Visualizing Billions of Data Points: Doing It Right

Alexey Zaytsev,

Skoltech, CDISE

18 January

Slides are based on http://go.continuum.io/visualizing-billions-data-points/

Overview

1. Visualizing big data — what is the problem?
2. Datashading
3. Datasets:
    - 10 million points of NYC Taxi data
    - 3 billion points of OpenStreetMap data
    - 300 million points of US Census data

Billions and billions…

HAIR BALL PROBLEM

Visualization of networks often results in hairballs.
They are beautiful and powerful.
But are they useful?

What we need is insight.
Not a picture.

https://networkscience.wordpress.com/2016/06/22/no-hairball-the-graph-drawing-experiment/

Big data magnifies small problems

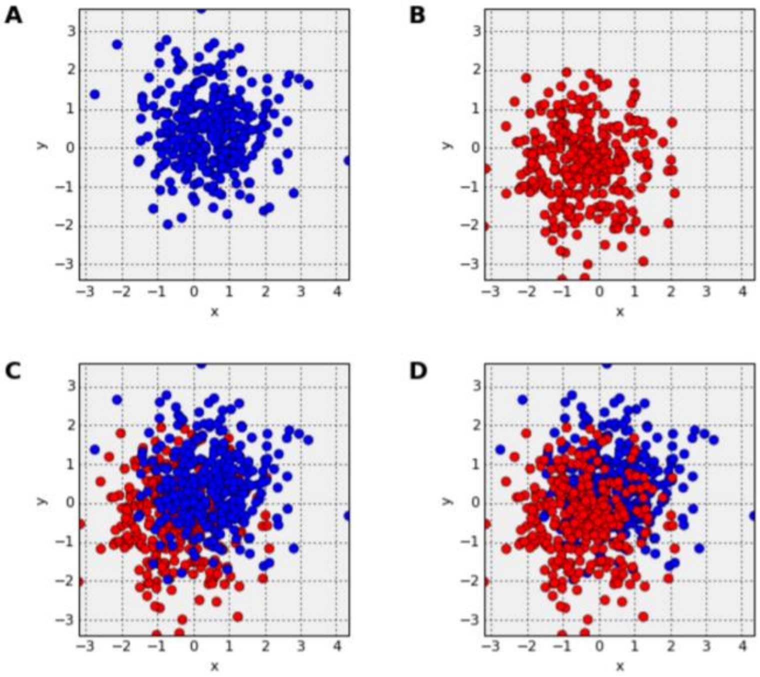
Big data presents storage and computation problems

More importantly, standard plotting tools have problems that are magnified by big data:

- Overdrawing/Overplotting
- Saturation
- Undersaturation
- Binning issues

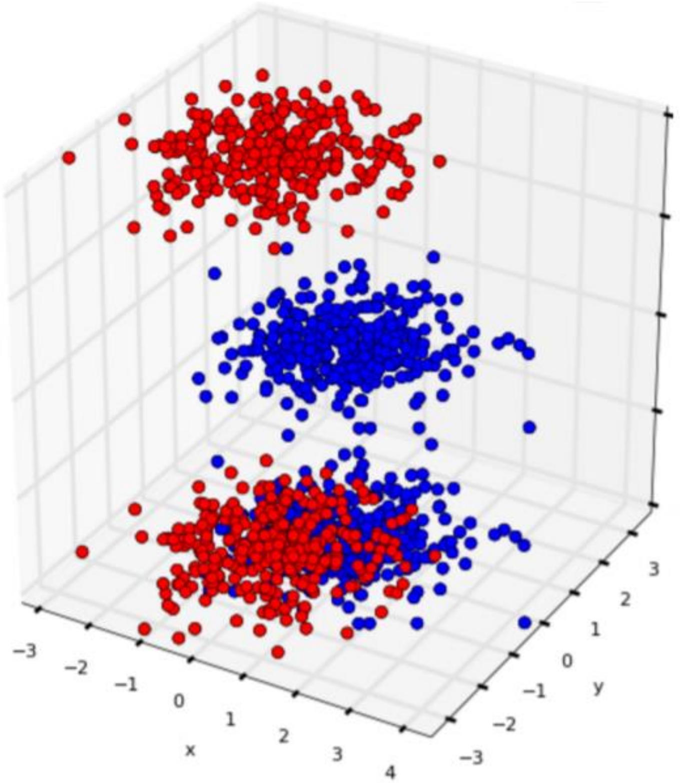
We’ll first explain these problems, and then present *datashading* technique to address them.

Overdrawning



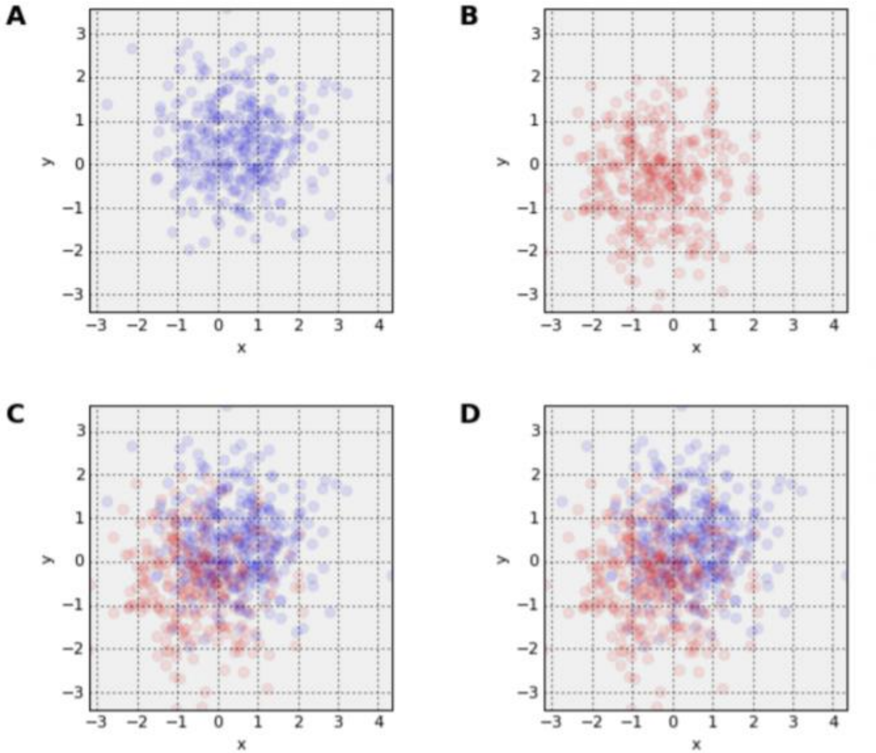
- For a scatterplot, the order in which points are drawn is very important
- The same distribution can look entirely different depending on plotting order
- Last data plotted overplots

Overdrawning



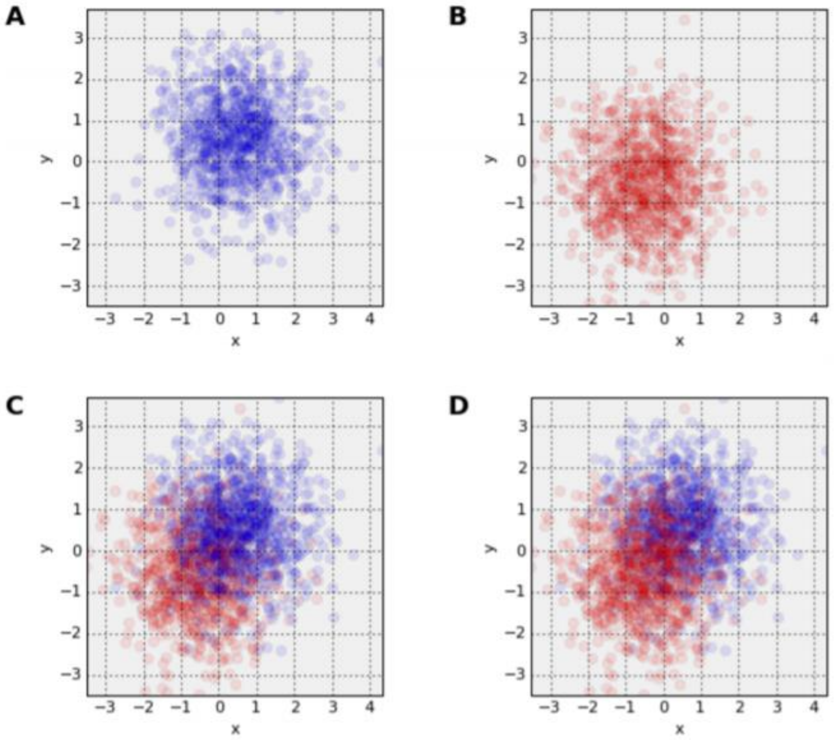
- Underlying issue is just occlusion
- Same problem happens with one category, but less obvious
- a Can prevent occlusion using transparency

Saturation



- For alpha = 0.1, up to 10 points can overlap before saturating the available brightness
- Now the order of plotting matters less
- After 10 points, first-plotted data still lost
- For one category, 10, 20, or 2000 points overlapping will look identical

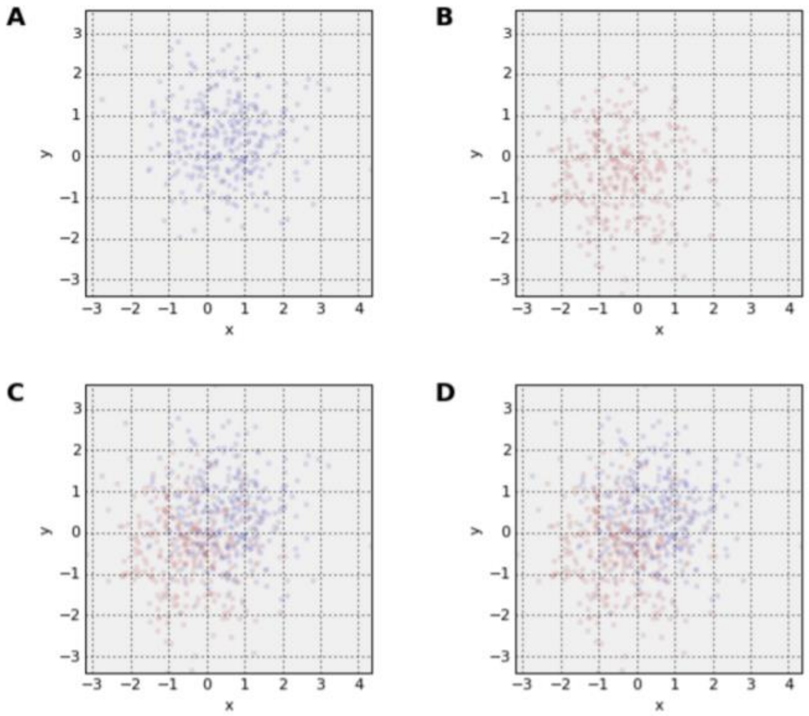
Saturation



Same alpha value, more points:

- Now is highly misleading
- alpha value depends on size, overlap of dataset
- Difficult-to-set parameter, hard to know when data is misrepresented

Saturation

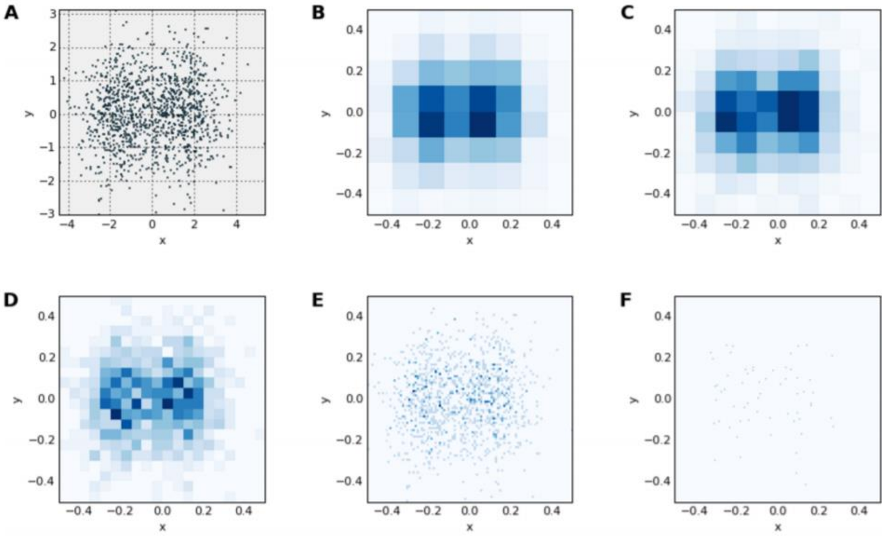


Can try to reduce point size to reduce overplotting and saturation

- Now points are hard to see, with no guarantee of avoiding problems
- Another difficult-to-set parameter
- For really big data, scatterplots start to become very inefficient: many points per pixel — may as well be binning by pixel

Binning

- Can use heatmap instead of scatter
- Avoids saturation by auto-ranging on bins
- Result independent of data size
- Here two merged normal distributions look very different at different binning
- Another difficult-to-set parameter



Main challenges while plotting big data

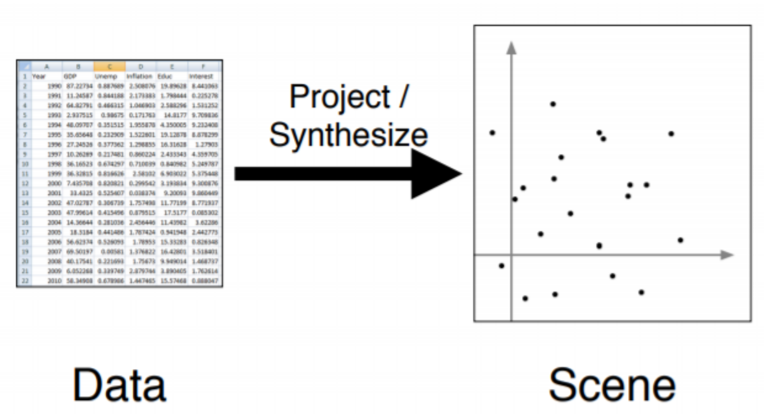
- When exploring really big data, the visualization is all you have — there's no way to look at each of the individual data points
- Common plotting problems can lead to completely incorrect conclusions based on misleading visualizations
- Slow processing makes trial and error approach ineffective

When data is large, you don't know when the plot is lying.

Datashading

- Flexible, configurable pipeline for automatic plotting
- Prevents overplotting, saturation, and undersaturation •
- Mitigates binning issues by interactivity, even of very large datasets on ordinary machines
- Allows rapid iteration of visual styles & configs, interactive selections and filtering, to support data exploration

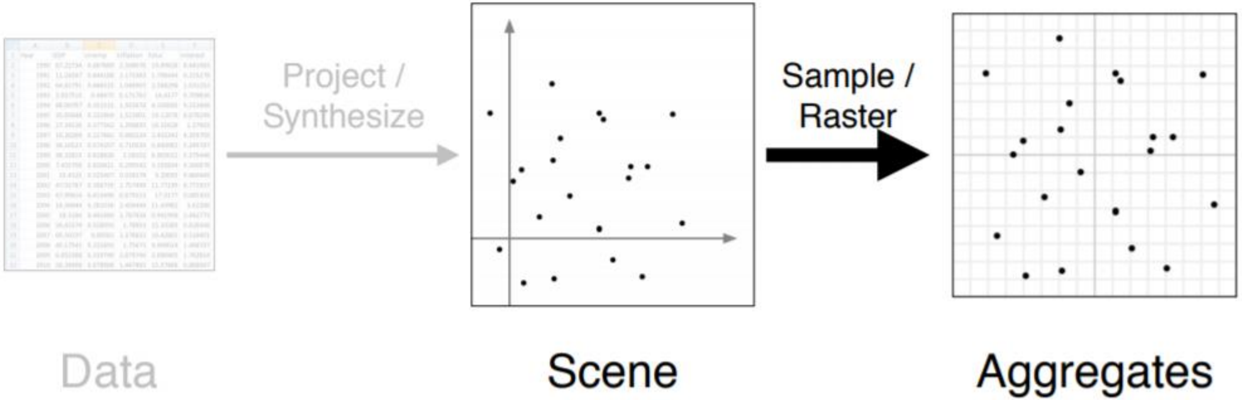
Datashading pipeline starts with projection



	A	B	C	D	E	F
1	Year	GDP	Unemp	Inflation	Educ	Interest
2	1990	87.22734	0.887689	2.508076	19.89628	8.443063
3	1991	11.24587	0.844188	2.173383	1.798444	0.225276
4	1992	64.82791	0.466315	1.046903	2.588296	1.531252
5	1993	2.937515	0.98675	0.171763	14.8177	9.709836
6	1994	48.09707	0.351515	1.955878	4.350005	9.232408
7	1995	35.65648	0.232909	1.522801	19.12878	8.878299
8	1996	27.24526	0.377562	1.298855	16.31628	1.27903
9	1997	10.26289	0.217481	0.860224	2.433343	4.359705
10	1998	36.16523	0.674297	0.710039	0.840982	5.249787
11	1999	36.32815	0.816626	2.58102	6.903022	5.375448
12	2000	7.435708	0.820821	0.299542	3.193834	9.300876
13	2001	33.4325	0.525407	0.038374	9.20093	9.860449
14	2002	47.02787	0.306739	1.757498	11.77199	8.771937
15	2003	47.99614	0.415496	0.879515	17.5177	0.085302
16	2004	14.36644	0.281036	2.456446	11.43982	3.62286
17	2005	18.3184	0.441486	1.787424	0.941948	2.442773
18	2006	56.62374	0.526093	1.78953	15.33283	0.826348
19	2007	69.50197	0.00581	1.376822	16.42801	3.518401
20	2008	40.17541	0.221693	1.75673	9.949014	1.468737
21	2009	6.052268	0.339749	2.879744	3.890465	1.762614
22	2010	58.34908	0.678986	1.447465	15.57468	0.888047

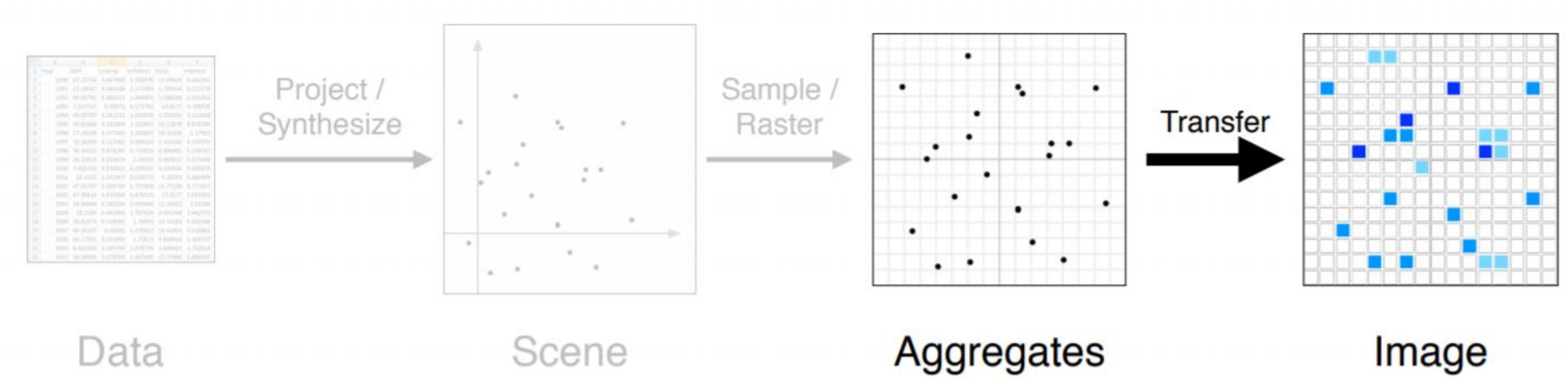
- Stage 1: select variables (columns) to project onto the screen
- Data often filtered at this stage

Datashading pipeline continues with aggregation



- Stage 2: Aggregate data into a fixed set of bins
- Each bin yields one or more scalars (total count, mean, stddev, etc.)

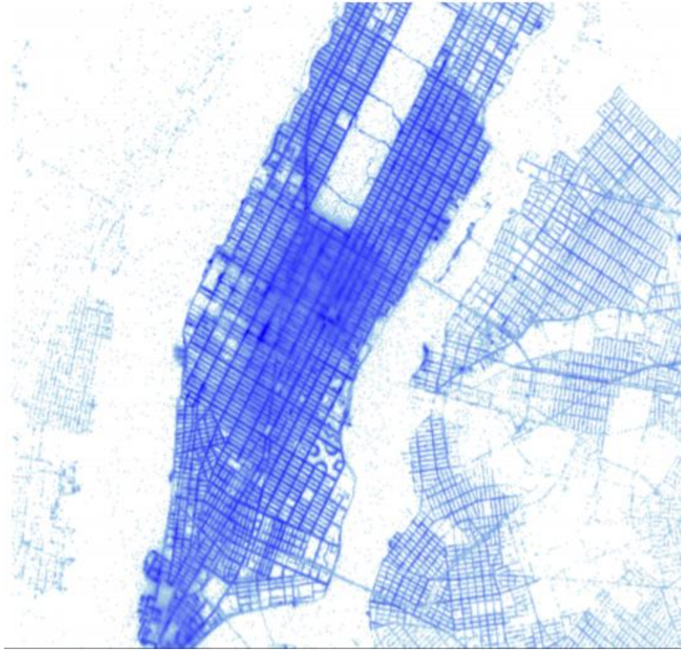
Datashading pipeline ends with transfer



- Stage 3: Transform data using one or more transfer functions, culminating in a function that yields a visible image
- Each stage can be replaced and configured separately

NYC taxi data

- Data for 10 million New York City taxi trips
- Even 100,000 points gets slow for scatterplot
- Parameters usually need adjusting for every zoom
- True relationships within data not visible in std plot

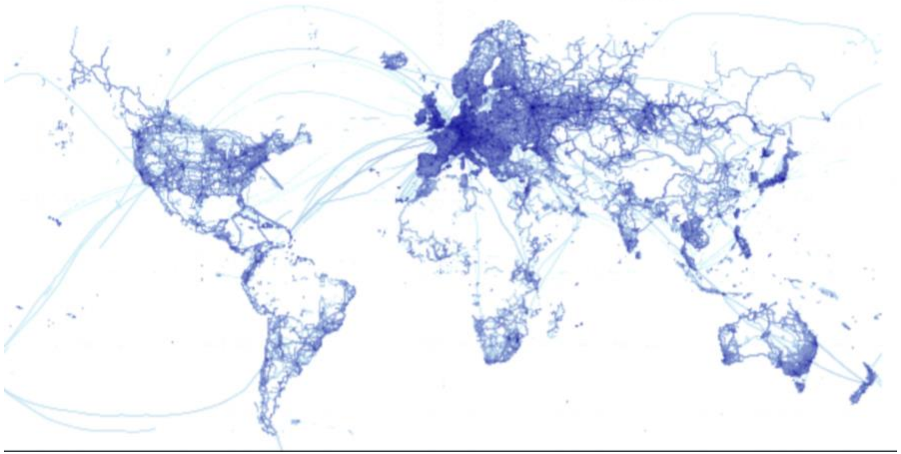


Datashading automatically reveals the entire dataset, including outliers, hot spots, and missing data

Billion OSM points

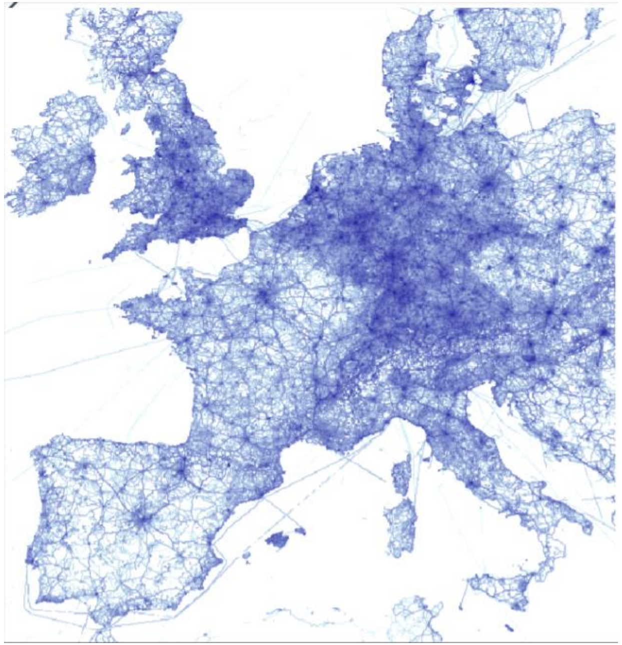
Open Street Map data:

- About 3 billion GPS coordinates
- https://blog.openstreetmap.org/2012/04/01/bulk-gps-point-data/
- This image was rendered in one minute on a standard MacBook with 16 GB RAM
- Renders in 7 seconds on a 128GB Amazon EC2 instance



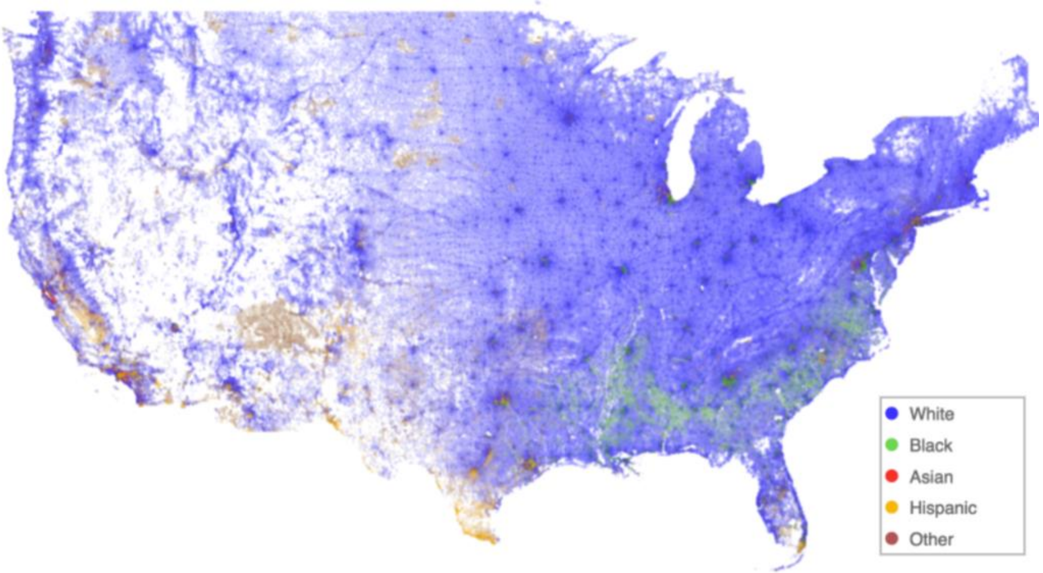
Billion OSM points

- OSM database is 40GB, larger than the 16GB RAM on this machine
- Fast out-of-core operation powered by: •
  - Numba: generates fast C and GPU code from Python source
  - Dask: Parallelizes tasks
- datashader source code is all Python



300 million points Census data

- One point per person
- 300 million total
- Categorized by race
- Datashading shows faithful distribution per pixel



Example of data exploration with datashader

http://datashader.org/topics/nyc_taxi.html