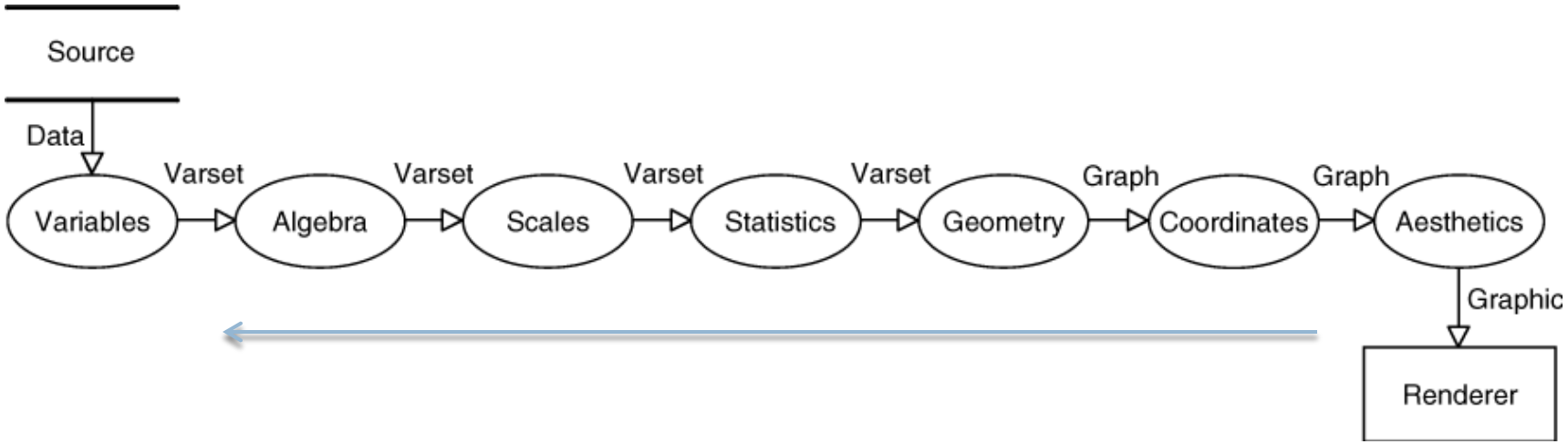
DATA VISUALIZATION

Leslie McIntosh and Connie Zabarovskaya

Today You Will Learn:

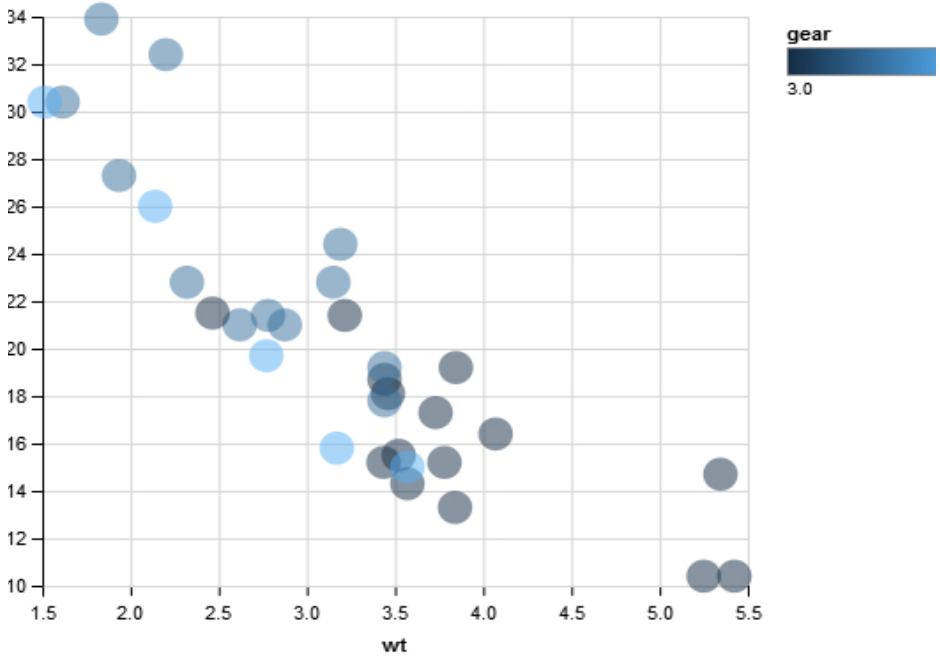
- What are Grammar of Graphics, GGVIS, rCharts and Shiny?
- How to create an interactive plot using ggvis
  - Explanation
  - Hands-On
- How to publish a ggvis plot online
  - Explanation
  - Hands-On
- How to create an interactive plot in rCharts
  - Explanation
  - Hands-On
- How to publish an rCharts plot online
  - Explanation
  - Hands-On
- Where to get more resources, tools, etc.

What is Grammar of Graphics?



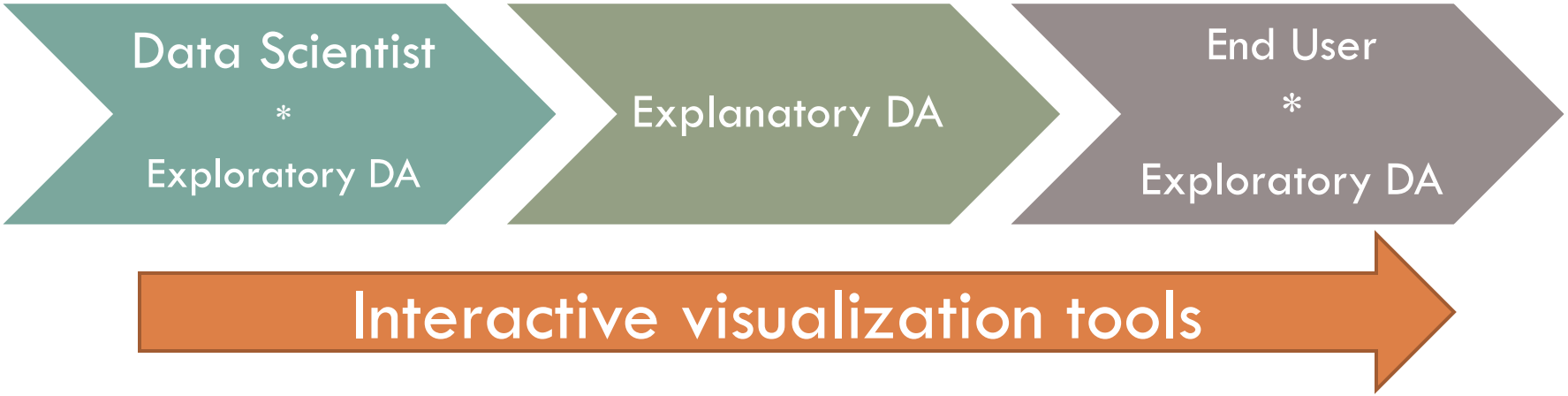
Leland Wilkinson: The Grammar of Graphics 2nd Edition

What is ggvis?



- a data visualization package,
- developed by Hadley Wickham
- allows for interactive graphics thanks to shiny integration

Why & How to Use ggvis?



- ggvis is great for both exploratory & explanatory data analysis
- syntax is similar to ggplot2
- works great with dplyr
- graphics can be published online using shiny

Installing ggvis

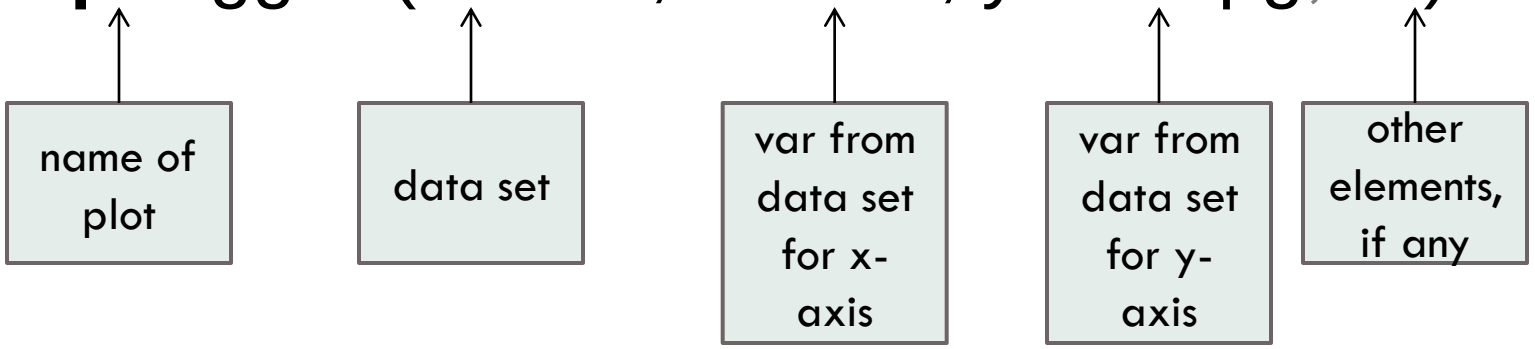
- Now available on CRAN: install.packages("ggvis")
- or install the development package:
  - install.packages("devtools") devtools::install_github("rstudio/ggvis", build_vignettes = FALSE)

Steps with ggvis

1. Intro to ggvis
2. Some examples with ggvis
3. Create a ggvis plot
4. Integrate it with a Shiny app
5. Publish it on a Shiny server/shinyapps.io

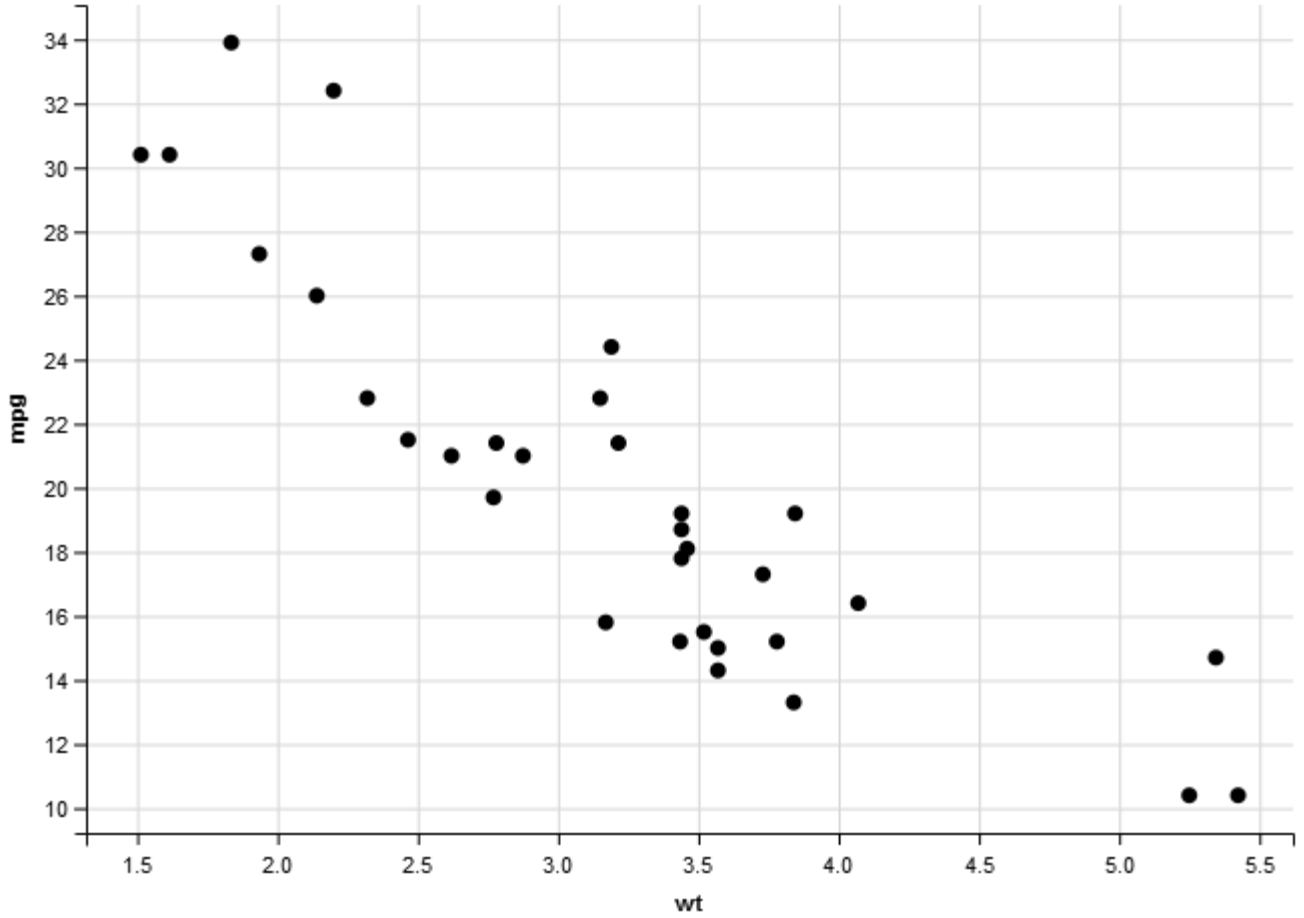
GGVIS Data Visualization (basic)

- > library(ggvis)
- call to ggvis
- > **p**<- ggvis(mtcars, x = ~wt, y = ~mpg, …)



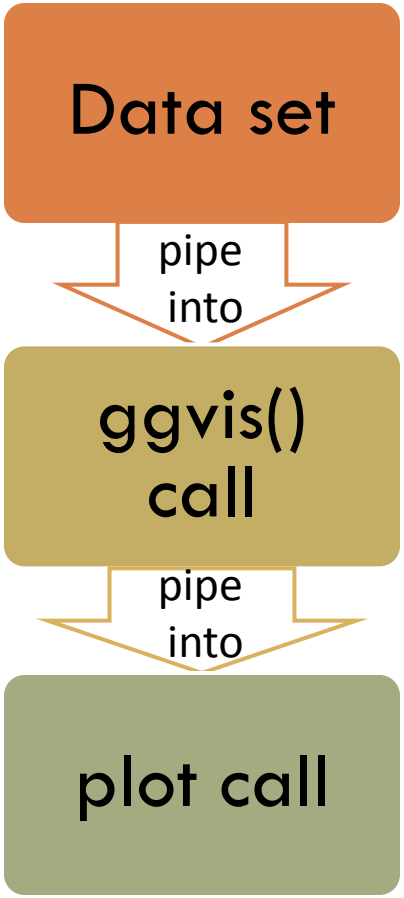
- to actually plot (display) the data
- > layer_points(**p**)

The Result



Adding Elements to the Plot

- to avoid nested functions and"temporary" variables use %>% (termed 'pipe')
- %>% takes value on the left-hand side and passes it to function or expression on the right-hand side
- *FYI: Pipe comes from the magrittr package (http://cran.r-project.org/web/packages/magrittr/magrittr.pdf)*

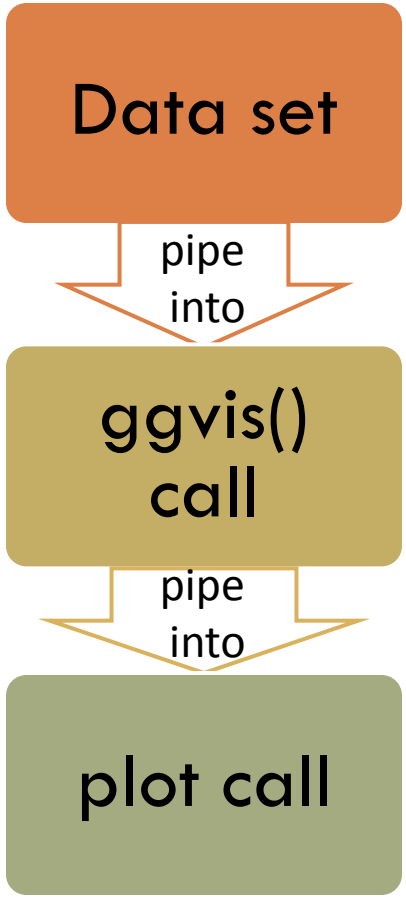


Adding Elements to the Plot

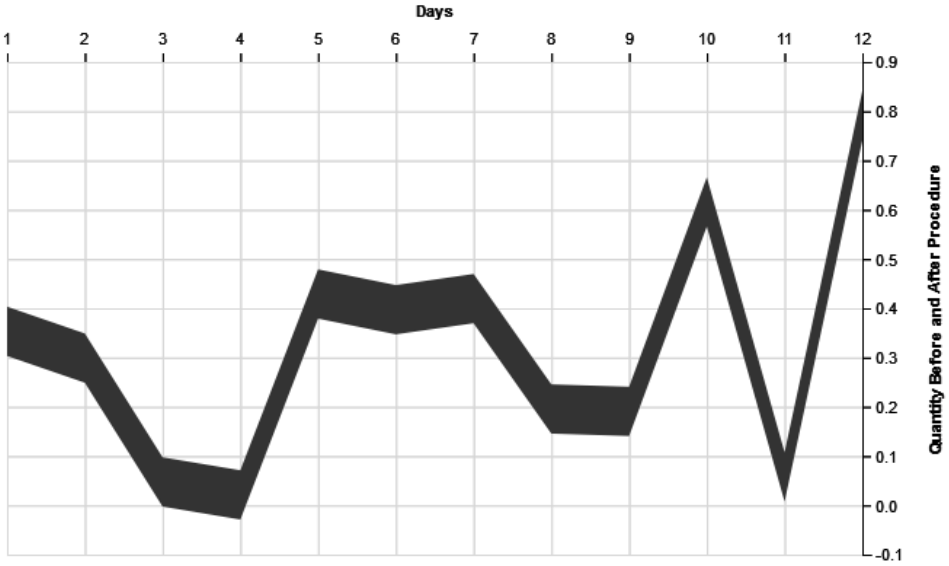
```
> mtcars %>%
ggvis(x = ~wt, y = ~mpg) %>%
layer_points()
```

OR

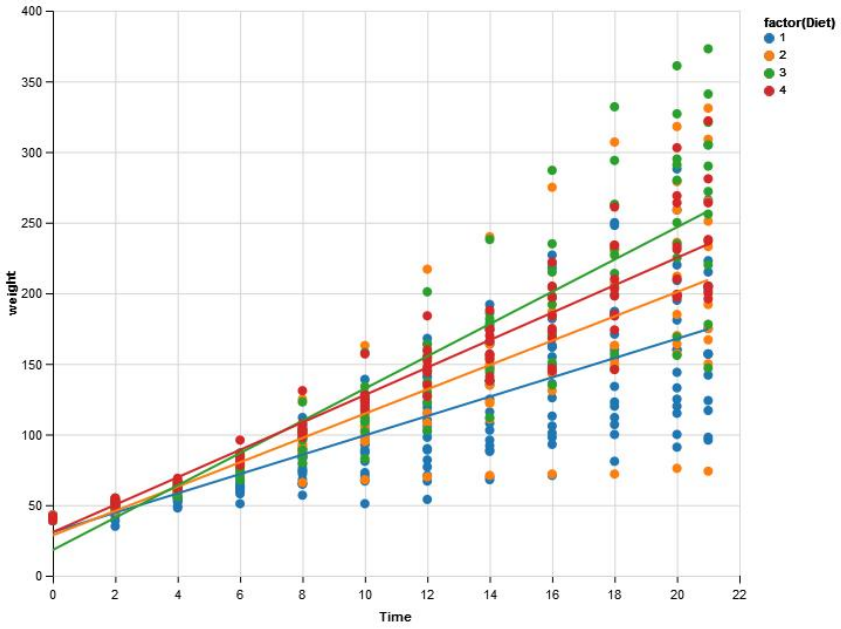
```
> mtcars %>%
ggvis(~wt, ~mpg) %>%
layer_points()
```



Some Examples




```
> df <- data.frame(x = 1:12, y = runif(12))
      df %>% ggvis(x = ~x, y = ~y, y2 = ~y - 0.1)
      %>% layer_ribbons() %>%
      add_axis("x", title = "Days", orient = "top")
      %>%
      add_axis("y", title = "Quantity Before and After
      Procedure", orient = "right", title_offset = 50)
```

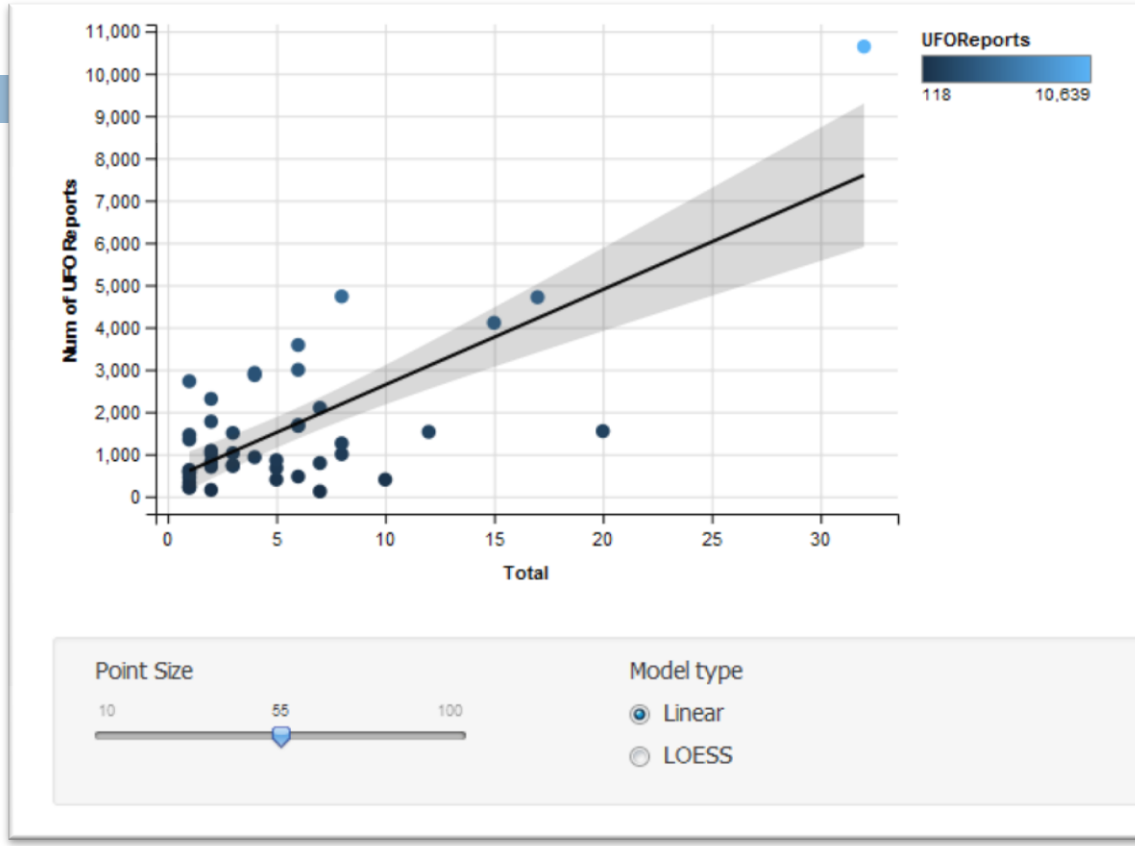



```
> ChickWeight %>%
    ggvis(x = ~Time, y = ~weight, fill =
  ~factor(Diet)) %>%
    layer_points() %>%
    group_by(Diet) %>%
  layer_model_predictions(model="lm",
  stroke = ~factor(Diet))
```

Hands-On Example

Copy and paste the code from:
http://rpubs.com/conniez/datavis

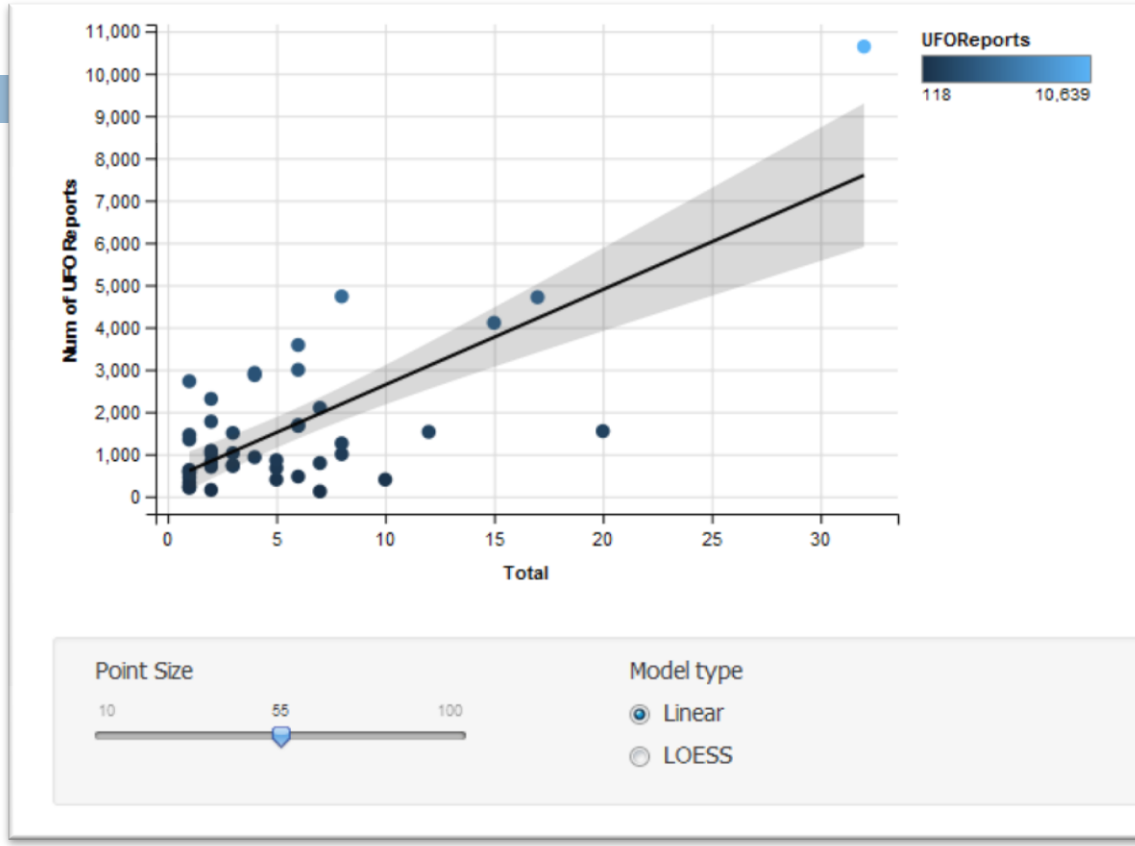
- if(!file.exists("./data")){dir.create("./data")}
- reportsUrl <- "https://github.com/ConnieZ/ufo_app/raw/master/data/ufo_reports.csv"
- download.file(reportsUrl, destfile="./data/ufo_reports.csv", mode="wb")
- milbaseUrl <- "https://github.com/ConnieZ/ufo_app/raw/master/data/milbases.csv"
- download.file(milbaseUrl, destfile="./data/milbases.csv", mode="wb")



- milbases <- read.csv("data/milbases.csv")
- reports <- read.csv("data/ufo_reports.csv")
- mergeddata <- merge(reports, milbases, by="State")

Hands-On Example

Copy and paste the code from:
http://rpubs.com/conniez/datavis



- mergeddata %>%

  ggvis(~Total, ~UFOReports) %>%

  layer_points(fill = ~UFOReports, size := input_slider(10, 100, label = "Point Size")) %>%

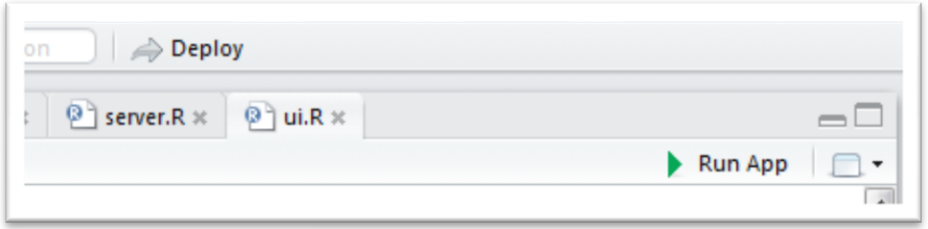
  layer_model_predictions(model = input_radiobuttons(choices = c("Linear" = "lm", "LOESS" = "loess"), selected = "lm",

  label = "Model type"), se = TRUE) %>%

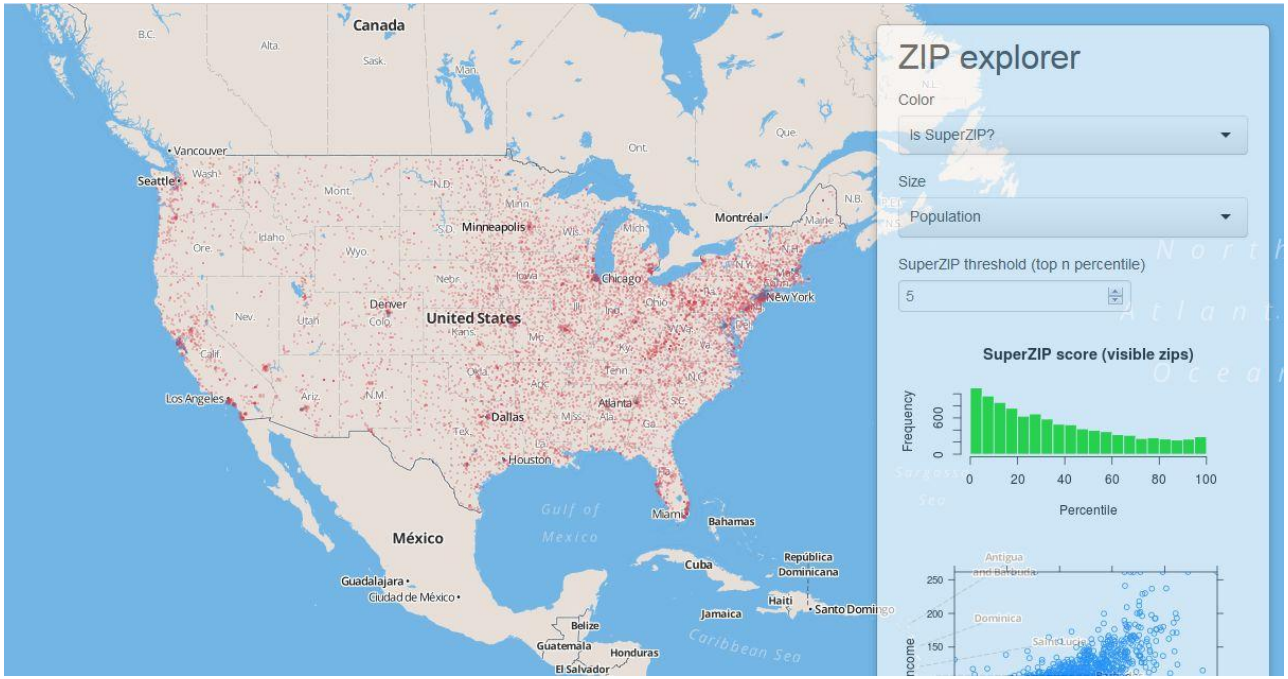
  add_axis("y", title = "Num of UFO Reports", orient = "left", title_offset = 50)

GGVIS Publishing Online

- You can publish your charts online using Shiny functionality. Add the script to a Shiny app and publish the app. Example here: http://ggvis.rstudio.com/interactivity.html#shiny-apps
- See good demo here: https://github.com/rstudio/ggvis/tree/master/demo/apps/basic
- Steps:
  - Create ui.R and server.R files for a Shiny app in a certain folder (folder name = short name of your app). Add data files in a folder inside of it
  - From Rstudio test your application by clicking "Run App" button in your text editor window of Rstudio.
  - If you have set up your Rstudio with access to Shiny or Shinyapps.io, you will also see the "Deploy" button to publish it online.
  - Another option:
    - runApp("myapp")
    - library(shinyapps)
    - deployApp("myapp")



Shiny Interactive Graphics



- Shiny is a web application framework for R, which allows you to create interactive web apps.
- Shiny can be used with any plotting tool in R

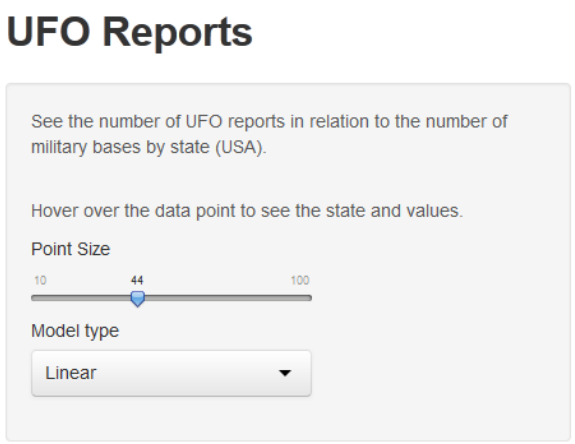
- Shiny Tutorial: http://shiny.rstudio.com/tutorial/lesson1/
- Shiny Cheatsheet: http://shiny.rstudio.com/articles/cheatsheet.html
- You can create Shiny apps by copying and modifying existing Shiny apps.

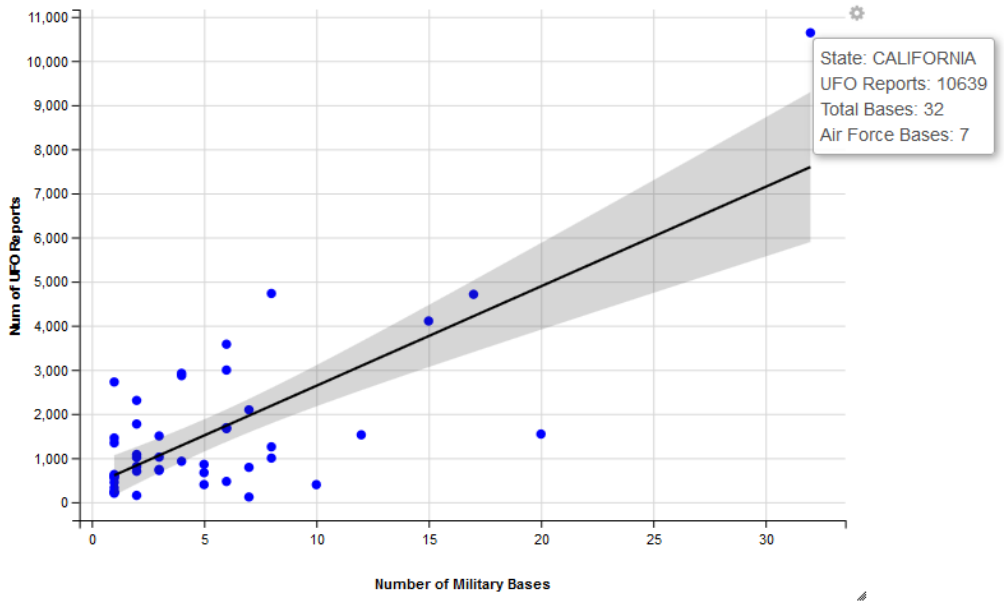
Shiny Online Publishing (web page)

- Shiny Server and Shiny Server Pro are currently only supported on Linux. They provide 64-bit binary installers for Ubuntu/Debian and Red Hat/CentOS.
- You can host your apps in the cloud for free with ShinyApps.io. Just create an account and link to your computer. There are some performance and memory limitations.

Let's Publish the GGVIS Plot Online

- See details: http://rpubs.com/conniez/datavis
- Steps:
  - Create an app folder in wd and copy the data folder into the app folder
  - Reconstruct the code for server.R and ui.R and save them into the app folder
  - Publish the app on shinyapps.io or shiny server





More Options with GGVIS

- Interactive Controls available:
  - input_slider(),
  - input_select() (dropdown menu)
  - input_checkbox(),
  - input_checkboxgroup() (multiple choice),
  - input_radiobuttons(),
  - input_text() and input_numeric() (only numerical input),
- Use "=" when assigning interactive controls to a var, ":=" is used only for static values
- Other layer options, besides layer_points() and layer_histograms():
  - layer_bars()
  - layer_ribbons() (filled space between two paths or a path and an axis)
  - layer_paths() (all points connected with a line)
  - layer_lines() (equivalent to *arrange(x) %>% layer_paths()*)
  - layer_smooths() (displays predictions with a line)
  - layer_rects() (rectangles)
  - layer_text() (displays text on the chart)
- To display multiple layers on one chart – pipe (%>%) them into each other, but include individual parameters (fill, size, span, etc.) inside their parentheses, instead of ggvis()

Some More Resources on GGVIS

- http://ggvis.rstudio.com/cookbook.html
- http://ggvis.rstudio.com/interactivity.html

More Resources

- Links to examples and tutorials on Slidify, rCharts, Shiny, etc.: http://www.r-bloggers.com/fantastic-presentations-from-r-using-slidify-and-rcharts/
- 100 interesting data sets: http://rs.io/2014/05/29/list-of-data-sets.html
- Google Dataset Engine: https://www.google.com/cse/publicurl?cx=002720237717066476899:v2wv26idk7m
- Missouri Data: https://data.mo.gov/

Thank you!

Questions?