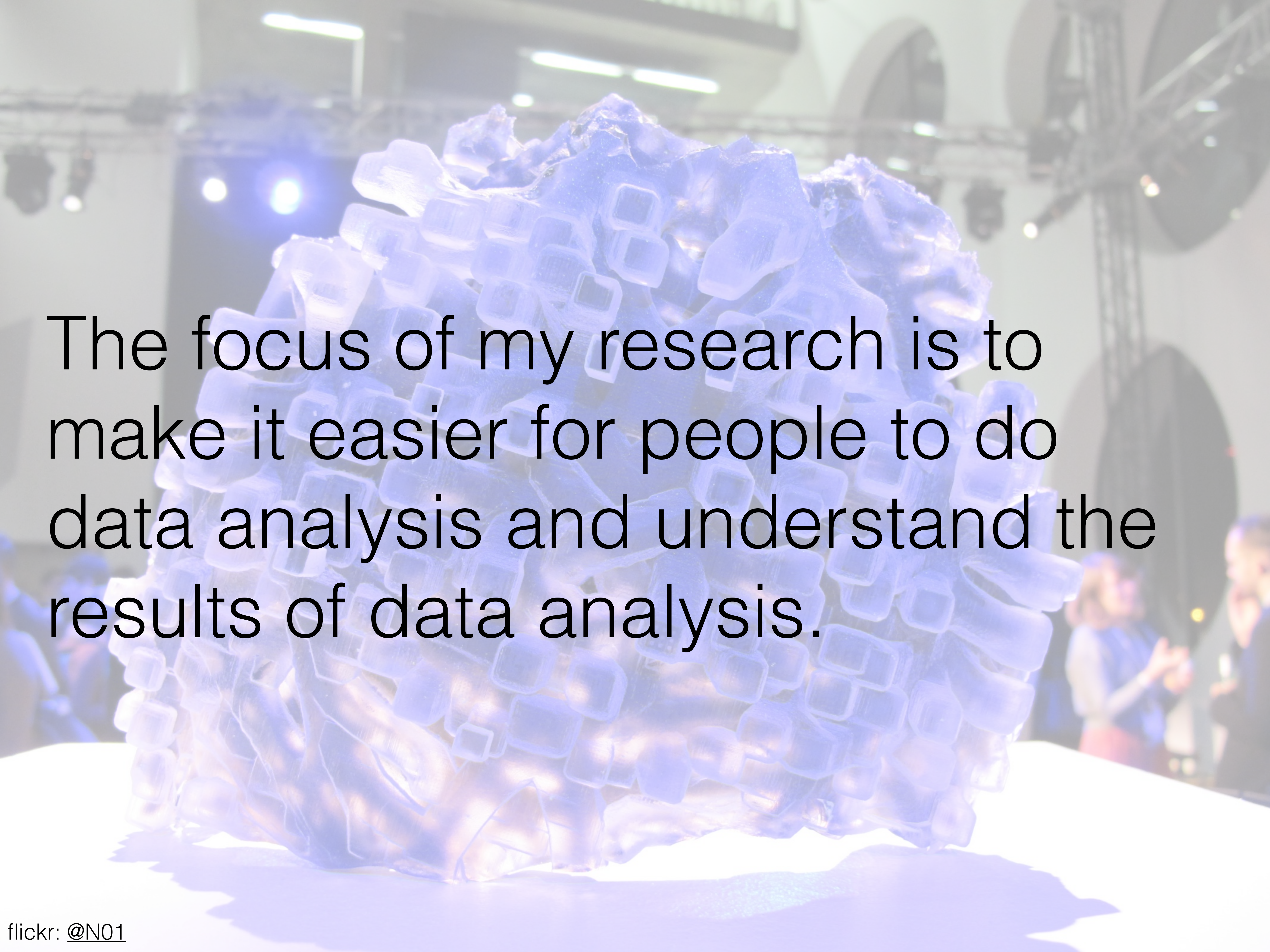
Interfacing with data

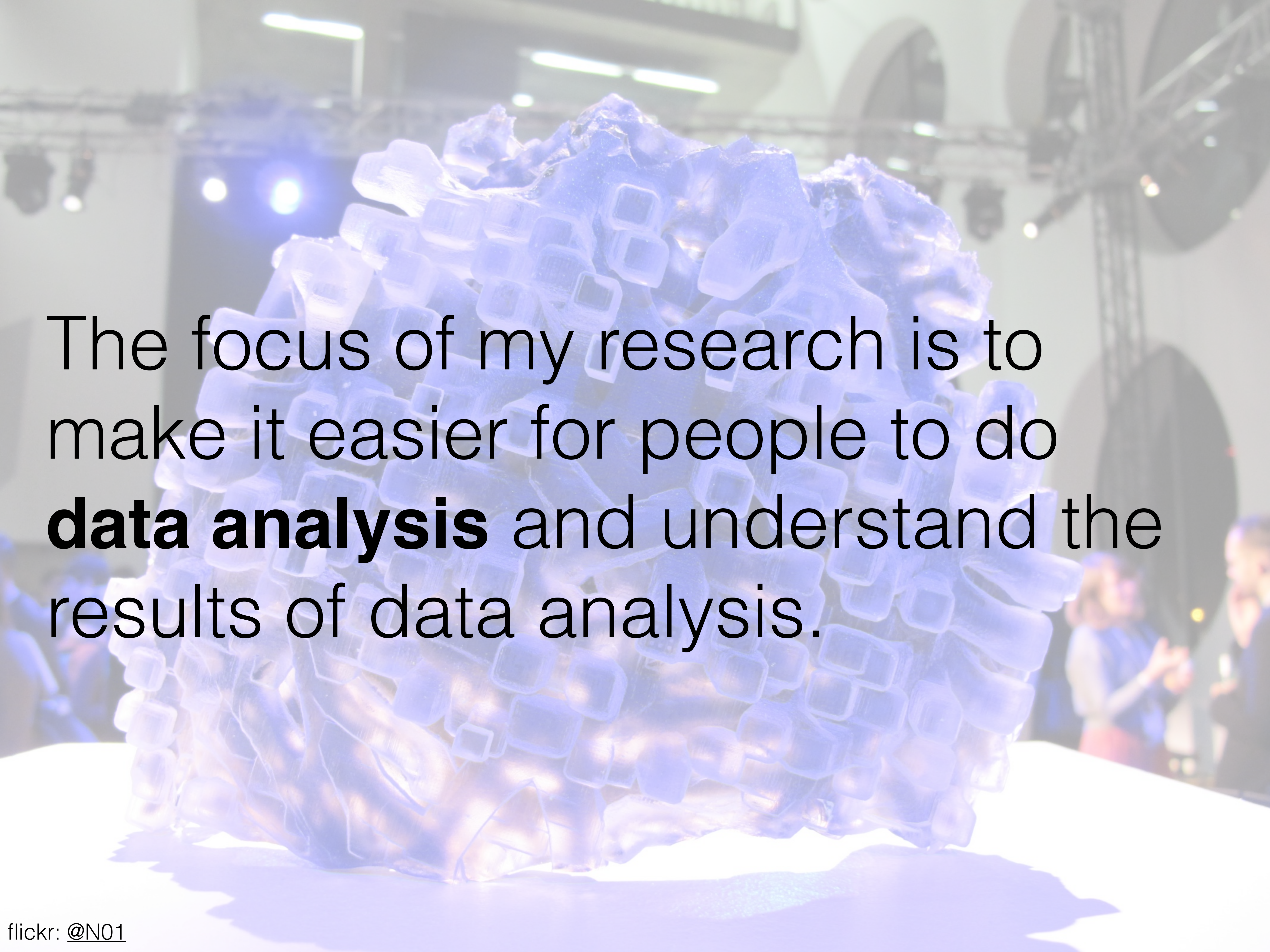
Amelia McNamara (@AmeliaMN)
Visiting Assistant Professor of Statistical and Data Sciences
Smith College Northampton, MA, USA

flickr: @N01



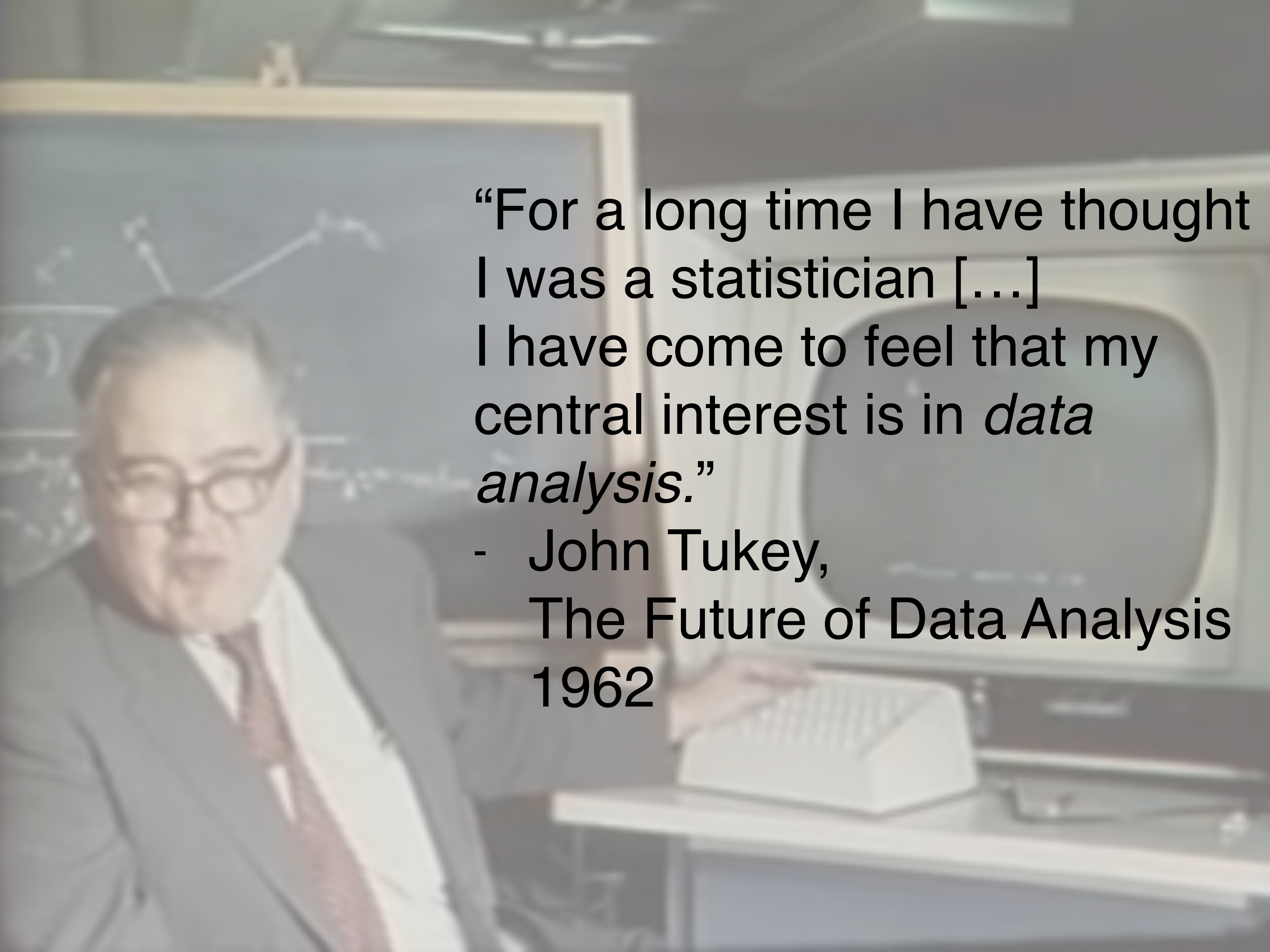
The focus of my research is to make it easier for people to do data analysis and understand the results of data analysis.

flickr: @N01



The focus of my research is to make it easier for people to do **data analysis** and understand the results of data analysis.

flickr: @N01


“For a long time I have thought
I was a statistician […]
I have come to feel that my
central interest is in *data
analysis*.”
- John Tukey,
The Future of Data Analysis
1962

CONTENTS

I INTRODUCTORY

II DIAGRAMS

III DISTRIBUTIONS

IV TESTS OF GOODNESS OF FIT, INDEPENDENCE AND HOMOGENEITY; WITH TABLE OF χ^2

V TESTS OF SIGNIFICANCE OF *MEANS*, DIFFERENCES OF MEANS, AND REGRESSION COEFFICIENTS

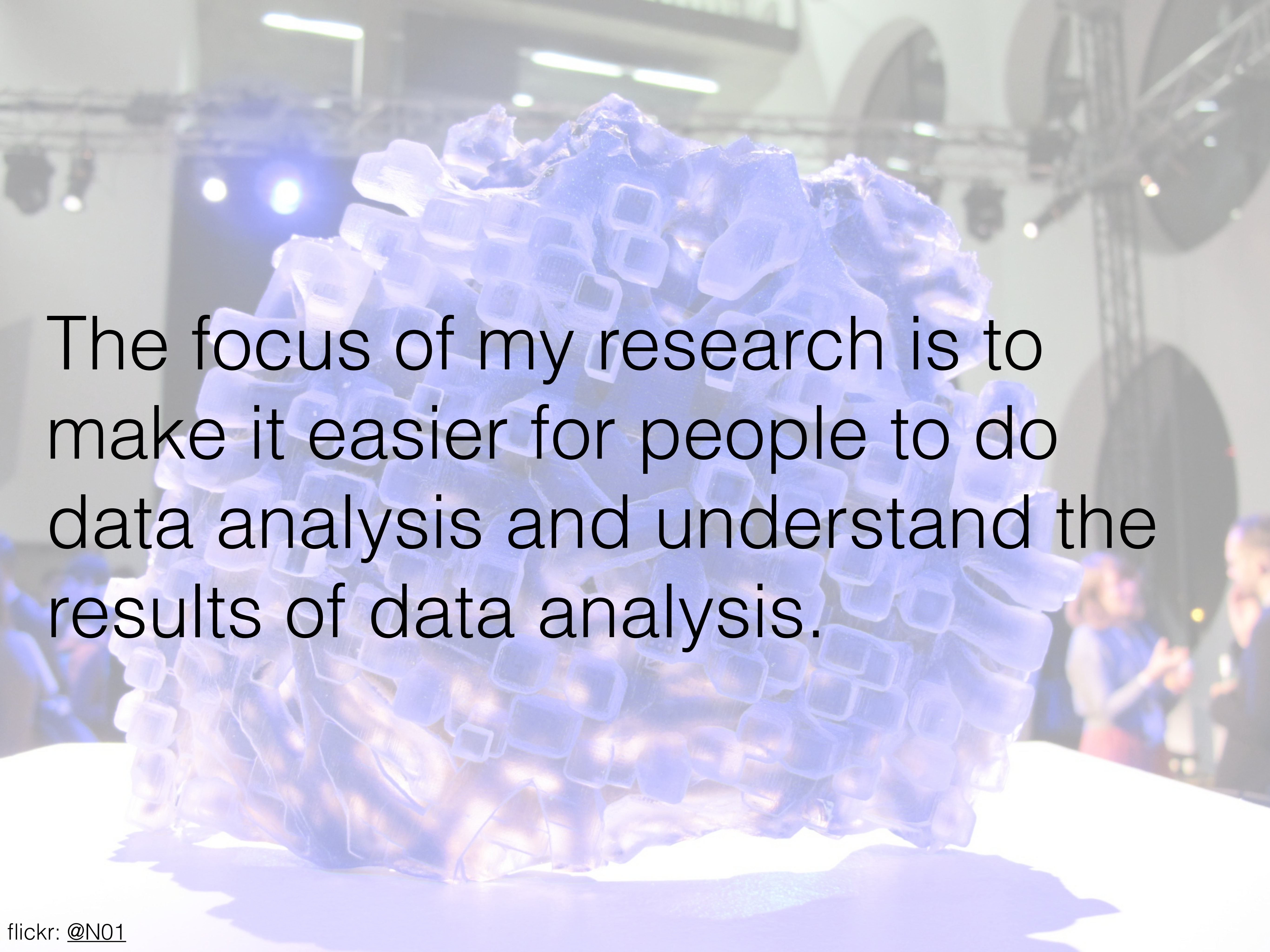
VI THE CORRELATION COEFFICIENT

VII INTRACLASS CORRELATIONS AND THE ANALYSIS OF VARIANCE

Images via Mark Hansen

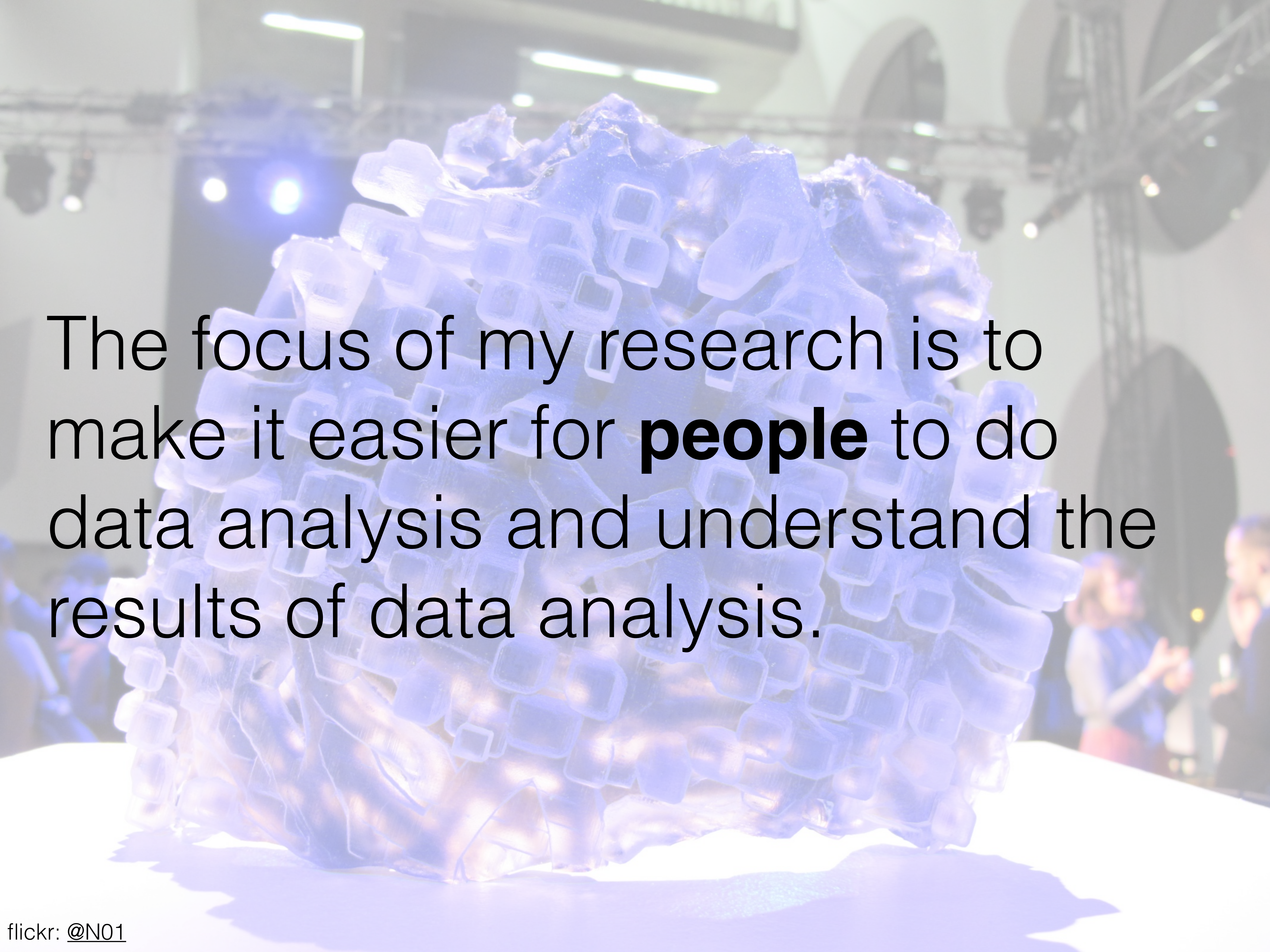
Contents

Images via Mark Hansen



The focus of my research is to make it easier for people to do data analysis and understand the results of data analysis.

flickr: @N01



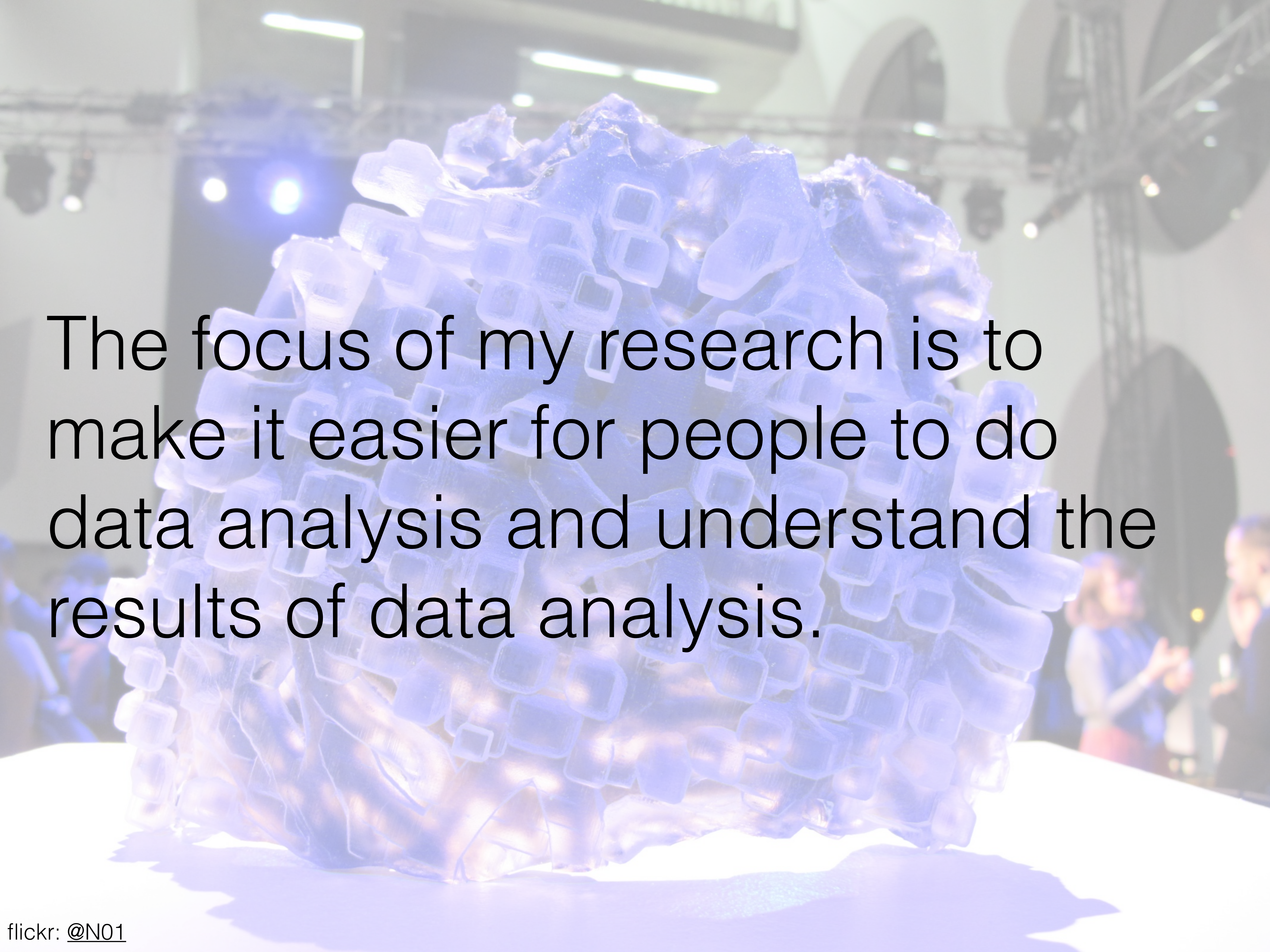
The focus of my research is to make it easier for **people** to do data analysis and understand the results of data analysis.

flickr: @N01

In particular

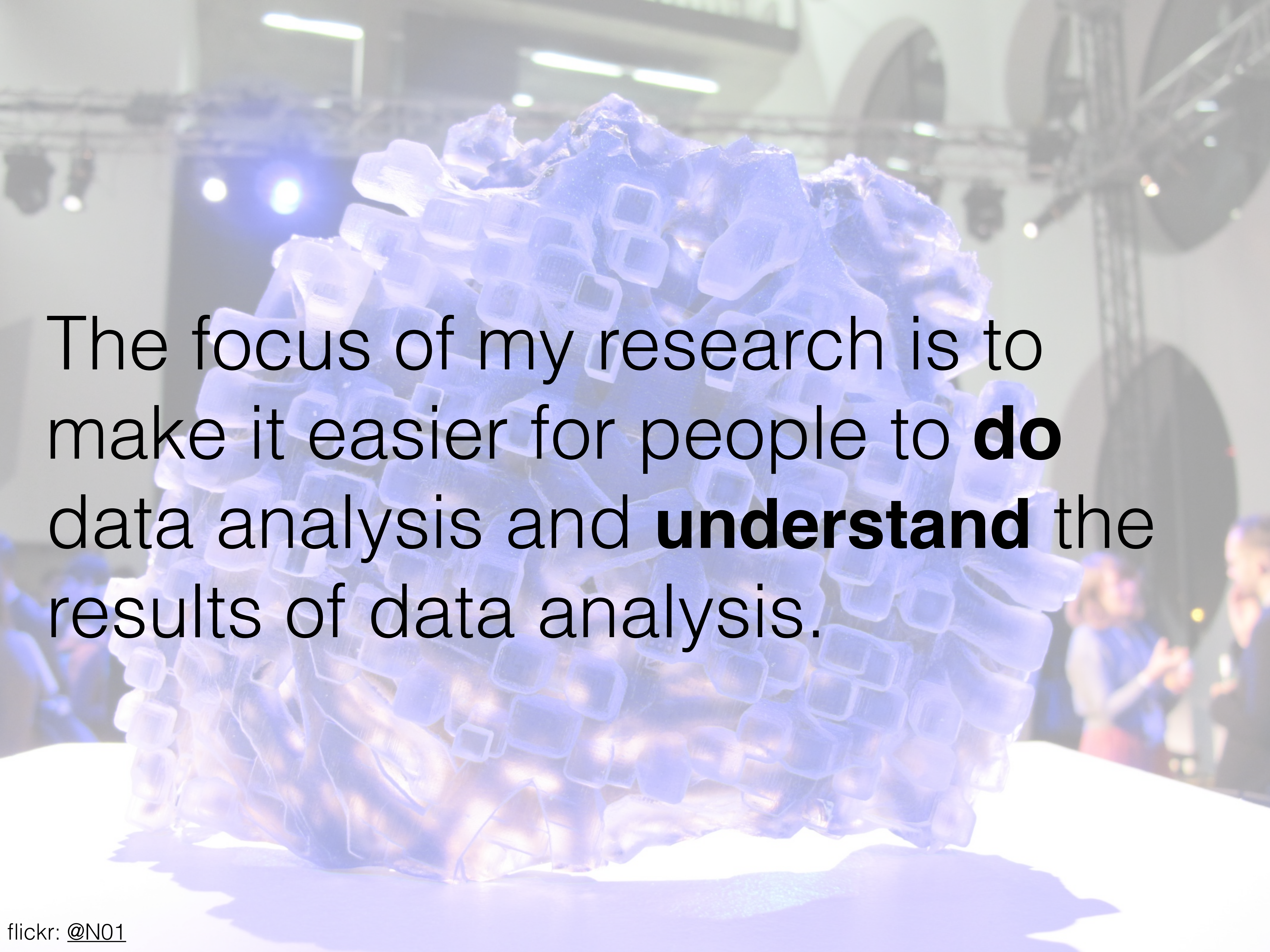
High school teachers/students

College students



The focus of my research is to make it easier for people to do data analysis and understand the results of data analysis.

flickr: @N01



The focus of my research is to make it easier for people to **do** data analysis and **understand** the results of data analysis.

flickr: @N01

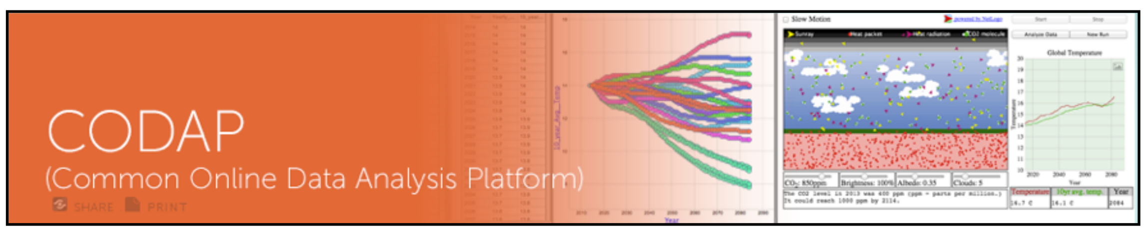
tools

Shiny
by RStudio

Open

TRIFACTA

+ a b l e a u



jmp
Statistical Discovery.™ From SAS.

DataDesk® 7

we can push the
boundary with
tools
and/or
curriculum

IDS
Introduction to Data Science

mobilize

Project MOSAIC
Modeling, Statistics, Calculus, and Computation

curriculum
(and, tools)

“We will be remiss in our duty to our students if we do not see that they learn to use the computer more easily, flexibly, and thoroughly than we ever have; we will be remiss in our duties to ourselves if we do not try to improve and broaden our own uses.”

- John Tukey,
The Technical Tools of Statistics
talking about the class of 1970

a modern statistical computing tool

- Accessibility
- Easy entry for novice users
- Data as a first-order persistent object
- Support for a cycle of exploratory and confirmatory analysis
- Flexible plot creation
- Support for randomization throughout
- Interactivity at every level
- Inherent visual documentation
- Simple support for narrative, publishing, and reproducibility
- Flexibility to build extensions

McNamara, Amelia. “Key Attributes of a Modern Statistical Computing Tool.” Preprint, http://bit.ly/ModernStatComputing.

tools designed for **learning** statistics are typically:

- graphical
- interactive
- intuitive
- supportive of EDA

but:

- don't support reproducibility
- can't handle real data

Fathom®

TinkerPlots™

StatKey to accompany *Statistics: Unlocking the Power of Data* by Lock, Lock, Lock, Lock, and Lock

Rossman/Chance Applet Collection

tools designed for **doing** statistics are typically:

- powerful
- flexible
- reproducible
- supportive of extensions

but:

- hard to get started using
- not interactive

Stata®

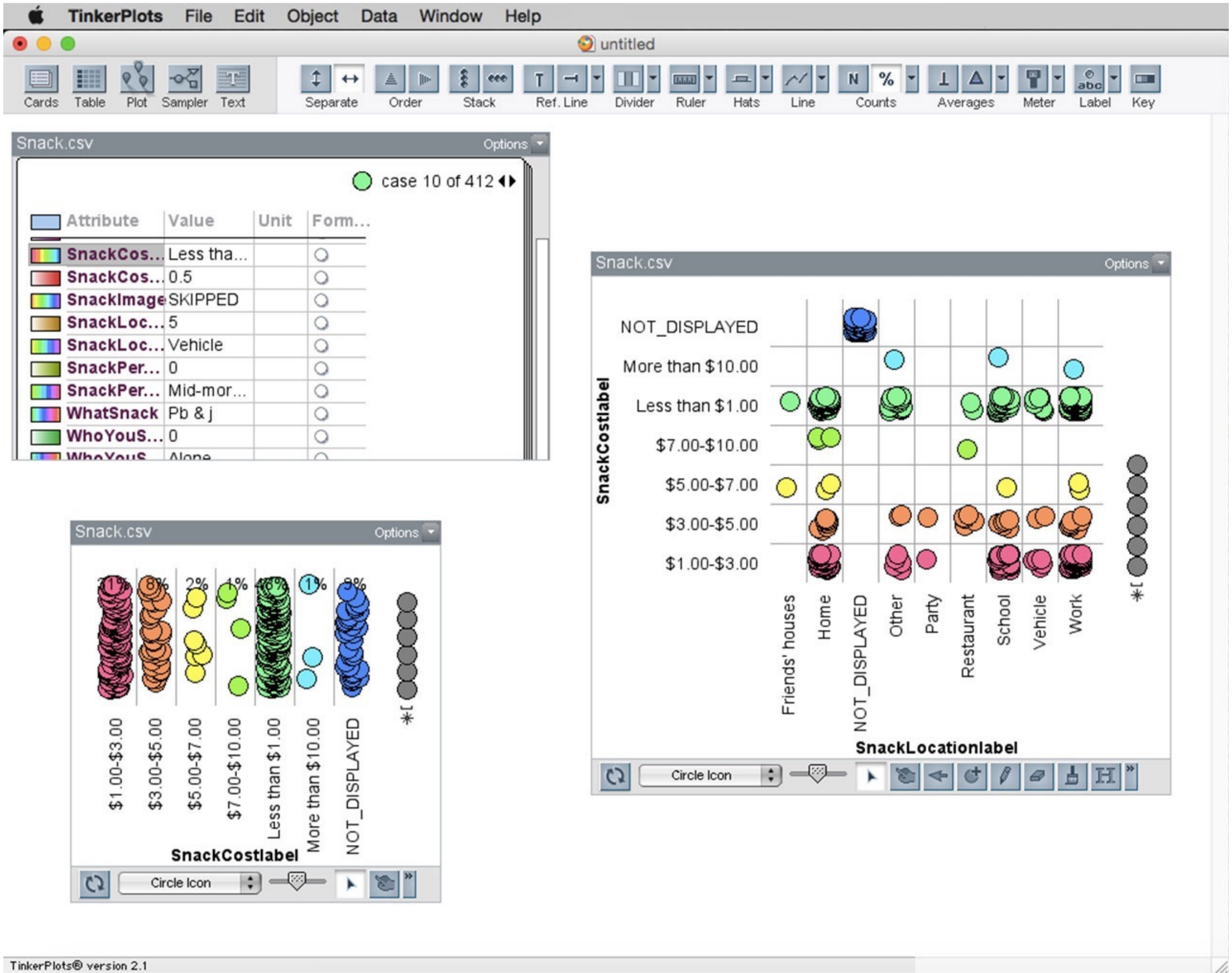
R

SPSS® AN IBM® COMPANY

sas® THE POWER TO KNOW®

MATLAB® The Language of Technical Computing

Easy entry— TinkerPlots



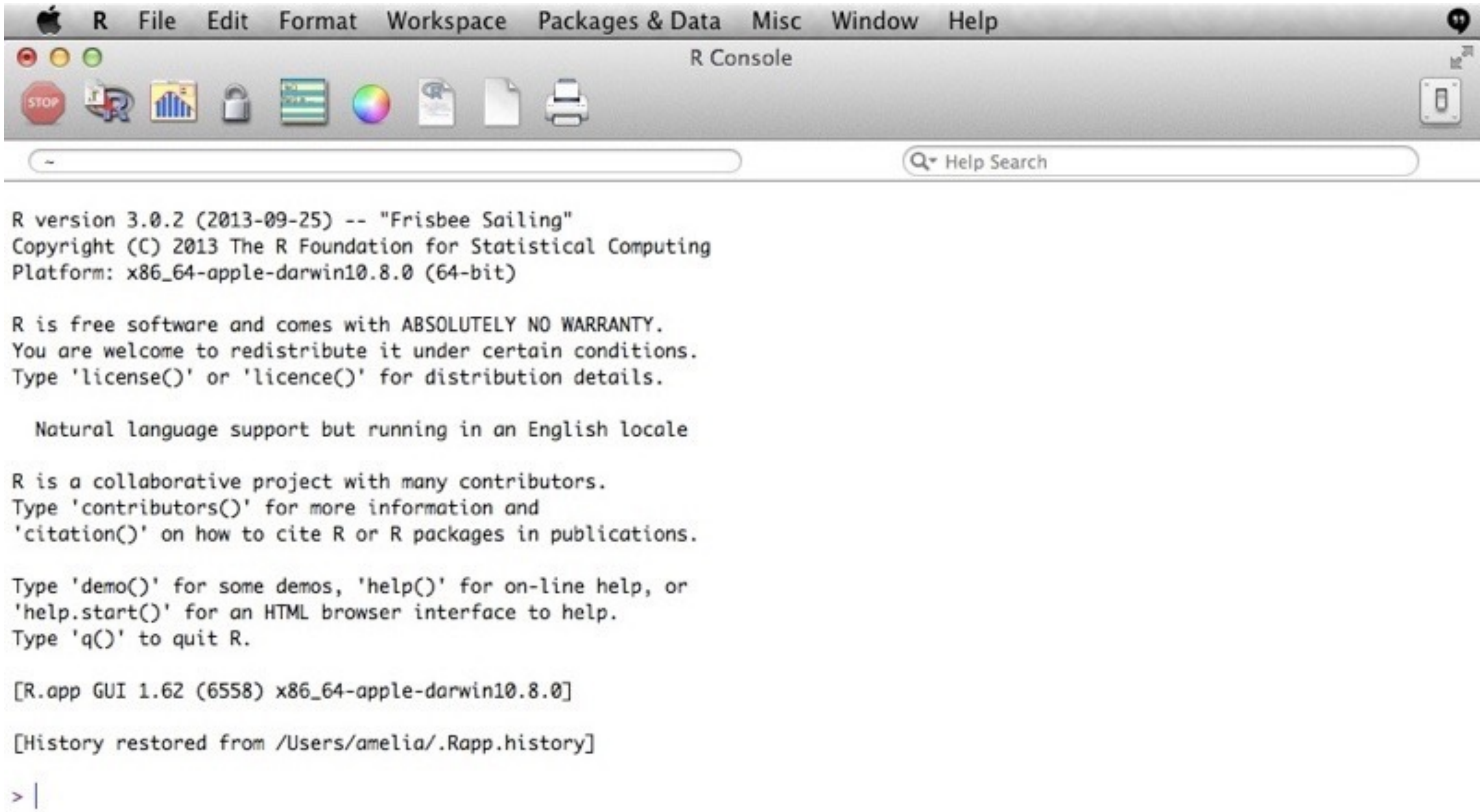
Extensible—

Terminal Shell Edit View Window Help

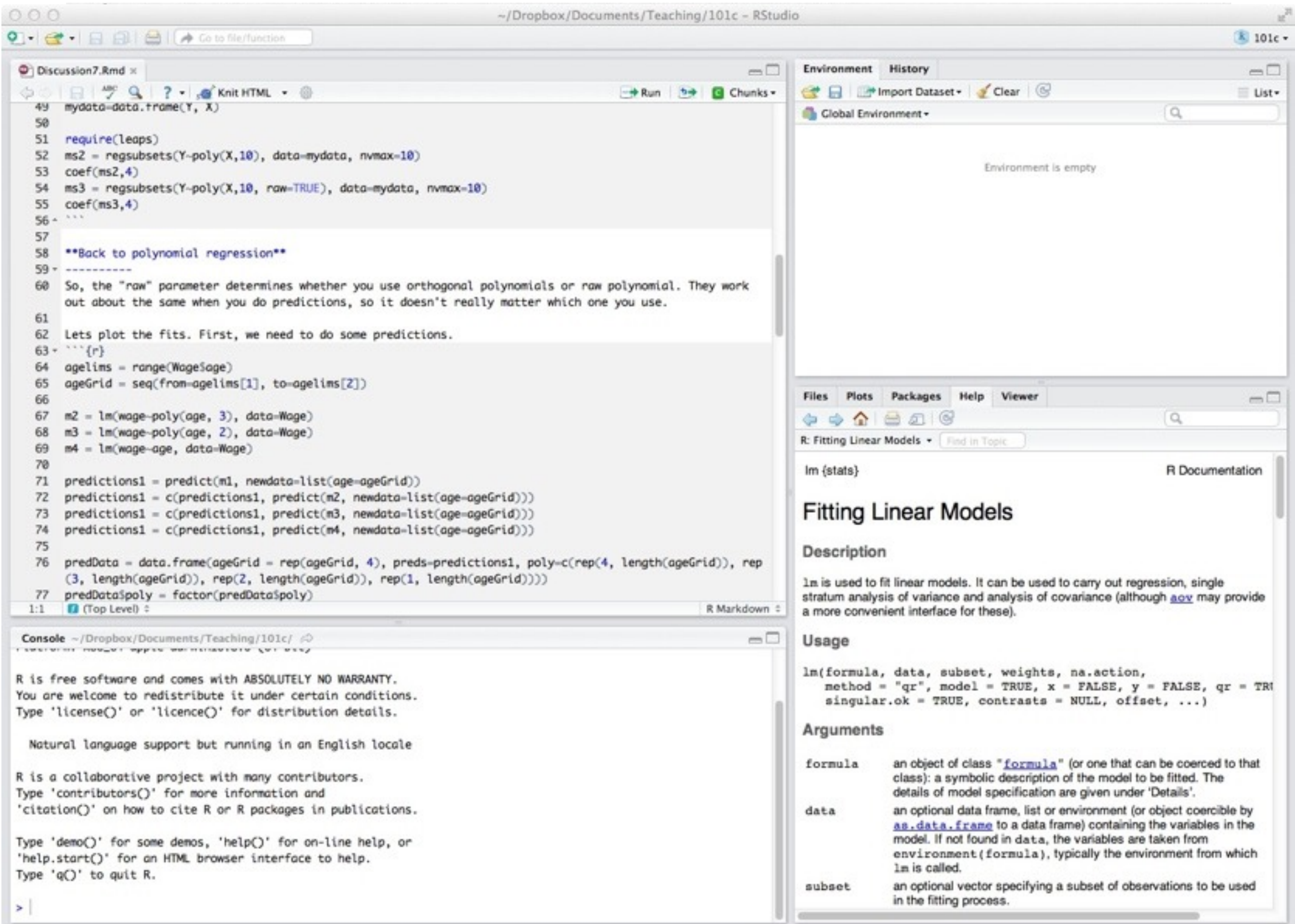
amelia — R — 80×24

```
Last login: Wed Apr 16 15:39:29 on ttys000
Amelias-MacBook-Air:~ amelia$ R

R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin10.8.0 (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.

  Natural language support but running in an English locale

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.

>
```

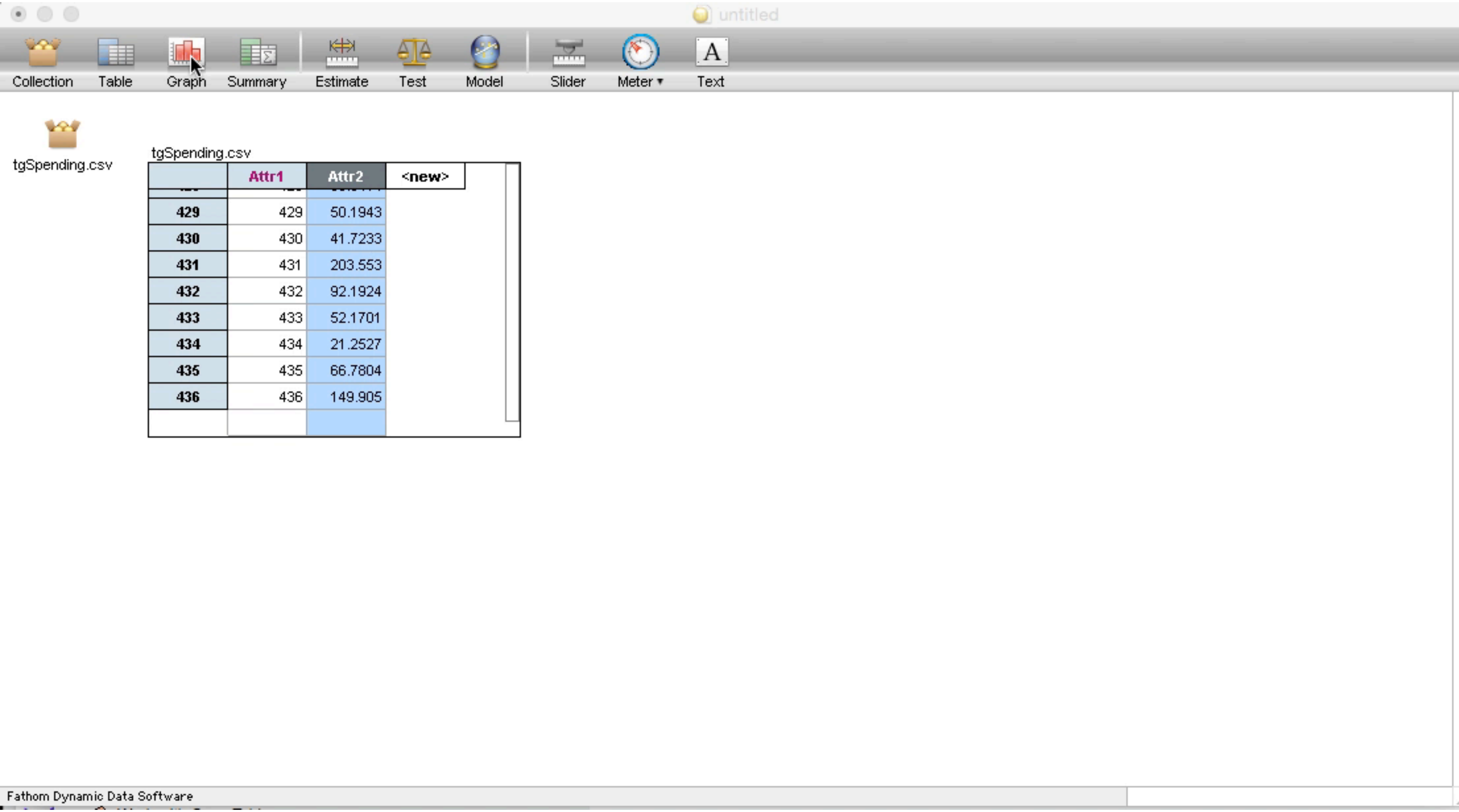
Extensible—


R File Edit Format Workspace Packages & Data Misc Window Help
R Console
STOP
Help Search
R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin10.8.0 (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.
Natural language support but running in an English locale
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.
[R.app GUI 1.62 (6558) x86_64-apple-darwin10.8.0]
[History restored from /Users/amelia/.Rapp.history]
>

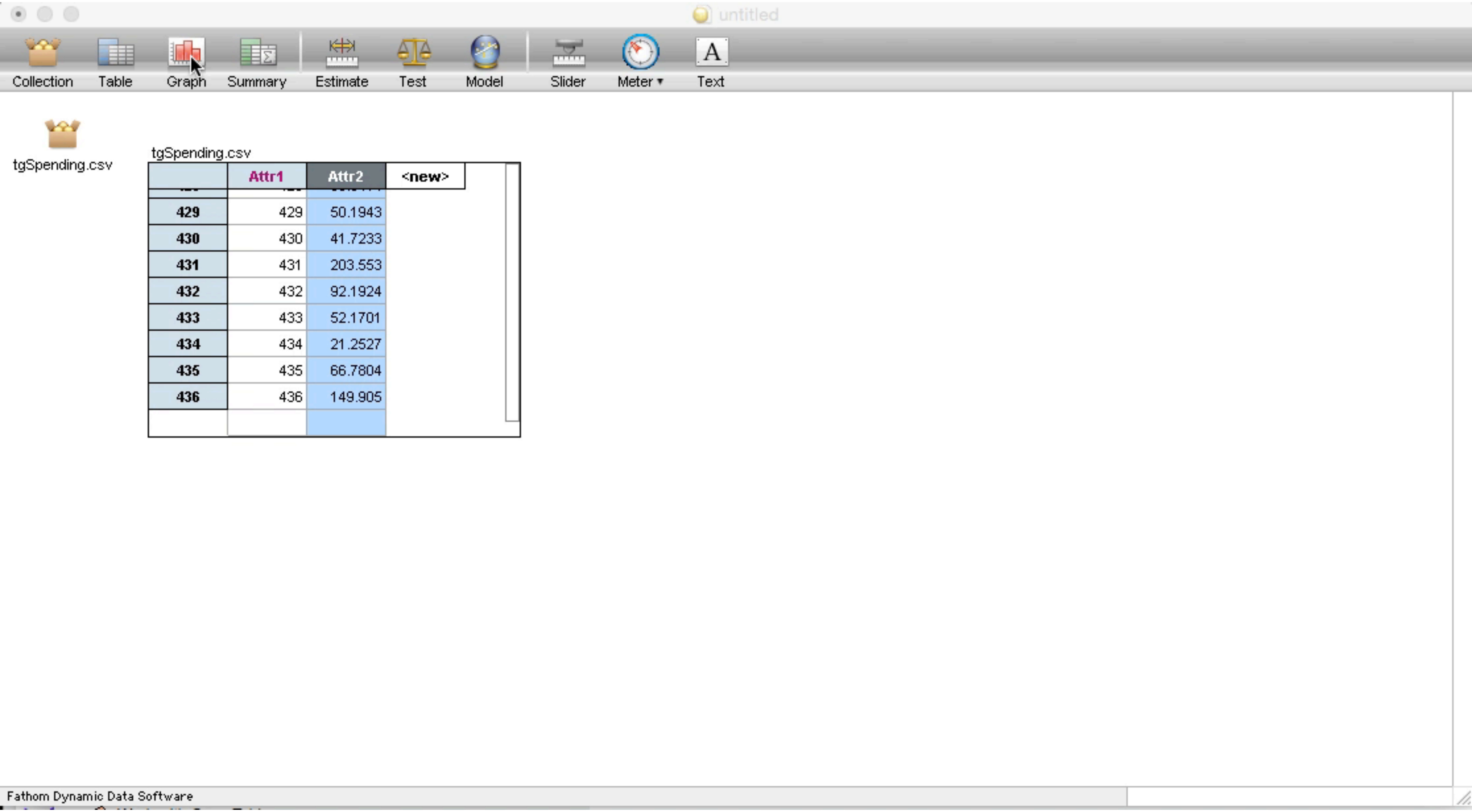
Extensible—


~/Dropbox/Documents/Teaching/101c – RStudio
Discussion7.Rmd
Knit HTML
Run
Chunks
mydata=data.frame(Y, X)
require(leaps)
ms2 = regsubsets(Y~poly(X,10), data=mydata, nvmax=10)
coef(ms2,4)
ms3 = regsubsets(Y~poly(X,10, raw=TRUE), data=mydata, nvmax=10)
coef(ms3,4)
Back to polynomial regression
So, the "raw" parameter determines whether you use orthogonal polynomials or raw polynomial. They work out about the same when you do predictions, so it doesn't really matter which one you use.
Lets plot the fits. First, we need to do some predictions.
```{r}
agelims = range(Wage$age)
ageGrid = seq(from=agelims[1], to=agelims[2])
m2 = lm(wage~poly(age, 3), data=Wage)
m3 = lm(wage~poly(age, 2), data=Wage)
m4 = lm(wage~age, data=Wage)
predictions1 = predict(m1, newdata=list(age=ageGrid))
predictions1 = c(predictions1, predict(m2, newdata=list(age=ageGrid)))
predictions1 = c(predictions1, predict(m3, newdata=list(age=ageGrid)))
predictions1 = c(predictions1, predict(m4, newdata=list(age=ageGrid)))
predData = data.frame(ageGrid = rep(ageGrid, 4), preds=predictions1, poly=c(rep(4, length(ageGrid)), rep(3, length(ageGrid)), rep(2, length(ageGrid)), rep(1, length(ageGrid))))
predData$poly = factor(predData$poly)
(Top Level)
R Markdown
Console ~/Dropbox/Documents/Teaching/101c/
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.
Natural language support but running in an English locale
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.
Environment
History
Import Dataset
Clear
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
R: Fitting Linear Models
lm {stats}
R Documentation
Fitting Linear Models
Description
lm is used to fit linear models. It can be used to carry out regression, single stratum analysis of variance and analysis of covariance (although aov may provide a more convenient interface for these).
Usage
lm(formula, data, subset, weights, na.action,
method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRU
singular.ok = TRUE, contrasts = NULL, offset, ...)
Arguments
formula
an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data
an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which lm is called.
subset
an optional vector specifying a subset of observations to be used in the fitting process.
```

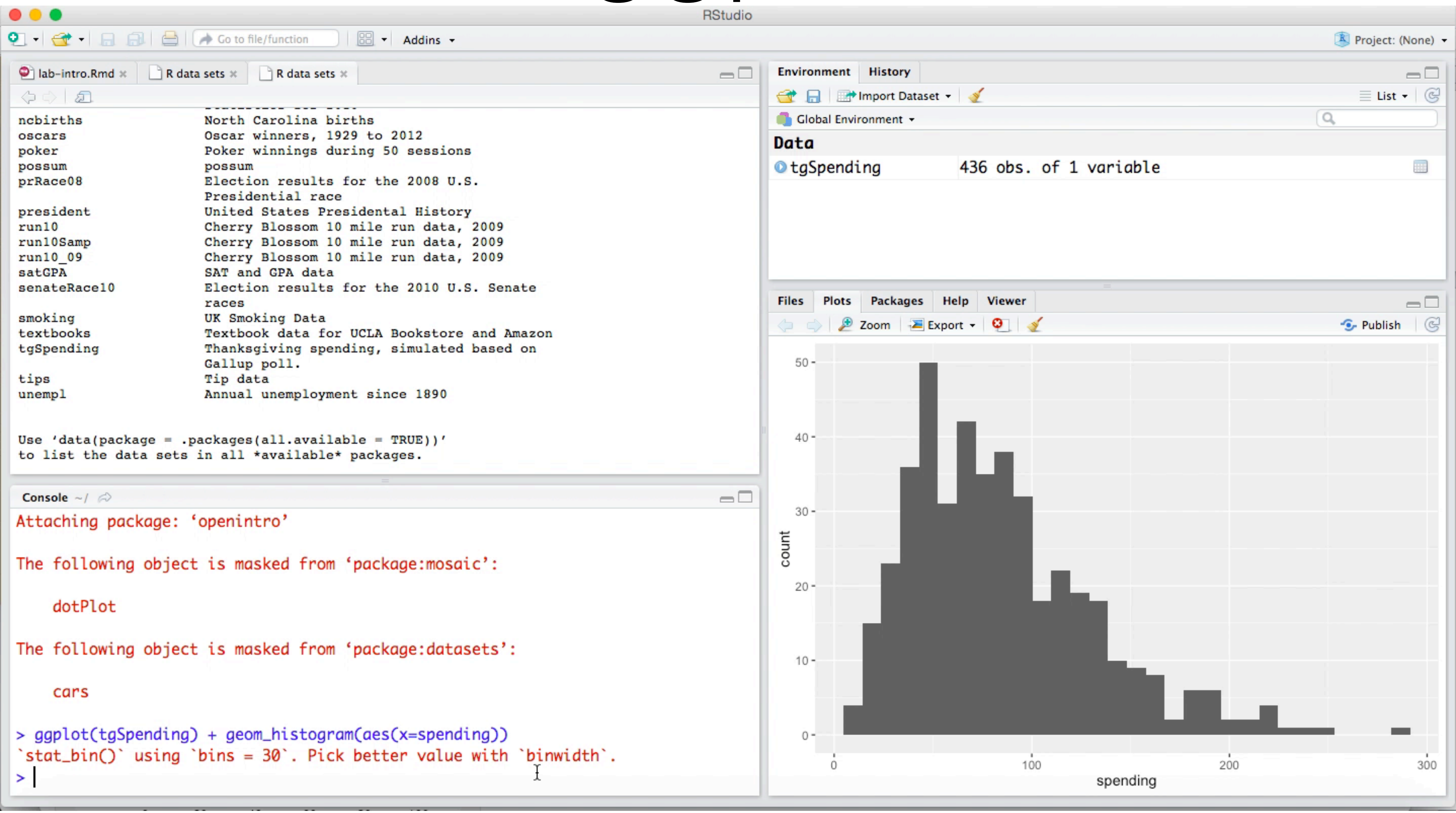
Interactivity— Fathom



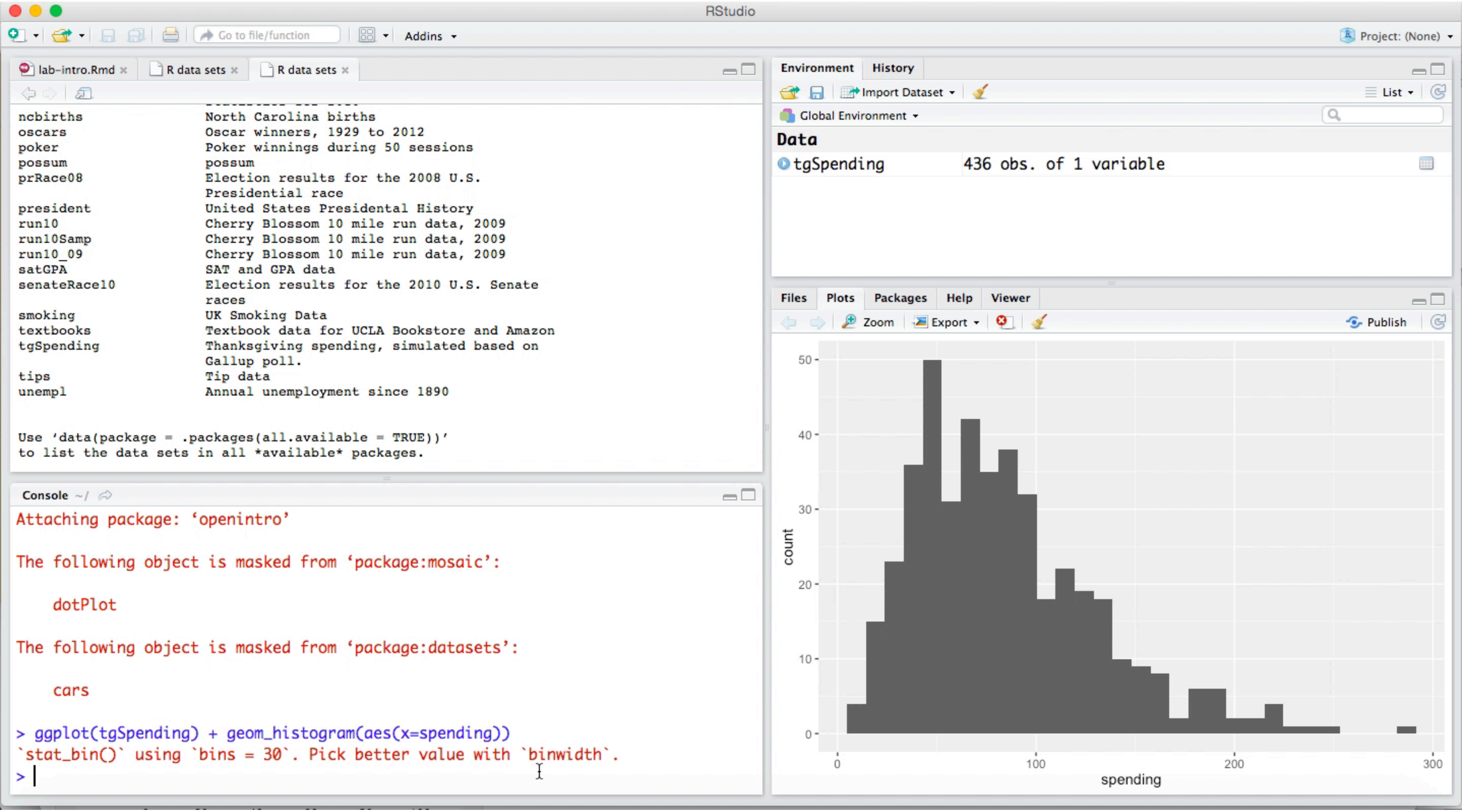
Interactivity— Fathom



R/ggplot2


RStudio
Go to file/function
Addins
Project: (None)
lab-intro.Rmd
R data sets
R data sets
ncbirths North Carolina births
oscars Oscar winners, 1929 to 2012
poker Poker winnings during 50 sessions
possum possum
prRace08 Election results for the 2008 U.S. Presidential race
president United States Presidental History
run10 Cherry Blossom 10 mile run data, 2009
run10Samp Cherry Blossom 10 mile run data, 2009
run10_09 Cherry Blossom 10 mile run data, 2009
satGPA SAT and GPA data
senateRace10 Election results for the 2010 U.S. Senate races
smoking UK Smoking Data
textbooks Textbook data for UCLA Bookstore and Amazon
tgSpending Thanksgiving spending, simulated based on Gallup poll.
tips Tip data
unempl Annual unemployment since 1890
Use 'data(package = .packages(all.available = TRUE))'
to list the data sets in all *available* packages.
Console ~/
Attaching package: 'openintro'
The following object is masked from 'package:mosaic':
dotPlot
The following object is masked from 'package:datasets':
cars
> ggplot(tgSpending) + geom_histogram(aes(x=spending))
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
>
Environment
History
Import Dataset
List
Global Environment
Data
tgSpending 436 obs. of 1 variable
Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
50
40
30
20
10
0
count
0
100
200
300
spending

R/ggplot2



RStudio
Go to file/function
Addins
Project: (None)
lab-intro.Rmd
R data sets
R data sets
ncbirths North Carolina births
oscars Oscar winners, 1929 to 2012
poker Poker winnings during 50 sessions
possum possum
prRace08 Election results for the 2008 U.S. Presidential race
president United States Presidental History
run10 Cherry Blossom 10 mile run data, 2009
run10Samp Cherry Blossom 10 mile run data, 2009
run10_09 Cherry Blossom 10 mile run data, 2009
satGPA SAT and GPA data
senateRace10 Election results for the 2010 U.S. Senate races
smoking UK Smoking Data
textbooks Textbook data for UCLA Bookstore and Amazon
tgSpending Thanksgiving spending, simulated based on Gallup poll.
tips Tip data
unempl Annual unemployment since 1890
Use 'data(package = .packages(all.available = TRUE))' to list the data sets in all *available* packages.
Console ~/
Attaching package: 'openintro'
The following object is masked from 'package:mosaic':
dotPlot
The following object is masked from 'package:datasets':
cars
> ggplot(tgSpending) + geom_histogram(aes(x=spending))
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
>
Environment
History
Import Dataset
List
Global Environment
Data
tgSpending 436 obs. of 1 variable
Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
count
spending
0
10
20
30
40
50
0
100
200
300

Tableau

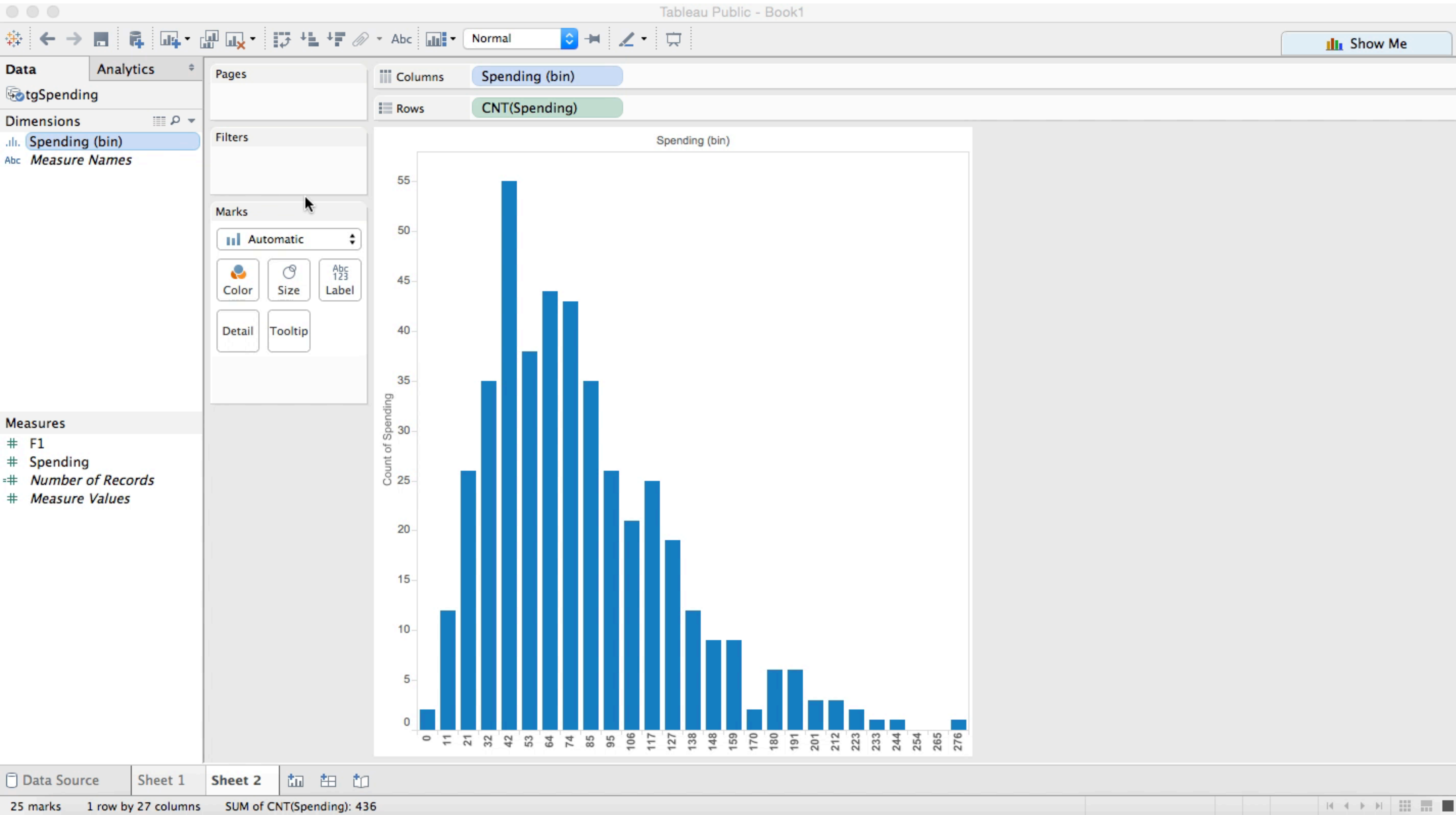


Tableau Public - Book1
Normal
Show Me
Data
Analytics
tgSpending
Dimensions
Spending (bin)
Measure Names
Measures
F1
Spending
Number of Records
Measure Values
Pages
Filters
Marks
Automatic
Color
Size
Label
Detail
Tooltip
Columns
Spending (bin)
Rows
CNT(Spending)
Spending (bin)
Count of Spending
55
50
45
40
35
30
25
20
15
10
5
0
0 11 21 32 42 53 64 74 85 95 106 117 127 138 148 159 170 180 191 201 212 223 233 244 254 265 276
Data Source
Sheet 1
Sheet 2
25 marks
1 row by 27 columns
SUM of CNT(Spending): 436

Tableau

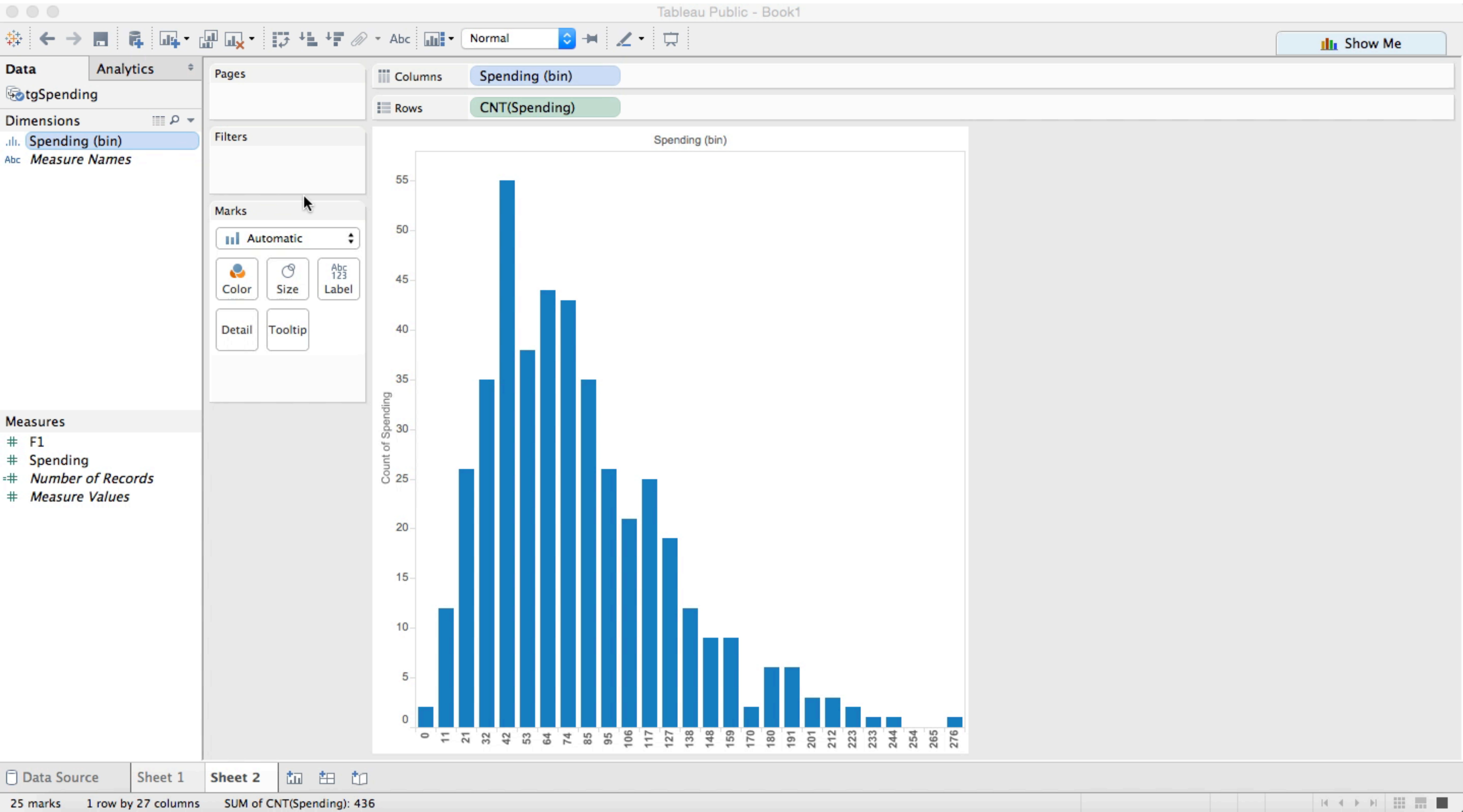


Tableau Public - Book1
Normal
Show Me
Data
Analytics
tgSpending
Dimensions
Spending (bin)
Measure Names
Measures
F1
Spending
Number of Records
Measure Values
Pages
Filters
Marks
Automatic
Color
Size
Label
Detail
Tooltip
Columns
Spending (bin)
Rows
CNT(Spending)
Spending (bin)
Count of Spending
0 5 10 15 20 25 30 35 40 45 50 55
0 11 21 32 42 53 64 74 85 95 106 117 127 138 148 159 170 180 191 201 212 223 233 244 254 265 276
Data Source
Sheet 1
Sheet 2
25 marks
1 row by 27 columns
SUM of CNT(Spending): 436

Gather your data

A histogram is based on a collection of data about a numeric variable. Our first step is to gather some values for that variable. The initial dataset we will consider consists of fuel consumption (in miles per gallon) from a sample of car models available in 1974 (yes, rather out of date). We can visualize the dataset as a pool of items, with each item identified by its value—which in theory lets us "see" all the items, but makes it hard to get the gestalt of the variable. What are some common values? Is there a lot of variation?

Sort into an ordered list

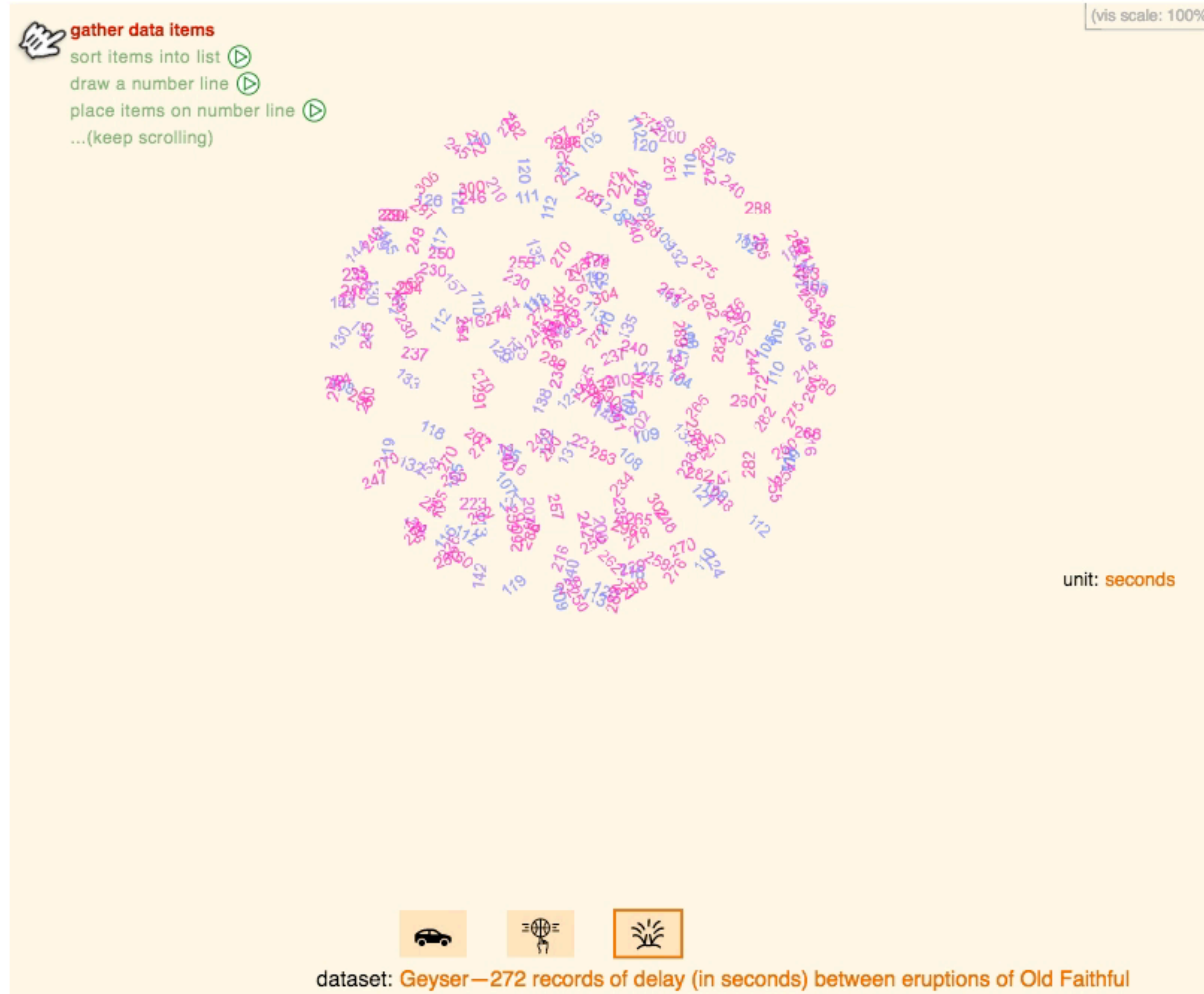
A useful first step towards describing the variable's distribution is to sort the items into a list. Now we can see the maximum value and the minimum value. Beyond that, it is hard to say much about the center, shape, and spread of the distribution. Part of the problem is that the list is completely filled; the space between any two items is the same, no matter how dissimilar their values may be. We need a way to see how the items relate to each other. Are they clustered around a few specific values? Is there one lonely item, with a value far removed from all the others?

Draw the number line

A common convention is to use a number line, on which higher values are displayed to the right and smaller (or negative) values to the left. We can draw a line representing all possible numbers between the minimum and maximum data values.

Add data to the number line

Now, we map each item to a dot at the appropriate point along the number line. In our visualization we draw the path followed by each item on its way from the list to the line, helping to reveal how adjacent list items end up close or far apart on the number line

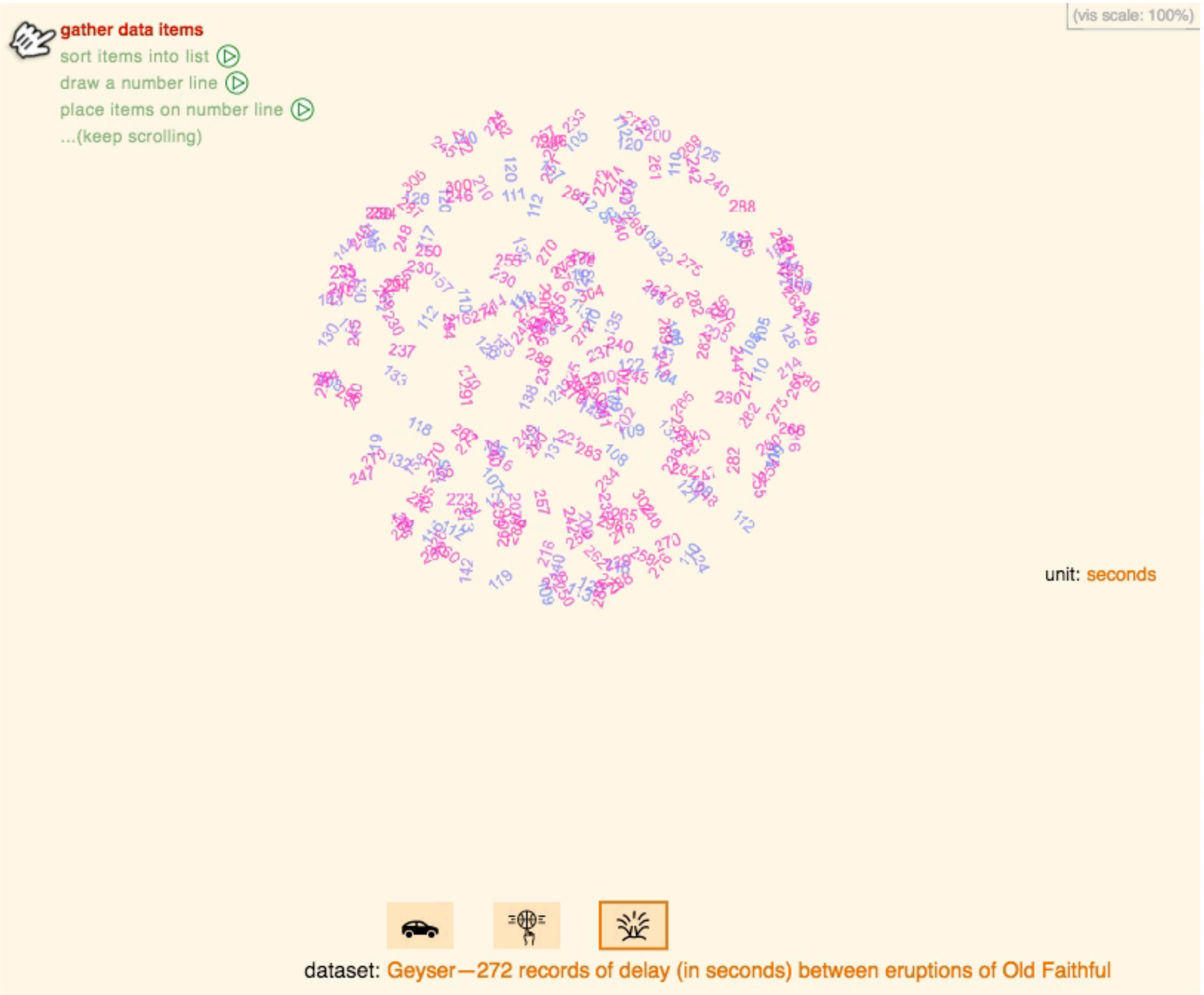


http://tinlizzie.org/histograms/

Exploring Histograms, an essay by Aran Lunzer and Amelia McNamara

Gather your data

A histogram is based on a collection of data about a numeric variable. Our first step is to gather some values for that variable. The initial dataset we will consider consists of fuel consumption (in miles per gallon) from a sample of car models available in 1974 (yes, rather out of date). We can visualize the dataset as a pool of items, with each item identified by its value—which in theory lets us "see" all the items, but makes it hard to get the gestalt of the variable. What are some common values? Is there a lot of variation?



Sort into an ordered list

A useful first step towards describing the variable's distribution is to sort the items into a list. Now we can see the maximum value and the minimum value. Beyond that, it is hard to say much about the center, shape, and spread of the distribution. Part of the problem is that the list is completely filled; the space between any two items is the same, no matter how dissimilar their values may be. We need a way to see how the items relate to each other. Are they clustered around a few specific values? Is there one lonely item, with a value far removed from all the others?

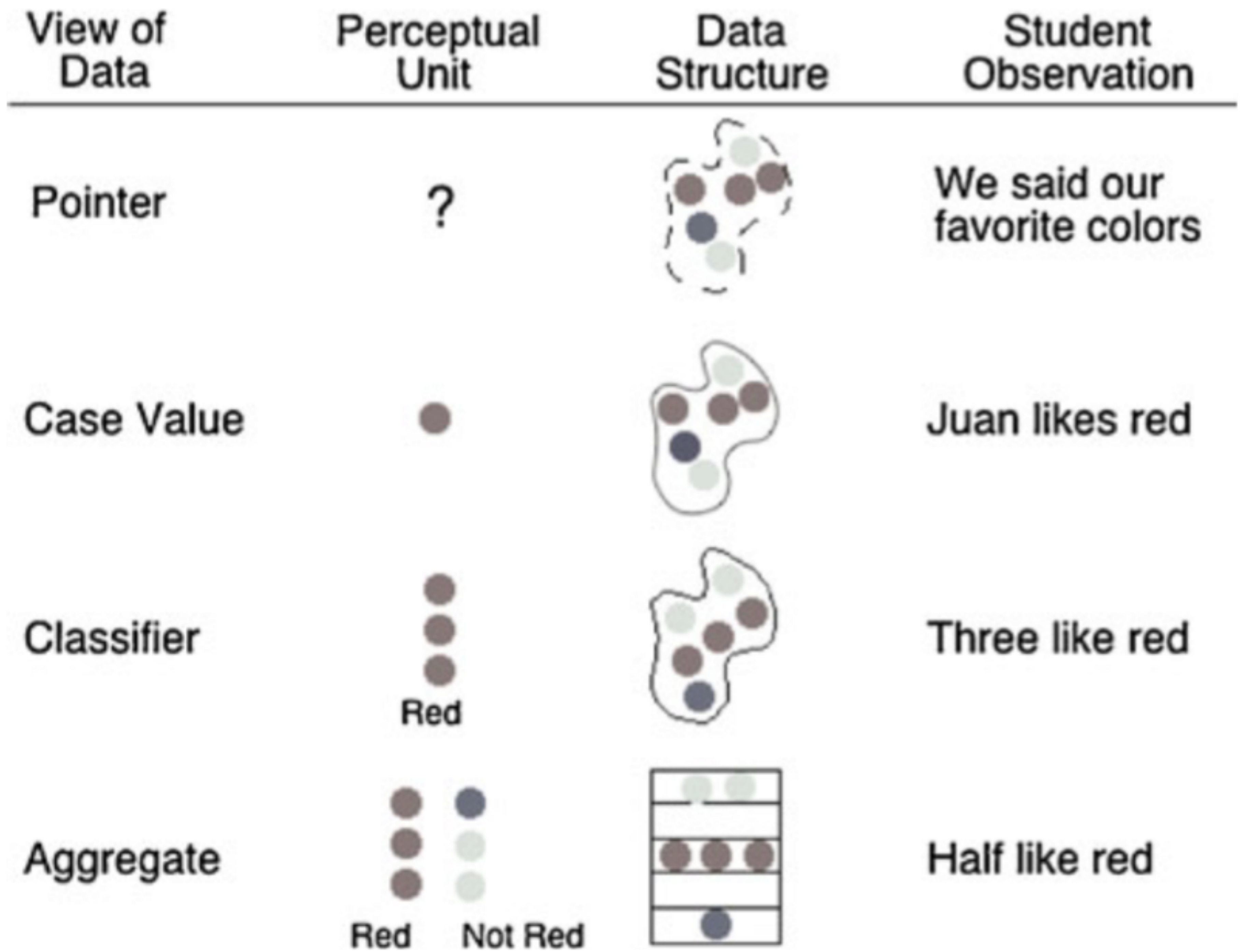
Draw the number line

A common convention is to use a number line, on which higher values are displayed to the right and smaller (or negative) values to the left. We can draw a line representing all possible numbers between the minimum and maximum data values.

Add data to the number line

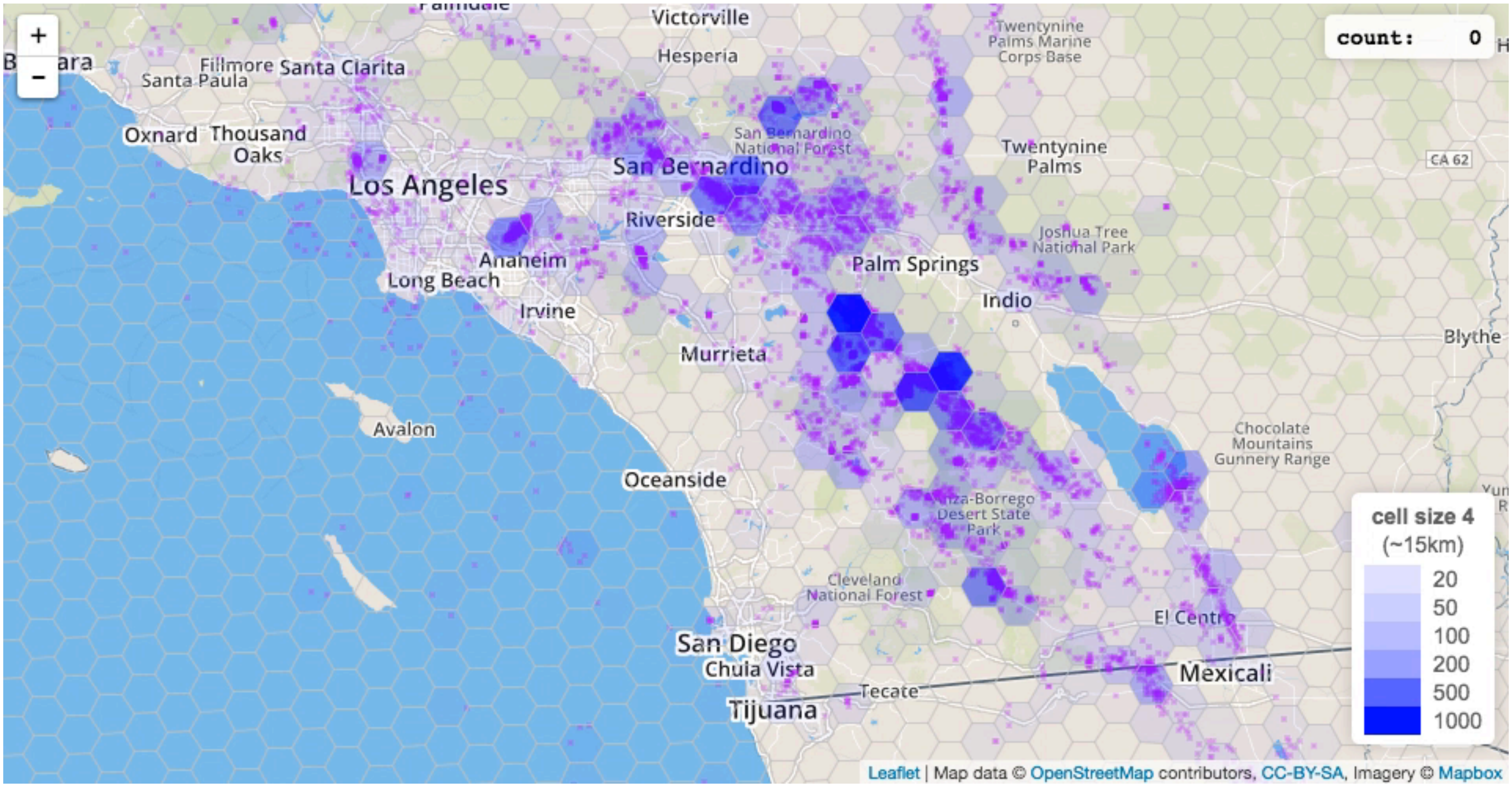
Now, we map each item to a dot at the appropriate point along the number line. In our visualization we draw the path followed by each item on its way from the list to the line, helping to reveal how adjacent list items end up close or far apart on the number line

http://tinlizzie.org/histograms/



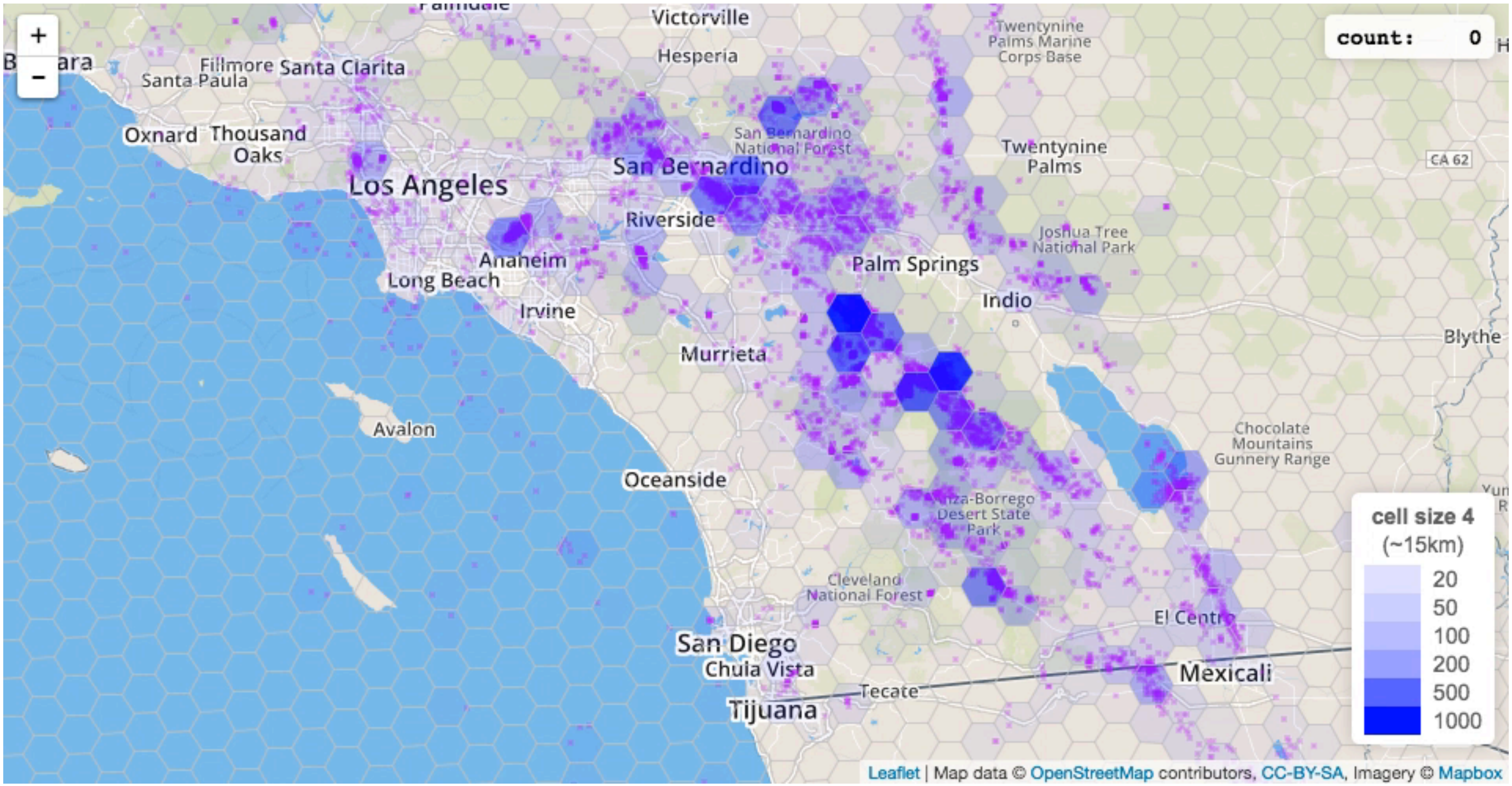
Konold, C. et al. "Data seen through different lenses." *Educational Studies in Mathematics*, 2014.

Spatial aggregation toy



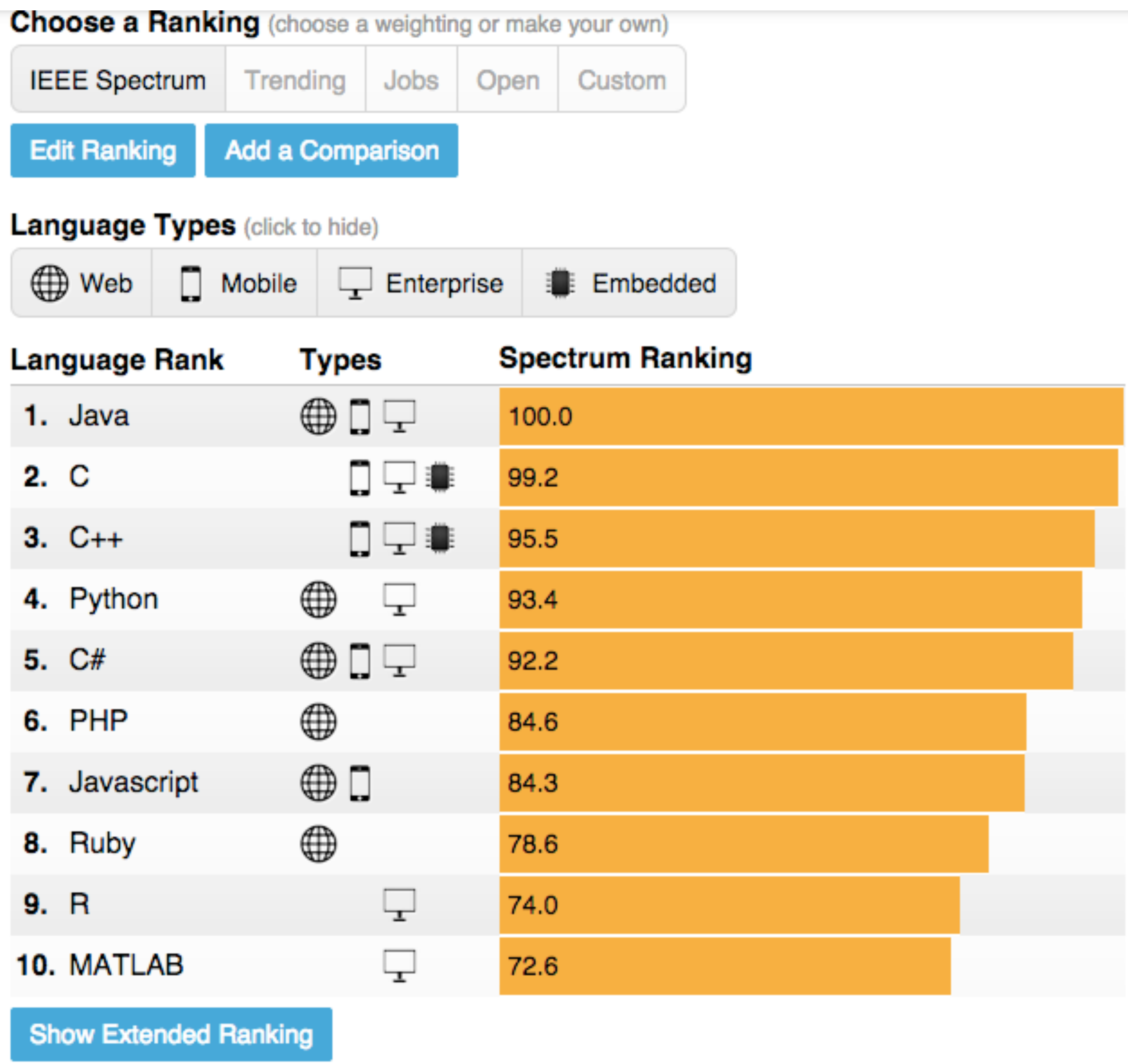
http://bit.ly/spatial_agg

Spatial aggregation toy



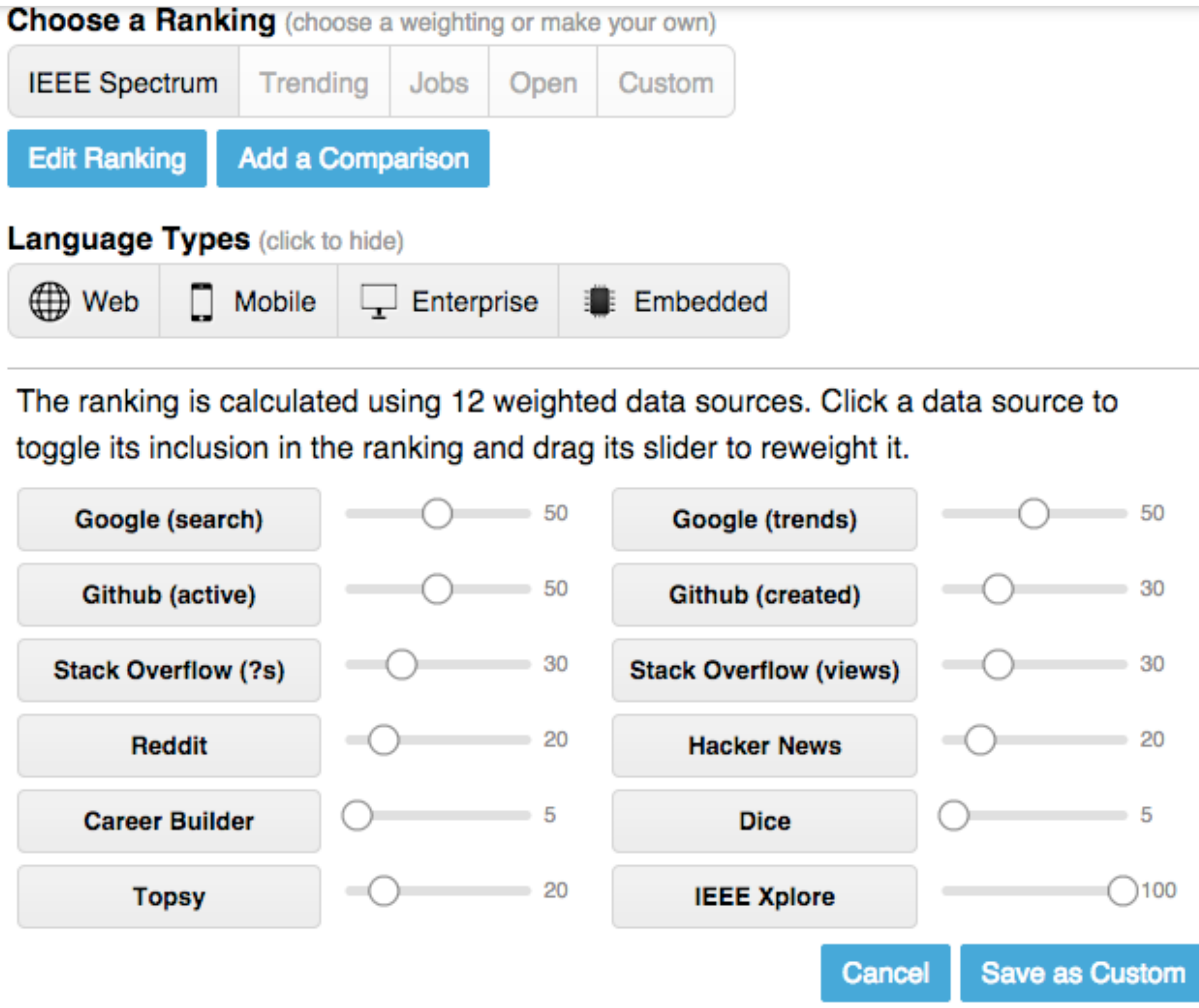
http://bit.ly/spatial_agg

Interactivity in published work

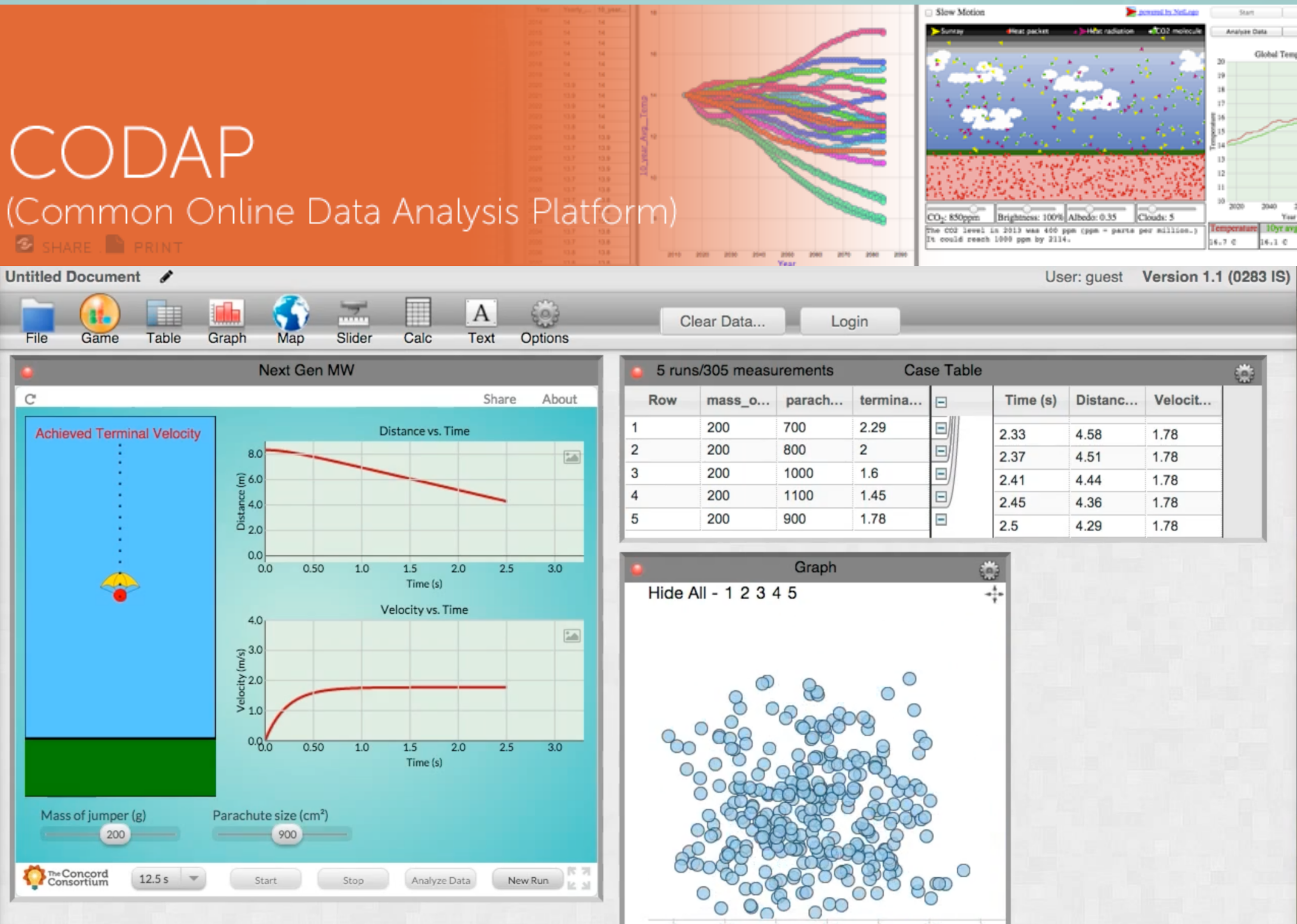


http://bit.ly/IEEE_languagerank

Interactivity in published work



http://bit.ly/IEEE_languagerank



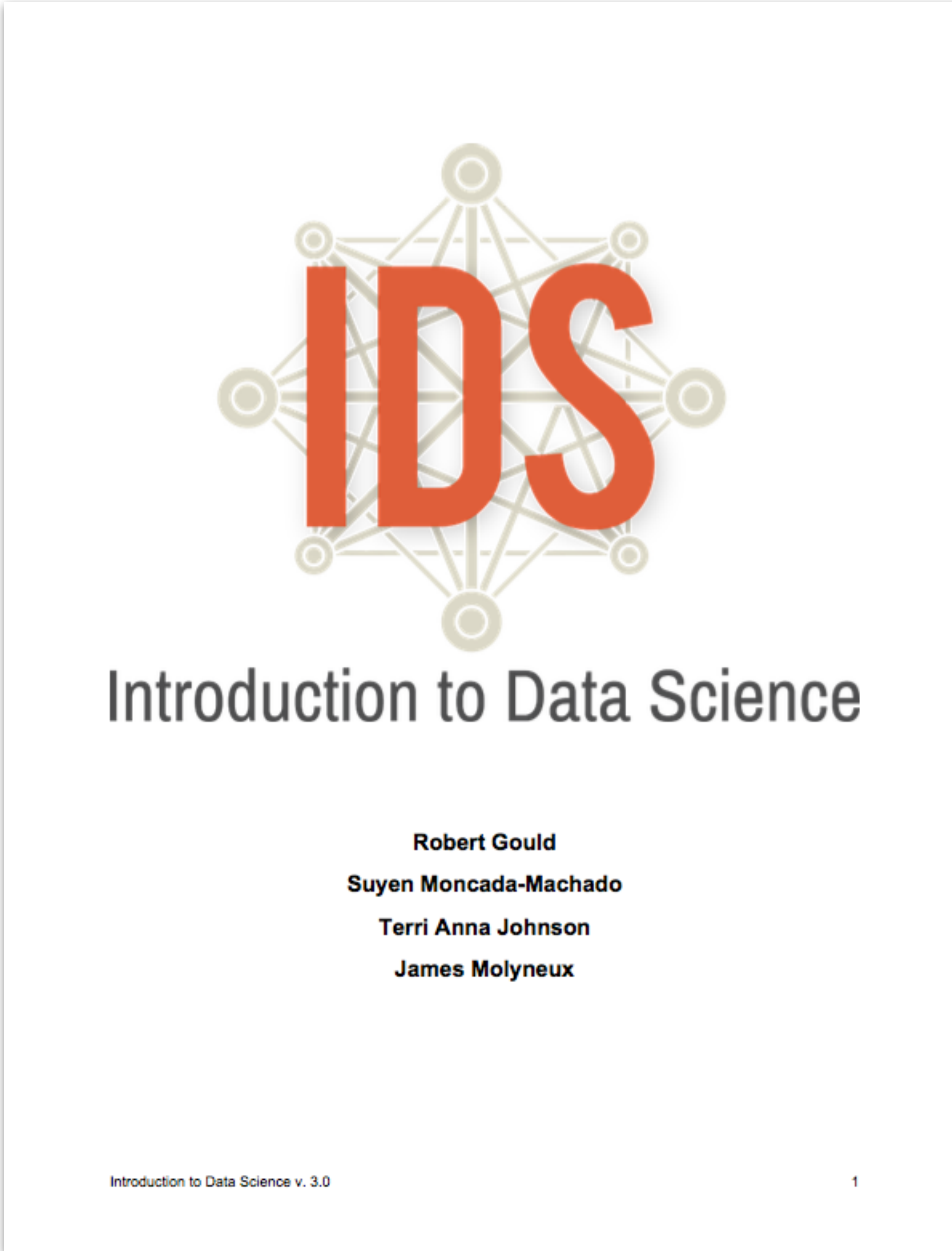
http://concord.org/projects/codap

Curriculum

mobilize

Our data. Our lives.

www.mobilizingcs.org/

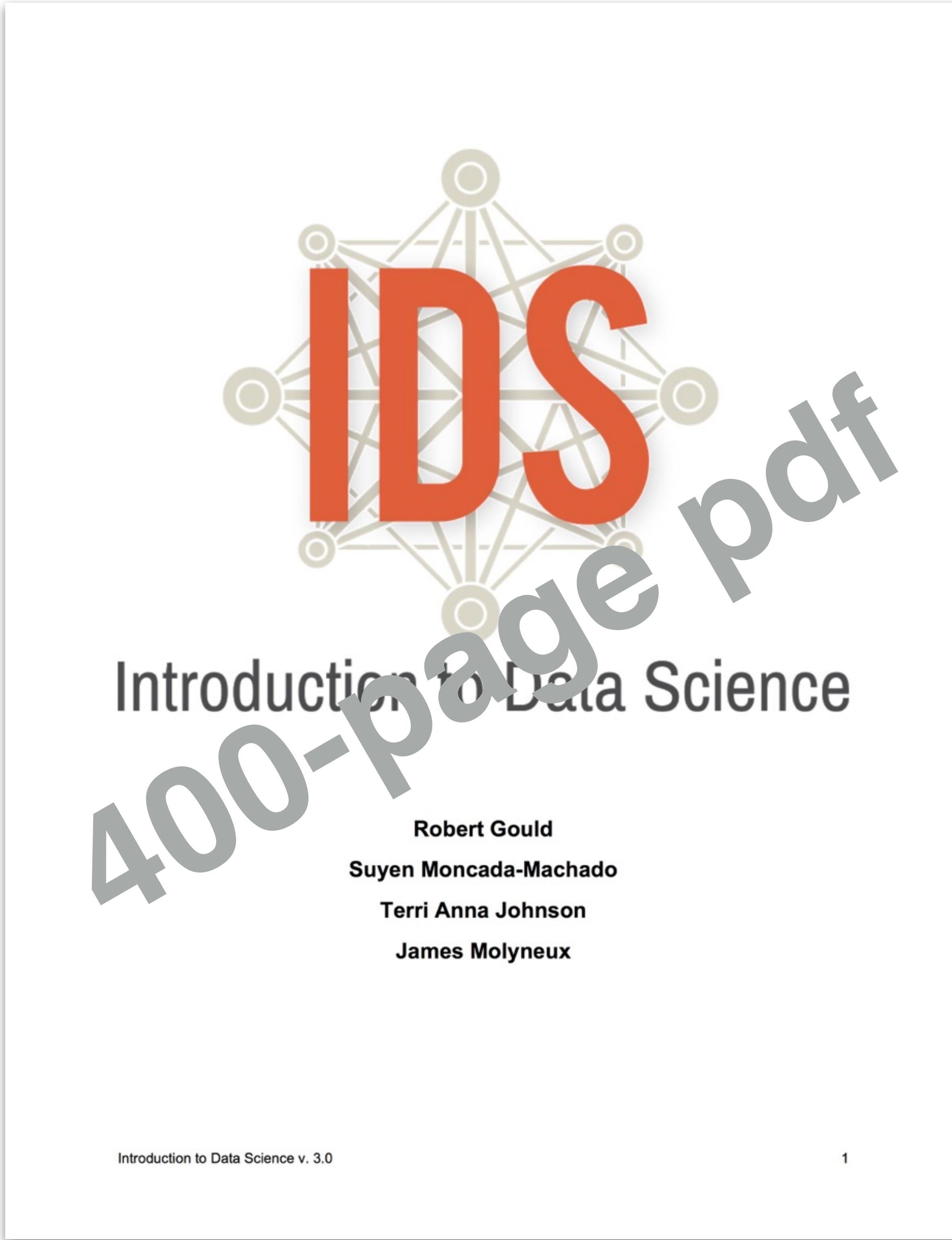

Introduction to Data Science

Robert Gould
Suyen Moncada-Machado
Terri Anna Johnson
James Molyneux

Introduction to Data Science v. 3.0 1

- Year-long course
- Validates Algebra II requirement
- “Data science”
- Taught in R within RStudio server
- Participatory sensing
- Content includes:
  - Exploratory data analysis
  - Randomization, simulation, bootstrapping
  - Simple linear regression, multiple regression
  - Decision trees, clustering, k-means

www.mobilizingcs.org/



IDS

Introduction to Data Science

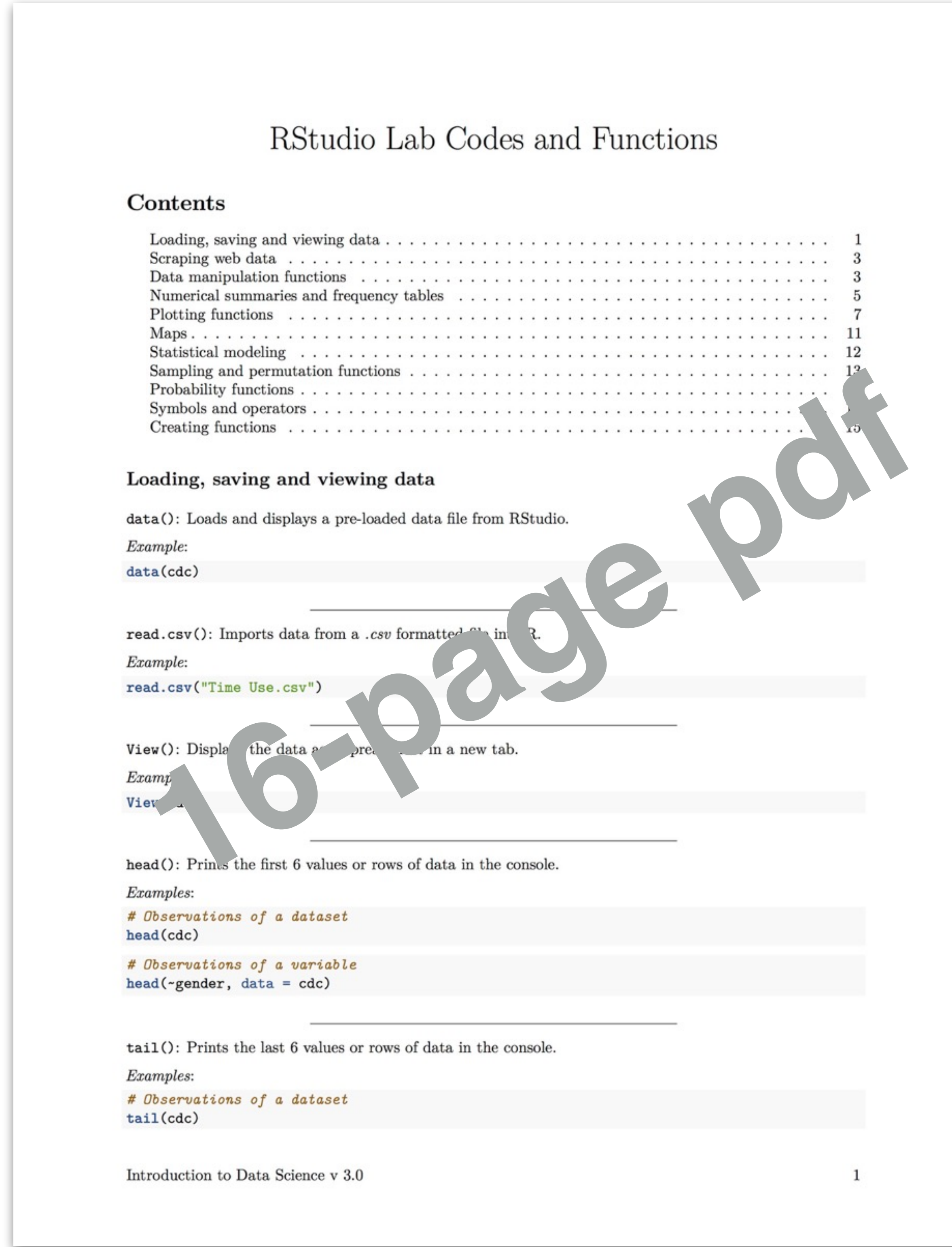
Robert Gould

Suyen Moncada-Machado

Terri Anna Johnson

James Molyneux

Introduction to Data Science v. 3.0 1



RStudio Lab Codes and Functions

Contents

Loading, saving and viewing data	1
Scraping web data	3
Data manipulation functions	3
Numerical summaries and frequency tables	5
Plotting functions	7
Maps	11
Statistical modeling	12
Sampling and permutation functions	13
Probability functions	
Symbols and operators	
Creating functions	15

Loading, saving and viewing data

`data()`: Loads and displays a pre-loaded data file from RStudio.

Example:

```
data(cdc)
```

`read.csv()`: Imports data from a *.csv* formatted file into R.

Example:

```
read.csv("Time Use.csv")
```

`View()`: Displays the data as a spreadsheet in a new tab.

Example:

```
View(cdc)
```

`head()`: Prints the first 6 values or rows of data in the console.

Examples:

```
# Observations of a dataset
head(cdc)
```

```
# Observations of a variable
head(~gender, data = cdc)
```

`tail()`: Prints the last 6 values or rows of data in the console.

Examples:

```
# Observations of a dataset
tail(cdc)
```

Introduction to Data Science v 3.0 1

DataCamp

{swirl}

Learn R, in R.

OpenIntro

IDS

Introduction to Data Science

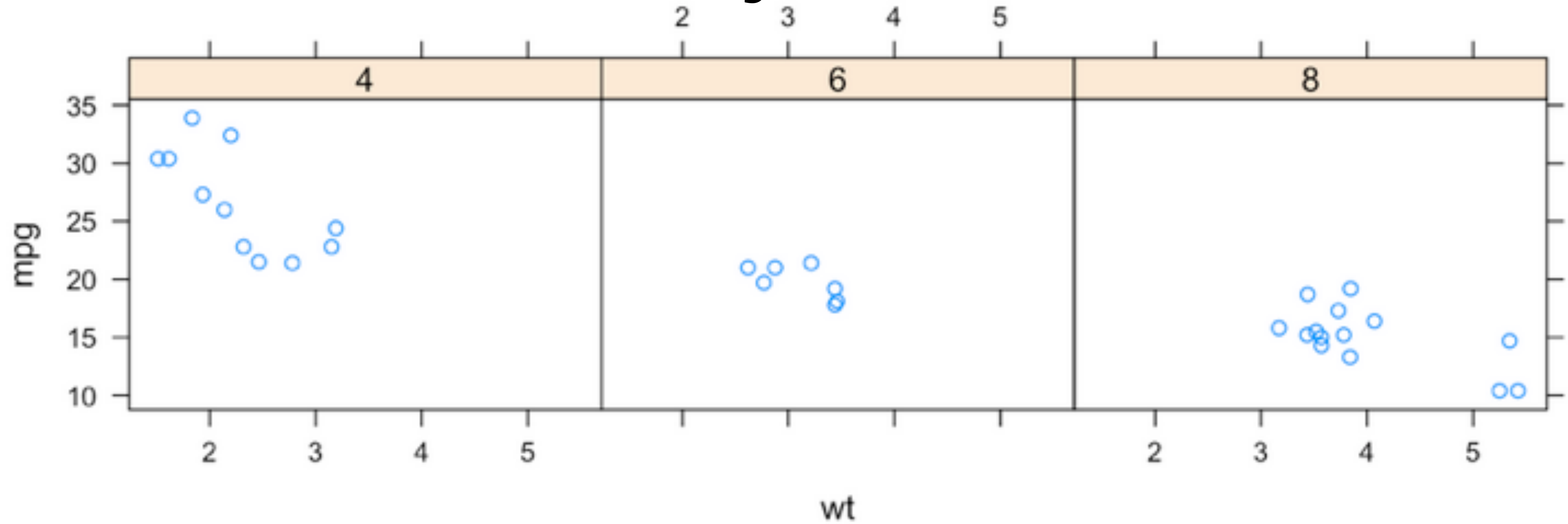
R Syntax

```
xyplot(mpg ~ wt | as.factor(cyl), data = mtcars)
```

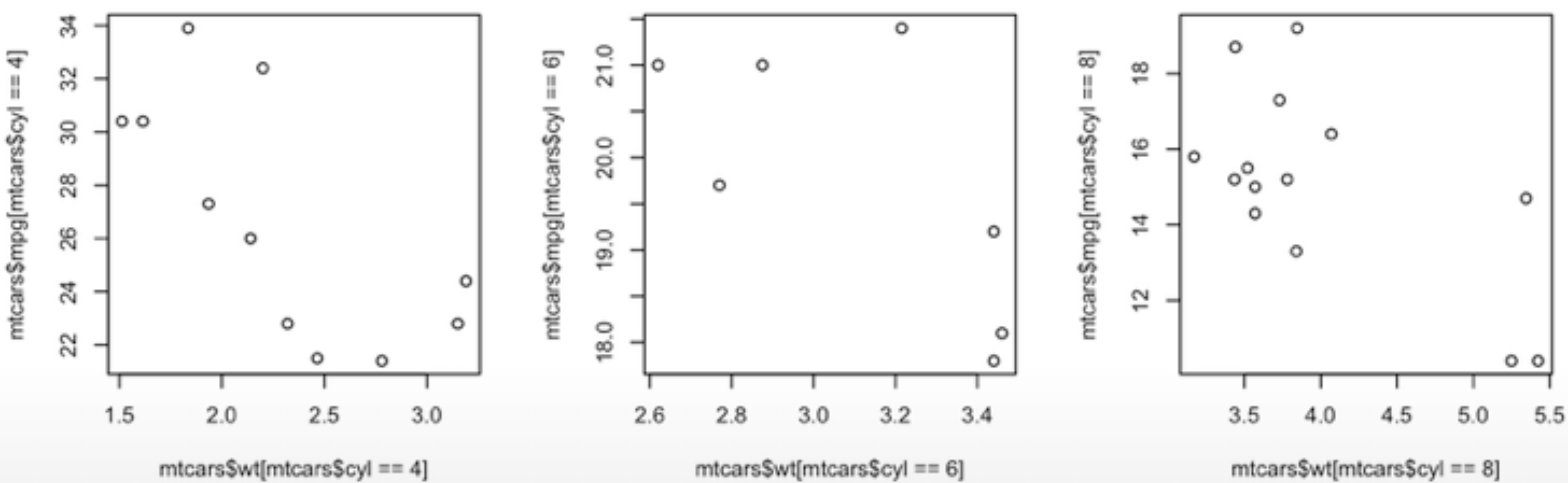
vs.

```
par(mfrow = c(1,3))
plot(mtcars$wt[mtcars$cyl == 4], mtcars$mpg[mtcars$cyl == 4])
plot(mtcars$wt[mtcars$cyl == 6], mtcars$mpg[mtcars$cyl == 6])
plot(mtcars$wt[mtcars$cyl == 8], mtcars$mpg[mtcars$cyl == 8])
```

R Syntax



4
6
8
mpg
wt



mtcars$mpg[mtcars$cyl == 4]
mtcars$wt[mtcars$cyl == 4]
mtcars$mpg[mtcars$cyl == 6]
mtcars$wt[mtcars$cyl == 6]
mtcars$mpg[mtcars$cyl == 8]
mtcars$wt[mtcars$cyl == 8]

Formula-based syntax

- `lattice` graphics
- `mosaic` package for statistics
- `mobilizr` additional functions

Want to check out the labs?

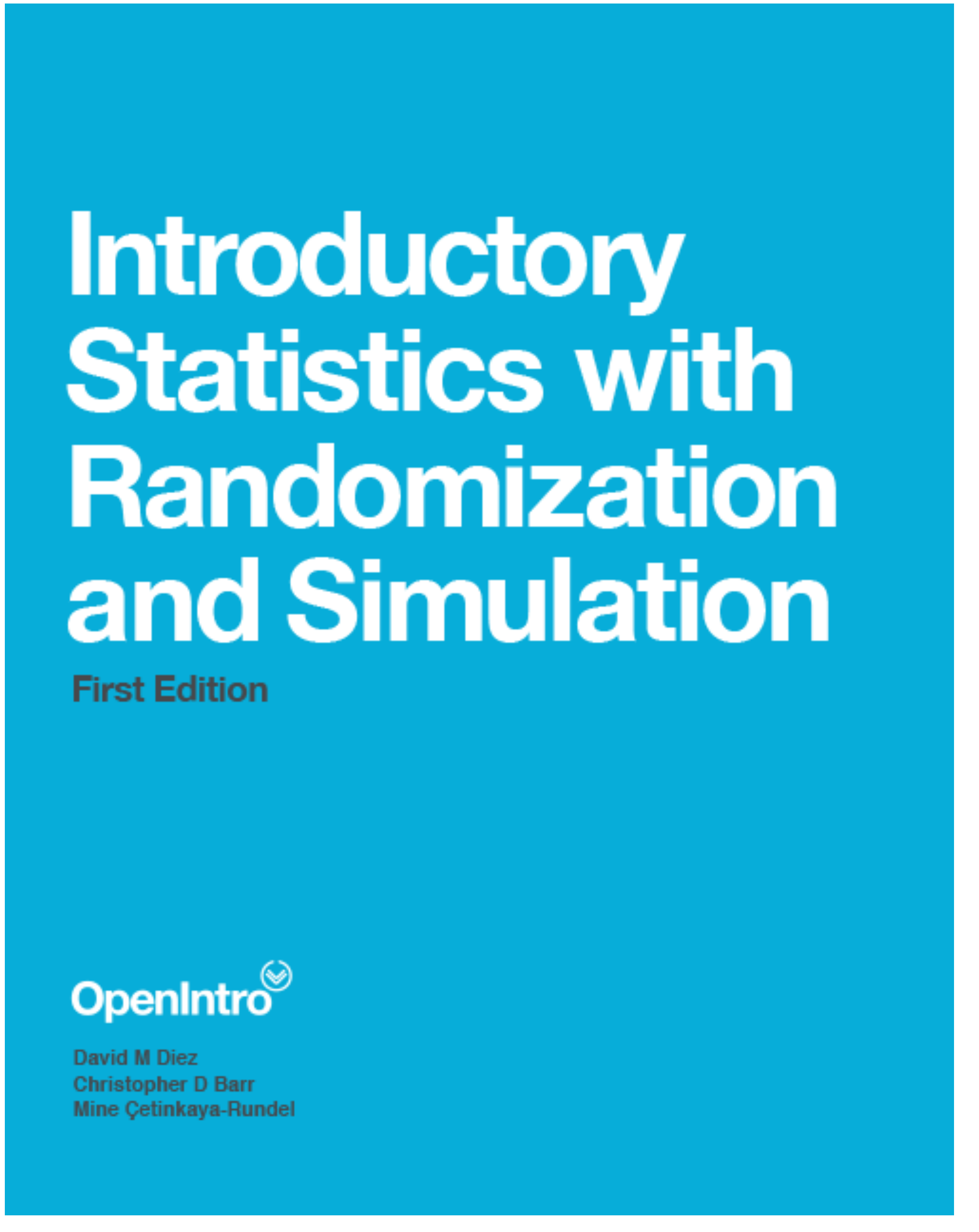
```
library(devtools)
install_github("mobilizingcs/mobilizr")
library(mobilizr)
load_labs()
```

http://stemforall2016.videohall.com/presentations/790

Data science at Smith

Introductory course

Introductory Statistics with Randomization and Simulation.
David M Diez, Christopher D Barr, Mine Çetinkaya-Rundel
www.openintro.org



R syntax comparison

Cheat Sheet

SMITH COLLEGE

Syntax

Syntax is the set of rules that govern what code works and doesn't work. Most programming languages offer one standardized syntax, but R has many.

Most people use some combination of all the syntaxes available to them.

1. Dollar sign syntax uses the dollar sign to locate a variable within a dataset. It is expected by most **base** R functions.
2. Formula syntax uses the **data=** argument at the end of a list of function arguments. The formula syntax is used by modeling functions like **lm()**, **lattice** graphics like **xyplot()**, and **mosaic** summary statistics like **mean()**.
3. Tidyverse syntax uses data as the first argument to function calls. It is used by the packages **dplyr** and **tidyr**, among others. The associated graphics library is **ggplot2**.

Dollar sign syntax

```
goal(data$x, data$y)
```

Summary statistics:

one continuous variable:
```
mean(mtcars$mpg)
```

one categorical variable:
```
table(mtcars$cyl)
```

two categorical variables:
```
table(mtcars$cyl, mtcars$am)
```

one continuous, one categorical:
```
mean(mtcars$mpg[mtcars$cyl==4])
mean(mtcars$mpg[mtcars$cyl==6])
mean(mtcars$mpg[mtcars$cyl==8])
```

Plotting:

one continuous variable:
```
hist(mtcars$disp)

boxplot(mtcars$disp)
```

one categorical variable:
```
barplot(table(mtcars$cyl))
```

two continuous variables:
```
plot(mtcars$disp, mtcars$mpg)
```

two categorical variables:
```
mosaicplot(table(mtcars$am, mtcars$cyl))
```

one continuous, one categorical:
```
histogram(mtcars$disp[mtcars$cyl==4])
histogram(mtcars$disp[mtcars$cyl==6])
histogram(mtcars$disp[mtcars$cyl==8])

boxplot(mtcars$disp[mtcars$cyl==4])
boxplot(mtcars$disp[mtcars$cyl==6])
boxplot(mtcars$disp[mtcars$cyl==8])
```

Wrangling:

subsetting:
```
mtcars[mtcars$mpg>30, ]
```

making a new variable:
```
mtcars$efficient[mtcars$mpg>30] <- TRUE
mtcars$efficient[mtcars$mpg<30] <- FALSE
```

Formula syntax

```
goal(y~x|z, data=data, group=w)
```

Summary statistics:

one continuous variable:
```
mosaic::mean(~mpg, data=mtcars)
```

one categorical variable:
```
mosaic::tally(~cyl, data=mtcars)
```

two categorical variables:
```
mosaic::tally(cyl~am, data=mtcars)
```

one continuous, one categorical:
```
mosaic::mean(mpg~cyl, data=mtcars)
```

tilde

Plotting:

one continuous variable:
```
lattice::histogram(~disp, data=mtcars)

lattice::bwplot(~disp, data=mtcars)
```

one categorical variable:
```
mosaic::bargraph(~cyl, data=mtcars)
```

two continuous variables:
```
lattice::xyplot(mpg~disp, data=mtcars)
```

two categorical variables:
```
mosaic::bargraph(~am, data=mtcars,
    group=cyl)
```

one continuous, one categorical:
```
lattice::histogram(~disp|cyl, data=mtcars)

lattice::bwplot(cyl~disp, data=mtcars)
```

Tidyverse syntax

```
data %>% goal(x)
```

Summary statistics:

one continuous variable:
```
mtcars %>% dplyr::summarize(mean(mpg))
```

one categorical variable:
```
mtcars %>% dplyr::group_by(cyl) %>%
    dplyr::summarize(n())
```

the pipe

two categorical variables:
```
mtcars %>% dplyr::group_by(cyl, am) %>%
    dplyr::summarize(n())
```

one continuous, one categorical:
```
mtcars %>% dplyr::group_by(cyl) %>%
    dplyr::summarize(mean(mpg))
```

Plotting:

one continuous variable:
```
ggplot2::qplot(x=mpg, data=mtcars,
    geom = "histogram")

ggplot2::qplot(y=disp, x=1, data=mtcars,
    geom="boxplot")
```

one categorical variable:
```
ggplot2::qplot(x=cyl, data=mtcars, geom="bar")
```

two continuous variables:
```
ggplot2::qplot(x=disp, y=mpg, data=mtcars,
    geom="point")
```

two categorical variables:
```
ggplot2::qplot(x=factor(cyl), data=mtcars, geom="bar")
    + facet_grid(.~am)
```

one continuous, one categorical:
```
ggplot2::qplot(y=disp, x=factor(cyl), data=mtcars,
    geom="boxplot")

ggplot2::qplot(x=disp, data=mtcars, geom = "histogram")
    + facet_grid(.~cyl)
```

Wrangling:

subsetting:
```
mtcars %>%
    dplyr::filter(mpg>30)
```

making a new variable:
```
mtcars <- mtcars %>%
    dplyr::mutate(efficient = if_else(mpg>30,
    TRUE, FALSE))
```

- Graphical literacy skills should be taught in the context of science and social science.
- Translating between representations may be beneficial
- Explicitly focus on the links between visual features and meaning.
- Make graph reading metacognitive

Shah, P. and Hoeffner, J. “Review of Graph Comprehension Research: Implications for Instruction.” *Educational Psychology Review*, 2002.

New faculty

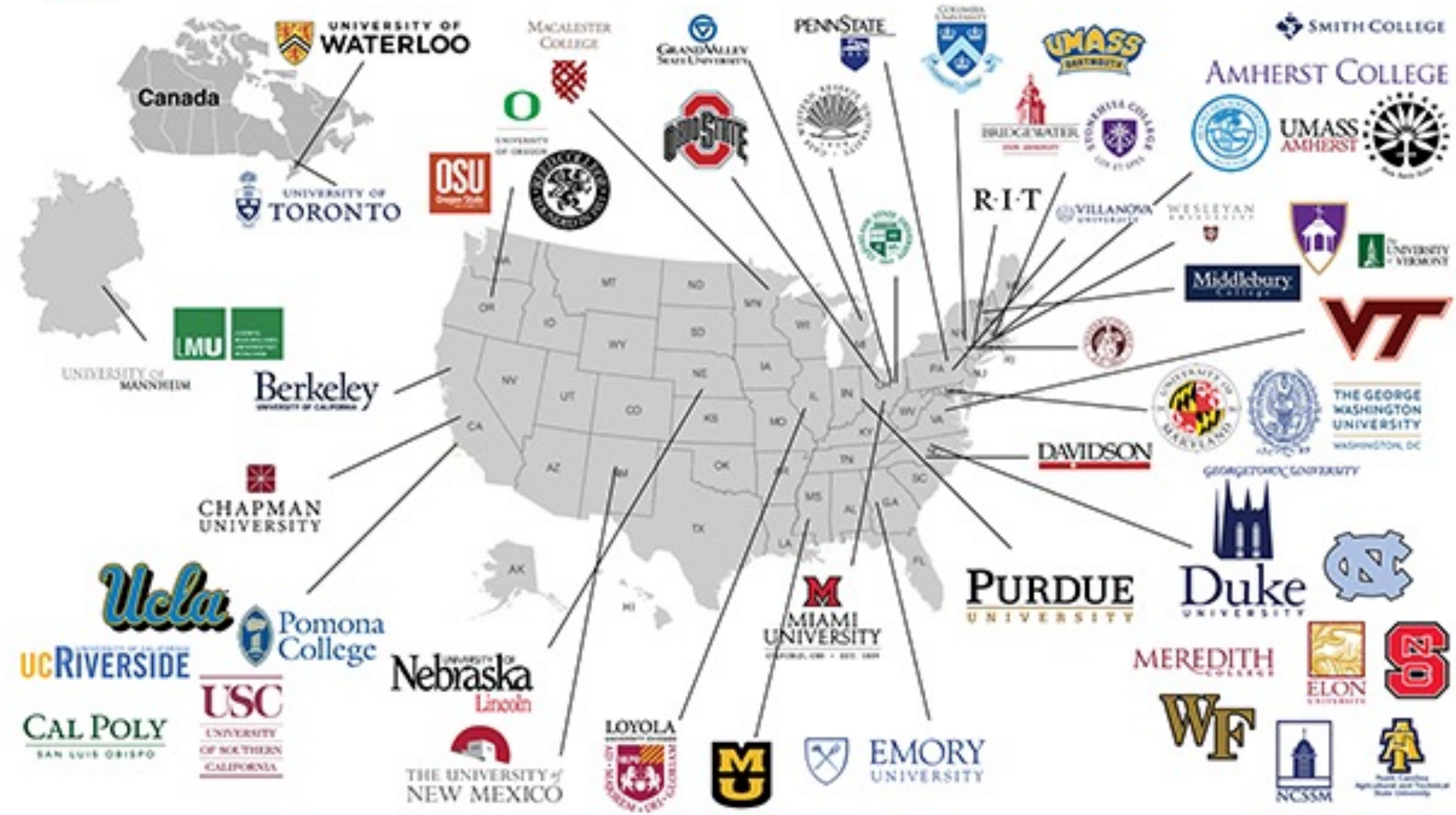
- SDS 136: Communicating with Data
- SDS 192: Introduction to Data Science
- SDS 235: Visual Analytics
- SDS 236: Data Journalism
- SDS 293: Machine Learning

New courses

ASA DataFest™

- Started at UCLA
- Now a national and international event
- Groups of undergraduate students compete to find insight in data
- Past data sponsors: LAPD, Kiva.com, eHarmony, Travelocity

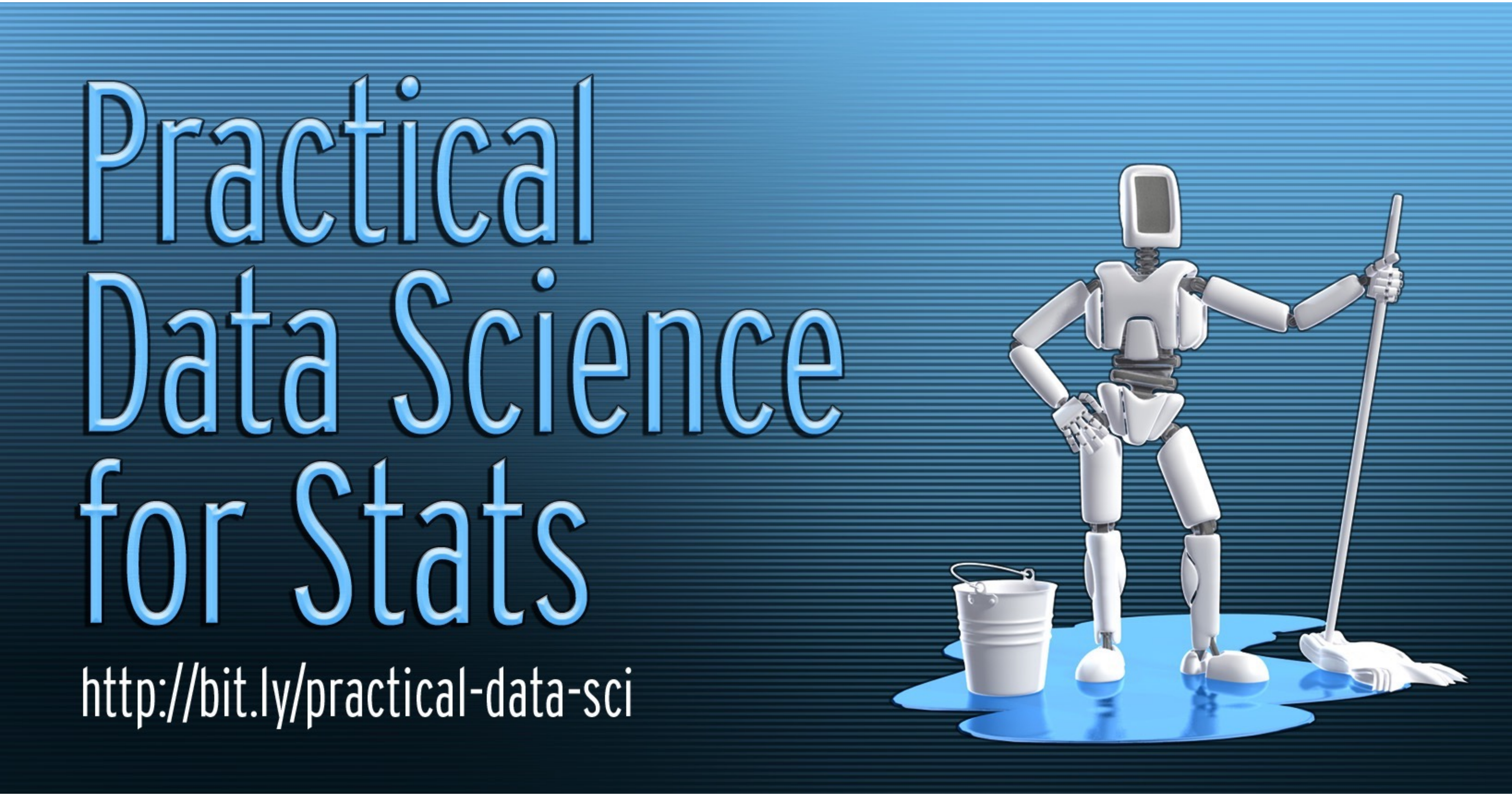
ASA DataFest™



http://ww2.amstat.org/education/datafest/

http://www.science.smith.edu/datafest/

Data Science as a Science



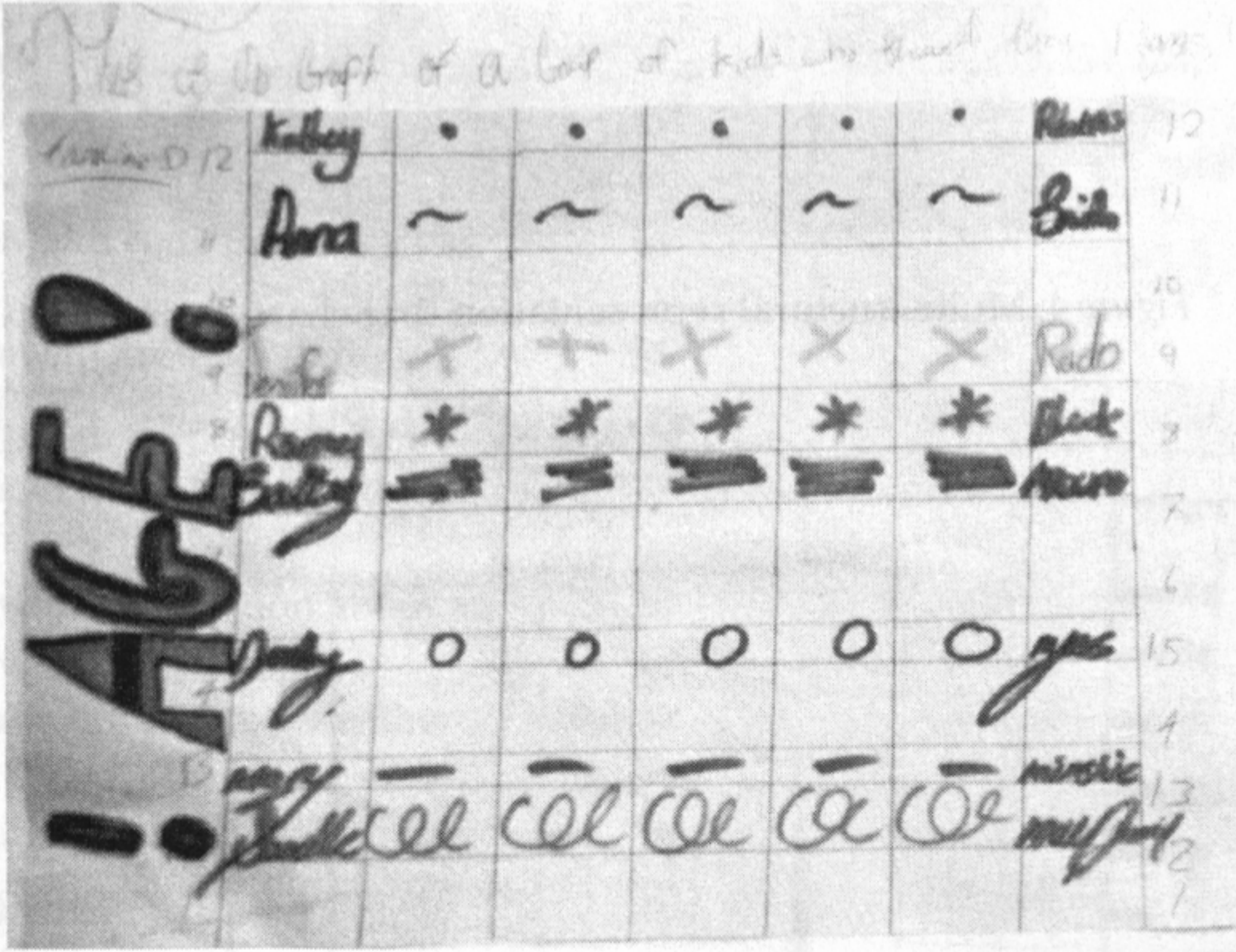


Figure 3. Prestructural representation from Group 5. (The caption at the top reads "This is a graph of a group of kids who showed their ages".)

Chick, H. and Watson, J. "Data representation and interpretation by primary school students working in groups." *Mathematics Education Research Journal,* 2001.

Thank you

Amelia McNamara (@AmeliaMN)

flickr: @N01