogram ----
ui <- fluidPage(


  # App title ----
  titlePanel("Hello Shiny!"),

  # Sidebar layout with input and output definitions ----
  sidebarLayout(


    # Sidebar panel for inputs ----
    sidebarPanel(


      # Input: Slider for the number of bins ----
      sliderInput(inputId = "bins",
                  label = "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30
    ),


    # Main panel for displaying outputs ----
    mainPanel(


      # Output: Histogram ----
      plotOutput(outputId = "distPlot")
    )
  )
)
```

Shiny: Interactive Output

```
# Define UI for app that draws a histogram ----  
ui <- fluidPage(  
  
  # App title ----  
  titlePanel("Hello Shiny!"),  
  
  # Sidebar layout with input and output definitions ----  
  sidebarLayout(  
  
    # Sidebar panel for inputs ----  
    sidebarPanel(  
  
      # Input: Slider for the number of bins ----  
      sliderInput(inputId = "bins",  
                  label = "Number of bins:",  
                  min = 1,  
                  max = 50,  
                  value = 30)  
  
    ),  
  
    # Main panel for displaying outputs ----  
    mainPanel(  
  
      # Output: Histogram ----  
      plotOutput(outputId = "distPlot")  
    )  
)
```

In the server, we decide what to render in the output

We put a plot output in the main panel

```
# Define server logic required to draw a histogram ----  
server <- function(input, output) {  
  
  # Histogram of the Old Faithful Geyser Data ----  
  # with requested number of bins  
  # This expression that generates a histogram is wrapped in a call  
  # to renderPlot to indicate that:  
  #  
  # 1. It is "reactive" and therefore should be automatically  
  # re-executed when inputs (input$bins) change  
  # 2. Its output type is a plot  
  output$distPlot <- renderPlot({  
    x   <- faithful$waiting  
    bins <- seq(min(x), max(x), length.out = input$bins + 1)  
  
    hist(x, breaks = bins, col = "#75AADB", border = "white",  
          xlab = "Waiting time to next eruption (in mins)",  
          main = "Histogram of waiting times")  
  })  
}
```

Plot re-renders each time this input is updated ('Reactivity')

This code is re-run each time the reactive input changes

Shiny: Interactive Output

- Output (e.g. plot) gets updated automatically when an input widget changes.
- Example app:
 - Slider input in the sidebar panel controls the number of bins
 - A plot output in the main panel renders the histogram
- Required steps:
 - Register the output object in ui.R and specify where to put the output plot.
 - server.R tells Shiny how to render the output. (a plot in this example)

Shiny: Interactive Output

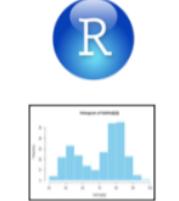
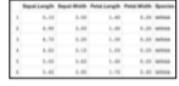
- Different pairs of **Output** and **Render** functions are available.
- Used to add R output to the UI

Output function	render function	creates
htmlOutput/uiOutput	renderUI	a Shiny tag object or HTML
imageOutput	renderImage	images (saved as a link to a source file)
plotOutput	renderPlot	plots
tableOutput	renderTable	data frame, matrix, other table like structures
textOutput	renderText	character strings
verbatimTextOutput	renderPrint	any printed output

Shiny: Interactive Output

- Different pairs of **Output** and **Render** functions are available.
- Used to add R output to the UI

Outputs - render*() and *Output() functions work together to add R output to the UI

	<code>DT::renderDataTable(expr, options, callback, escape, env, quoted)</code>		<code>dataTableOutput(outputId, icon, ...)</code>
	<code>renderImage(expr, env, quoted, deleteFile)</code>		<code>imageOutput(outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline)</code>
	<code>renderPlot(expr, width, height, res, ..., env, quoted, func)</code>		<code>plotOutput(outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline)</code>
	<code>renderPrint(expr, env, quoted, func, width)</code>		<code>verbatimTextOutput(outputId)</code>
	<code>renderTable(expr, ..., env, quoted, func)</code>		<code>tableOutput(outputId)</code>
foo	<code>renderText(expr, env, quoted, func)</code>		<code>textOutput(outputId, container, inline)</code>
	<code>renderUI(expr, env, quoted, func)</code>		<code>uiOutput(outputId, inline, container, ...)</code> & <code>htmlOutput(outputId, inline, container, ...)</code>

Shiny: Where to start?

- Many more important concepts!
 - **Reactivity, extensions,...**
- **Shiny Starting Points**
 - Shiny portal site:
 - [Tutorial \(get started\)](#) (Written tutorials: [starting point](#))
 - [Articles \(go deeper\)](#)
 - [Cheat Sheet](#)
 - [Shiny User Showcase:](#)
 - Shiny Apps for the Enterprise
 - Industry Specific Shiny Apps
 - Shiny Apps as Analytics Tools
 - Shiny Apps that Extend Shiny
 - Shiny Apps with Popular Appeal
 - Shiny Apps for Teaching
 - [Shiny Examples](#)

Discussion

- Dashboard with R in the background
 - easy to implement different features
 - possible to automate the entire workflow
 - easy to integrate statistical models
 - visually appealing
 - interactive

References / Source Materials

- **Rmarkdown**
 - <https://rmarkdown.rstudio.com/>
 - <https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf>
- **Flexdashboard**
 - <https://rmarkdown.rstudio.com/flexdashboard/>
 - <https://rmarkdown.rstudio.com/flexdashboard/using.html>
 - <https://rmarkdown.rstudio.com/flexdashboard/examples.html>
- **Htmlwidgets**
 - <http://www.htmlwidgets.org/>
- **CrossTalk**
 - <https://rstudio.github.io/crosstalk/>
- **Shiny**
 - <https://shiny.rstudio.com/>
 - <https://rstudio.github.io/shinydashboard/>
 - Shiny Tutorial Start: <https://shiny.rstudio.com/tutorial/written-tutorial/lesson1/>

Contact us

- Rudradev Sengupta: rsengup4@its.jnj.com
- Ziv Shkedy: ziv.shkedy@uhasselt.be