Gerrymandering and computational redistricting:
conceptual, software, data challenges

Olivia Guest
http://oliviaguest.com

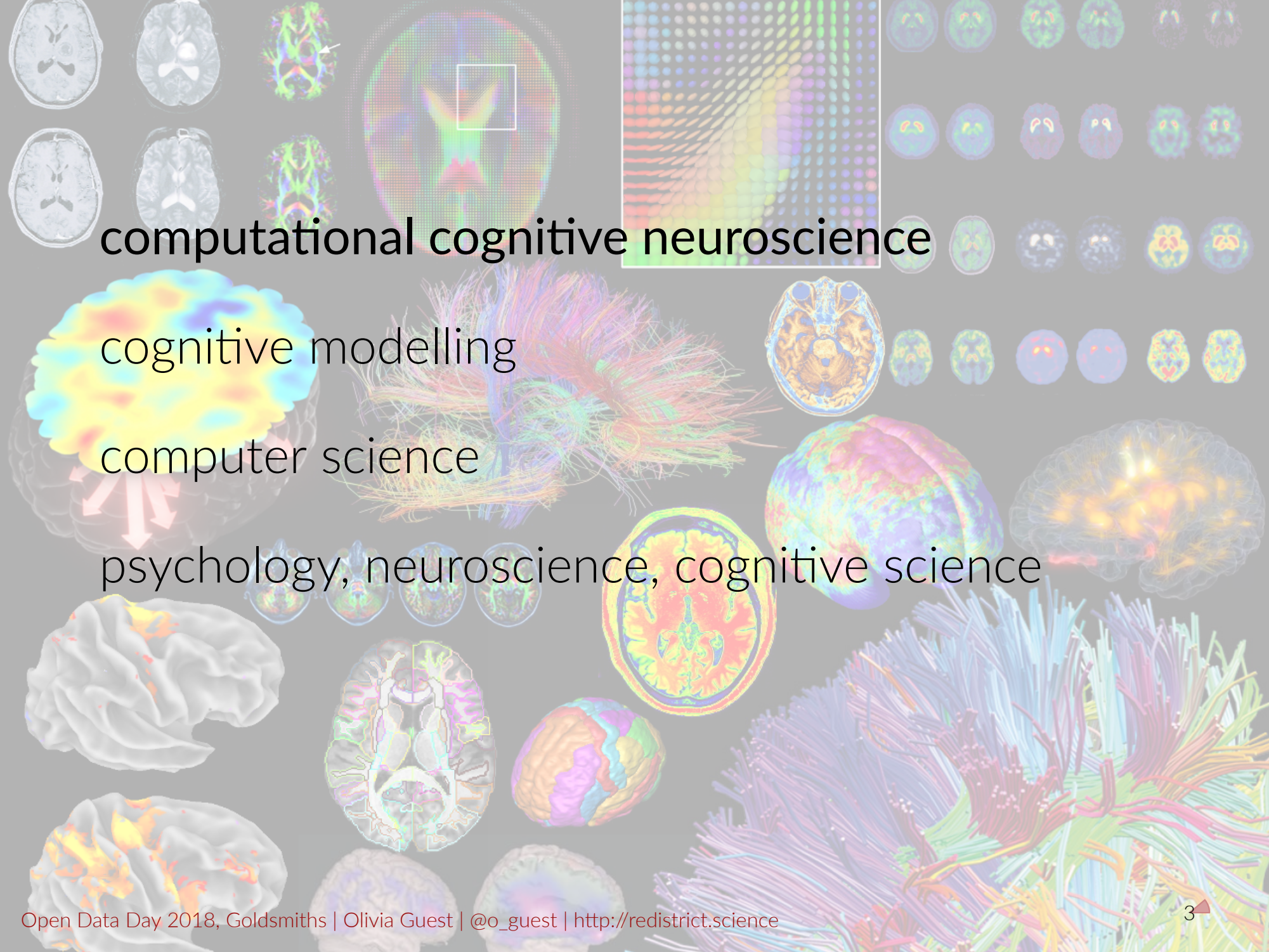
March 3, 2018
Open Data Day, Goldsmiths

Collaborator

Bradley C. Love

University College London &
The Alan Turing Institute

http://bradlove.org



computational cognitive neuroscience

cognitive modelling

computer science

psychology, neuroscience, cognitive science

Talk Outline

Part 1: Science

Talk Outline

Part 1: Science

Part 2: Conceptual, software, data challenges

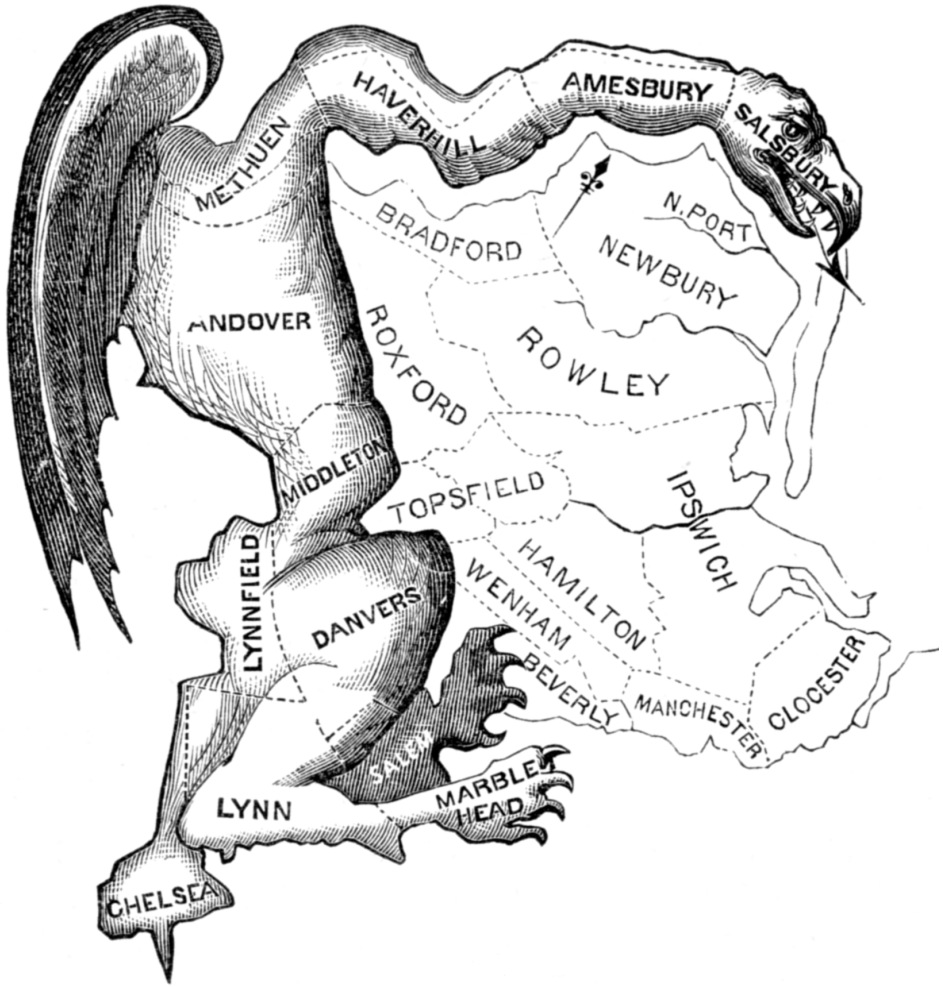
“how it was done”

What is gerrymandering?

(re)draw districts to benefit party

corruption

complex shape

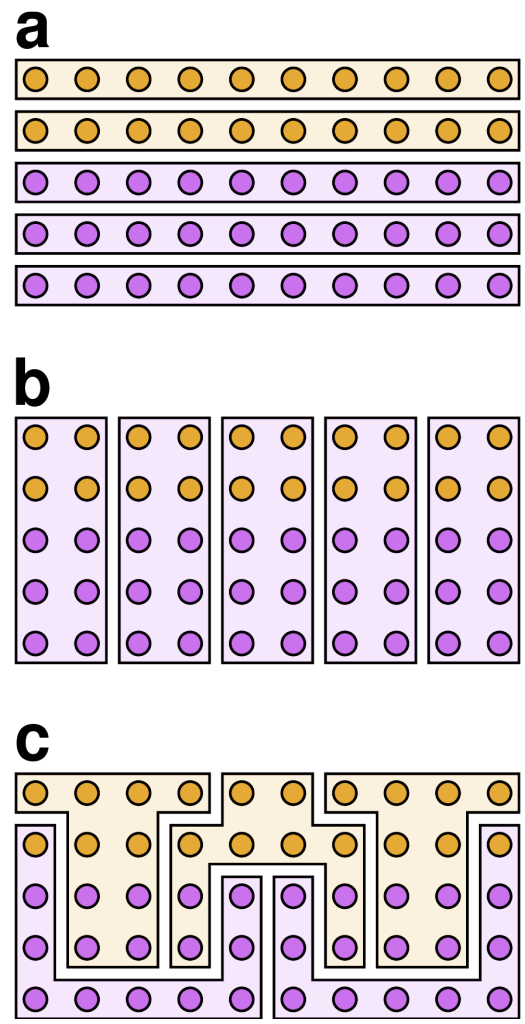


What is gerrymandering?

compact, fair

compact, not fair

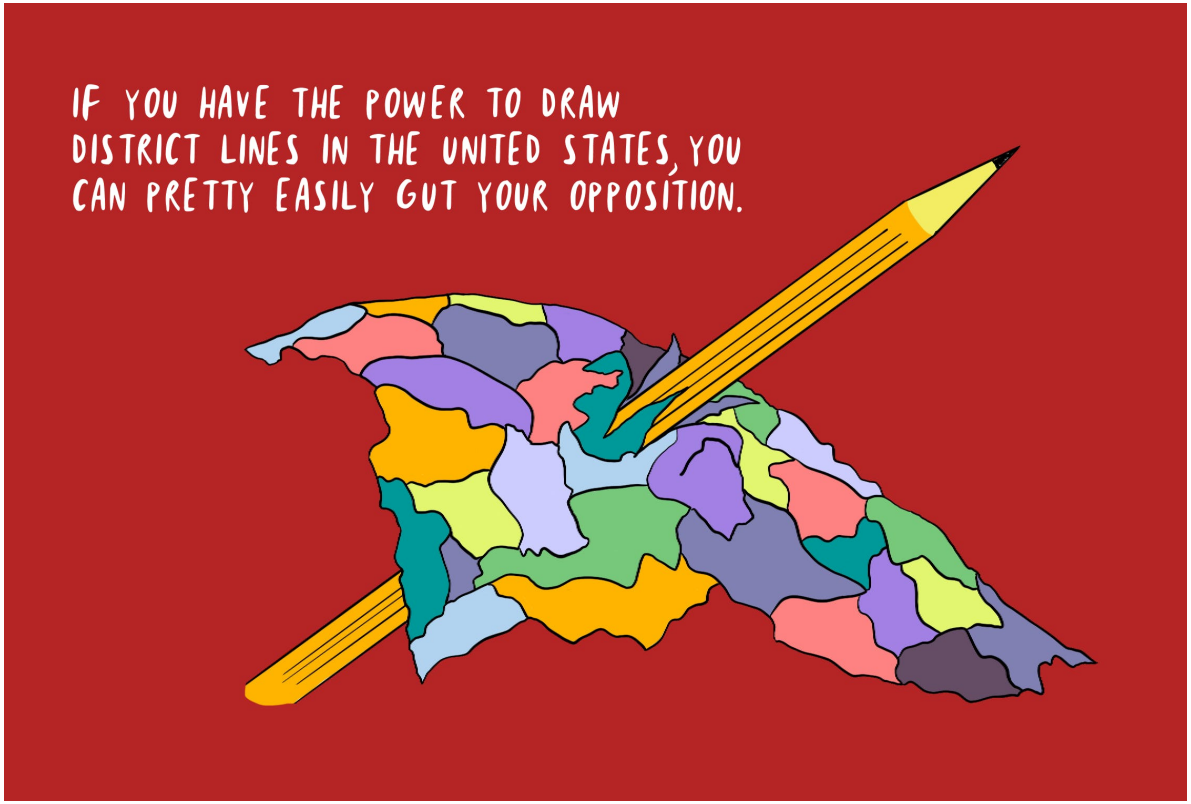
not compact, not fair



What is gerrymandering?

17 R states:
72% of seats;
52% of vote

6 D states:
71% of the seats;
56% of vote



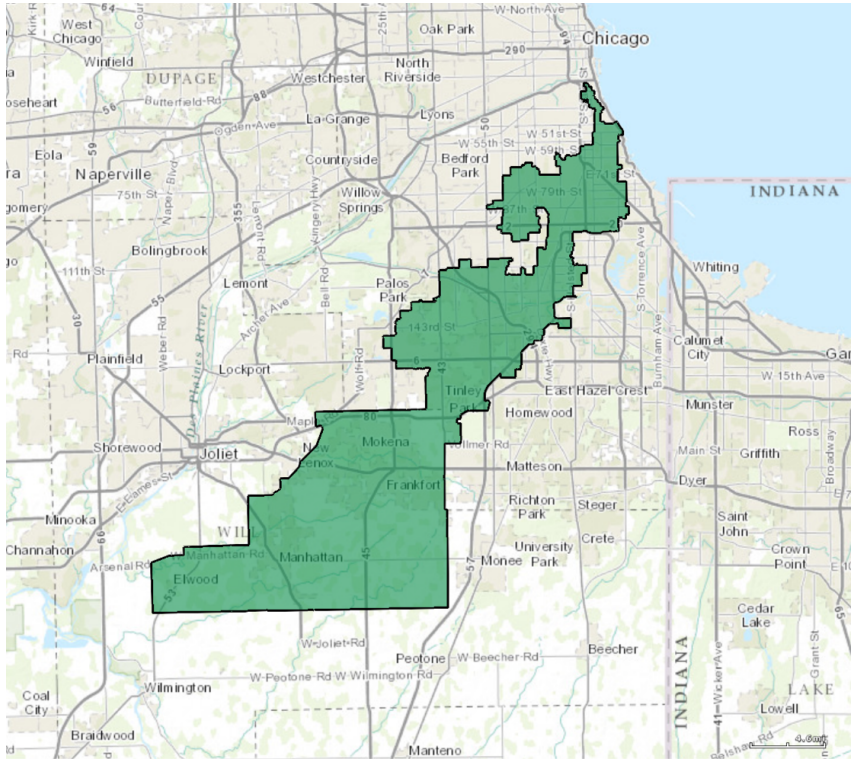
by Olivia Walch, @oliviawalch

What is gerrymandering?

2000: Obama lost in IL

opponent allowed to draw maps

…carved Obama's house out!



Levitt, Justin, The Legal Context for Scientific Redistricting Analysis (March 28, 2011). NEW TECHNOLOGY AND THE OLD PROBLEMS OF REDISTRICTING, Forthcoming; Loyola-LA Legal Studies Paper No. 2011-13. Available at SSRN: https://ssrn.com/abstract=1798005 or http://dx.doi.org/10.2139/ssrn.1798005

Computational Redistricting

humans debate
districting criteria

machines draw
the district
boundaries

open source, open
data
don't get tired

Psychological Perspective

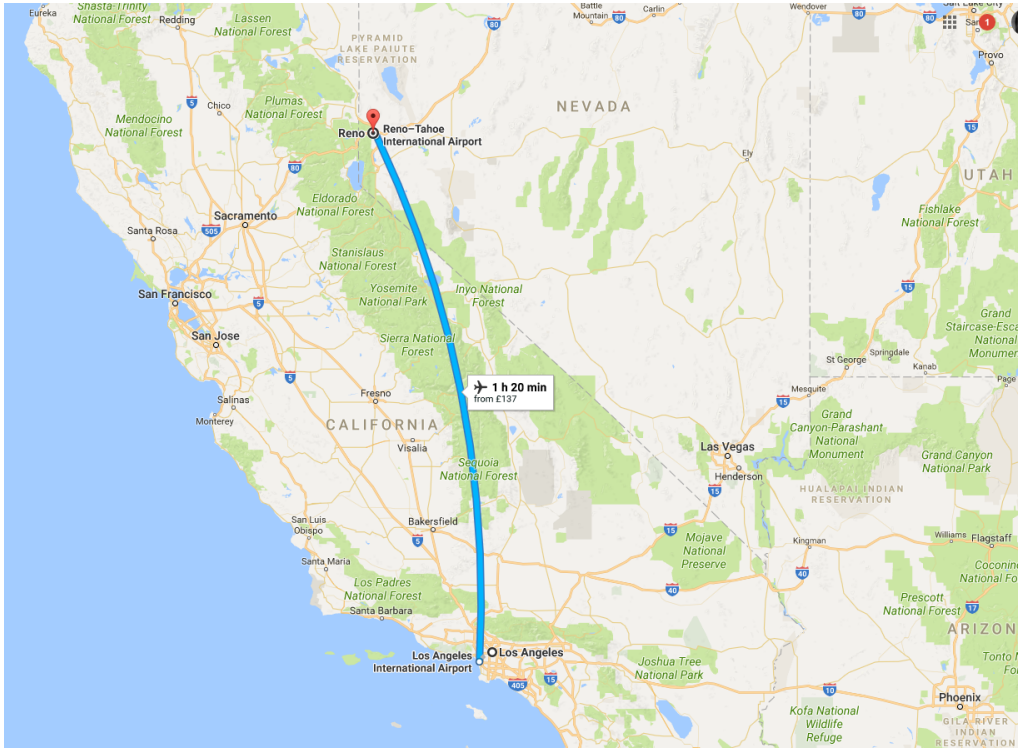
humans get
confused by the
world

Myers, 2002. Intuition: Its Powers and Perils.

Psychological Perspective

humans get confused by the world

Reno east of LA



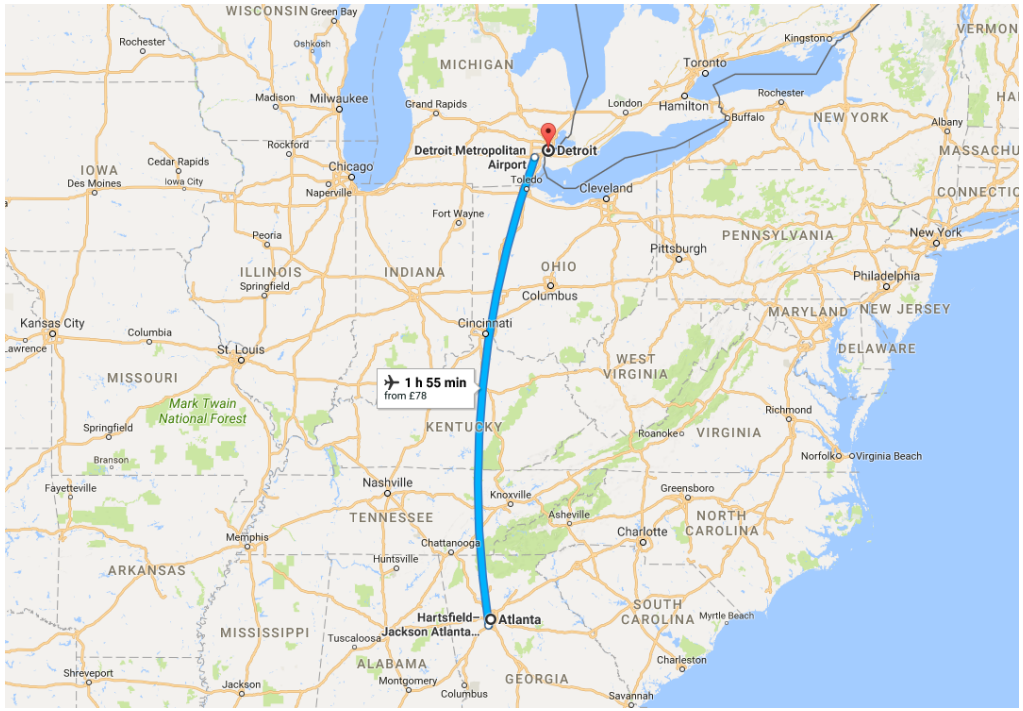
Myers, 2002. Intuition: Its Powers and Perils.

Psychological Perspective

humans get confused by the world

Reno east of LA

Altanta east of Detroit



Myers, 2002. Intuition: Its Powers and Perils.

Analysis

Improvement as func of number of districts and (number of districts)^2

$R^2 = 0.550, F(2, 40) = 24.47, p \approx 0$

Analysis

Improvement as func of number of districts and (number of districts)^2

$R^2 = 0.550, F(2, 40) = 24.47, p \approx 0$

20% improvement per state on average

Analysis

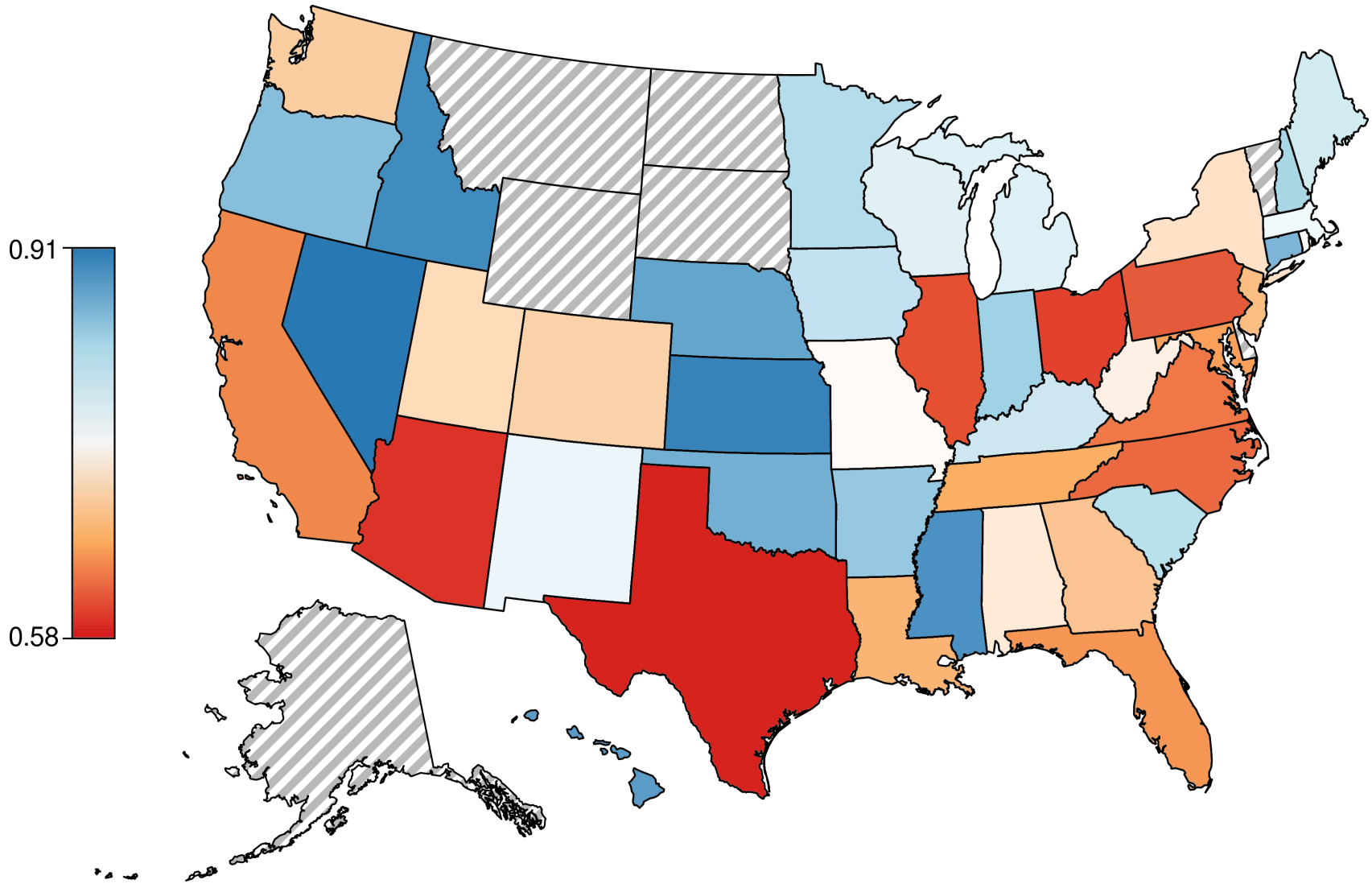
Improvement as func of number of districts and (number of districts)^2

$R^2 = 0.550, F(2, 40) = 24.47, p \approx 0$

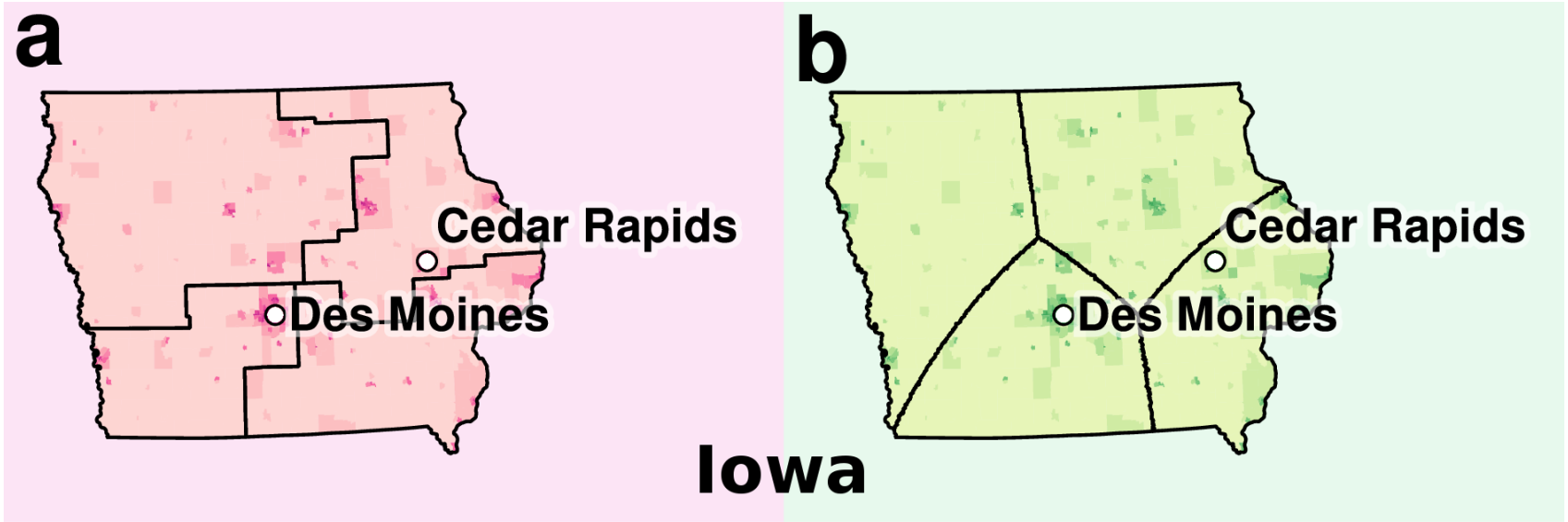
20% improvement per state on average

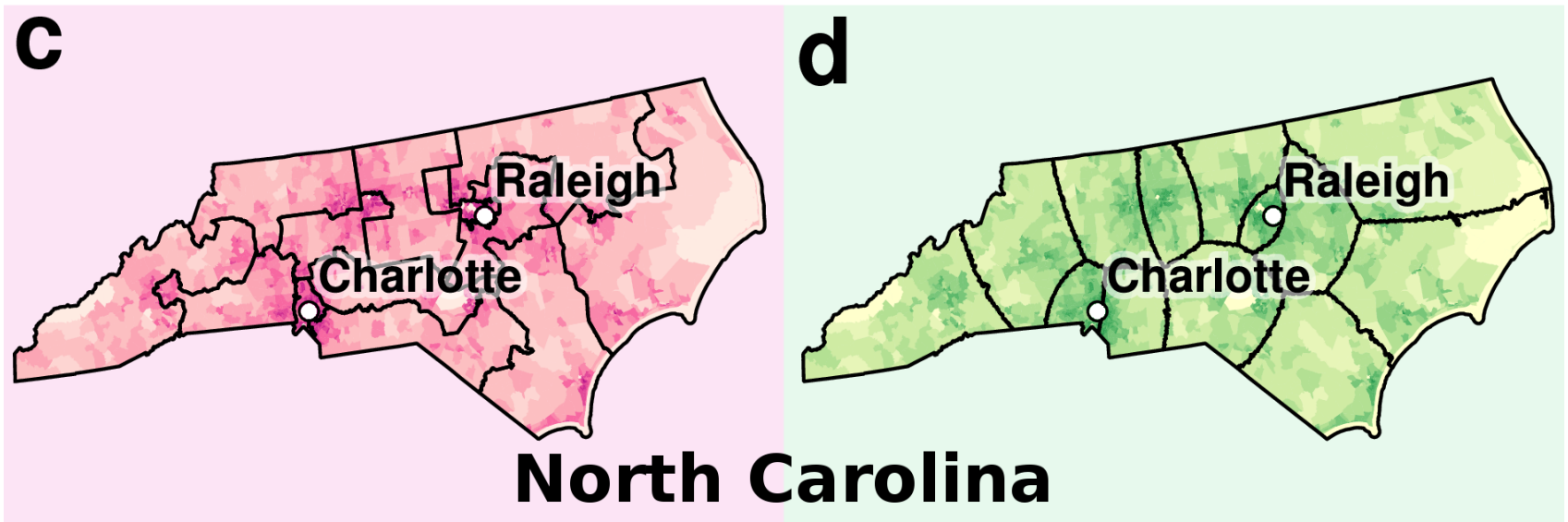
cognitive demands may tax humans

Our Results

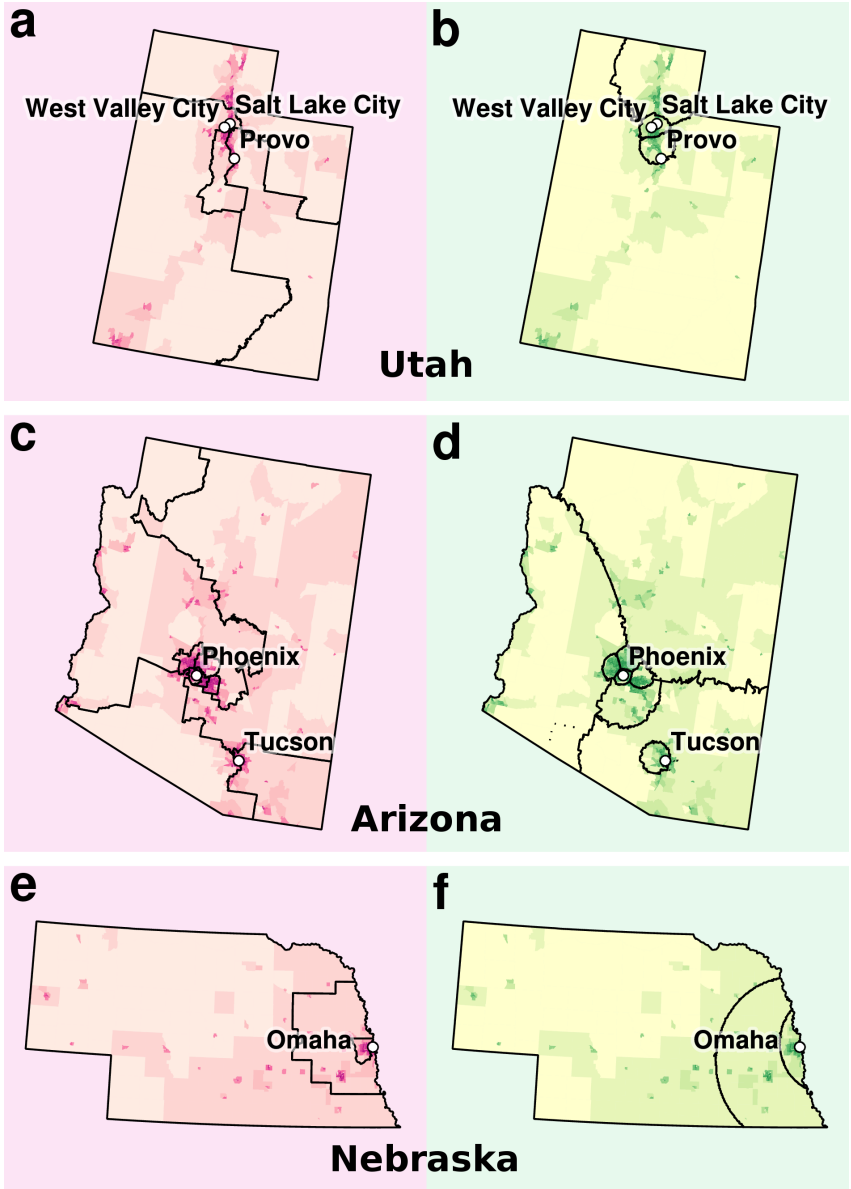


Our Results


a
Cedar Rapids
Des Moines
b
Cedar Rapids
Des Moines
Iowa


c
Raleigh
Charlotte
d
Raleigh
Charlotte
North Carolina

Our Results


a
West Valley City
Salt Lake City
Provo
b
West Valley City
Salt Lake City
Provo
Utah
c
Phoenix
Tucson
d
Phoenix
Tucson
Arizona
e
Omaha
f
Omaha
Nebraska

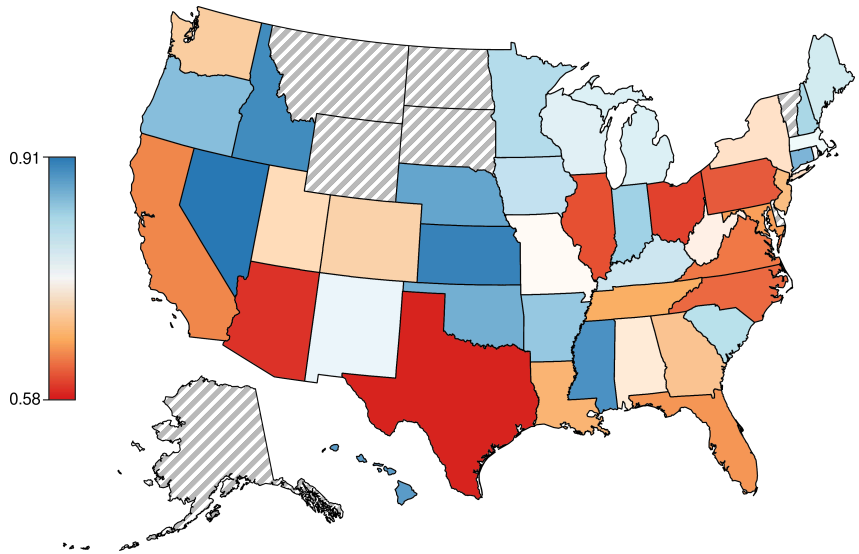
More Results

larger states → more improvement

complex for humans

gerrymandering may be accidental

Arizona is worst state



Discussion

political, legal, academic debate on criteria

Discussion

political, legal, academic debate on criteria

computers do the grunt work

Discussion

political, legal, academic debate on criteria

computers do the grunt work

open software/data for transparency

End of Part 1

Conceptual Challenges

understand US:
legal system,
politics

Conceptual Challenges

understand US: legal system, politics

why are they like this?

Conceptual Challenges

understand US: legal system, politics

other research: disparate, unconsolidated

why are they like this?

Conceptual Challenges

understand US: legal system, politics

other research: disparate, unconsolidated

algorithm development***

why are they like this?

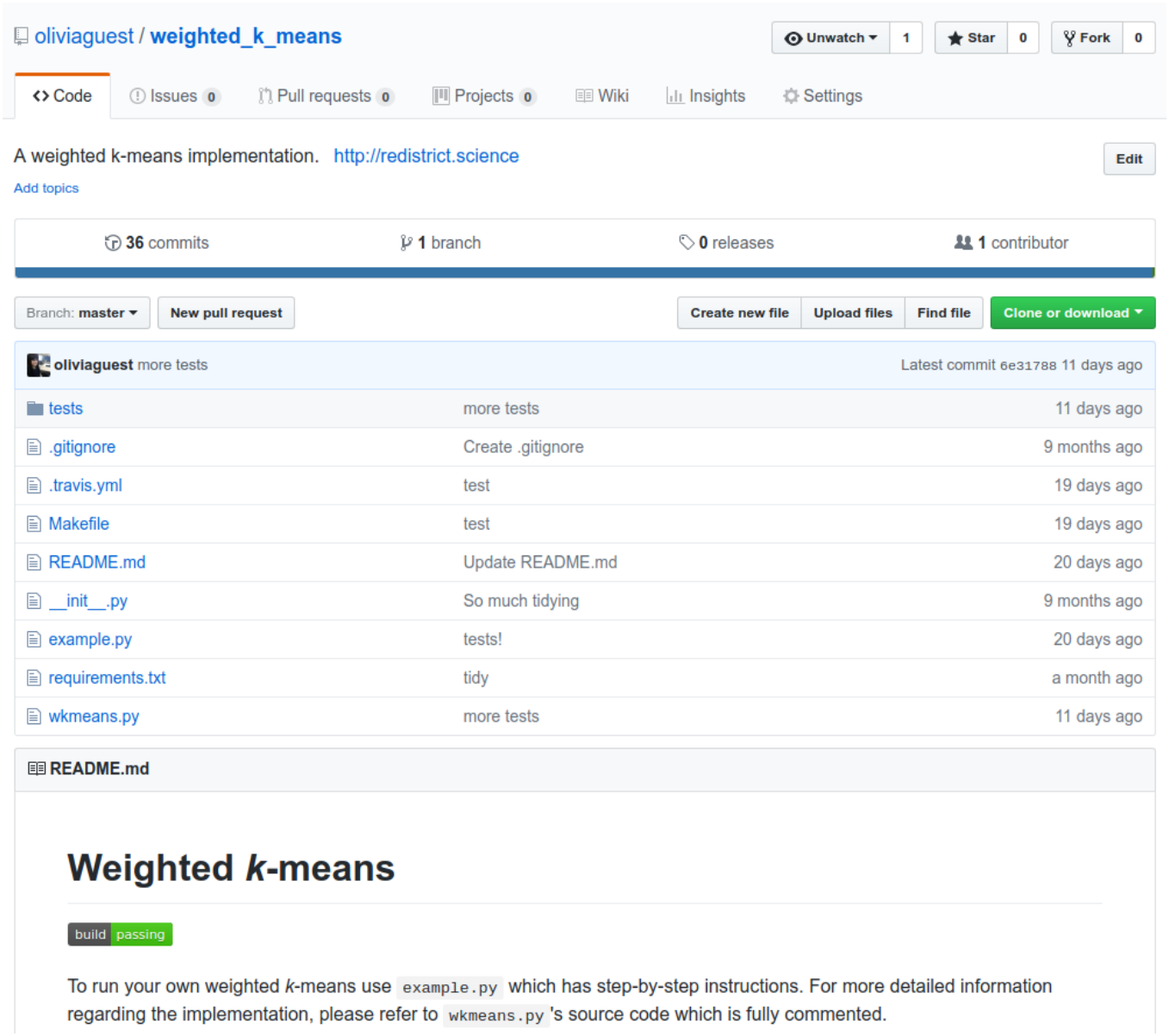
Our Algorithm

Weighted *k*-means:
clusters = districts, points = census blocks

The weight for cluster *i* is

$$w_i = \frac{|C_i|^\alpha}{\sum_{j=1}^{K} |C_j|^\alpha}$$

Weighted *k*-means



oliviaguest / **weighted_k_means**

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

A weighted k-means implementation. http://redistrict.science

Edit

Add topics

36 commits | **1** branch | **0** releases | **1** contributor

Branch: **master** | New pull request | Create new file | Upload files | Find file | Clone or download

oliviaguest more tests — Latest commit 6e31788 11 days ago

tests	more tests	11 days ago
.gitignore	Create .gitignore	9 months ago
.travis.yml	test	19 days ago
Makefile	test	19 days ago
README.md	Update README.md	20 days ago
__init__.py	So much tidying	9 months ago
example.py	tests!	20 days ago
requirements.txt	tidy	a month ago
wkmeans.py	more tests	11 days ago

README.md

Weighted *k*-means

build passing

To run your own weighted *k*-means use `example.py` which has step-by-step instructions. For more detailed information regarding the implementation, please refer to `wkmeans.py` 's source code which is fully commented.

Our Metric

$$\text{Improvement Ratio} = \frac{\text{pairwise distances clusters}}{\text{pairwise distances districts}}$$

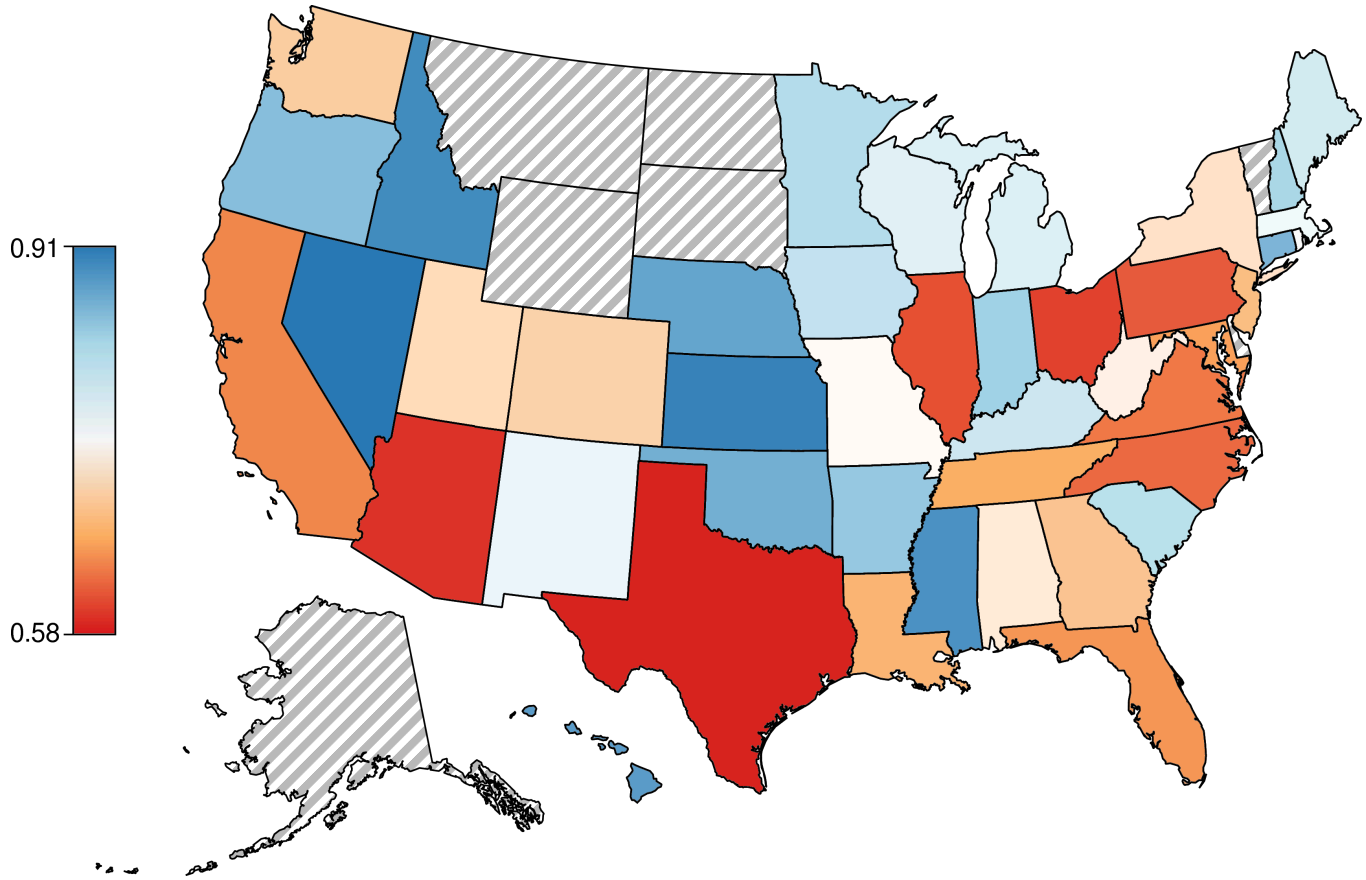
Our Metric

$$\text{Improvement Ratio}_{SC} = \frac{50\ \text{km}}{60\ \text{km}} = 0.833$$

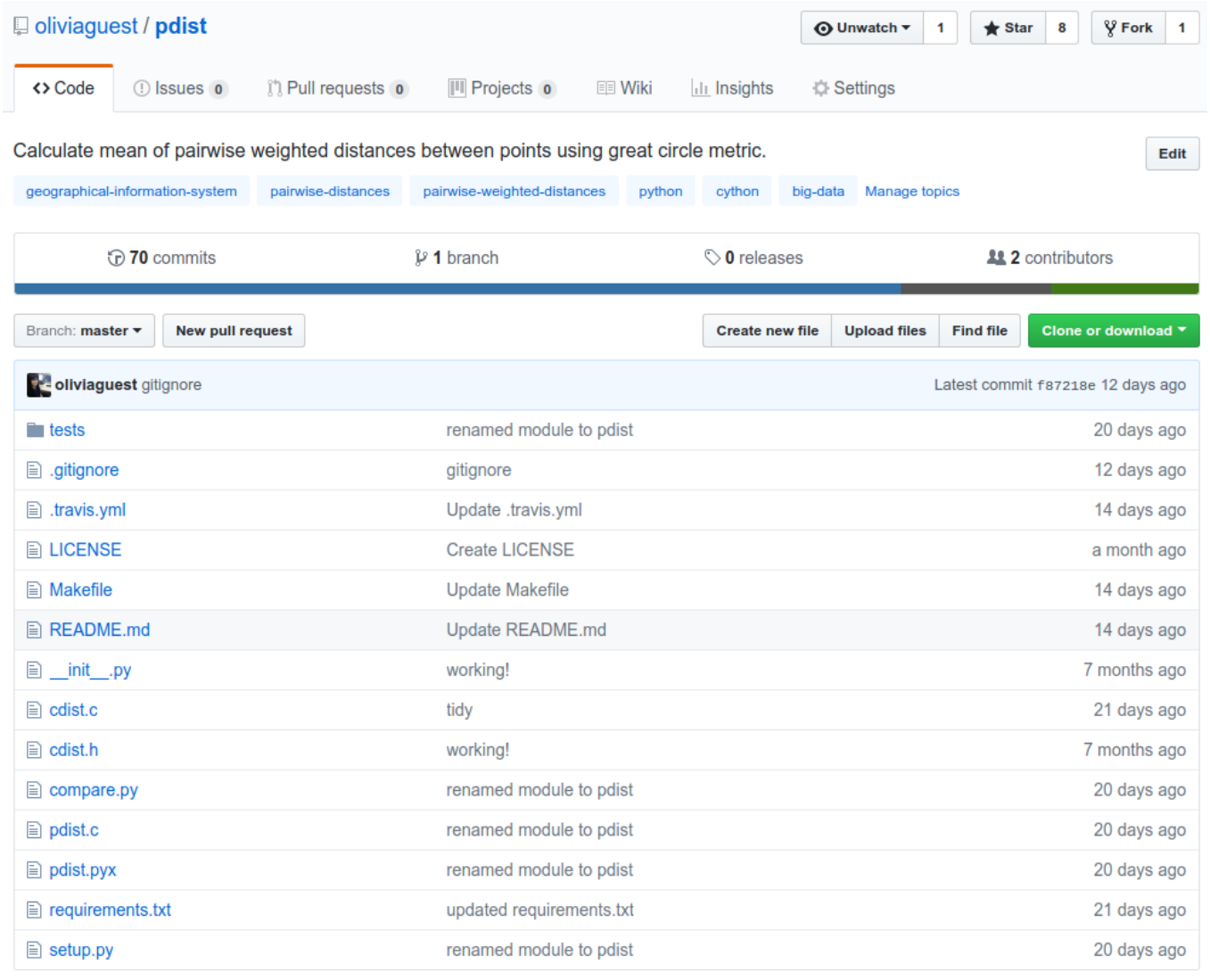
Our Metric

$$\text{Improvement Ratio}_{TX} = \frac{60\text{ km}}{100\text{ km}} = 0.6$$

Our Metric



Pairwise Distances Metric



oliviaguest / pdist

Unwatch 1 | Star 8 | Fork 1

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

Calculate mean of pairwise weighted distances between points using great circle metric. Edit

geographical-information-system | pairwise-distances | pairwise-weighted-distances | python | cython | big-data | Manage topics

70 commits | 1 branch | 0 releases | 2 contributors

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

oliviaguest gitignore — Latest commit f87218e 12 days ago

tests	renamed module to pdist	20 days ago
.gitignore	gitignore	12 days ago
.travis.yml	Update .travis.yml	14 days ago
LICENSE	Create LICENSE	a month ago
Makefile	Update Makefile	14 days ago
README.md	Update README.md	14 days ago
__init__.py	working!	7 months ago
cdist.c	tidy	21 days ago
cdist.h	working!	7 months ago
compare.py	renamed module to pdist	20 days ago
pdist.c	renamed module to pdist	20 days ago
pdist.pyx	renamed module to pdist	20 days ago
requirements.txt	updated requirements.txt	21 days ago
setup.py	renamed module to pdist	20 days ago

Pairwise Distances Metric

README.md

pdist

build passing codecov 100% license CC0 Follow @o_guest 3k

Calculate the mean of the pairwise weighted distances between points using the great circle metric 🌐 for a very big dataset without running out of RAM 💣 and/or waiting till the end of the universe. 😂

Usage

Mean of Pairwise Weighted Distances Using Great Circle

This code takes a set of 2D data points `X` and calculates the mean of the pairwise weighted distances between points using the great circle metric. It offers extensive speedup over Python-only implementations, so it is useful when dealing with *very big data*.

To call use:

```
mean_distances = c_mean_dist(X, weights)
```

where `X` are your data points, and `weights` are the weights or counts (depending on how you want to conceptualise them). Weights affect the mean of the pairwise distances in the same as including more of the point which the weight corresponds to. So if a data point with value (0, 1) has a weight of 2, the average pairwise distances will be affected in the same way as if you had added another data point with value (0, 1) to `X` and had set both their weights to 1.

Great Circle Distance

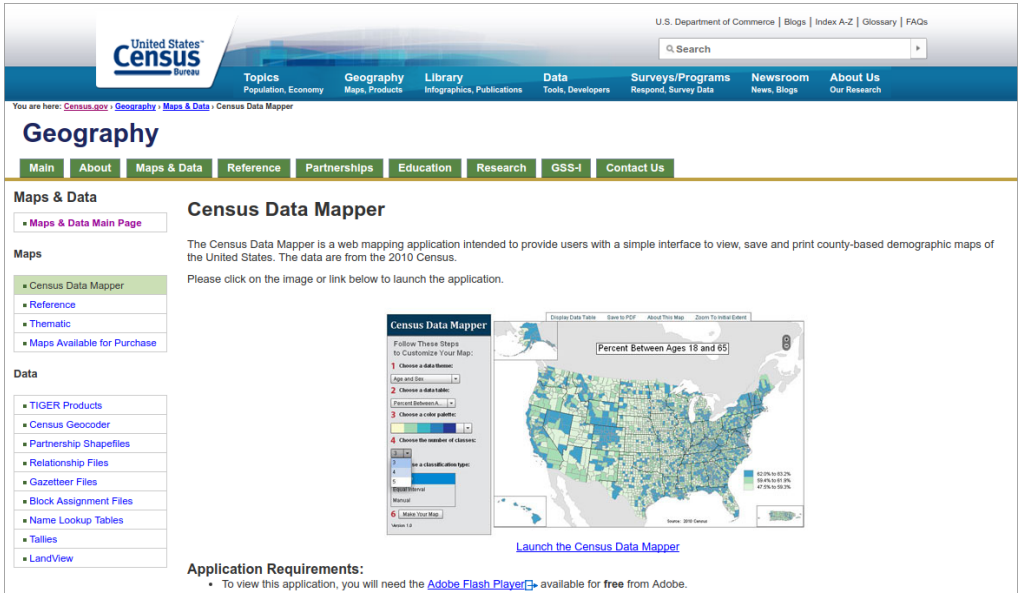
It also implements great circle, also known as orthodromic or geodesic, distance metric faster than GeoPy in `cdist`.

Example

For an example of both functions see `compare.py`.

Software & Data Challenges

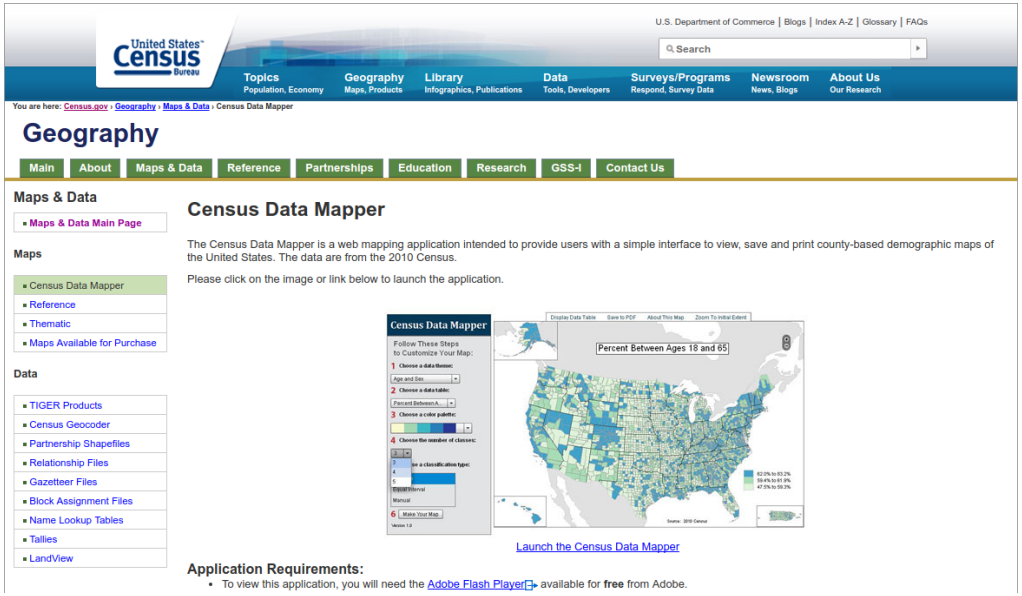
US Census Bureau
GIS data, API



Software & Data Challenges

US Census Bureau
GIS data, API

equations into
code

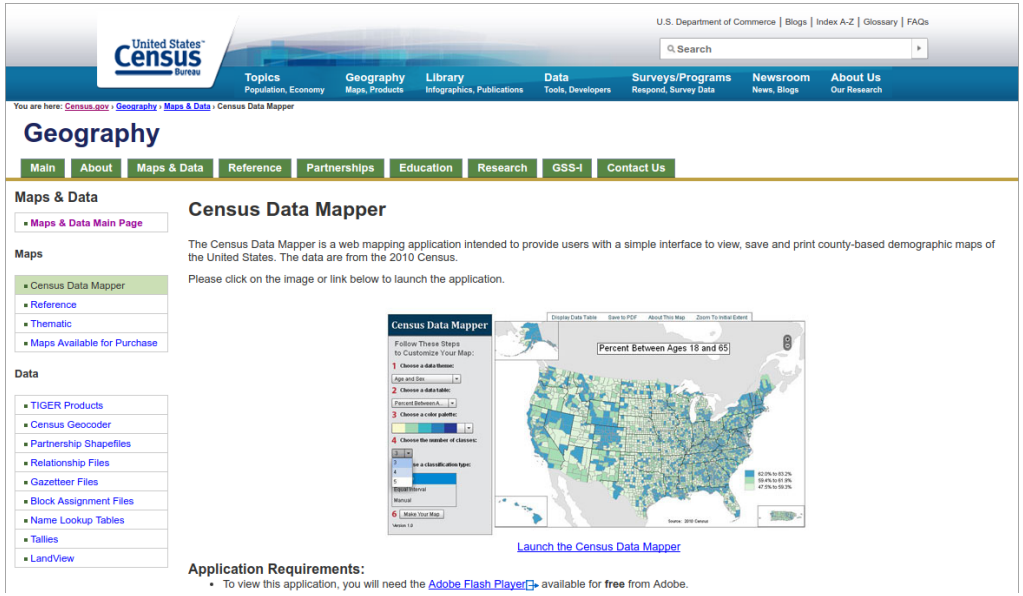


Software & Data Challenges

US Census Bureau
GIS data, API

equations into code

11 million census blocks (geometry and population)

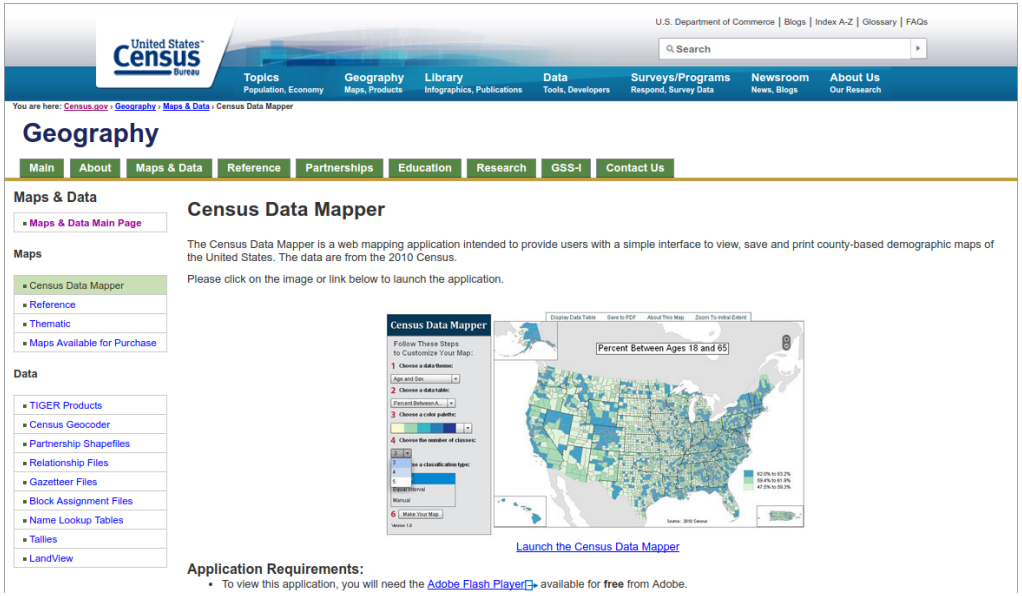


Software & Data Challenges

US Census Bureau
GIS data, API

equations into code

11 million census blocks (geometry and population)
OMG!!!



Software & Data Challenges

interactive map

collect data on our solutions vs reality

share 10s of gigs of data

Gerrymandering and Computational Redistricting

Olivia Guest, Frank J. Kanayet, and Bradley C. Love

Share Tweet

Welcome

Partisan gerrymandering poses a threat to democracy. One solution is using computers to draw the district lines instead of relying on politicians. To learn more about gerrymandering and our computer-based solution, please read "*Gerrymandering and Computational Redistricting*" by Olivia Guest, Frank J. Kanayet, and Bradley C. Love. This interactive website accompanies that paper and shows how computers can improve the districting process.

The data and code used for these simulations can be found on the OSF repository osf.io/5fepu and the redistrict github repository.

Maps

Below are two interactive maps showing the existing districts (left column) and the results from our weighted *k*-means algorithm (right column) . The colors used are to facilitate viewing and are randomly assigned to each congressional district — except in the case of states with at-large districts(i.e., a single district), which have no color.

Moving your cursor over a state highlights its border in black. Clicking on a state zooms both maps in, allowing you to see the state's districts close up for both reality and our solution. To find a specific location click on the magnifying glass, on the top left of both maps, to open up a search box.

Existing Solution

Clustering Solution

Interactive Map

Asked on Twitter

Interactive Map

Asked on Twitter

Leaflet & convert data from GeoJSON to TopoJSON

Leaflet

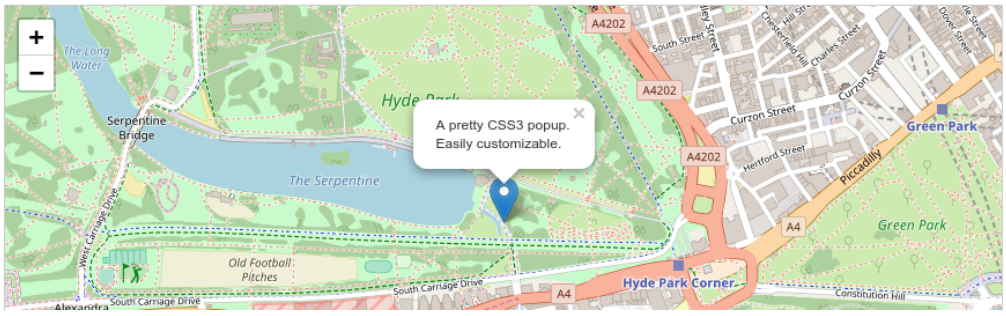
an open-source JavaScript library
for mobile-friendly interactive maps

Overview Tutorials Docs Download Plugins Blog

Aug 8, 2017 — Leaflet 1.2.0 has been released.

Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps. Weighing just about 38 KB of JS, it has all the mapping features most developers ever need.

Leaflet is designed with *simplicity*, *performance* and *usability* in mind. It works efficiently across all major desktop and mobile platforms, can be extended with lots of plugins, has a beautiful, easy to use and well-documented API and a simple, readable source code that is a joy to contribute to.

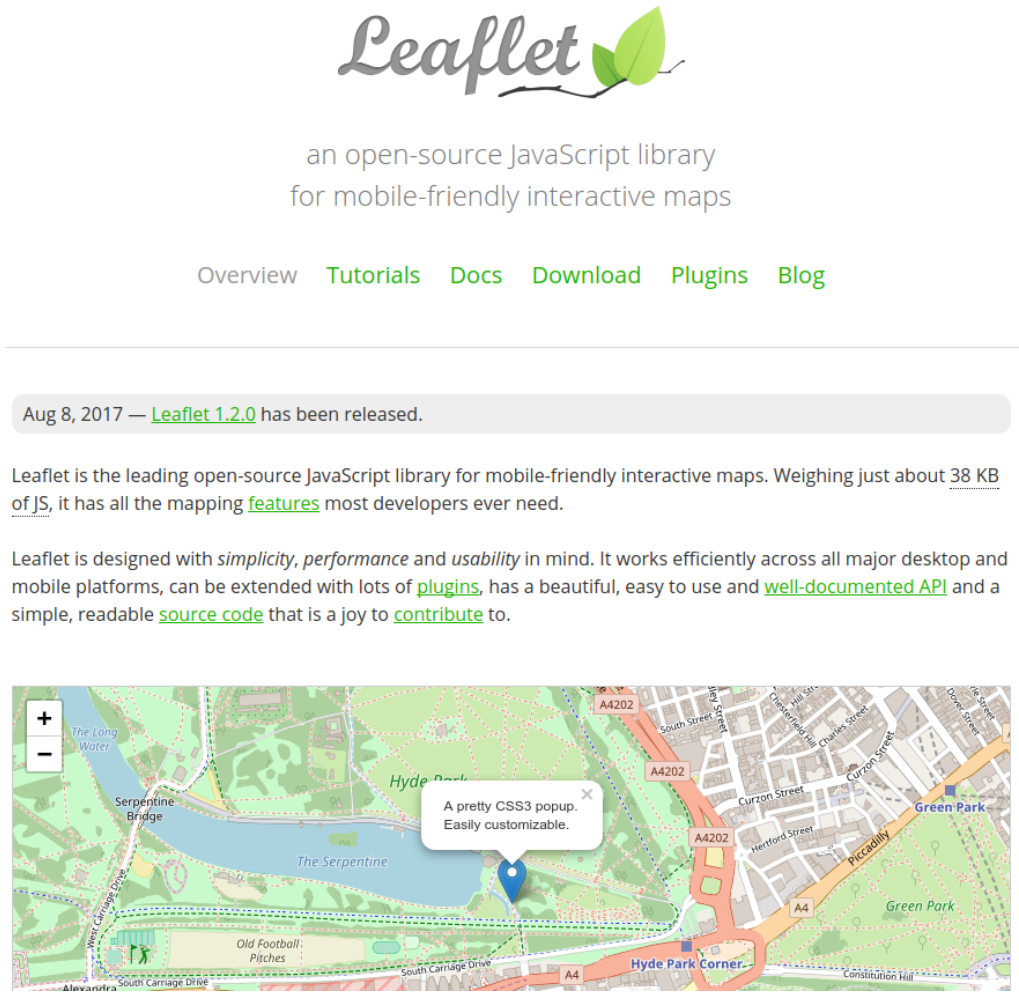


Interactive Map

Asked on Twitter

Leaflet & convert data from GeoJSON to TopoJSON

redistrict.science



General Lessons

Twitter is amazing

Knowing who to ask

Online open source/data communities

The End! Thank you!