Introduction to R Shiny

Aaron Geller

What is Shiny and why should I use it?

Shiny is an R package for creating interactive web applications without needing to know HTML, CSS or JavaScript.

Why use Shiny?

- Exploratory data analysis
- Contextual learning and exploration (e.g., data on a map)
- Dashboards
- Tools for your colleagues
- Teaching
- And more... see the Shiny gallery!

Before starting in Shiny

- Think conceptually about what you want to create. Maybe make a sketch or outline.
- Consider both
  - **functionality** (e.g., how do I want to interactively manipulate the data) and
  - **form** (e.g., what layout is going to be the most straightforward to the user).
- Create a working static version of your figure (or table) first (without Shiny).

Basic Shiny App Structure

app.R

```r
library(shiny)

ui <- fluidPage()

server <- function(input, output){}

shinyApp(ui, server)
```

Basic Shiny App Structure

app.R

```
library(shiny)

ui <- fluidPage()

server <- function(input, output){}

shinyApp(ui, server)
```

- Load the Shiny library so the functions we need are available.

Basic Shiny App Structure

app.R

```r
library(shiny)

ui <- fluidPage()

server <- function(input, output){}

shinyApp(ui, server)
```

- Load the Shiny library so the functions we need are available.
- The user interface (UI) displays widgets for manipulating data (input) and figures, tables, etc. (output).

Basic Shiny App Structure

app.R

```
library(shiny)

ui <- fluidPage()

server <- function(input, output){}

shinyApp(ui, server)
```

- Load the Shiny library so the functions we need are available.
- The user interface (UI) displays widgets for manipulating data (input) and figures, tables, etc. (output).
- The server contains a series of R statements to define what the app does.

Basic Shiny App Structure

app.R

```
library(shiny)

ui <- fluidPage()

server <- function(input, output){}

shinyApp(ui, server)
```

- Load the Shiny library so the functions we need are available.
- The user interface (UI) displays widgets for manipulating data (input) and figures, tables, etc. (output).
- The server contains a series of R statements to define what the app does.
- Create the Shiny App.

How to run a Shiny app

Option 1: from a terminal

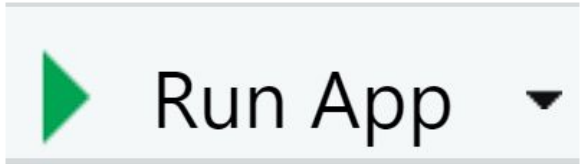
- Within a directory that has your app.R file, run the following within an R session:

```
library(shiny)
runApp("app.R")
```

- This should automatically open a browser to the URL where your local version of the app is hosted

Option 2: from R Studio

- Launch R Studio and then open your application file `("app.R")`
- Click the Run App button.



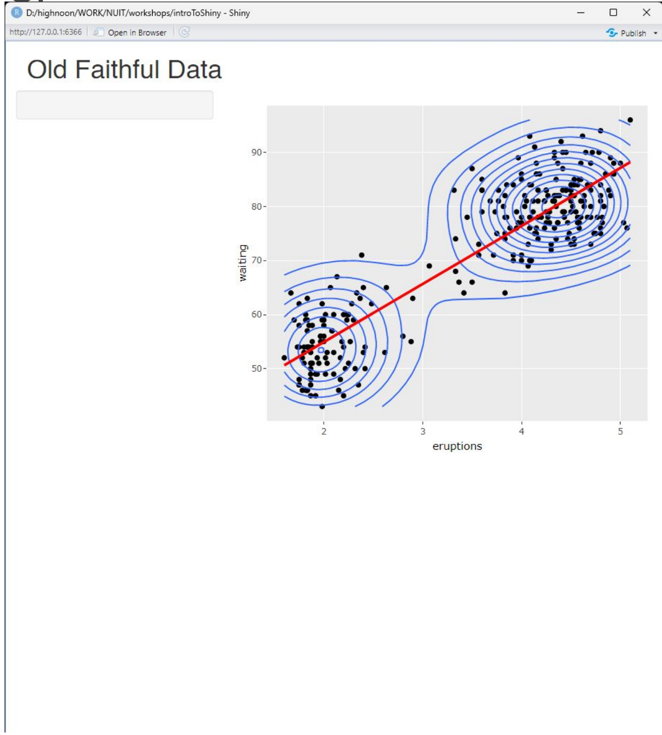
- This should automatically open a window showing your local version of the app.

Examples

1. Just a plot, to show how to start the server
2. Add inputs that control the plot content
3. Add a text box showing the fit details
4. Add dropdown to choose which marginal plots to show
5. Add `conditionalPanel` to control displaying `marginalFormat`
6. Add slider to control `binwidth` for histograms

Example 1

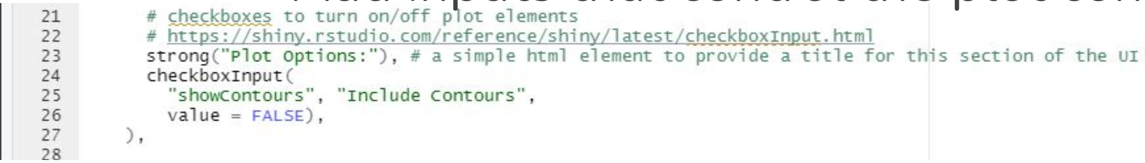
Just a plot, to show how to start the server


D:/highnoon/WORK/NUIT/workshops/introToShiny - Shiny

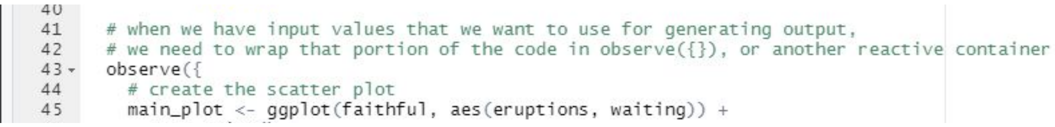
http://127.0.0.1:6366

Example 2

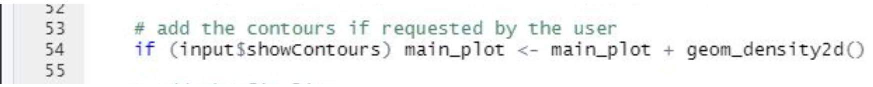
Add inputs that control the plot content



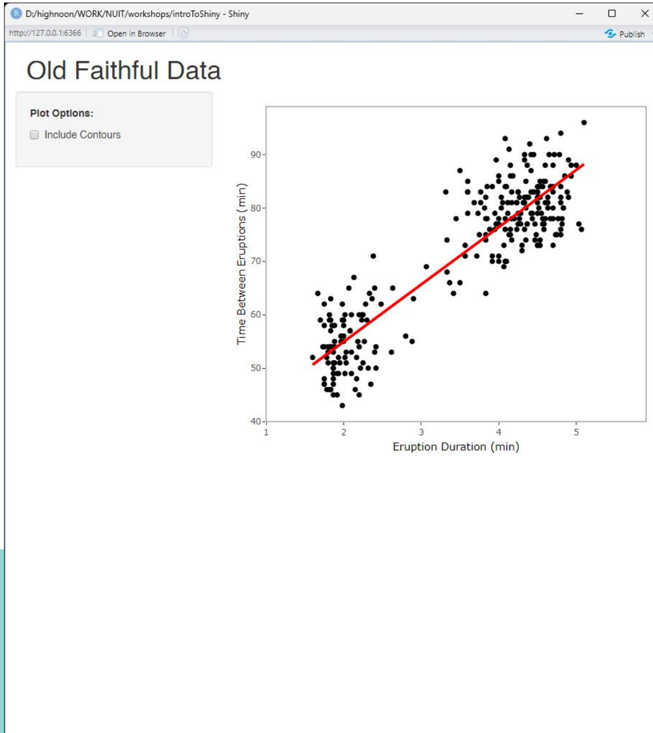
```
    # checkboxes to turn on/off plot elements
    # https://shiny.rstudio.com/reference/shiny/latest/checkboxInput.html
    strong("Plot Options:"), # a simple html element to provide a title for this section of the UI
    checkboxInput(
      "showContours", "Include Contours",
      value = FALSE),
  ),
```



```
# when we have input values that we want to use for generating output,
# we need to wrap that portion of the code in observe({}), or another reactive container
observe({
  # create the scatter plot
  main_plot <- ggplot(faithful, aes(eruptions, waiting)) +
```



```
  # add the contours if requested by the user
  if (input$showContours) main_plot <- main_plot + geom_density2d()
```



Old Faithful Data

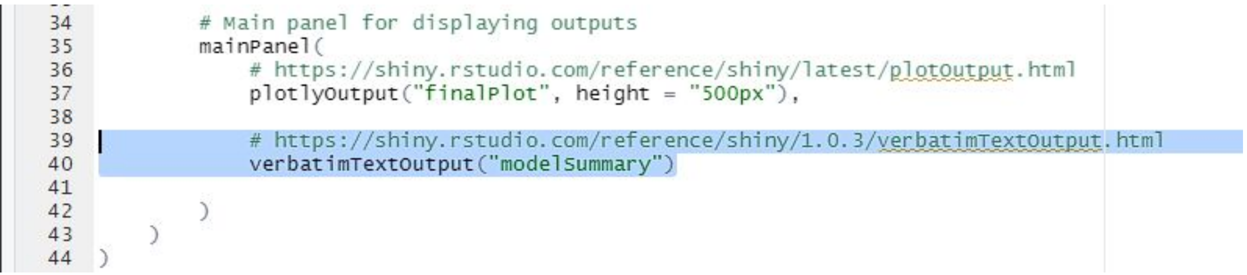
Plot Options:

☐ Include Contours

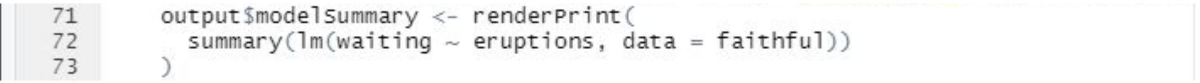
- Exercise 2.1: add input to turn line on/off (default on)
- Exercise 2.2: change `sidebarLayout` to `verticalLayout`
- Exercise 2.3: change `sidebarPanel` to `wellPanel`

Example 3

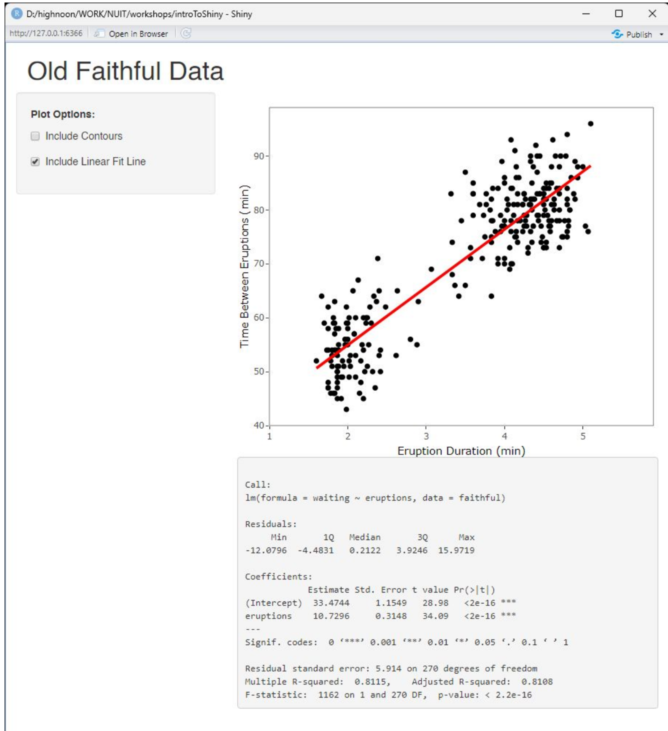
Add a text box showing the fit details



```r
    # Main panel for displaying outputs
    mainPanel(
        # https://shiny.rstudio.com/reference/shiny/latest/plotOutput.html
        plotlyOutput("finalPlot", height = "500px"),

        # https://shiny.rstudio.com/reference/shiny/1.0.3/verbatimTextOutput.html
        verbatimTextOutput("modelSummary")

    )
  )
)
```



```r
  output$modelSummary <- renderPrint(
    summary(lm(waiting ~ eruptions, data = faithful))
  )
```

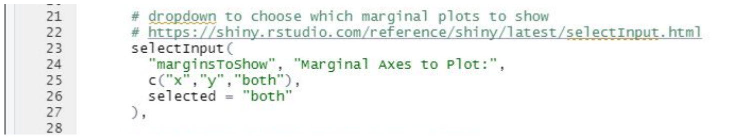


Plot Options: Include Contours [ ], Include Linear Fit Line [✓]

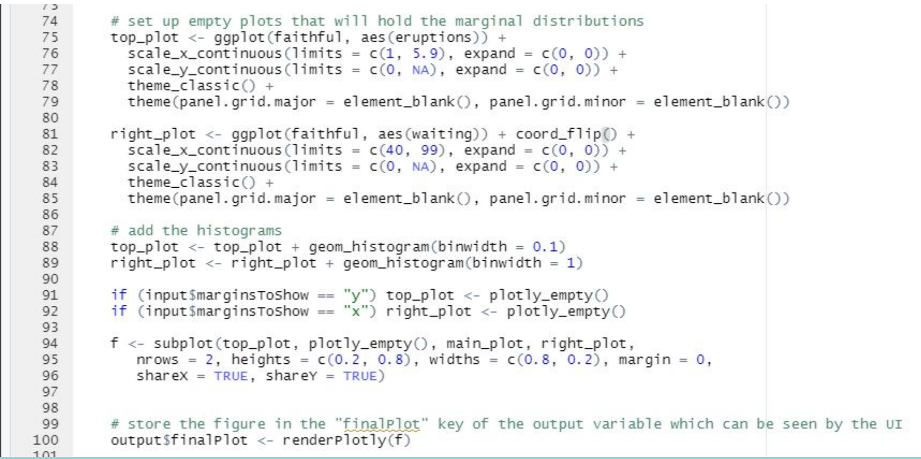
```

Call:
lm(formula = waiting ~ eruptions, data = faithful)

Residuals:
     Min       1Q   Median       3Q      Max 
-12.0796  -4.4831   0.2122   3.9246  15.9719 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  33.4744     1.1549   28.98   <2e-16 ***
eruptions    10.7296     0.3148   34.09   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.914 on 270 degrees of freedom
Multiple R-squared:  0.8115,	Adjusted R-squared:  0.8108 
F-statistic:  1162 on 1 and 270 DF,  p-value: < 2.2e-16
```

Example 4

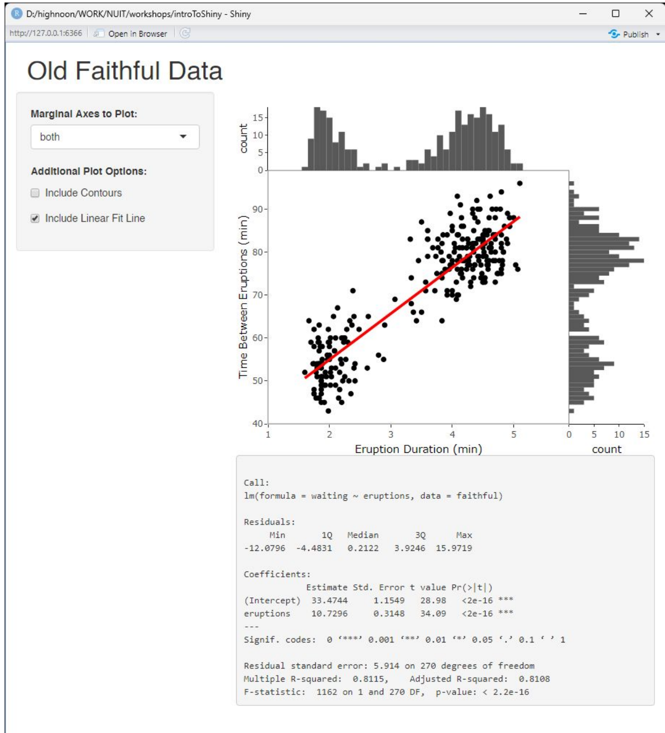
Add dropdown to choose marginal plots



```r
    # dropdown to choose which marginal plots to show
    # https://shiny.rstudio.com/reference/shiny/latest/selectInput.html
    selectInput(
      "marginsToShow", "Marginal Axes to Plot:",
      c("x","y","both"),
      selected = "both"
    ),
```



```r
  # set up empty plots that will hold the marginal distributions
  top_plot <- ggplot(faithful, aes(eruptions)) +
    scale_x_continuous(limits = c(1, 5.9), expand = c(0, 0)) +
    scale_y_continuous(limits = c(0, NA), expand = c(0, 0)) +
    theme_classic() +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())

  right_plot <- ggplot(faithful, aes(waiting)) + coord_flip() +
    scale_x_continuous(limits = c(40, 99), expand = c(0, 0)) +
    scale_y_continuous(limits = c(0, NA), expand = c(0, 0)) +
    theme_classic() +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())

  # add the histograms
  top_plot <- top_plot + geom_histogram(binwidth = 0.1)
  right_plot <- right_plot + geom_histogram(binwidth = 1)

  if (input$marginsToShow == "y") top_plot <- plotly_empty()
  if (input$marginsToShow == "x") right_plot <- plotly_empty()

  f <- subplot(top_plot, plotly_empty(), main_plot, right_plot,
     nrows = 2, heights = c(0.2, 0.8), widths = c(0.8, 0.2), margin = 0,
     shareX = TRUE, shareY = TRUE)


  # store the figure in the "finalPlot" key of the output variable which can be seen by the UI
  output$finalPlot <- renderPlotly(f)
```



Old Faithful Data

Marginal Axes to Plot:

both

Additional Plot Options:

☐ Include Contours

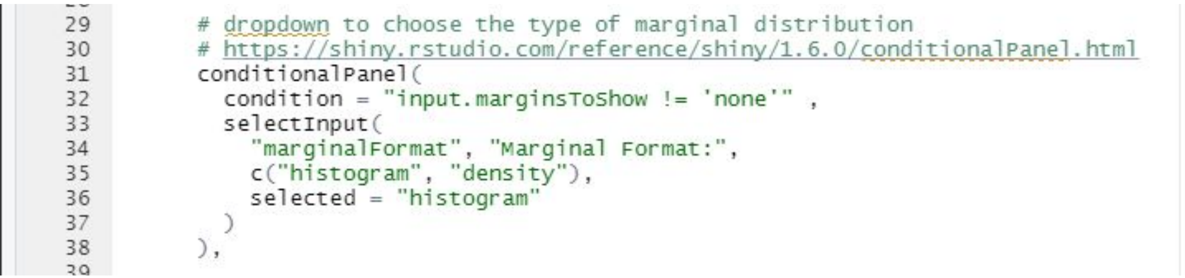
☑ Include Linear Fit Line

```
Call:
lm(formula = waiting ~ eruptions, data = faithful)

Residuals:
     Min       1Q   Median       3Q      Max 
-12.0796  -4.4831   0.2122   3.9246  15.9719 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  33.4744     1.1549   28.98   <2e-16 ***
eruptions    10.7296     0.3148   34.09   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.914 on 270 degrees of freedom
Multiple R-squared:  0.8115,	Adjusted R-squared:  0.8108 
F-statistic:  1162 on 1 and 270 DF,  p-value: < 2.2e-16
```

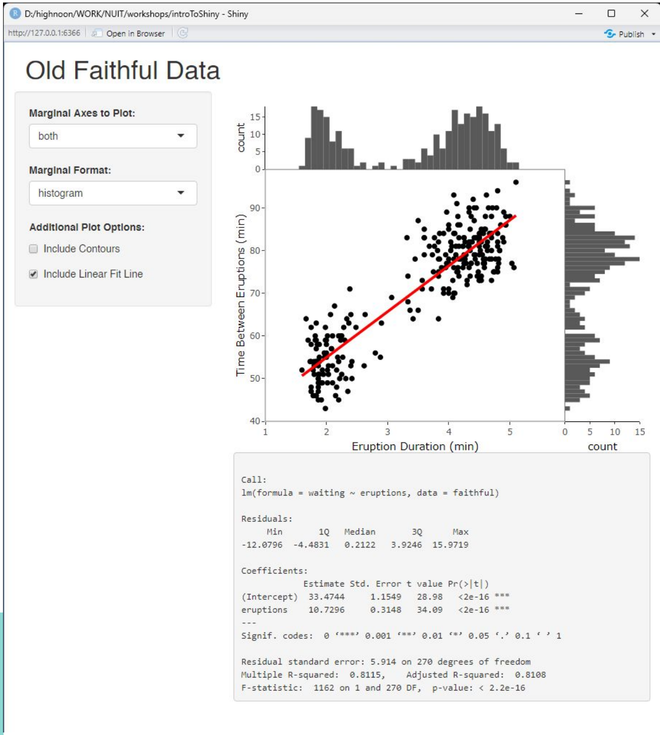
- Exercise 4.1: add dropdown to choose `histogram` vs. `density`
- Exercise 4.2: add option to choose `none` for `marginsToShow`

Example 5

Add `conditionalPanel` to control displaying `marginalFormat`



```r
# dropdown to choose the type of marginal distribution
# https://shiny.rstudio.com/reference/shiny/1.6.0/conditionalPanel.html
conditionalPanel(
  condition = "input.marginsToShow != 'none'" ,
  selectInput(
    "marginalFormat", "Marginal Format:",
    c("histogram", "density"),
    selected = "histogram"
  )
),
```


D:/highnoon/WORK/NUIT/workshops/introToShiny - Shiny
http://127.0.0.1:6366
Open in Browser
Publish
count
Time Between Eruptions (min)
Eruption Duration (min)
count

Old Faithful Data

Marginal Axes to Plot: both

Marginal Format: histogram

Additional Plot Options:

- [ ] Include Contours
- [x] Include Linear Fit Line

```
Call:
lm(formula = waiting ~ eruptions, data = faithful)

Residuals:
     Min       1Q   Median       3Q      Max 
-12.0796  -4.4831   0.2122   3.9246  15.9719 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  33.4744     1.1549   28.98   <2e-16 ***
eruptions    10.7296     0.3148   34.09   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.914 on 270 degrees of freedom
Multiple R-squared:  0.8115,	Adjusted R-squared:  0.8108 
F-statistic:  1162 on 1 and 270 DF,  p-value: < 2.2e-16
```

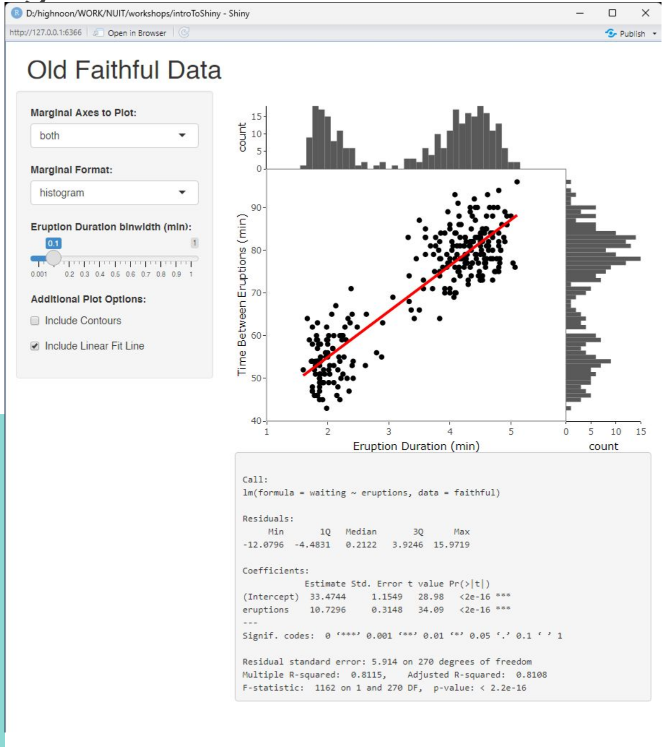
- Exercise 5.1: use `conditionalPanel` to show fit output only when showing the fit line

Example 6

Add slider to control `binwidth` for the histogram on the x axis

```
# slider to choose the bin size for the x axis (only relevant for histogram margin type)
# https://shiny.rstudio.com/reference/shiny/latest/sliderInput.html
sliderInput("xbinwidth", "Eruption Duration binwidth (min):",
            min = 1e-3, max = 1, value = 0.1),
```

```
            top_plot <- top_plot + geom_histogram(binwidth = input$xbinwidth)
```


D:/highnoon/WORK/NUIT/workshops/introToShiny - Shiny
http://127.0.0.1:6366
Open in Browser
Publish
count
Time Between Eruptions (min)
Eruption Duration (min)
count

Old Faithful Data

Marginal Axes to Plot: both

Marginal Format: histogram

Eruption Duration binwidth (min): 0.1

Additional Plot Options:

- [ ] Include Contours
- [✓] Include Linear Fit Line

```
Call:
lm(formula = waiting ~ eruptions, data = faithful)

Residuals:
     Min       1Q   Median       3Q      Max
-12.0796  -4.4831   0.2122   3.9246  15.9719

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  33.4744     1.1549   28.98   <2e-16 ***
eruptions    10.7296     0.3148   34.09   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.914 on 270 degrees of freedom
Multiple R-squared:  0.8115,    Adjusted R-squared:  0.8108
F-statistic:  1162 on 1 and 270 DF,  p-value: < 2.2e-16
```

- Exercise 6.1: add slider to control the `binwidth` on the y axis
- Exercise 6.2: use `conditionalPanel` to only show these sliders for the histogram marginal format
  - *Bonus*: use multiple conditions within the `conditionalPanel` to only show these sliders for histogram format and when the correct marginal panel is displayed

Hosting on shinyapps.io

First, sign up for an account on shinyapps.io

Option 1: from a terminal

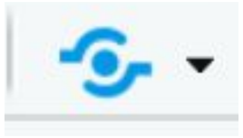
- Within a directory above the one that has your app.R file, run the following within an R session:

```
library(rsconnect)
deployApp(“dirName”)
```

- replace “`dirName`” with the name of the directory that contains app.R.
- This will create a url similar to: `https://[you].shinyapps.io/[dirName]`

Option 2: from R Studio

- Launch R Studio and then open your application file `(“app.R”)`
- Click the publish button.



- This will open a GUI window to select the file(s) to publish and other options, and will create the URL