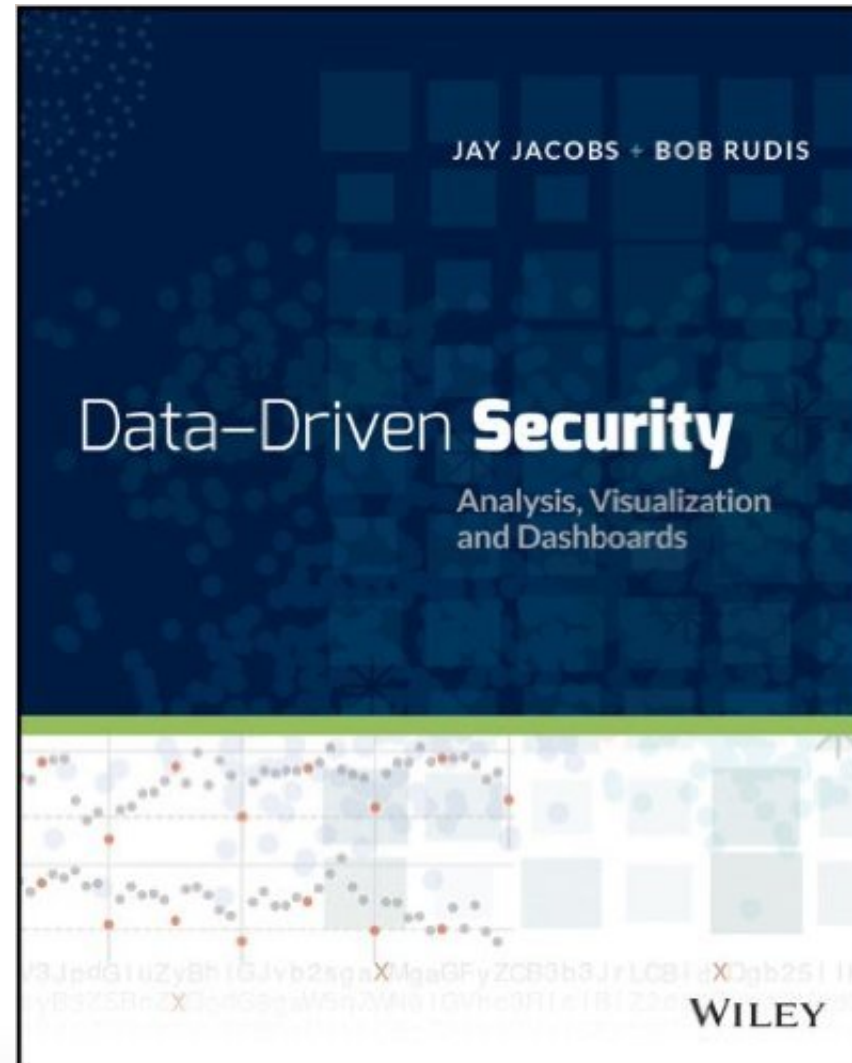


<http://dds.ec/> (blog, podcast and...)



Authored/contributed to 12 CRAN packages

| Package Name | Updated | Authors | Title | Vignettes | Task View |
|---------------------------------|------------|--|--|-----------|---------------------------------|
| cdcfluview | 2015-08-09 | Bob Rudis (@hrbrmstr) | Retrieve U.S. Flu Season Data from the CDC FluView Portal | | |
| cymruservices | 2015-07-28 | Bob Rudis [aut, cre] | Query Team Cymru IP Address, Autonomous System Number (ASN), Border Gateway Protocol (BGP), Bogon and Malware Hash Data Services | | |
| docxtractr | 2015-08-29 | Bob Rudis [aut, cre] | Extract Data Tables from Microsoft Word Documents | | |
| ggthemes | 2015-07-01 | Jeffrey B. Arnold [aut, cre], Gergely Daroczi [c... | Extra Themes, Scales and Geoms for 'ggplot2' | 1 | |
| iptools | 2015-07-23 | Bob Rudis [aut, cre], Oliver Key... | Manipulate, Validate and Resolve IP Addresses | 2 | |
| longurl | 2015-08-21 | Bob Rudis [aut, cre] | Expand Short URLs Using the 'LongURL' API | | |
| metricsgraphics | 2015-06-14 | Bob Rudis [aut, cre], Ali Almossawi [ctb, cph] (...) | Create Interactive Charts with the JavaScript 'MetricsGraphics' Library | 1 | |
| RBerkeley | 2015-07-29 | Jeffrey A. Ryan [aut, cre], Bob Rudis [ctb] | Oracle 'Berkeley DB' Interface for R | 1 | |
| slackr | 2014-09-08 | Bob Rudis (@hrbrmstr) & Jay Jacobs (@jayjacobs) | Send messages, images, R objects and files to Slack.com channels/users | | WebTechnologies |
| statebins | 2014-08-27 | Bob Rudis (@hrbrmstr) | statebins is an alternative to choropleth maps for USA States | | |
| urltools | 2015-08-31 | Oliver Keyes [aut, cre], Jay Jacobs [aut, cre], ... | Vectorised Tools for URL Handling and Parsing | 1 | WebTechnologies |
| viridis | 2015-09-14 | Simon Garnier [aut, cre], Noam Ross [ctb, cph] (...) | Matplotlib Default Color Map | 1 | |
| waffle | 2015-03-23 | Bob Rudis | Create Waffle Chart Visualizations in R | | |



Where Am I?

- <http://rud.is/b> Less infosec, more R & vis
- <http://twitter.com/hrbrmstr>
- <http://github.com/hrbrmstr>
- <http://stackoverflow.com/users/1457051/hrbrmstr>
- bob@rudis.net (if you like waiting for responses)



How You May View R

```
head(pressure, 3)
```

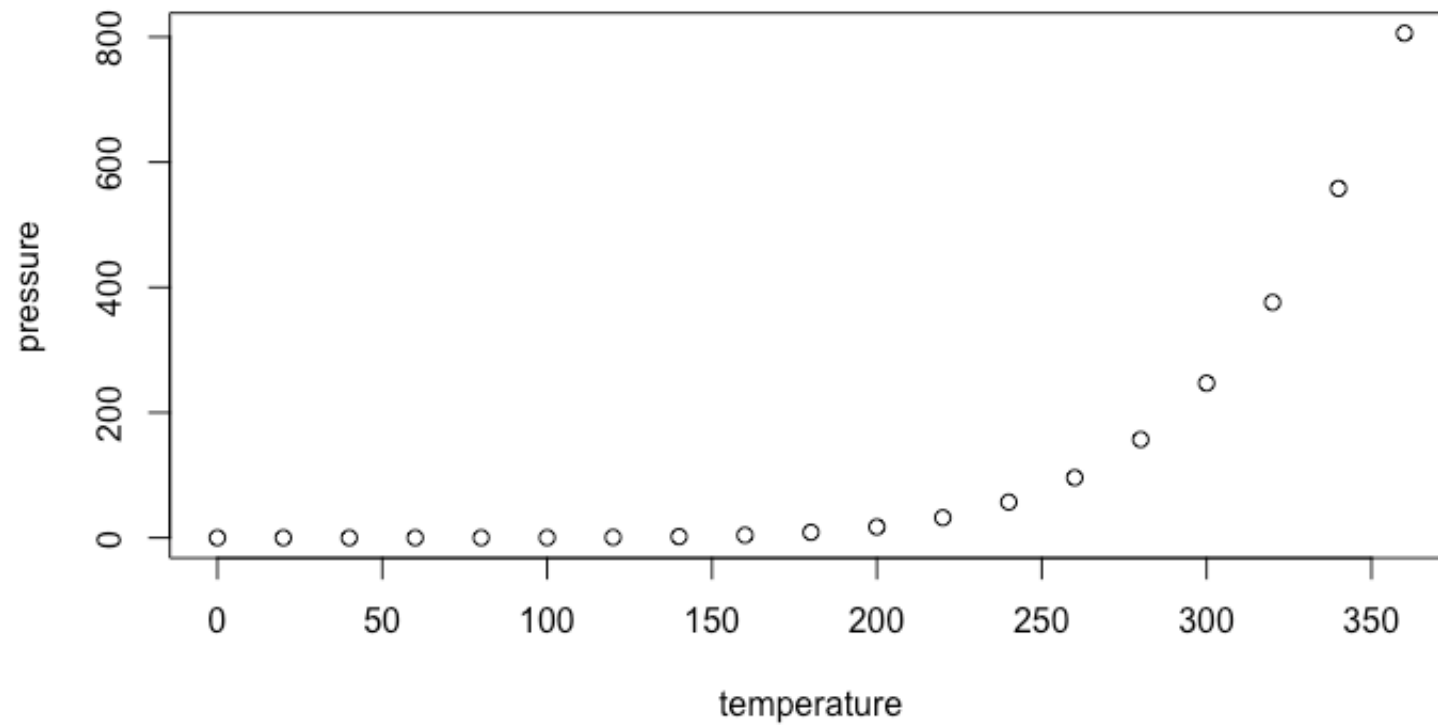
```
##    temperature pressure
## 1           0    0.0002
## 2          20    0.0012
## 3          40    0.0060
```

```
summary(pressure)
```

```
##    temperature      pressure
## Min.      :  0    Min.      : 0.0002
## 1st Qu.: 90    1st Qu.:  0.1800
## Median :180    Median :  8.8000
## Mean   :180    Mean   :124.3367
## 3rd Qu.:270    3rd Qu.:126.5000
## Max.   :360    Max.   :806.0000
```



How You May View R



How [I Hope] You Will View R



```
z <- seq(-10, 10, 0.01)
scatterplot3js(cos(z), sin(z), z, color=rainbow(length(z)))
```



How [I Hope] You Will View R




```
forceNetwork(Links = MisLinks, Nodes = MisNodes, Source = "source",  
             Target = "target", Value = "value", NodeID = "name",  
             Group = "group", opacity = 0.4)
```





```
dest <- factor(sprintf("%.2f:%.2f", flights[,3], flights[,4]))
freq <- sort(table(dest), decreasing=TRUE)
frequent_destinations <- names(freq)[1:10]
idx <- dest %in% frequent_destinations
frequent_flights <- flights[idx, ]
latlong <- unique(frequent_flights[,3:4])
earth <- system.file("images/world.jpg", package="threejs")
globejs(img=earth, lat=latlong[,1], long=latlong[,2],
        arcs=frequent_flights, arcsHeight=0.3, arcsLwd=2,
        arcsColor="#ffff00", arcsOpacity=0.15, atmosphere=TRUE)
```



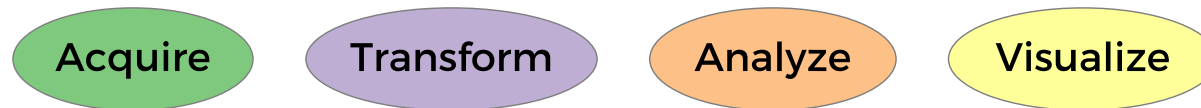
What is R?

What is R?

- R is a programming language
- R is statistical software
- R is an environment for interactive data analysis+visualization
- R is a community



What is R's Relationship with TGW?



- R can help you **access/acquire, clean and reformat data**
- R lets you **statistically analyze** data to find **insights**
- R enables **rapid, iterative prototyping** of visualizations to help **communicate** those insights
- R helps make those steps **organized** and **repeatable/reproducible**

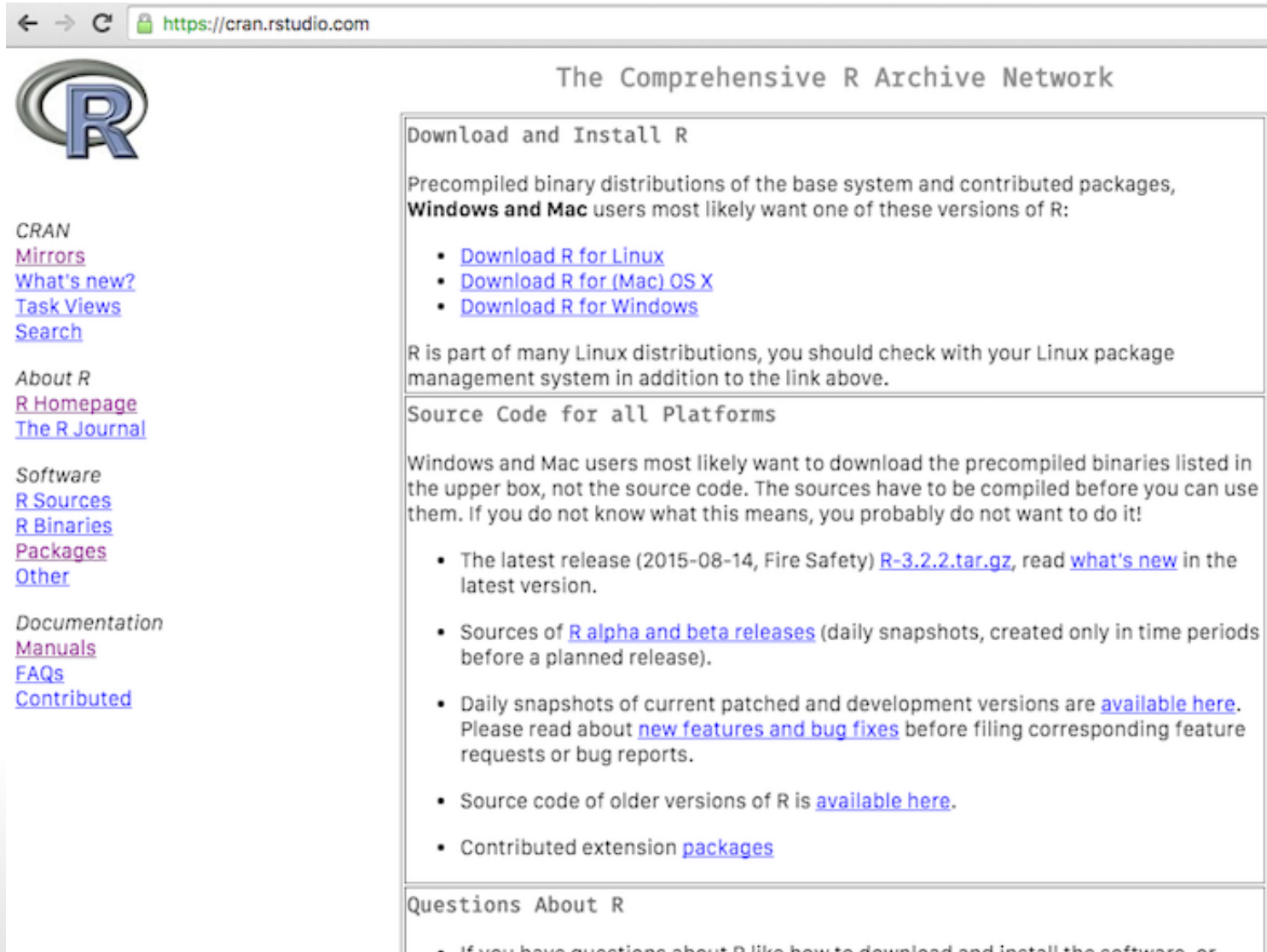


2013. SESAME WORKSHOP. ALL RIGHTS RESERVED



Getting Started with R

<http://cran.rstudio.com>



The screenshot shows the CRAN website in a web browser. The browser's address bar displays <https://cran.rstudio.com>. The website features the R logo on the left, a navigation menu with links to CRAN resources, and a main content area titled "The Comprehensive R Archive Network". The main content is divided into three sections: "Download and Install R", "Source Code for all Platforms", and "Questions About R".

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2015-08-14, Fire Safety) [R-3.2.2.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

Questions About R

- If you have questions about R like how to download and install the software, or

Getting Started with R/RStudio

<https://www.rstudio.com/products/rstudio/download/>

RStudio

ProductsResourcesPricingAbout UsBlog

Download RStudio

RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.

If you run R on a Linux server and want to enable users to remotely access RStudio using a web browser [please download RStudio Server](#).


Do you need support or a commercial license? [Check out our commercial offerings](#)

RStudio Desktop 0.99.484 — [Release Notes](#)

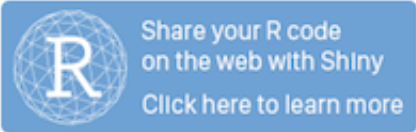
RStudio requires R 2.11.1 (or higher). If you don't already have R, you can download it [here](#).

Installers for Supported Platforms

| Installers | Size | Date | MD5 |
|---|---------|------------|----------------------------------|
| RStudio 0.99.484 - Windows Vista/7/8/10 | 73.9 MB | 2015-09-08 | 84eea3025538c811c0542c195c2f16e3 |
| RStudio 0.99.484 - Mac OS X 10.6+ (64-bit) | 56.2 MB | 2015-09-08 | a0ce0ad1f983d134b394358e3f4485e2 |
| RStudio 0.99.484 - Ubuntu 12.04+/Debian 8+ (32-bit) | 77.4 MB | 2015-09-08 | fe0c5d879c128c5d3d035bec73150fcc |
| RStudio 0.99.484 - Ubuntu 12.04+/Debian 8+ (64-bit) | 83.9 MB | 2015-09-08 | ee2a2ab6fce06e3936afd4b5968f7d0c |
| RStudio 0.99.484 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (32-bit) | 76.8 MB | 2015-09-08 | 61dcfadd2eb5135e2ff0482dcae3e385 |



CURIOUS
WITH COMPANIES
UPGRADED?



Share your R code
on the web with Shiny
[Click here to learn more](#)



RStudio

The screenshot displays the RStudio environment with three main panels:

- Source Editor (Left):** Contains an R script with the following code:

```
1 library(threejs)
2
3 z <- seq(-10, 10, 0.01)
4 x <- cos(z)
5 y <- sin(z)
6
7 scatterplot3js(x, y, z,
8               color=rainbow(length(z)))
9
```

Syntax-aware & data-structure aware editor
- Viewer (Top Right):** Displays a 3D plot of a helix curve, colored with a rainbow gradient, plotted against x, y, and z axes.
- Console (Bottom Right):** Shows the execution output of the script:

```
> library(threejs)
>
> z <- seq(-10, 10, 0.01)
> x <- cos(z)
> y <- sin(z)
>
> scatterplot3js(x, y, z,
+               color=rainbow(length(z)))
>
```

Built-in browser/viewer

Instant feedback



RStudio

ex1.R x mtcars x

Filter

| | mpg | cyl | disp | hp | drat | wt | qsec | vs | am | gear | car |
|---------------------|------|-----|-------|-----|------|-------|-------|----|----|------|-----|
| Mazda RX4 | 21.0 | 6 | 160.0 | 110 | 3.90 | 2.620 | 16.46 | 0 | 1 | 4 | |
| Mazda RX4 Wag | 21.0 | 6 | 160.0 | 110 | 3.90 | 2.875 | 17.02 | 0 | 1 | 4 | |
| Datsun 710 | 22.8 | 4 | 108.0 | 93 | 3.85 | 2.320 | 18.61 | 1 | 1 | 4 | |
| Hornet 4 Drive | 21.4 | 6 | 258.0 | 110 | 3.08 | 3.215 | 19.44 | 1 | 0 | 3 | |
| Hornet Sportabout | 18.7 | 8 | 360.0 | 175 | 3.15 | 3.440 | 17.02 | 0 | 0 | 3 | |
| Valiant | 18.1 | 6 | 225.0 | 105 | 2.76 | 3.460 | 20.22 | 1 | 0 | 3 | |
| Duster 360 | 14.3 | 8 | 360.0 | 245 | 3.21 | 3.570 | 15.84 | 0 | 0 | 3 | |
| Merc 240D | 24.4 | 4 | 146.7 | 62 | 3.69 | 3.190 | 20.00 | 1 | 0 | 4 | |
| Merc 230 | 22.8 | 4 | 140.8 | 95 | 3.92 | 3.150 | 22.90 | 1 | 0 | 4 | |
| Merc 280 | 19.2 | 6 | 167.6 | 123 | 3.92 | 3.440 | 18.30 | 1 | 0 | 4 | |
| Merc 280C | 17.8 | 6 | 167.6 | 123 | 3.92 | 3.440 | 18.90 | 1 | 0 | 4 | |
| Merc 450SE | 16.4 | 8 | 275.8 | 180 | 3.07 | 4.070 | 17.40 | 0 | 0 | 3 | |
| Merc 450SL | 17.3 | 8 | 275.8 | 180 | 3.07 | 3.730 | 17.60 | 0 | 0 | 3 | |
| Merc 450SLC | 15.2 | 8 | 275.8 | 180 | 3.07 | 3.780 | 18.00 | 0 | 0 | 3 | |
| Cadillac Fleetwood | 10.4 | 8 | 472.0 | 205 | 2.93 | 5.250 | 17.98 | 0 | 0 | 3 | |
| Lincoln Continental | 10.4 | 8 | 460.0 | 215 | 3.00 | 5.424 | 17.82 | 0 | 0 | 3 | |
| Chrysler Imperial | 14.7 | 8 | 440.0 | 230 | 3.23 | 5.345 | 17.42 | 0 | 0 | 3 | |
| Fiat 128 | 32.4 | 4 | 78.7 | 66 | 4.08 | 2.200 | 19.47 | 1 | 1 | 4 | |
| Honda Civic | 30.4 | 4 | 75.7 | 52 | 4.93 | 1.615 | 18.52 | 1 | 1 | 4 | |
| Toyota Corolla | 33.9 | 4 | 71.1 | 65 | 4.22 | 1.835 | 19.90 | 1 | 1 | 4 | |
| Toyota Corona | 21.5 | 4 | 120.1 | 97 | 3.70 | 2.465 | 20.01 | 1 | 0 | 3 | |
| Dodge Challenger | 15.5 | 8 | 318.0 | 150 | 2.76 | 3.520 | 16.87 | 0 | 0 | 3 | |
| AMC Javelin | 15.2 | 8 | 304.0 | 150 | 3.15 | 3.435 | 17.30 | 0 | 0 | 3 | |
| Camaro Z28 | 13.3 | 8 | 350.0 | 245 | 3.73 | 3.840 | 15.41 | 0 | 0 | 3 | |
| Pontiac Firebird | 19.2 | 8 | 400.0 | 175 | 3.08 | 3.845 | 17.05 | 0 | 0 | 3 | |
| Fiat X1-9 | 27.3 | 4 | 79.0 | 66 | 4.08 | 1.935 | 18.90 | 1 | 1 | 4 | |
| Porsche 914-2 | 26.0 | 4 | 120.3 | 91 | 4.43 | 2.140 | 16.70 | 0 | 1 | 5 | |
| Lotus Europa | 30.4 | 4 | 95.1 | 113 | 3.77 | 1.513 | 16.90 | 1 | 1 | 5 | |
| Ford Pantera L | 15.8 | 8 | 351.0 | 264 | 4.22 | 3.170 | 14.50 | 0 | 1 | 5 | |
| Ferrari Dino | 19.7 | 6 | 145.0 | 175 | 3.62 | 2.770 | 15.50 | 0 | 1 | 5 | |
| Maserati Bora | 15.0 | 8 | 301.0 | 335 | 3.54 | 3.570 | 14.60 | 0 | 1 | 5 | |

Showing 1 to 32 of 32 entries

History

Environment Files Plots Packages Help Viewer

Import Dataset

Global Environment

| Name | Type | Length | Size | Value |
|--------|------------|--------|---------|-------------------------------------|
| mtcars | data.frame | 11 | 6.6 KB | 32 obs. of 11 variables |
| x | numeric | 2001 | 15.7 KB | num [1:2001] -0.839 -0.844 -0.85... |
| y | numeric | 2001 | 15.7 KB | num [1:2001] 0.544 0.536 0.527 0... |
| z | numeric | 2001 | 15.7 KB | num [1:2001] -10 -9.99 -9.98 -9... |

Console ~ /Development/

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 ~/Development/.RData]

> library(threejs)
>
> z <- seq(-10, 10, 0.01)
> x <- cos(z)
> y <- sin(z)
>
> scatterplot3js(x, y, z,
+ color=rainbow(length(z)))
>
> data(mtcars)
> View(mtcars)
>



R is Familiar

- Dynamic (like JavaScript & Python)
- Has variables (like JavaScript & Python)
- ...functions (like JavaScript & Python)
- ...loops (like JavaScript & Python)
- ...regular expressions (like JavaScript & Python)
- ...and, help from friends (packages) (like Node or Python modules)



R is Different

It's "vectorized" (think `map()` or `[for]`)

```
a <- 1:10  
sum(a)
```

```
## [1] 55
```



R is Different

It *really* likes something called "data frames" (Python does too, now)

```
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1         5.1         3.5         1.4         0.2   setosa
## 2         4.9         3.0         1.4         0.2   setosa
## 3         4.7         3.2         1.3         0.2   setosa
## 4         4.6         3.1         1.5         0.2   setosa
## 5         5.0         3.6         1.4         0.2   setosa
## 6         5.4         3.9         1.7         0.4   setosa
```



R is Different

It has an affinity for arcane punctuation:

```
`huh?` <- iris$Sepal.Length[[2]] / iris[2,1] * 3 %>% sqrt()  
print(`huh?`)
```

```
## [1] 1.732051
```



R is Different

And, complex+efficient algorithms can be confusing:

```
dat <- readLines(textConnection(" 3   weeks,   2   days,  4 hours
 4 week,  6 days, 12 hours
4 day, 3 hours
 7 hours
8  hour"))

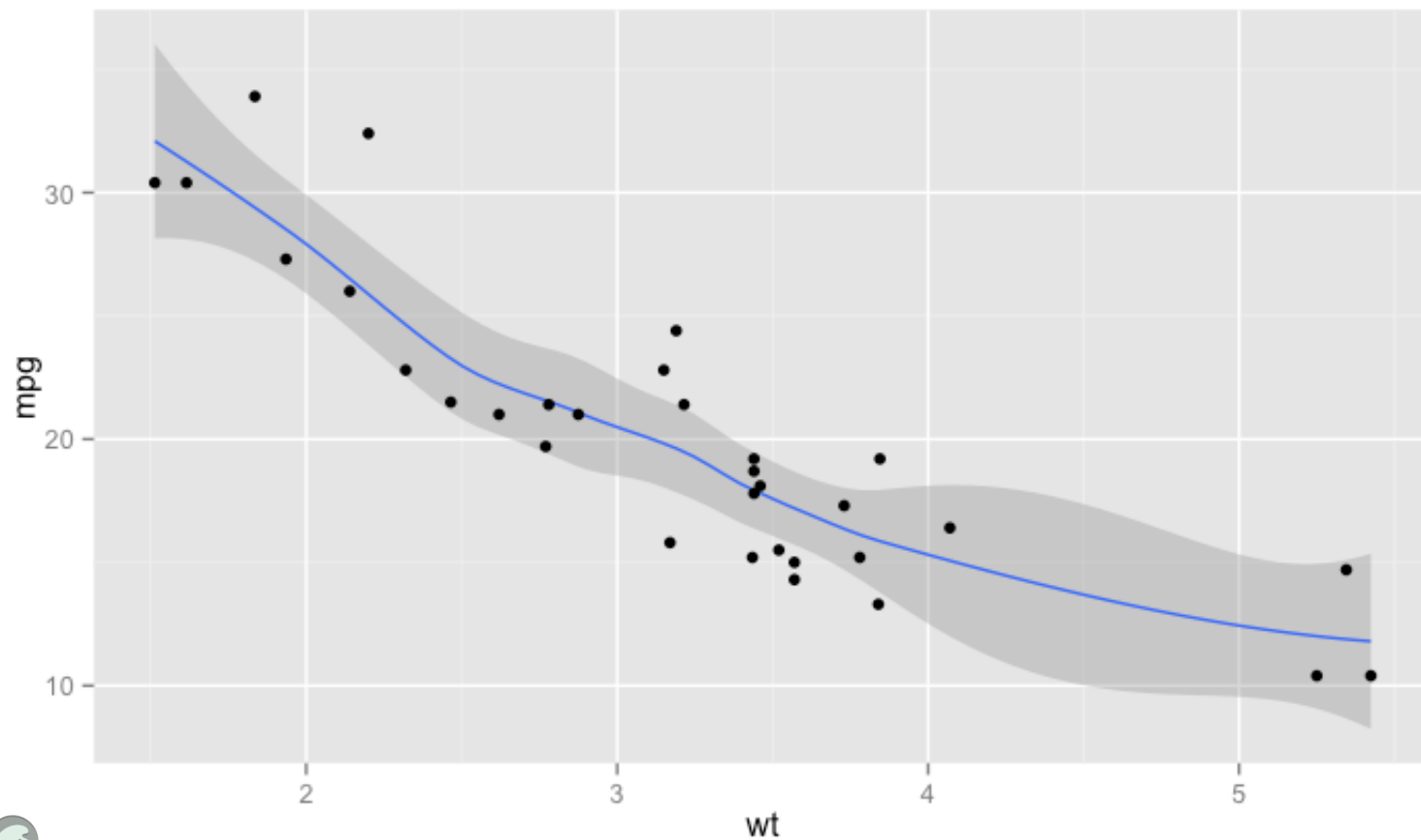
sapply(str_split(str_trim(dat), "[ ]*"), function(x) {
  sum(sapply(x, function(y) {
    bits <- str_split(str_trim(y), "[ ]+")[[1]]
    duration(as.numeric(bits[1]), bits[2])
  })) / 3600
})
```

```
## [1] 556 828 99 7 8
```



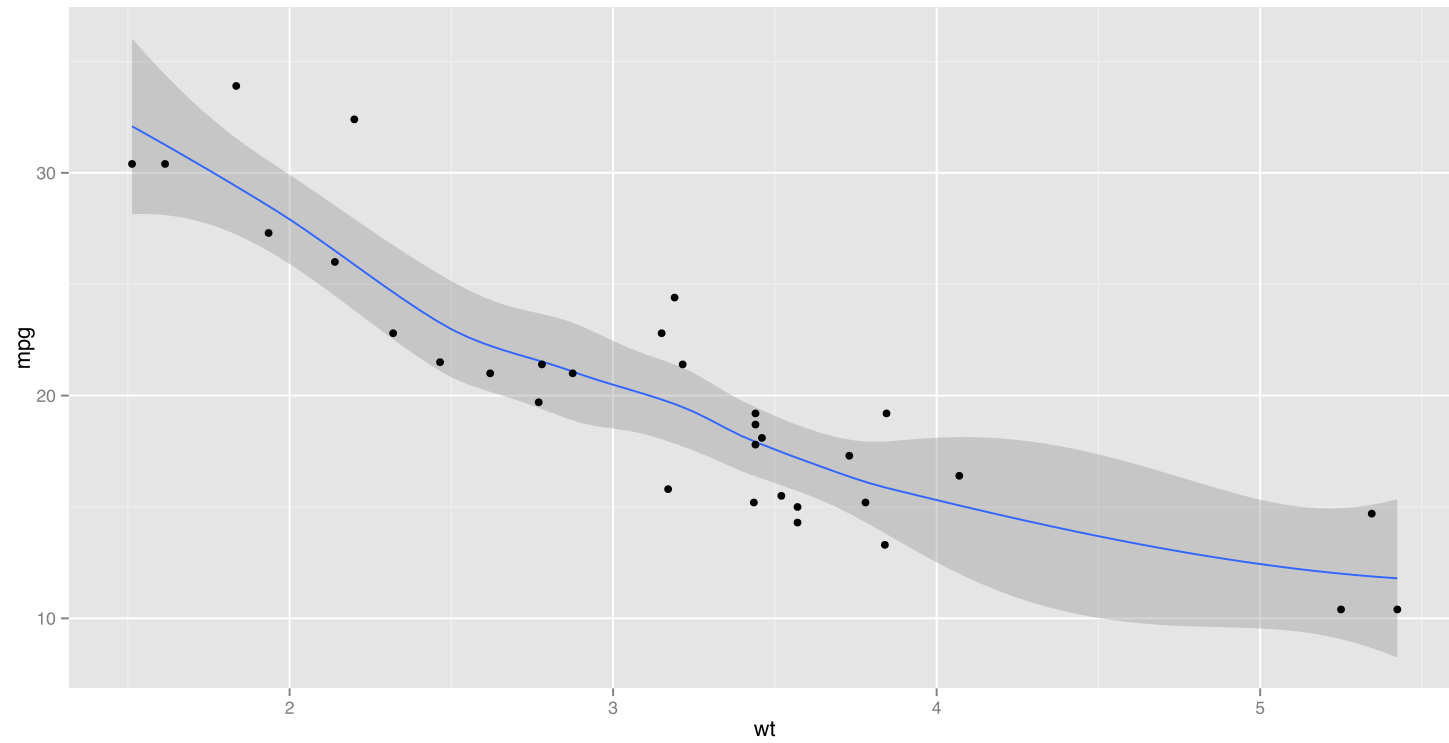
R & The Graphical Web

```
library(ggplot2)
g1 <- ggplot(mtcars, aes(x=wt, y=mpg)) + geom_smooth() + geom_point()
print(g1)
```



This is all it takes to turn that plot into an editable/usable SVG graphic:

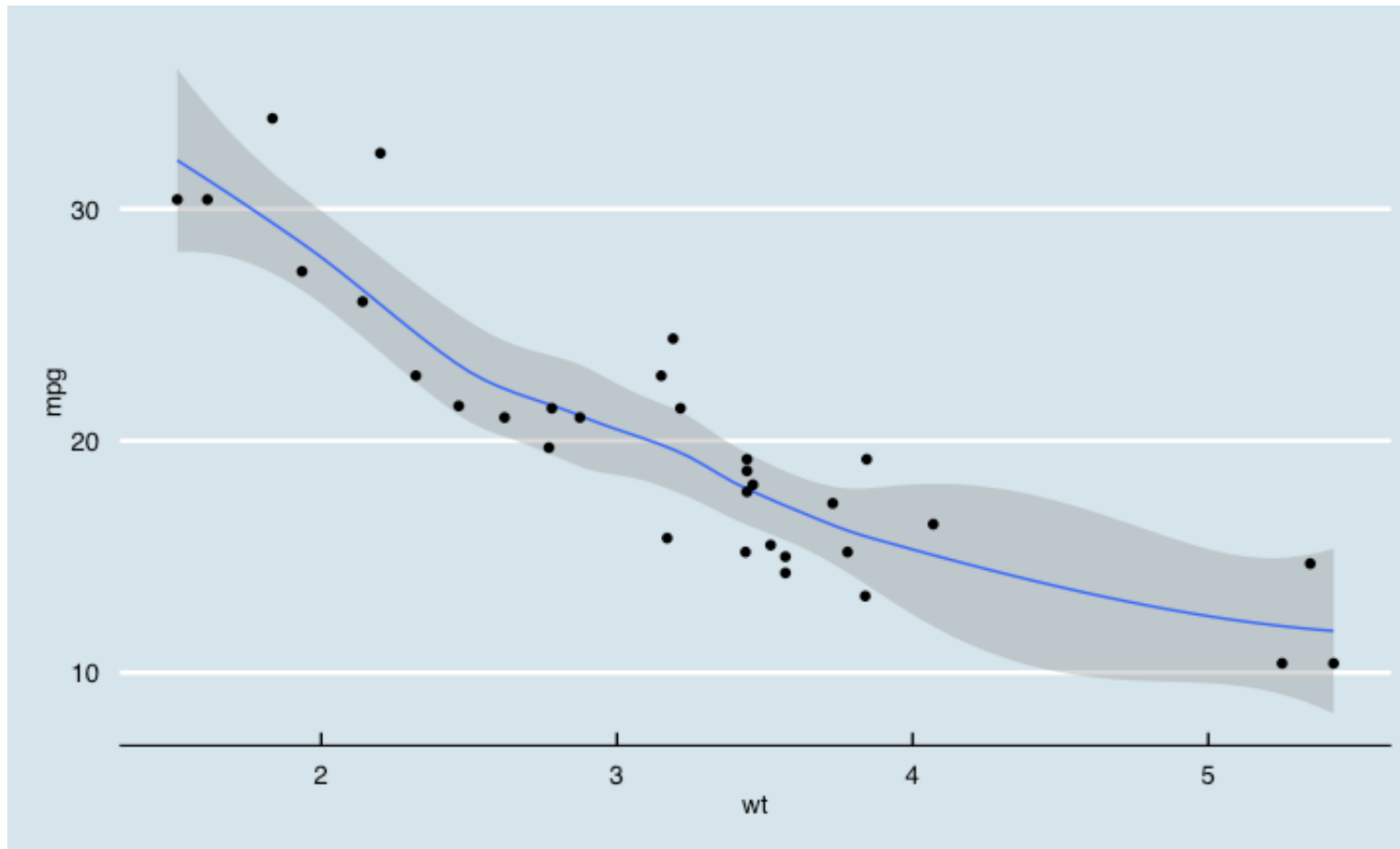
```
ggsave(g1, "img/g1.svg")
```



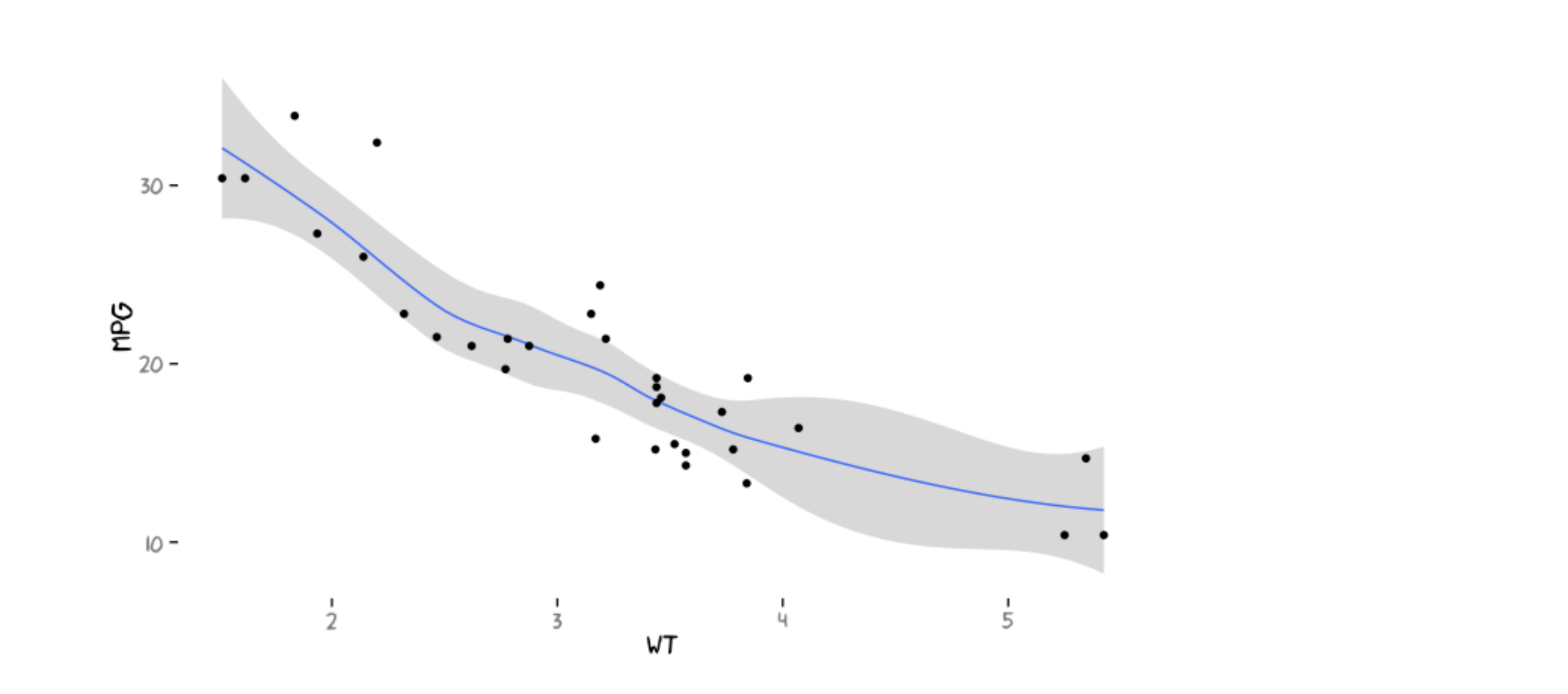
```
<?xml version="1.0" encoding="UTF-8"?>
<svg xmlns="http://www.w3.org/2000/svg"
      xmlns:xlink="http://www.w3.org/1999/xlink"
      width="819pt" height="425pt"
      viewBox="0 0 819 425" version="1.1">
<defs>
<g>
<symbol overflow="visible" id="glyph0-0">
<path style="stroke:none;"
      d="M 0.3125 0 L 0.3125 -6.875 L 5.765625 -6.875 L 5.765625 0 Z M 4.90625 -0.859375 L 4.90625 0.859375 L 5.765625 0.859375 L 5.765625 -0.859375 Z" />
</symbol>
<symbol overflow="visible" id="glyph0-1">
<path style="stroke:none;"
      d="M 0.921875 -4.75 L 0.921875 -5.390625 C 1.523438 -5.453125 1.945312 -5.550781 2.1875 -5.550781 L 2.1875 -4.75 L 0.921875 -4.75 Z" />
</symbol>
</g>
</svg>
```



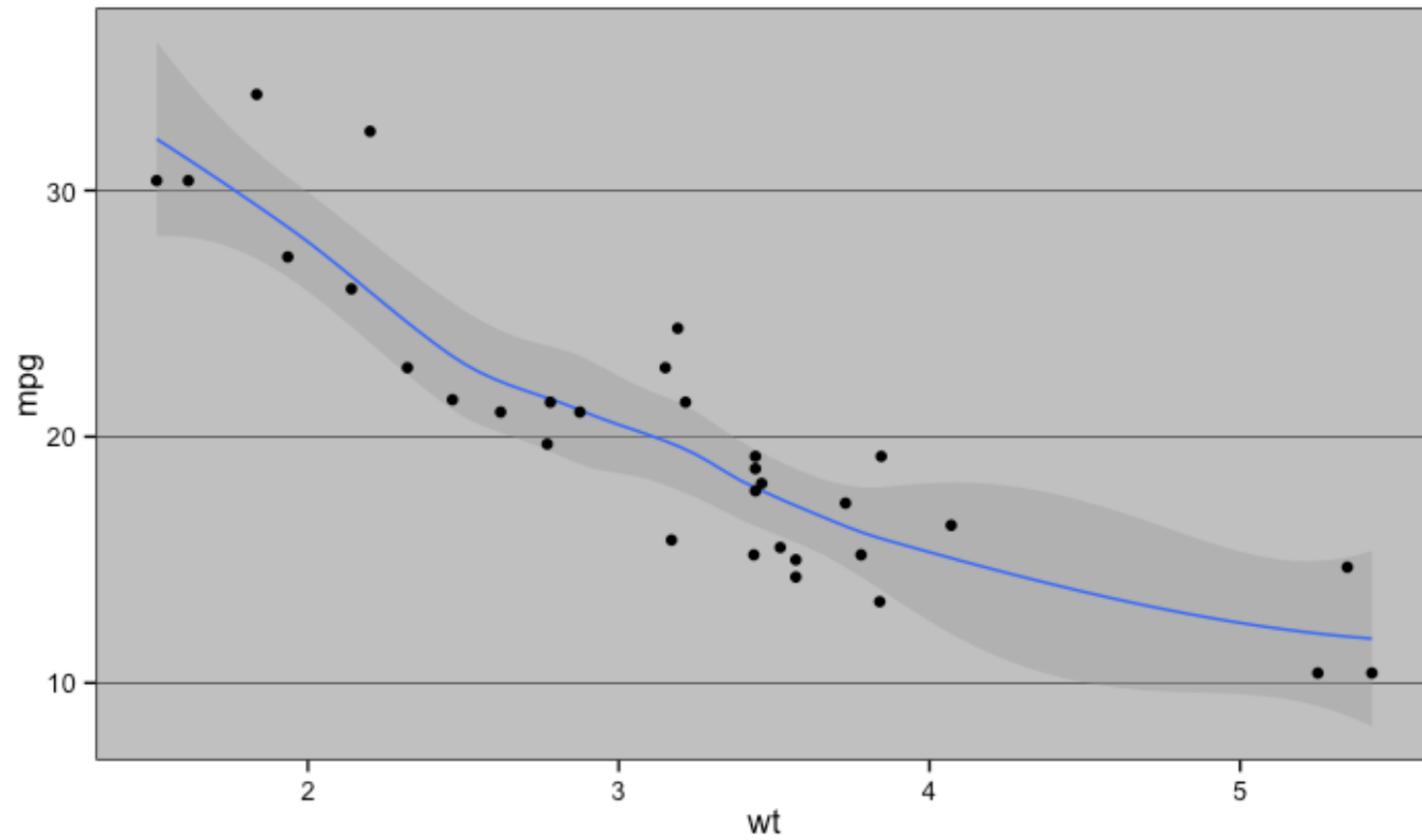
```
library(ggthemes)
g1 + theme_economist()
```



```
library(xkcd)
g1 + theme_xkcd()
```



```
g1 + theme_excel()
```



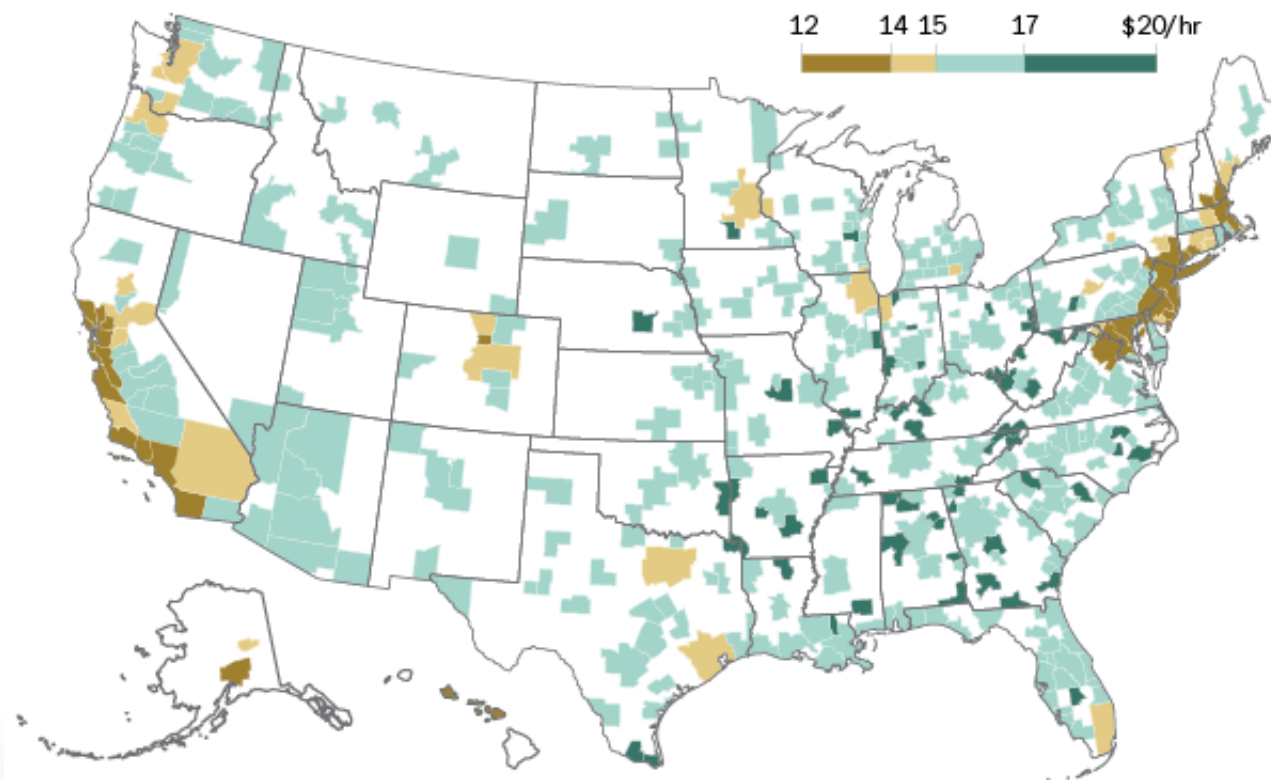
Doing Real Work

Doing Real Work

<http://bit.ly/pewmapdemo+>

Where Paychecks Stretch the Most, and Least

Estimated real purchasing power of a national \$15 hourly wage, by metropolitan area

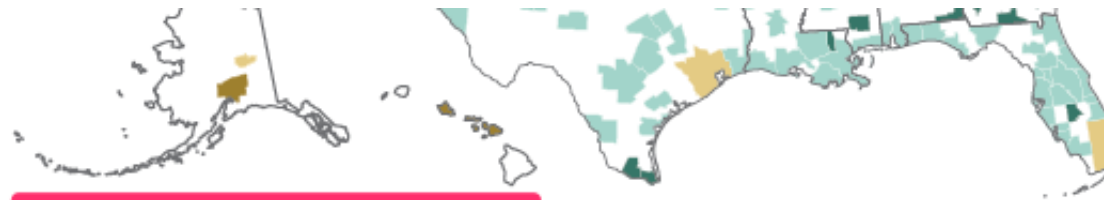


Note: Based on 2013 regional price parities for metropolitan statistical areas.
Source: Bureau of Economic Analysis, Pew Research Center analysis

PEW RESEARCH CENTER

Doing Real Work

Let's find this data!



Note: Based on 2013 regional price parities for metropolitan statistical areas.
Source: Bureau of Economic Analysis, Pew Research Center analysis

PEW RESEARCH CENTER

Doing Real Work

Let's find this data!

U.S. Department of Commerce

Bureau of Economic Analysis

Home

National

International

Regional

Industry

Interactive Data

National Data

Industry Data

International Data

GDP & Personal Income

Fixed Assets

GDP-by-industry

Input-Output

Int'l Trade

Regional Data

GDP & Personal Income

Table

Major Area

+

States

Metropolitan Statistical Area

State Metro/Nonmetro portions

Next Step



Doing Real Work

Let's find this data!

U.S. Department of Commerce

Bureau of Economic Analysis

Home

National

International

Regional

Industry

Interactive Data

National Data

Industry Data

International Data

GDP & Personal Income

Fixed Assets

GDP-by-industry

Input-Output

Int'l Transactions, Services, & IIP

Dir

Regional Data

GDP & Personal Income

Table

Major Area

Area/Statistic +

Select one or more areas, a statistic, and a unit of measure

Area

All Areas

United States (Nonmetropolitan Portion)

Abilene, TX (Metropolitan Statistical Area)

Akron, OH (Metropolitan Statistical Area)

Albany, GA (Metropolitan Statistical Area)

Albany, OR (Metropolitan Statistical Area)

Multiple selections allowed.

Doing Real Work

Let's find this data!

U.S. Department of Commerce

Bureau of Economic Analysis

Home

National

International

Regional

Industry

Interactive Data

National Data

Industry Data

International Data

GDP & Personal Income

Fixed Assets

GDP-by-industry

Input-Output

International Trade

Regional Data

GDP & Personal Income

Table

Major Area

Area/Statistic

Time Period +

Time Period

All Years

2013

2012

2011

2010

2009

2008

i

Multiple selections allowed.

Next Step

Doing Real Work

Almost done!

Regional Data

Sign In

Register

GDP & Personal Income

Table

Major Area

Area/Statistic

Time Period

Selected Data +

RPP1 [Regional Price Parities](#)

[RPPs: All items](#) (index)

Metropolitan Statistical Area

MODIFY

CHART

MAP

DOWNLOAD

PRINT

| GeoFips | GeoName | 2013 |
|---------|---|-------|
| 00999 | United States (Nonmetropolitan Portion) | 87.7 |
| 10180 | Abilene, TX (Metropolitan Statistical Area) | 91.3 |
| 10420 | Akron, OH (Metropolitan Statistical Area) | 88.9 |
| 10500 | Albany, GA (Metropolitan Statistical Area) | 84.3 |
| 10540 | Albany, OR (Metropolitan Statistical Area) | 93.5 |
| 10580 | Albany-Schenectady-Troy, NY (Metropolitan Statistical Area) | 99.0 |
| 10740 | Albuquerque, NM (Metropolitan Statistical Area) | 97.1 |
| 10780 | Alexandria, LA (Metropolitan Statistical Area) | 88.4 |
| 10900 | Allentown-Bethlehem-Easton, PA-NJ (Metropolitan Statistical Area) | 100.0 |

Doing Real Work

Done! *(kinda)*

actions, Services, & IIP

Direct Investment & MNEs

GDP & Personal Income

Download

XLS

Download your table in Excel format.

CSV

Download your table in CSV format.

PDF

Download your table in PDF format.

Close

Personal Income

Personal Income

Major Metropolitan Statistical Areas

Regional Data

Items (in millions of dollars)

an Statistical Area

United States

Abilene, TX

Akron, OH

Albany, GA (Metropolitan Statistical Area)

84.3



Doing Real Work

Done! *(kinda)*

| | | |
|-------------------------------|---|-------|
| RPP1 Regional Price Parities | | |
| RPPs: All Items (Index) | | |
| Bureau of Economic Analysis | | |
| Metropolitan Statistical Area | | |
| GeoFips | GeoName | 2013 |
| 00999 | United States (Nonmetropolitan Portion) | 87.7 |
| 10180 | Abilene, TX (Metropolitan Statistical Area) | 91.3 |
| 10420 | Akron, OH (Metropolitan Statistical Area) | 88.9 |
| 10500 | Albany, GA (Metropolitan Statistical Area) | 84.3 |
| 10540 | Albany, OR (Metropolitan Statistical Area) | 93.5 |
| 10580 | Albany-Schenectady-Troy, NY (Metropolitan Statistical Area) | 99.0 |
| | | |



```
dat <- read.csv("data/download.csv", skip=4, header=TRUE, stringsAsFactors=FALSE)
dat <- head(dat, -2)
head(dat)
```

```
##      GeoFips                                          GeoName
## 1      00999                      United States (Nonmetropolitan Portion)
## 2      10180                Abilene, TX (Metropolitan Statistical Area)
## 3      10420                Akron, OH (Metropolitan Statistical Area)
## 4      10500                Albany, GA (Metropolitan Statistical Area)
## 5      10540                Albany, OR (Metropolitan Statistical Area)
## 6      10580 Albany-Schenectady-Troy, NY (Metropolitan Statistical Area)
##      X2013
## 1      87.7
## 2      91.3
## 3      88.9
## 4      84.3
## 5      93.5
## 6      99.0
```



```
dat$GeoName <- gsub(" \\(Metropolitan Statistical Area\\)", "",  
                    dat$GeoName)  
dat$GeoFips <- sprintf("%05d", as.numeric(dat$GeoFips))  
head(dat)
```

| ## | GeoFips | GeoName | X2013 |
|------|---------|---|-------|
| ## 1 | 00999 | United States (Nonmetropolitan Portion) | 87.7 |
| ## 2 | 10180 | Abilene, TX | 91.3 |
| ## 3 | 10420 | Akron, OH | 88.9 |
| ## 4 | 10500 | Albany, GA | 84.3 |
| ## 5 | 10540 | Albany, OR | 93.5 |
| ## 6 | 10580 | Albany-Schenectady-Troy, NY | 99.0 |



Doing Real Work

Appendix A - RegionalData (statistics by state, county, and MSA)

The new datasets *RegionalIncome* and *RegionalProduct* have more statistics and industry detail than the *RegionalData* dataset. See Appendices I and J. Although *RegionalData* is still valid, we encourage users to switch to the more comprehensive datasets *RegionalIncome* and *RegionalProduct*.

The *RegionalData* dataset contains estimates from the Regional Economic Accounts. These include estimates of GDP by state and metropolitan area; estimates of personal income and employment by state, metropolitan area, and county; and regional price parities by state and MSA.

| RegionalData Request Parameters | | | | | | |
|---------------------------------|--------|--------------------------------------|----------|--------------------------|------------------------|---------|
| Parameter Name | Type | Description | Required | Multiple Values Accepted | "All" value | Default |
| KeyCode | String | The code for the statistic requested | Yes | No | | |
| GeoFips | String | The state, county or MSA code | No | Yes | STATE or COUNTY or MSA | STATE |
| Year | String | Year requested | No | Yes | ALL | ALL |

Doing Real Work

`http://bea.gov/api/data/?UserID=xxxxxx-xxxx-xxxx-xxxx-xxxxxxxx&
method=GetData&datasetname=RegionalData&KeyCode=RPPALL_MI&
Year=2013&&ResultFormat=json"`

```
64     "Ordinal": "7",  
65     "Name": "DataValue",  
66     "DataType": "numeric",  
67     "IsValue": "1"  
68   }],  
69   "Data": [{  
70     "GeoFips": "10180",  
71     "GeoName": "Abilene, TX (Metropolitan Statistical Area)",  
72     "Code": "RPPALL_MI",  
73     "TimePeriod": "2013",  
74     "CL_UNIT": "IDX",  
75     "UNIT_MULT": "0",  
76     "DataValue": "91.3"  
77   }, {  
78     "GeoFips": "10420",  
79     "GeoName": "Akron, OH (Metropolitan Statistical Area)",  
80     "Code": "RPPALL_MI",  
81     "TimePeriod": "2013",  
82     "CL_UNIT": "IDX",  
83     "UNIT_MULT": "0",  
84     "DataValue": "88.9"  
85   }, {  
86     "GeoFips": "10500",  
87     "GeoName": "Albany, GA (Metropolitan Statistical Area)",  
88     "Code": "RPPALL_MI",  
89     "TimePeriod": "2013",  
90     "CL_UNIT": "IDX",
```

Doing Real Work

```
library(jsonlite)
dat <- readJSON("that horrible URL")
```

```
dat <- dat$BEAAPI$Results$Data
dat$X2013 <- as.numeric(dat$DataValue)
dat$GeoName <- gsub("\\(Metropolitan Statistical Area\\)", "",
                  dat$GeoName)
dat$GeoFips <- sprintf("%05d", as.numeric(dat$GeoFips))
head(dat[,c(1,2,8)])
```

| ## | GeoFips | GeoName | X2013 |
|------|---------|-----------------------------|-------|
| ## 1 | 10180 | Abilene, TX | 91.3 |
| ## 2 | 10420 | Akron, OH | 88.9 |
| ## 3 | 10500 | Albany, GA | 84.3 |
| ## 4 | 10540 | Albany, OR | 93.5 |
| ## 5 | 10580 | Albany-Schenectady-Troy, NY | 99.0 |
| ## 6 | 10740 | Albuquerque, NM | 97.1 |



Doing Real Work

```
library(httr)
response <- GET("http://bea.gov/api/data/",
               query=list(
                 UserID=Sys.getenv("BEA_API_TOKEN"),
                 method="GetData",
                 datasetname="RegionalData",
                 KeyCode="RPPALL_MI",
                 Year="2013",
                 ResultFormat="json"
               ))
dat <- fromJSON(content(response, as="text"))
```



Same cleanup as we did in the raw URL version

```
dat <- dat$BEAAPI$Results$Data
dat$X2013 <- as.numeric(dat$DataValue)
dat$GeoName <- gsub("\\(Metropolitan Statistical Area\\)", "",
                  dat$GeoName)
dat$GeoFips <- sprintf("%05d", as.numeric(dat$GeoFips))
head(dat[,c(1,2,8)])
```

| ## | GeoFips | GeoName | X2013 |
|------|---------|-----------------------------|-------|
| ## 1 | 10180 | Abilene, TX | 91.3 |
| ## 2 | 10420 | Akron, OH | 88.9 |
| ## 3 | 10500 | Albany, GA | 84.3 |
| ## 4 | 10540 | Albany, OR | 93.5 |
| ## 5 | 10580 | Albany-Schenectady-Troy, NY | 99.0 |
| ## 6 | 10740 | Albuquerque, NM | 97.1 |



Doing Real Work

```
library(noncensus)
data(counties)
xlate <- data.frame(fipscounty=sprintf("%s%s", counties$state_fips,
                                       counties$county_fips),
                   cbsa=counties$CBSA,
                   stringsAsFactors=FALSE)
dat <- merge(dat[,c(1,2,8)], xlate[,c("cbsa", "fipscounty")],
             by.x="GeoFips", by.y="cbsa")
head(dat)
```

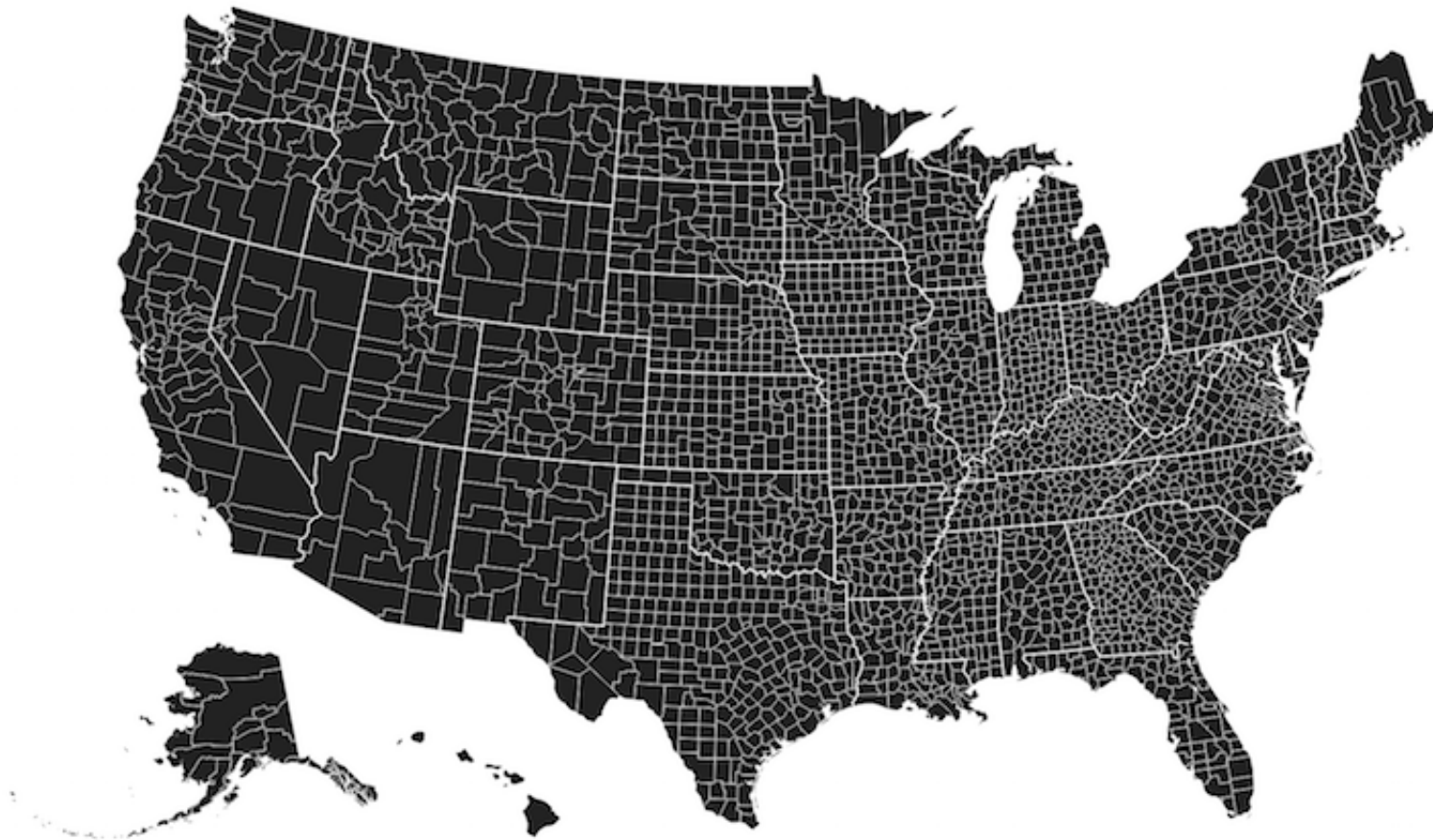
| ## | GeoFips | GeoName | X2013 | fipscounty |
|------|---------|-------------|-------|------------|
| ## 1 | 10180 | Abilene, TX | 91.3 | 48441 |
| ## 2 | 10180 | Abilene, TX | 91.3 | 48253 |
| ## 3 | 10180 | Abilene, TX | 91.3 | 48059 |
| ## 4 | 10420 | Akron, OH | 88.9 | 39133 |
| ## 5 | 10420 | Akron, OH | 88.9 | 39153 |
| ## 6 | 10500 | Albany, GA | 84.3 | 13177 |



Doing Real Work

mbostock's block #4090848 November 16, 2012

U.S. States TopoJSON



What's the Plan?

- Display a US map county choropleth with the counties filled according the RPP value
- The counties are not all represented and we don't need a billion small polygons left over so we'll outline the states for context
- We'd like to add contextual information via popup to show the discrete data value and the name of the metro area
- A legend would be good



Options (without R)

- Raw JS or jQuery + CSS
- Straight D3
- Kartograph
- Datamaps
- Leaflet [Ex: <http://leafletjs.com/examples/choropleth-example.html>]
(Perfect data. ~170 lines)



Getting Work Done : R + Leaflet

Get map data

```
library(rgdal)
URL <- "http://bl.ocks.org/mbostock/raw/4090846/us.json"
fil <- sprintf("data/%s", basename(URL))
if (!file.exists(fil)) download.file(URL, fil)

# read state borders from the file
states <- readOGR(fil, "states", stringsAsFactors=FALSE,
                  verbose=FALSE)

# read county borders from the file
county <- readOGR(fil, "counties", stringsAsFactors=FALSE,
                  verbose=FALSE)
```



Getting Work Done : R + Leaflet

We don't want to display all the counties, so we'll subtract out the ones that aren't in our data set.

```
rpp_counties <- subset(county, id %in% dat$fipscounty)
rpp_counties <- merge(rpp_counties, dat,
                      by.x="id",
                      by.y="fipscounty",
                      all.x=TRUE)
```



Getting Work Done : R + Leaflet

We need to setup the color scale (really similar to how you'd do it in JS)

```
library(leaflet)
pal <- colorBin("BrBG", range(rpp_counties$X2013), bins=5)
rpp_counties$color <- pal(rpp_counties$X2013)
```

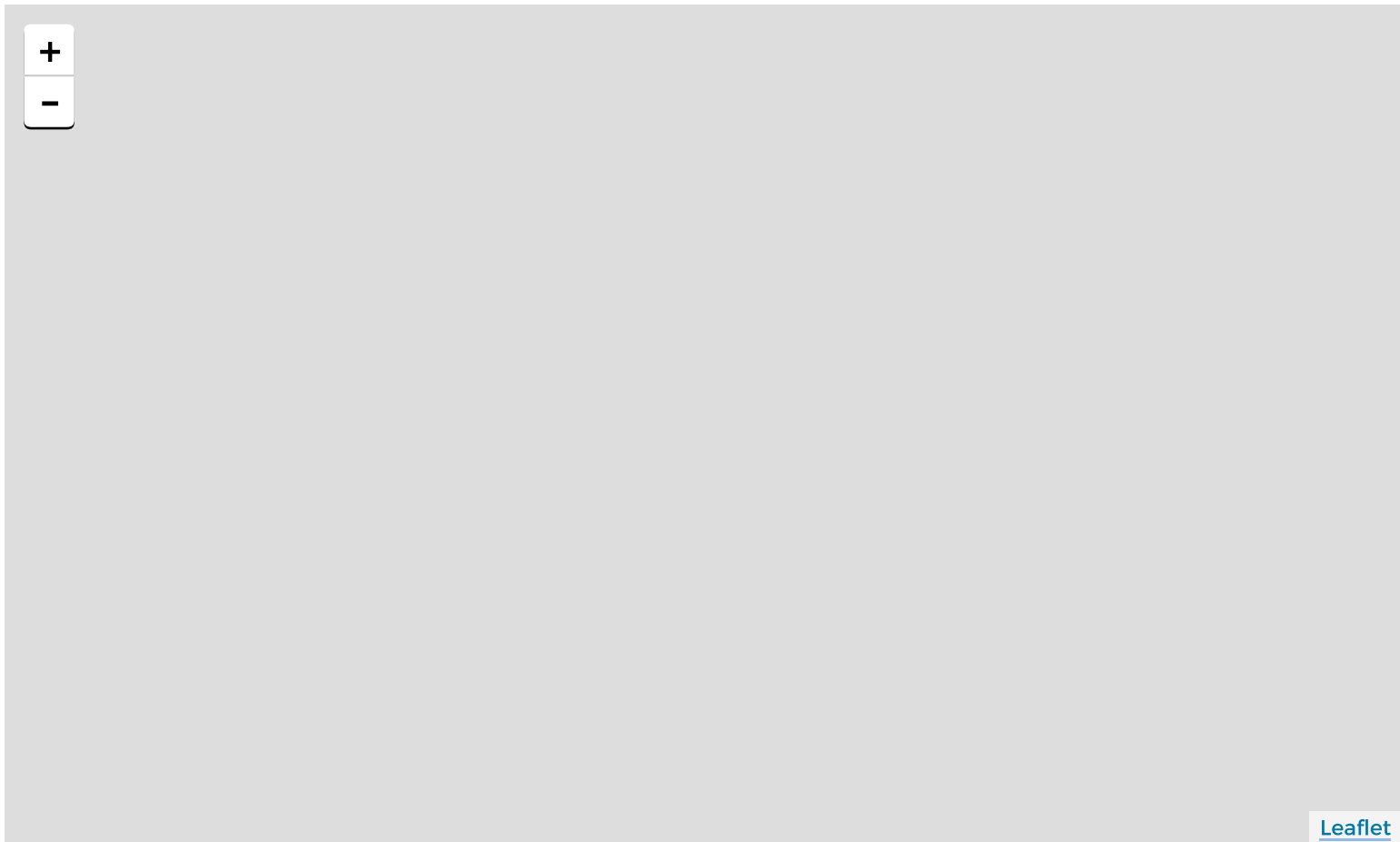


Getting Work Done : R + Leaflet

```
leaflet() %>%  
  addProviderTiles("Acetate.terrain") %>%  
  addPolygons(data=rpp_counties, weight=0.25,  
    fillColor=~color, color="black", fillOpacity=1,  
    popup=~sprintf("In %s, <span style='font-weight:700'>%s</span> has the purchasing  
      htmlEscape(GeoName),  
      htmlEscape(dollar(X2013)))) %>%  
  addPolygons(data=states, weight=0.5, fillColor="white", fillOpacity=0, fill=FALSE, color="#52  
  addLegend(position="bottomright", pal=pal, values=rpp_counties$X2013, labFormat=labelFormat("  
  setView(-74.0059, 40.7127, 6)
```



Where Paychecks Stretch The Most/Least



Doing Real Work

~170 lines pure leaflet/javascript

vs

~60 lines of R

...and the R version can be instantly used to get new BEA data sets where the leaflet one "cheated" and merged the data prior to the HTML example.



A Bit More About Getting Data In

Getting Data Into R

General

- built-in support for CSV/TSV/general delimited & fixed-width
- `readr` / `rio` faster & more robust compatibility
- `readxl` (and others) for raw Excel reading
- `googlesheets`
- `data.table` (large data)
- numerous packages to read statistical data files



Getting Data Into R

Web Scraping / API

- `httr` (like curl command line but better)
- `rvest` (more structured web page scraping)
- `jsonlite` (JSON)
- XML / `xml2` (XML)
- `Rselenium` (headless browser & DOM scraping)
- V8 (the V8 engine in R)



Getting Data Into R

Database

- dplyr
- RPostgreSQL
- RMySQL
- rredis
- mongolite
- RSQLite



Getting Data Into R

- Almost every useful public API covered
- Virtually every "big data" store including AWS/S3
- If something is missing, complain on Twitter and there'll be a package in a week



Crunching Stats

Real World R: Crunching Stats

<http://www.verizonenterprise.com/DBIR/2015/>



Real World R : Crunching Stats

200,000 incidents/breaches

~3,000 data elements per record

~150 lines of statistical analysis



Real World R: Crunching Stats

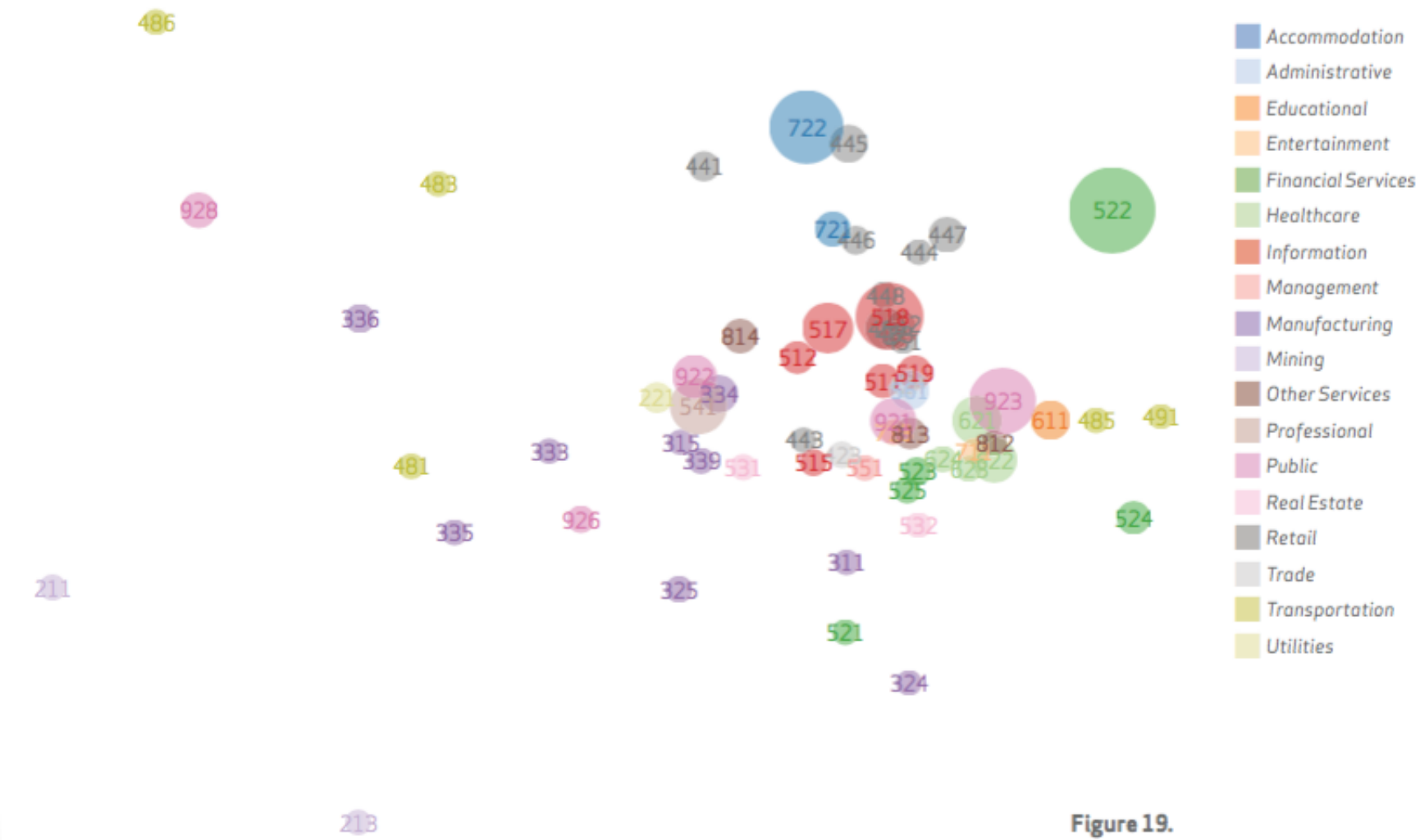


Figure 19.

Clustering on breach data
across industries

28 To look up the three-digit NAICS codes, visit: census.gov/eos/www/naics/index.html

Real World R: Crunching Stats

<http://vz-risk.github.io/dbir/2015/19/>



Getting Data Out of R

Getting Data Out of R

- files (CSV/JSON/XML)
- database write
- S3 upload
- API "put"



Getting Data Out of R

OpenCPU

Hello World! Basic JSON RPC

```
curl https://public.opencpu.org/ocpu/library/stats/R/rnorm/json \  
-H "Content-Type: application/json" -d '{"n":3, "mean": 10, "sd":10}'
```

```
[4.9829, 6.3104, 11.411]
```

This maps to the following request

```
#library(jsonlite)  
args <- fromJSON('{"n":3, "mean": 10, "sd":10}')
```

```
output <- do.call(stats::rnorm, args)  
toJSON(output)
```

Which is equivalent to this function call

```
rnorm(n=3, mean=10, sd=10)
```



Getting Data Out of R

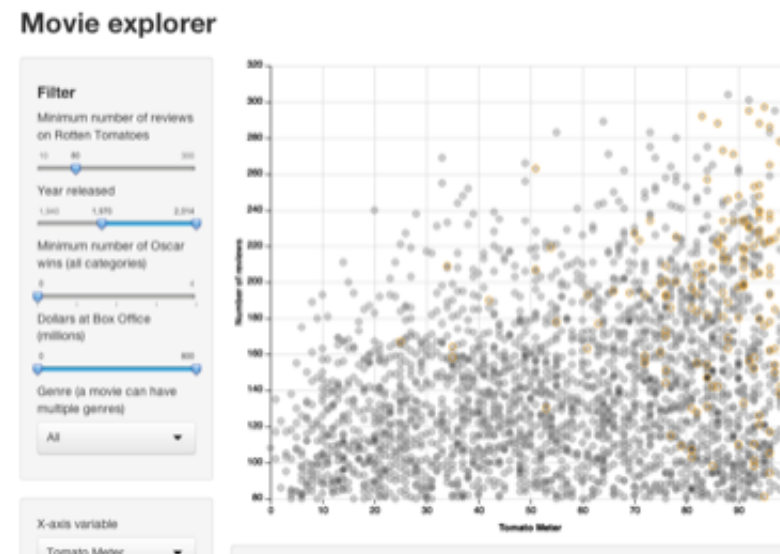
- plumber <http://plumber.trestletech.com/> (think "Flask")
- httpuv <https://github.com/rstudio/httpuv> (basic web & websocket server)



Getting Data and Visualizations Out of R

Getting Data and Visualizations Out of R

Shiny <http://shiny.rstudio.com/>



<http://shiny.rstudio.com/gallery/movie-explorer.html>

HTML Widgets

- htmlwidgets <http://www.htmlwidgets.org/>
- The widget gallery <http://hafen.github.io/htmlwidgetsgallery/> (i've got 3!)
- You've already seen one! (leaflet)

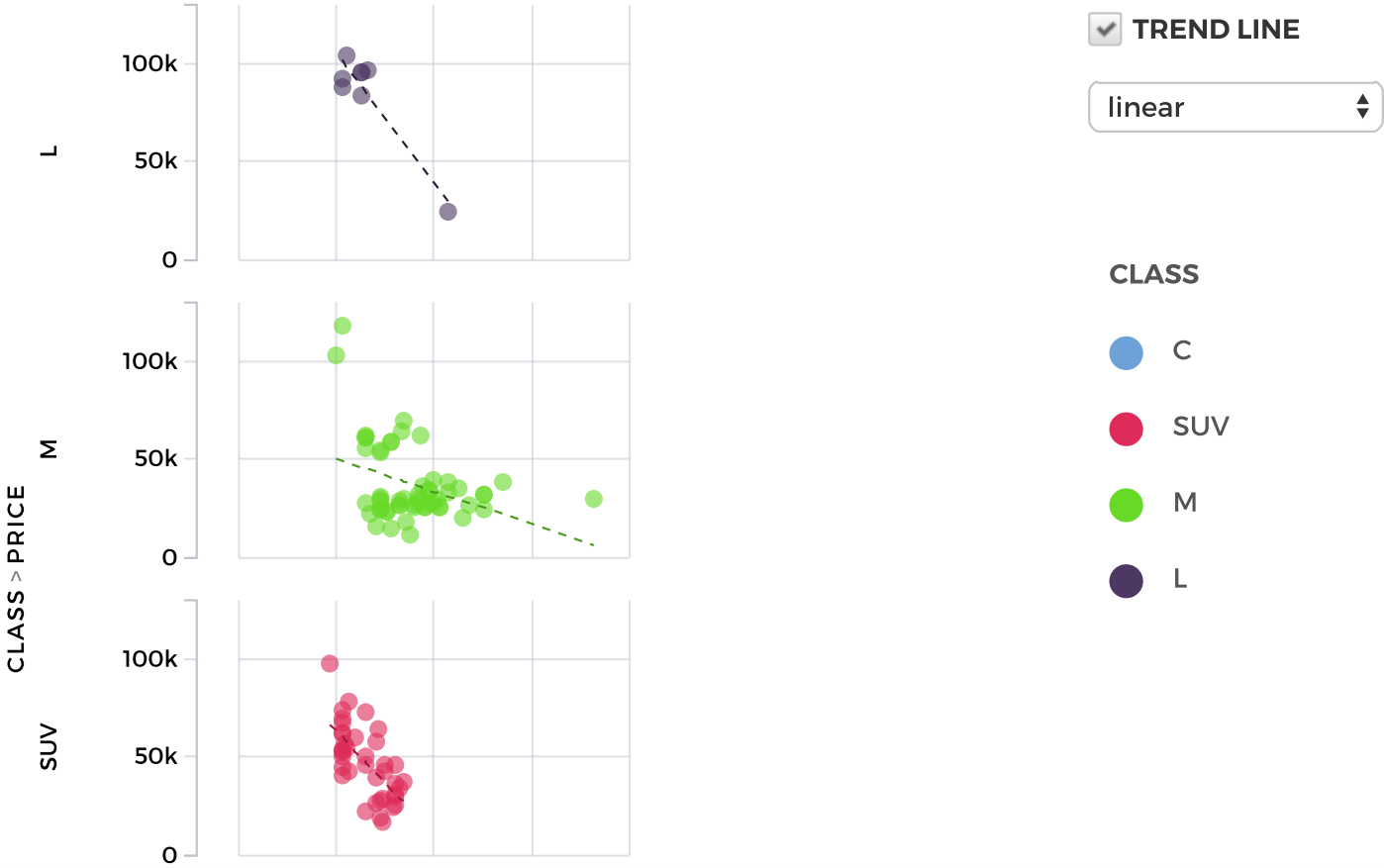


htmlwidgets

```
library(taucharts)
data(cars_data)
tauchart(cars_data) %>%
  tau_point("milespergallon", c("class", "price"), color="class") %>%
  tau_trendline() %>% tau_legend()
```



htmlwidgets



htmlwidgets

- `devtools::create("/path/to/new/package")`
- `setwd("/path/to/new/package")`
- **or use RStudio**
- `htmlwidgets::scaffoldWidget()`



htmlwidgets

- `devtools::build()`
- `devtools::install()`
- **or use RStudio**



htmlwidgets

```
library(widgety)
widgety("Hello, world!")
```

