Introducing Shiny

Jonas Schöley

jschoeley@health.sdu.dk

Max-Planck Odense Center on the Biodemography of Aging

Department of Public Health
University of Southern Denmark

What’s “Shiny”?

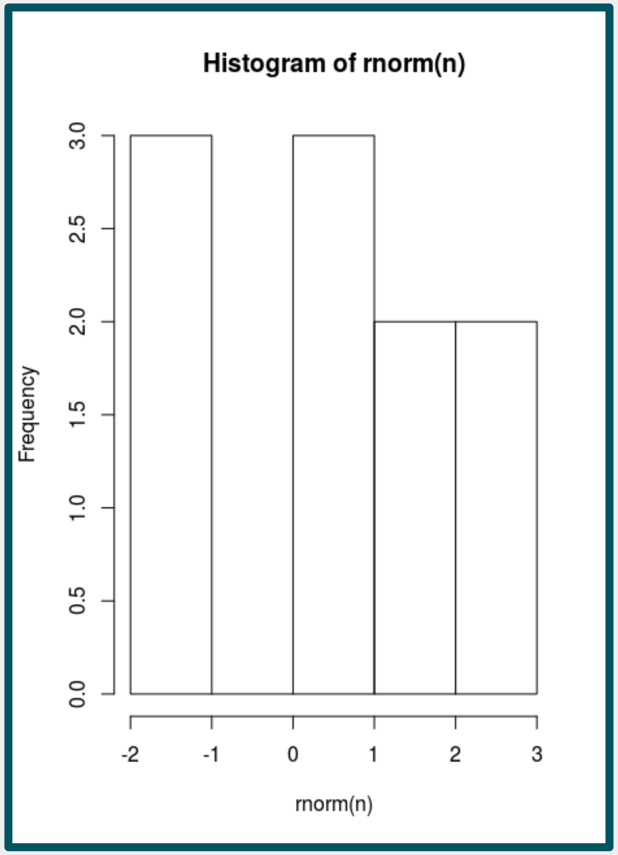
1 A graphical user-interface for your R program
2 Your R program as a web-page
3 A web-application framework for R

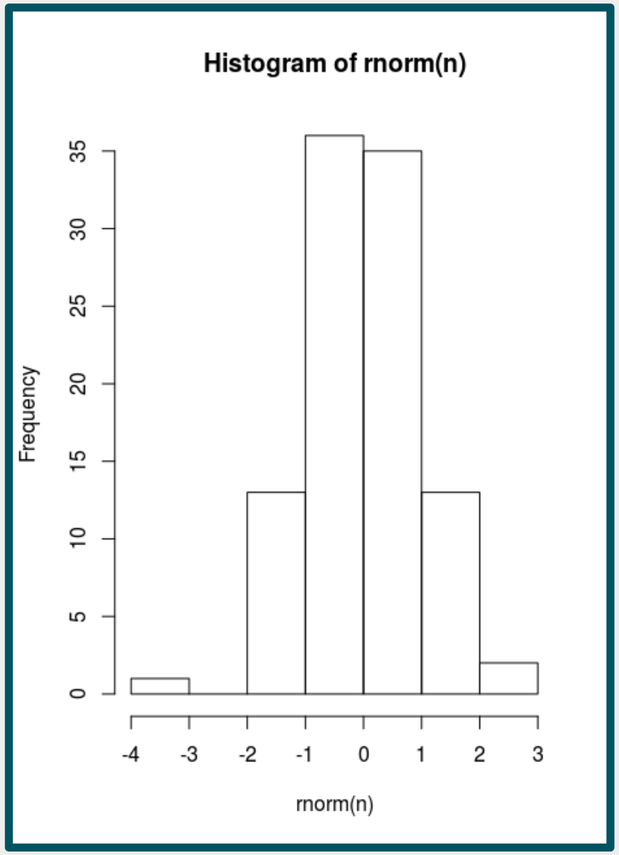
A graphical user-interface for your R program

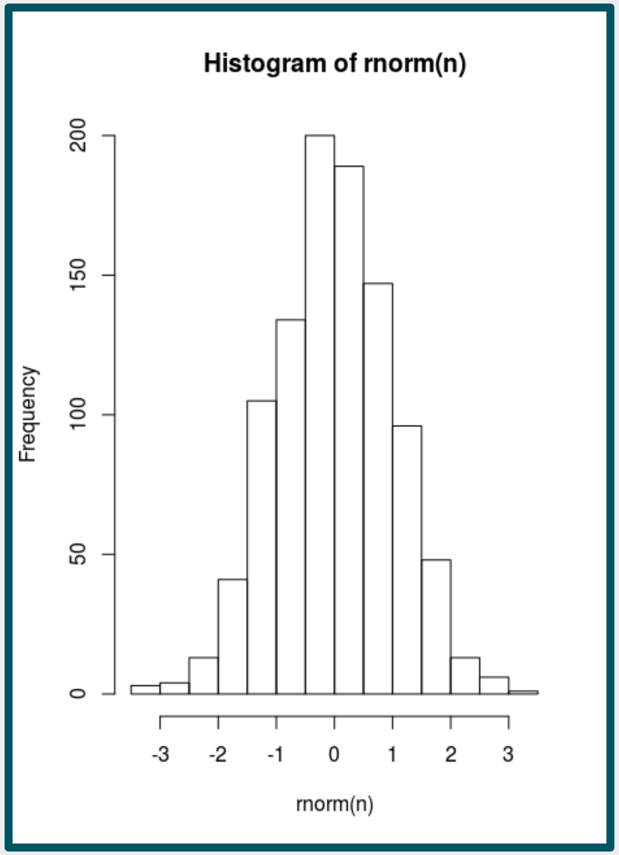
```
n = 10
hist(rnorm(n))
```

```
n = 100
hist(rnorm(n))
```

```
n = 1000
hist(rnorm(n))
```

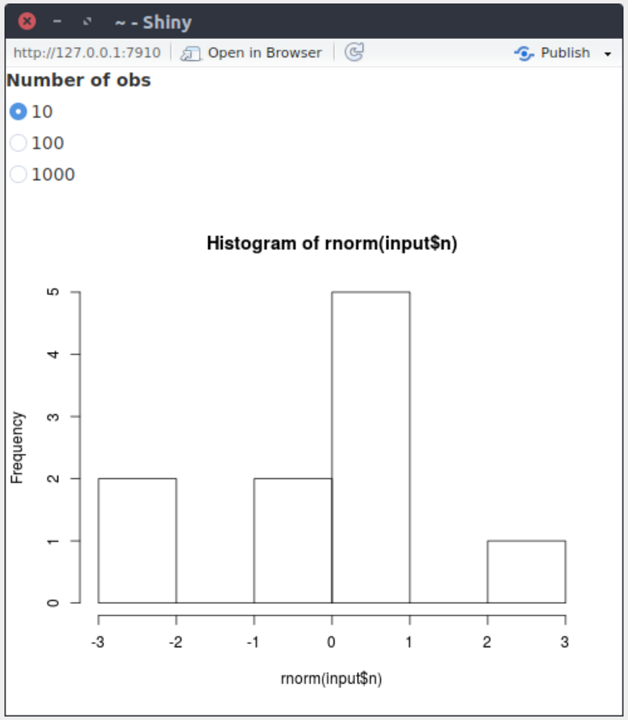






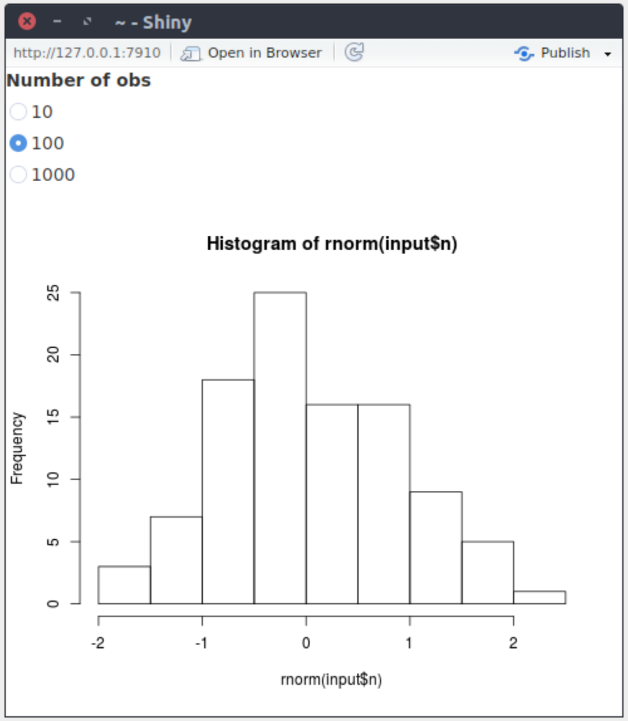
A graphical user-interface for your R program

```
library(shiny)
ui <- bootstrapPage(
  radioButtons(inputId = "n", label = "Number of obs", choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)
server <- function(input, output) {output$plot <- renderPlot(hist(rnorm(input$n)))}
shinyApp(ui = ui, server = server)
```



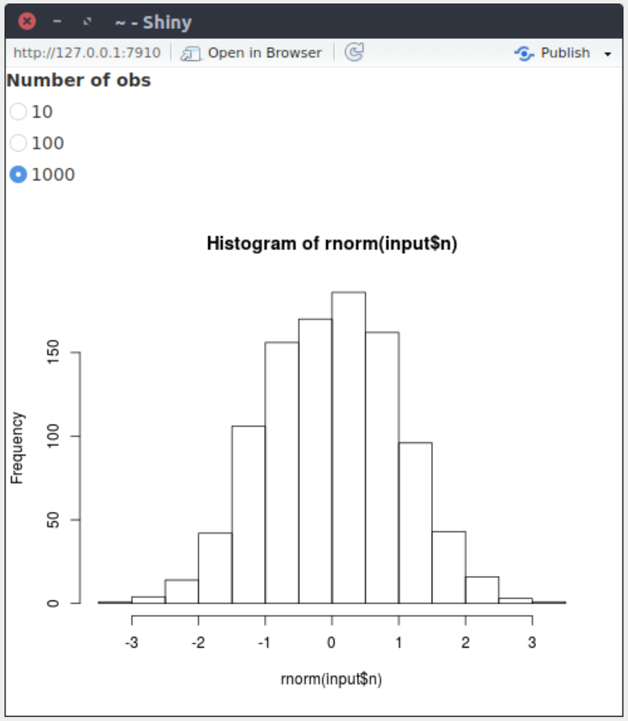
Number of obs

[✓ 10] 100 1000



Number of obs

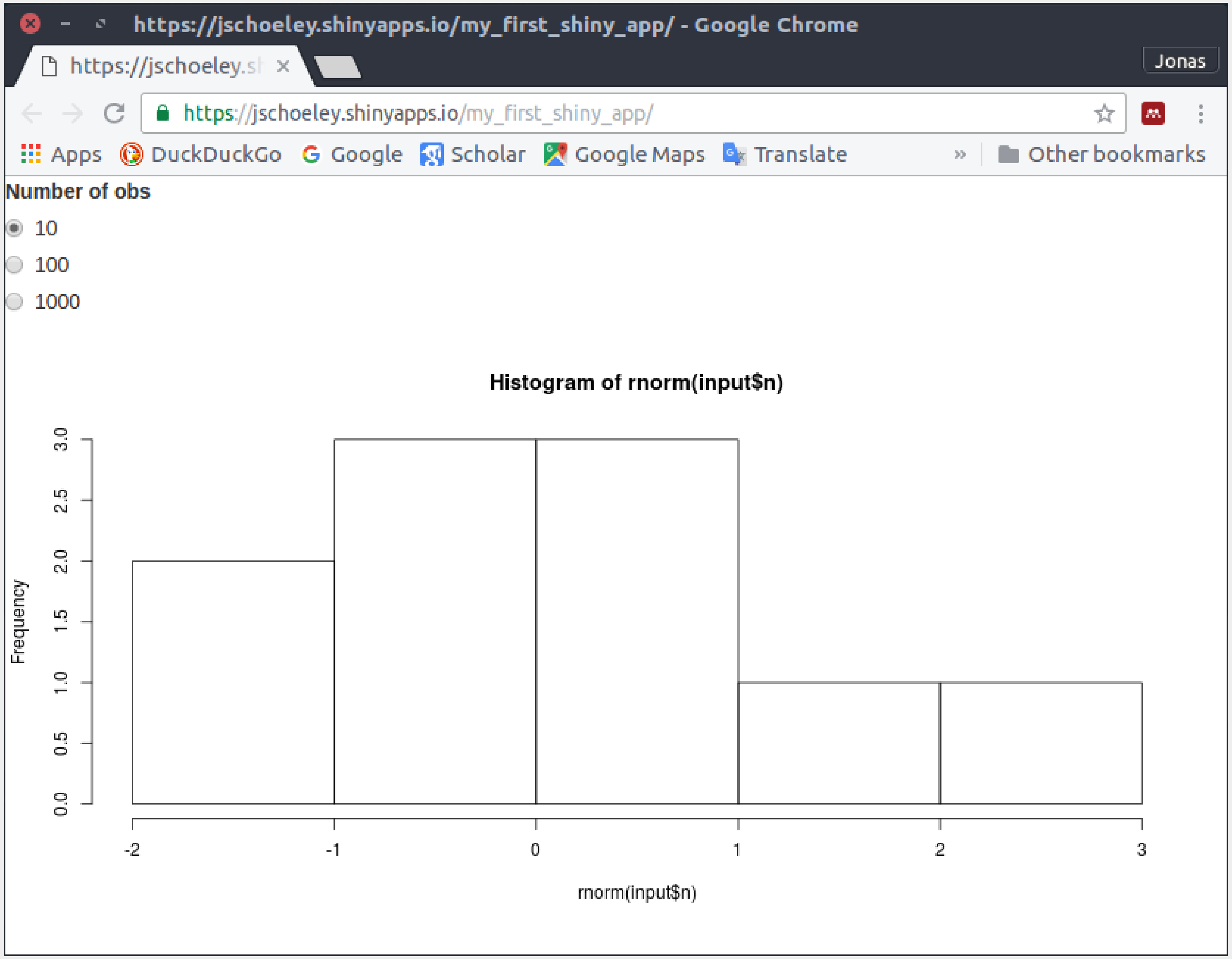
10 [✓ 100] 1000



Number of obs

10 100 [✓ 1000]

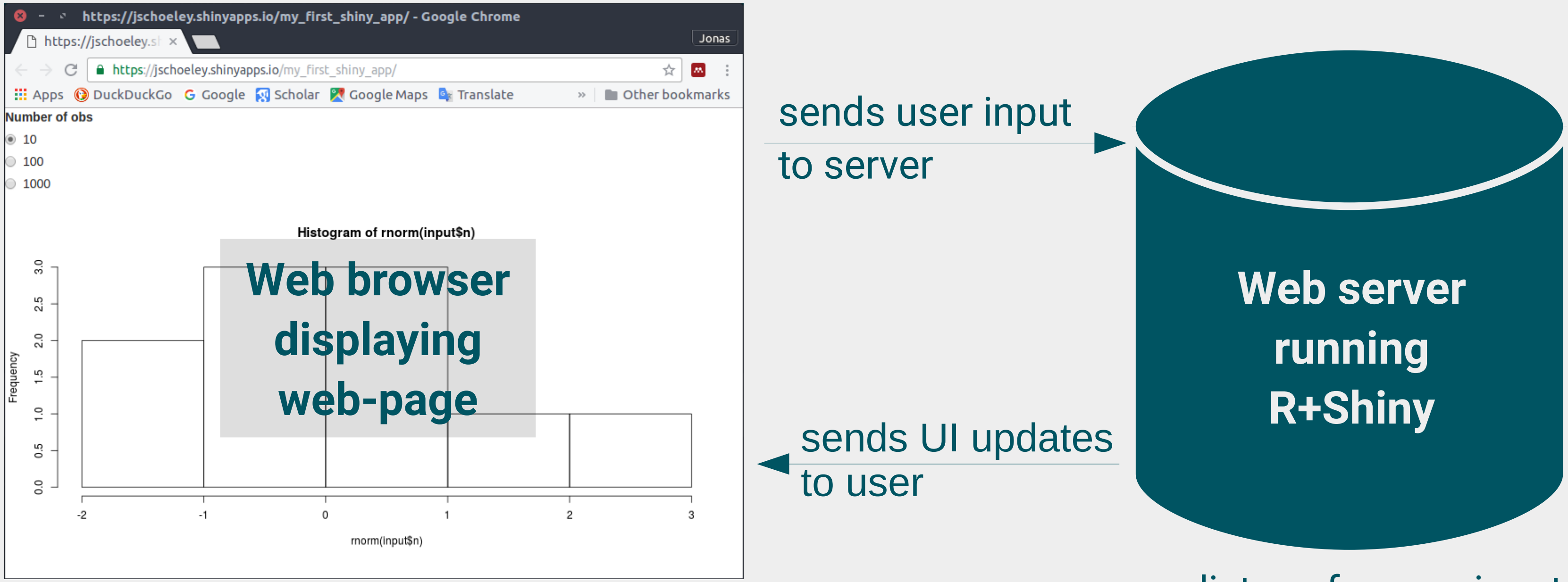
Your R program as a web-page


https://jschoeley.shinyapps.io/my_first_shiny_app/ - Google Chrome
https://jschoeley.s
Jonas
https://jschoeley.shinyapps.io/my_first_shiny_app/
Apps
DuckDuckGo
Google
Scholar
Google Maps
Translate
Other bookmarks
Histogram of rnorm(input$n)
Frequency
0.0
0.5
1.0
1.5
2.0
2.5
3.0
-2
-1
0
1
2
3
rnorm(input$n)

Number of obs

● 10
○ 100
○ 1000

Shiny: A web-application framework for R



https://jschoeley.shinyapps.io/my_first_shiny_app/ - Google Chrome
https://jschoeley.shinyapps.io/my_first_shiny_app/
Apps DuckDuckGo Google Scholar Google Maps Translate Other bookmarks
Number of obs
10
100
1000
Histogram of rnorm(input$n)
Frequency
rnorm(input$n)
Web browser displaying web-page
sends user input to server
sends UI updates to user
Web server running R+Shiny

- listens for updates from server
- renders web-page

- listens for user input
- performs R calculations
- builds web-page

Programing Shiny

1 Building a user interface (UI)
2 Connecting R code with the UI

Programing Shiny

```r
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Programing Shiny: Building a User Interface

1 Building a user interface (UI)

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Programing Shiny: Building a User Interface

Create an empty web-page
learn about more sophisticated layouts at
shiny.rstudio.com/articles/layout-guide.html

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

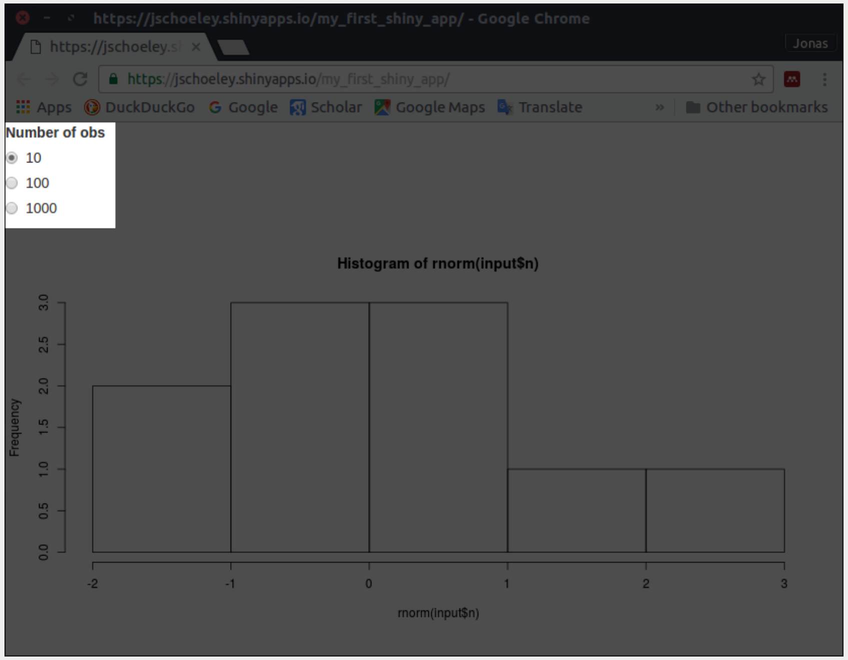
Programing Shiny: Building a User Interface

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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


https://jschoeley.shinyapps.io/my_first_shiny_app/ - Google Chrome
https://jschoeley.s
Jonas
https://jschoeley.shinyapps.io/my_first_shiny_app/
Apps
DuckDuckGo
Google
Scholar
Google Maps
Translate
Other bookmarks
Histogram of rnorm(input$n)
Frequency
0.0
0.5
1.0
1.5
2.0
2.5
3.0
-2
-1
0
1
2
3
rnorm(input$n)

Number of obs
[✓ 10] 100 1000

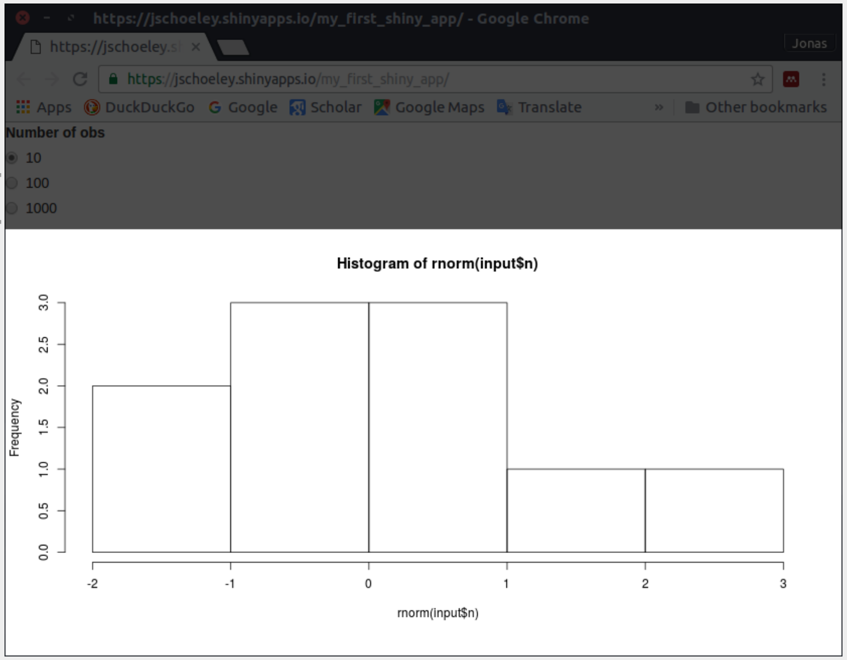
Programing Shiny: Building a User Interface

```r
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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


https://jschoeley.shinyapps.io/my_first_shiny_app/ - Google Chrome
https://jschoeley.shinyapps.io/my_first_shiny_app/
Apps DuckDuckGo Google Scholar Google Maps Translate Other bookmarks
Number of obs
10
100
1000
Histogram of rnorm(input$n)
Frequency
rnorm(input$n)

Programing Shiny: Building a User Interface

Don’t forget the comma
the UI specification is one large nested function call with different UI elements as parameters

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Programing Shiny: Connecting R to the UI

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

2 Connecting R code with the UI

Programing Shiny: Connecting R to the UI

R communicates with the UI using 2 lists

the input list contains the values of all input elements of the UI, the output list contains the objects which R sends to the UI

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Programing Shiny: Connecting R to the UI

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Programing Shiny: Connecting R to the UI

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Programing Shiny: Connecting R to the UI

The “actual” R code

R calculates new output once new input arrives

```
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Programing Shiny: Wrapping Things Up

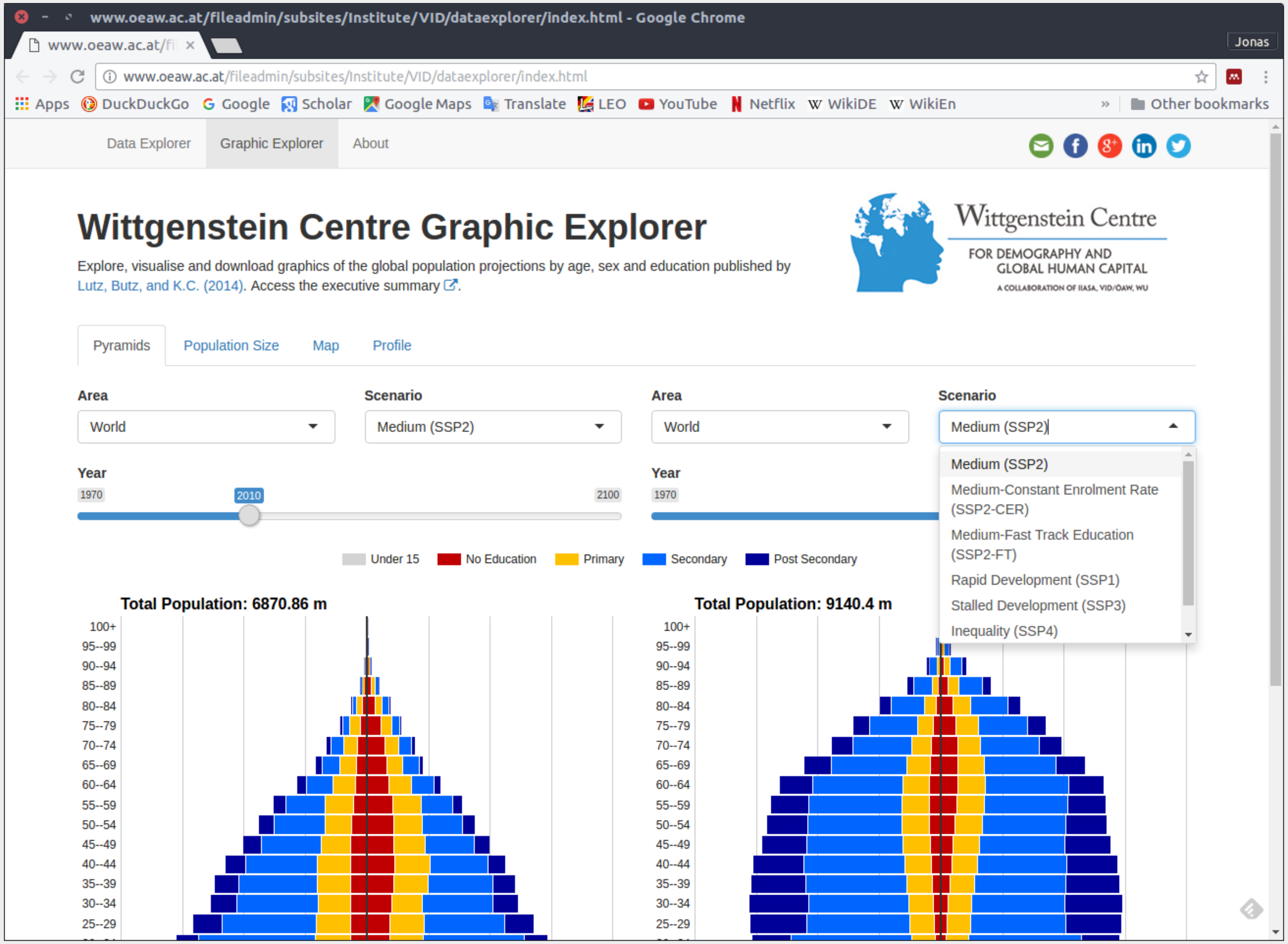
```r
library(shiny)

ui <- bootstrapPage(
  radioButtons(inputId = "n",
               label = "Number of obs",
               choices = list(10, 100, 1000)),
  plotOutput(outputId = "plot")
)

server <- function(input, output) {
  output$plot <- renderPlot(
    hist(rnorm(input$n))
  )
}

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

Who Uses It?



www.oeaw.ac.at/fileadmin/subsites/Institute/VID/dataexplorer/index.html - Google Chrome

Data Explorer | Graphic Explorer | About

Wittgenstein Centre Graphic Explorer

Explore, visualise and download graphics of the global population projections by age, sex and education published by Lutz, Butz, and K.C. (2014). Access the executive summary.

Wittgenstein Centre FOR DEMOGRAPHY AND GLOBAL HUMAN CAPITAL A COLLABORATION OF IIASA, VID/ÖAW, WU

Pyramids | Population Size | Map | Profile

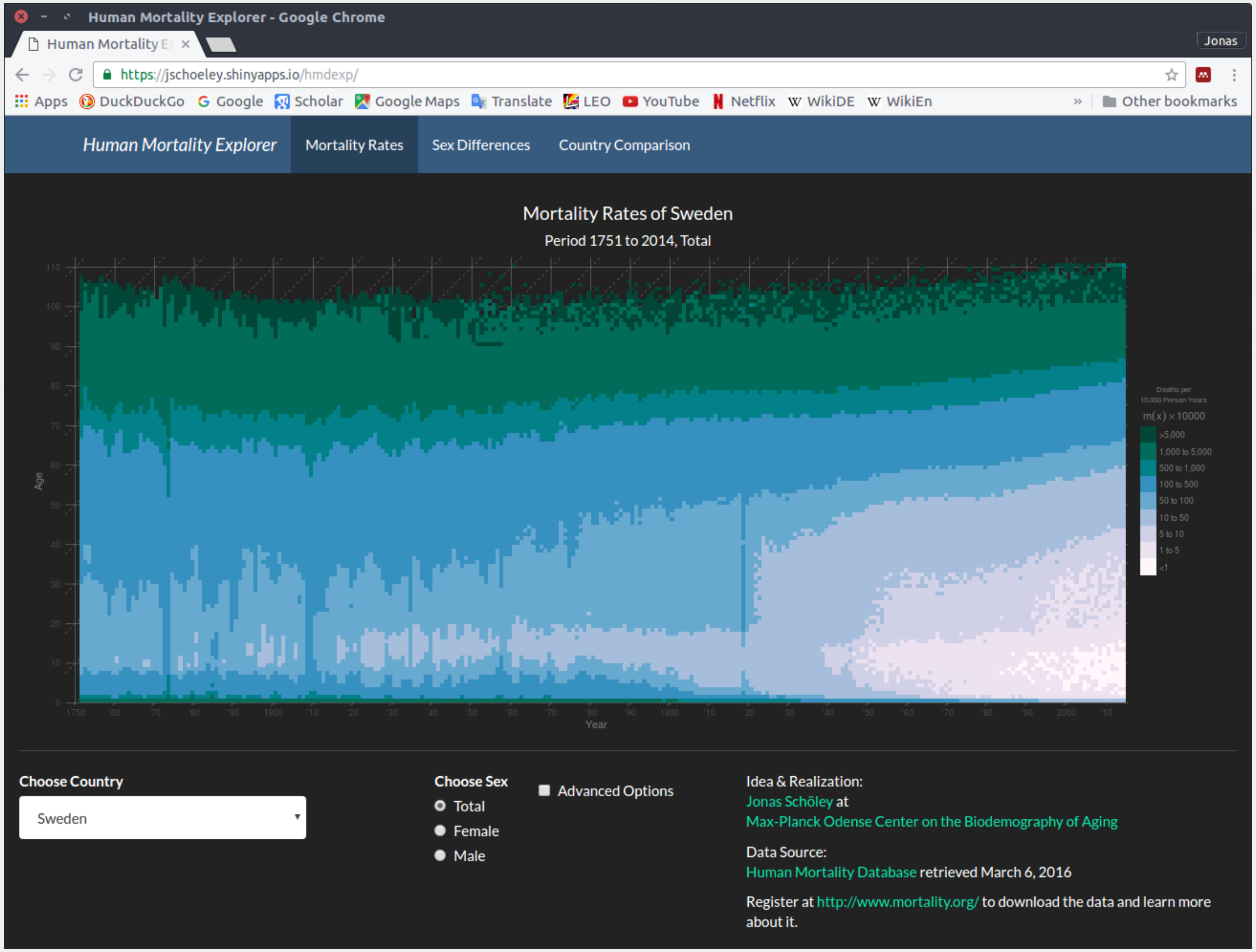
Area: World | Scenario: Medium (SSP2) | Area: World | Scenario: Medium (SSP2)

Year: 2010 (1970–2100) | Year: 1970

Scenario options: Medium (SSP2); Medium-Constant Enrolment Rate (SSP2-CER); Medium-Fast Track Education (SSP2-FT); Rapid Development (SSP1); Stalled Development (SSP3); Inequality (SSP4)

Wittgenstein Centre for Demography and Global Human Capital, (2015). Wittgenstein Centre Data Explorer Version 1.2. Available at: http://www.wittgensteincentre.org/dataexplorer

Who Uses It?


Human Mortality Explorer - Google Chrome
Human Mortality E ×
Jonas
https://jschoeley.shinyapps.io/hmdexp/
Apps DuckDuckGo Google Scholar Google Maps Translate LEO YouTube Netflix WikiDE WikiEn » Other bookmarks
Human Mortality Explorer
Mortality Rates
Sex Differences
Country Comparison
Mortality Rates of Sweden
Period 1751 to 2014, Total
Age
Year
Deaths per 10,000 Person Years
m(x) × 10000
>5,000
1,000 to 5,000
500 to 1,000
100 to 500
50 to 100
10 to 50
5 to 10
1 to 5
<1

Choose Country

Sweden

Choose Sex

- [✓ Total]
- Female
- Male

Advanced Options

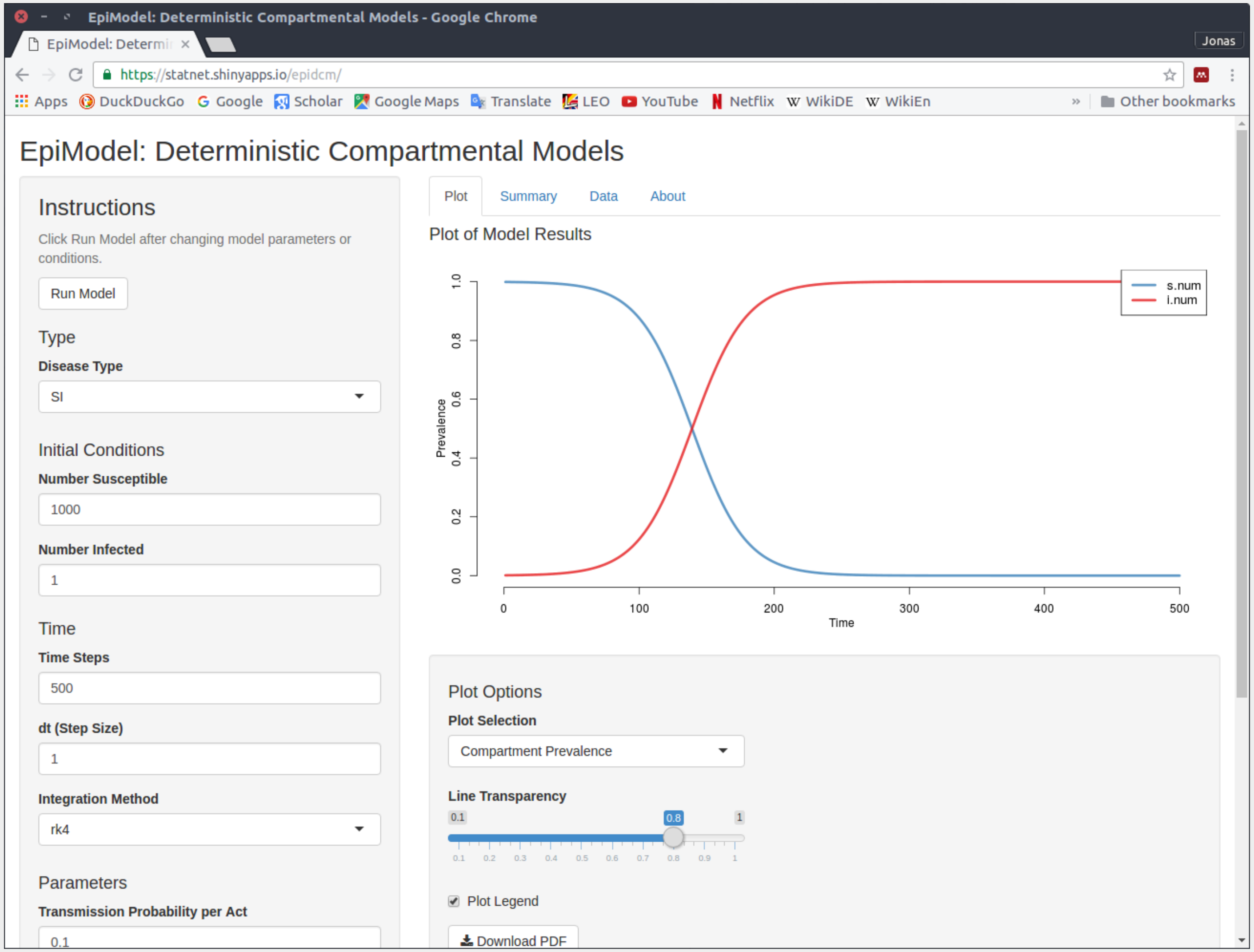
Idea & Realization:
Jonas Schöley at
Max-Planck Odense Center on the Biodemography of Aging

Data Source:
Human Mortality Database retrieved March 6, 2016

Register at http://www.mortality.org/ to download the data and learn more about it.

Jonas Schöley (2016): “The Human Mortality Explorer”. https://jschoeley.shinyapps.io/hmdexp/

Who Uses It?



EpiModel: Deterministic Compartmental Models

Instructions

Click Run Model after changing model parameters or conditions.

Run Model

Type

Disease Type: SI

Initial Conditions

Number Susceptible: 1000

Number Infected: 1

Time

Time Steps: 500

dt (Step Size): 1

Integration Method: rk4

Parameters

Transmission Probability per Act: 0.1

Plot | Summary | Data | About

Plot of Model Results

Plot Options

Plot Selection: Compartment Prevalence

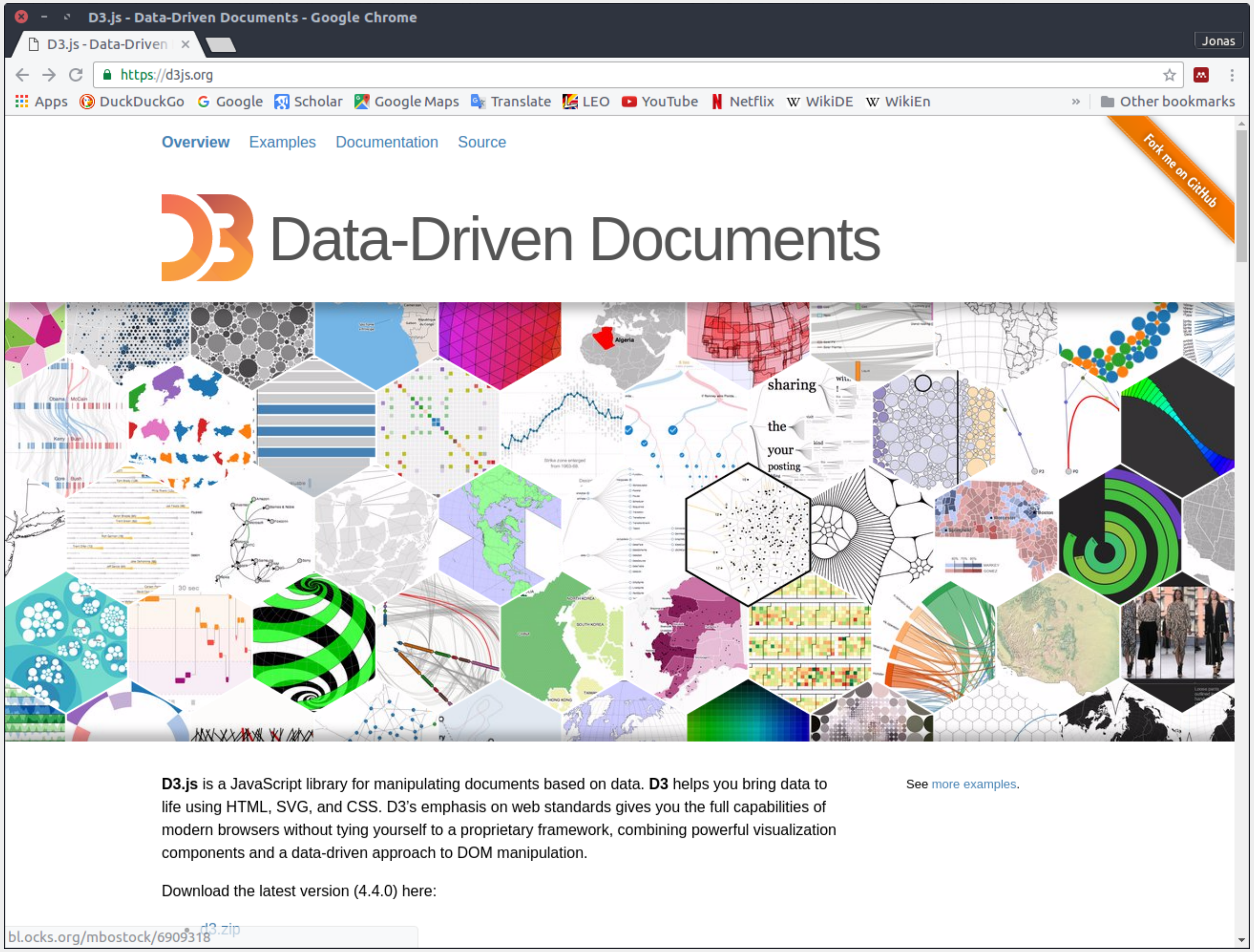
Line Transparency: 0.8

✓ Plot Legend

Download PDF

EpiModel: Deterministic Compartmental Models. https://statnet.shinyapps.io/epidcm/

Alternatives: D3.js



D3.js is a JavaScript library for manipulating documents based on data. **D3** helps you bring data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

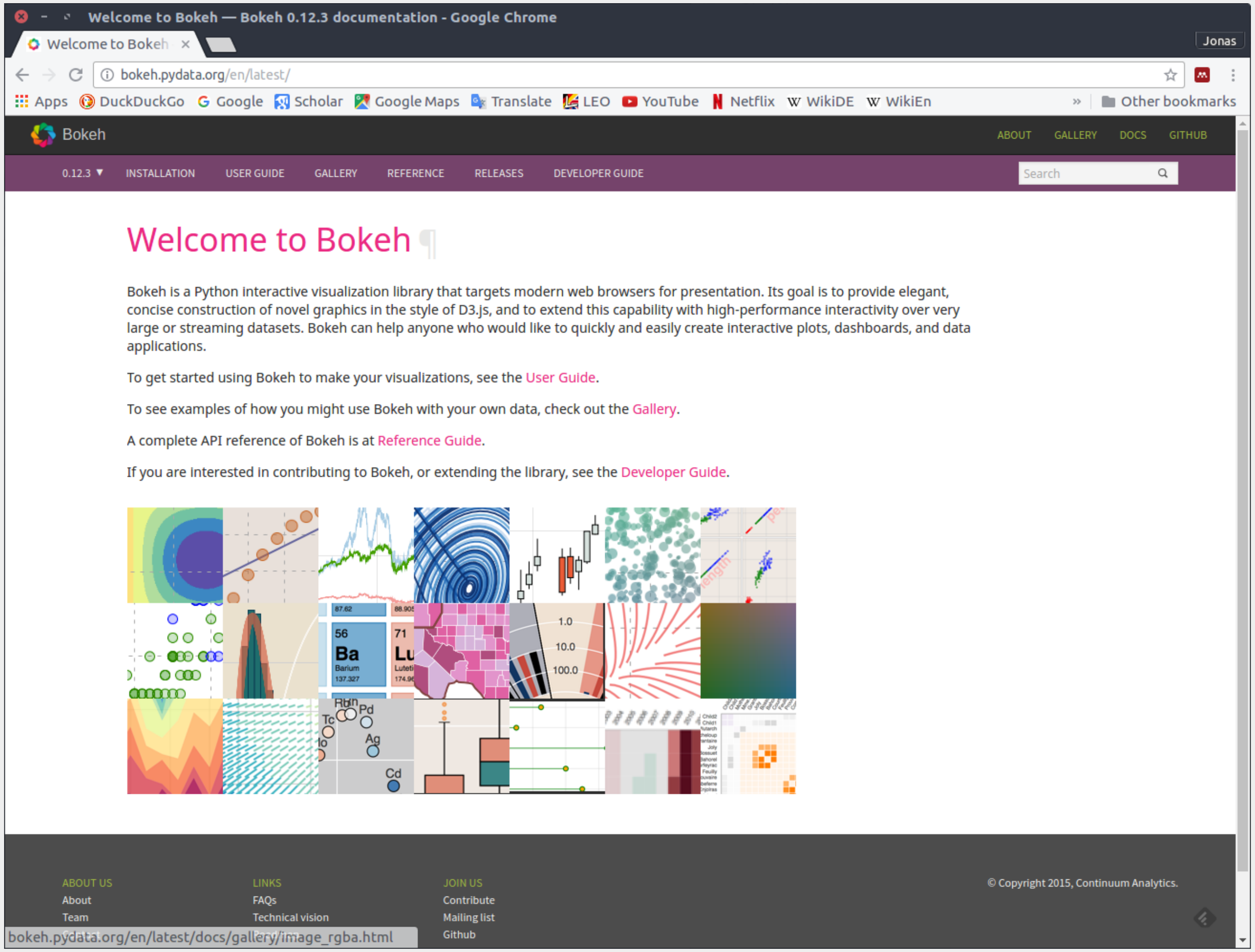
See more examples.

Download the latest version (4.4.0) here:

- d3.zip

+ fast
+ does not require R/Shiny web server
+ can be easily embedded into an existing web-page
+ large user base
- requires knowledge of JavaScript, HTML, CSS
- does not provide statistical computing features of R

Alternatives: Bokeh



Welcome to Bokeh — Bokeh 0.12.3 documentation - Google Chrome

bokeh.pydata.org/en/latest/

Bokeh — ABOUT GALLERY DOCS GITHUB

0.12.3 ▼ INSTALLATION USER GUIDE GALLERY REFERENCE RELEASES DEVELOPER GUIDE

Welcome to Bokeh

Bokeh is a Python interactive visualization library that targets modern web browsers for presentation. Its goal is to provide elegant, concise construction of novel graphics in the style of D3.js, and to extend this capability with high-performance interactivity over very large or streaming datasets. Bokeh can help anyone who would like to quickly and easily create interactive plots, dashboards, and data applications.

To get started using Bokeh to make your visualizations, see the User Guide.

To see examples of how you might use Bokeh with your own data, check out the Gallery.

A complete API reference of Bokeh is at Reference Guide.

If you are interested in contributing to Bokeh, or extending the library, see the Developer Guide.

ABOUT US: About, Team

LINKS: FAQs, Technical vision

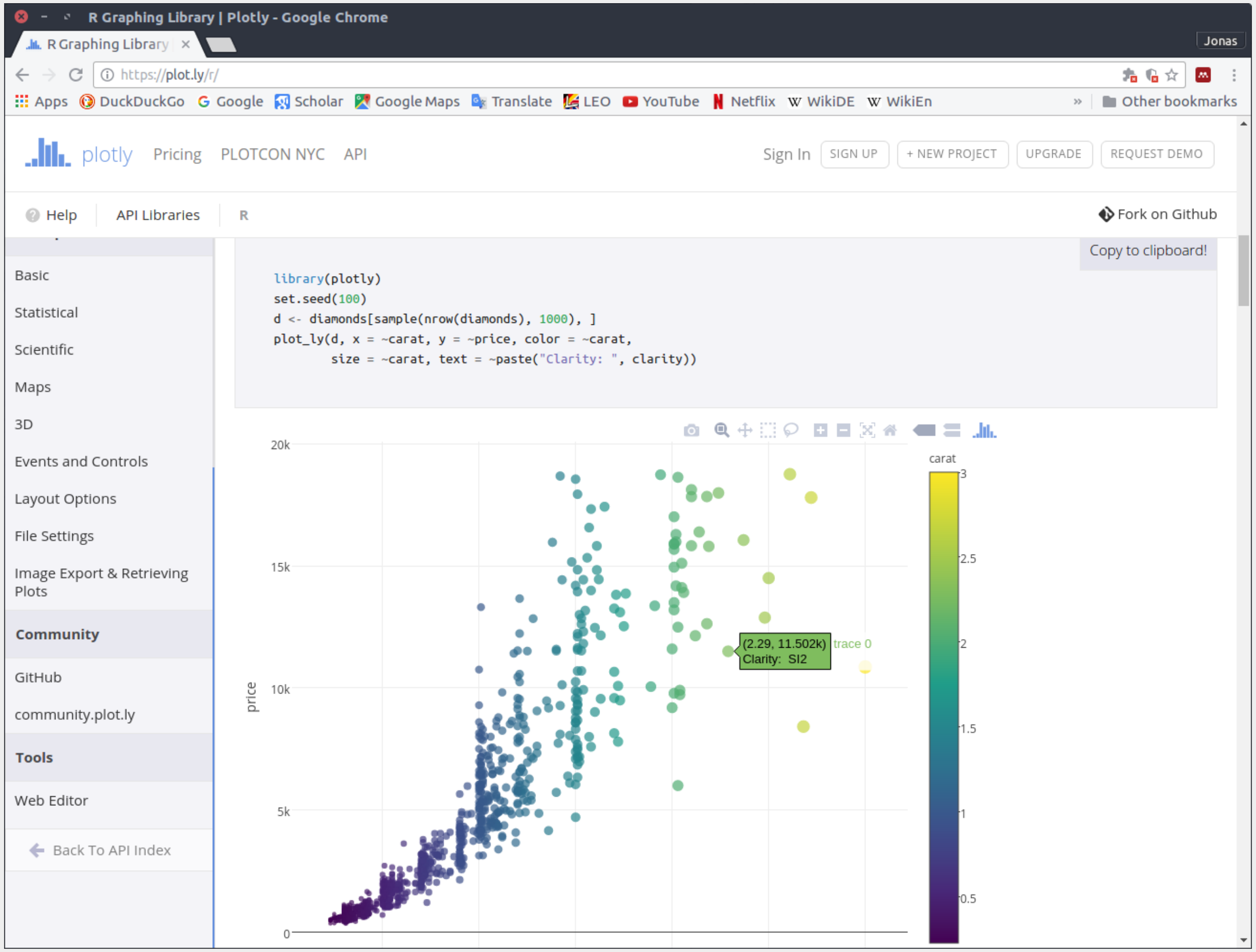
JOIN US: Contribute, Mailing list, Github

© Copyright 2015, Continuum Analytics.

bokeh.pydata.org/en/latest/docs/gallery/image_rgba.html

+ Python
- does not provide statistical computing features of R

Alternatives: plotly



```r
library(plotly)
set.seed(100)
d <- diamonds[sample(nrow(diamonds), 1000), ]
plot_ly(d, x = ~carat, y = ~price, color = ~carat,
        size = ~carat, text = ~paste("Clarity: ", clarity))
```

+ very user-friendly
+ provides basic interactivity for an existing ggplot without any user-effort (ggplotly)
- web-server offerings are commercial

Slides available at

github.com/jschoeley/2016-mpidr-intro_shiny

Jonas Schöley

jschoeley@health.sdu.dk

Twitter: @rettungstweet