# Visualization (for Data Science)

Prof. Joydeep Ghosh
ECE/UT
www.ideal.ece.utexas.edu/~ghosh
ghosh@ece.utexas.edu

# Hans Gosling, Gapminder and Bubble Charts

- Hans Rosling's 200 Countries, 200 Years, 4 Minutes - The Joy of Stats - BBC Four

Download Tool from https://www.gapminder.org/tools-offline/

# Interactive Charts in Browser

- TOOLS:  GapMinder Provides Bubble Charts and a few others
  - Also a nice tutorial and lot of datasets to try out

  - Guide for  bring your data using Gapminder offline app

- Google Charts includes Bubble Charts
  - 3 numeric values (axes, size), 1 categorical/numeric (color) + animation
  - Gallery of Other Charts

- Can also create an moving bubble chart in Excel, using Visual Basic for Applications (**VBA**)

# Modern Web-Based Visualization

- Interactive, often Javascript based
- http://d3js.org/
- **D3.js** is a JavaScript library for manipulating documents based on data. **D3** helps you bring data to life using HTML, SVG and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.
- Gallery at: https://github.com/mbostock/d3/wiki/Gallery
- There is a great free online book to learn D3

- Webinar and ebook: http://it-ebooks.info/book/1265/
  - **D3** allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. For example, you can use D3 to generate an HTML table from an array of numbers. Or, use the same data to create an interactive *Scalable Vector Graphics* (*SVG*) bar chart with smooth transitions and interaction.
- http://nvd3.org/
  - Simpler than D3.js

# D3js Examples:

- https://bost.ocks.org/mike/sankey/
  - Sankey diagrams visualize the magnitude of flow between nodes in a network.
- Hierarchy: http://vizuly.io/product/weighted-tree/?demo=d3js
- Localized Zoom-in https://bost.ocks.org/mike/fisheye/
- Interactive Axis selection: http://charts.animateddata.co.uk/whatmakesushappy/

# R/Python interactive visualizations

- Dynamic (Python):
- BOKEH:  http://bokeh.pydata.org/en/latest/
  - Gallery shows source code
- Plotly This is a really cool library that makes it easy to quickly create D3 style charts in Python (also works with R). One issue is all charts are publicly hosted on their servers (unless you pay), so not free if you have confidential data.

- Dynamic (R): Shiny from Rstudio
- http://shiny.rstudio.com/
  - Again gallery shows code: (server.R, ui.R)
    - http://shiny.rstudio.com/gallery/retirement-simulation.html

- Also see
  - http://setosa.io/ (e.g Simpson's paradox, PCA visuals etc)
  - http://www.highcharts.com/
  - http://nvd3.org/
    - Simpler than D3.js

# Bokeh

- **Beautiful Python Visualizations: An Interview with Bryan Van de Ven, Bokeh Core Developer**
  - http://www.kdnuggets.com/2017/08/interview-bryan-van-de-ven-bokeh.html
- Resources pointed to:
- GitHub: https://github.com/Bokeh/bokeh
  Documentation: http://bokeh.pydata.org/en/latest
  Example Apps: https://demo.bokehplots.com
  Tutorials: http://nbviewer.jupyter.org/github/bokeh/bokeh-notebooks/blob/master/index.ipynb
  Mailing List: https://groups.google.com/a/continuum.io/forum/#!forum/bokeh
  Gitter Chat: https://gitter.im/bokeh/bokeh

# Bokeh Examples

# Visualizing Very Large Data Sets in Python

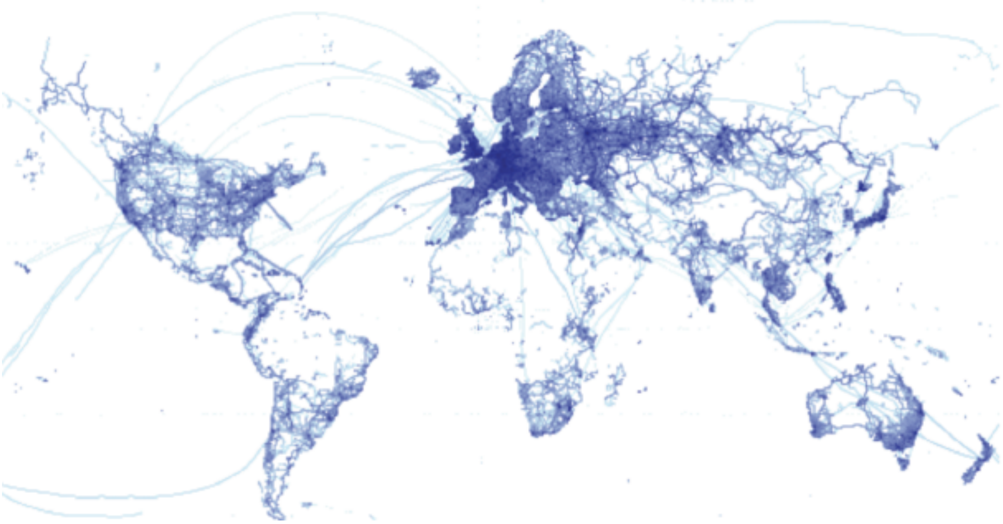
- **Datashading**: rendering truly big data
- **HoloViews:** reproducible scientific visualization
- **Dask/Numba**: scaling up

Bednar’s Talk https://www.youtube.com/watch?v=6m3CFbKmK_c
(16:20..)
Peter Wong’s Talk:
https://www.youtube.com/watch?v=fB3cUrwxMVY&t=797s

Docs and example notebooks at
Github.com/bokeh/datashader

# Visualizing Very Large Data Sets

Because Datashader decouples the data-processing from the visualization, it can handle arbitrarily large data



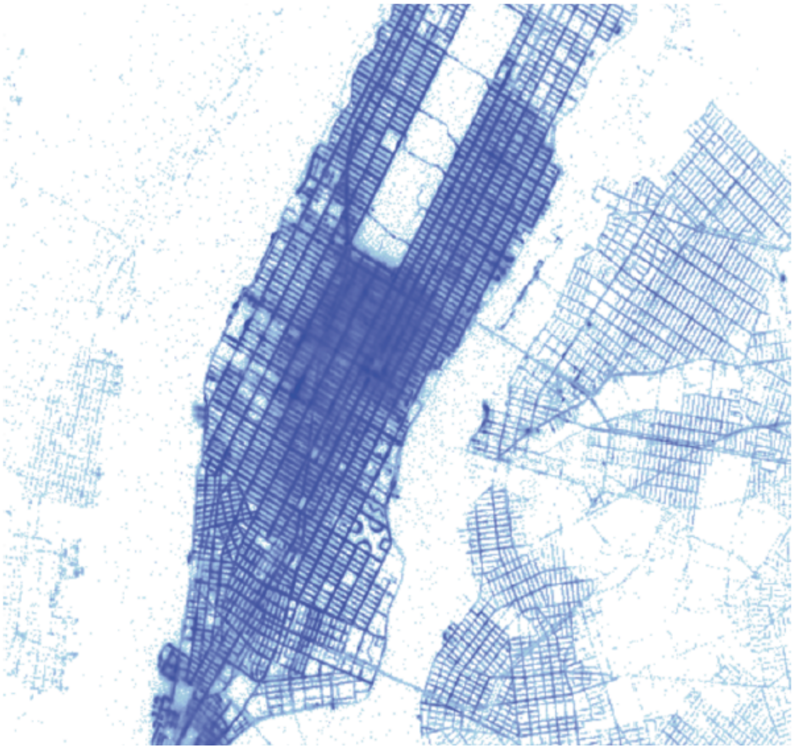
E.g. Open Street Map data:

- 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

*Courtesy: Jim Bednar, Continuum. Also see*
*http://www.slideshare.net/continuumio/visualizing-a-billion-points-w-bokeh-datashader*

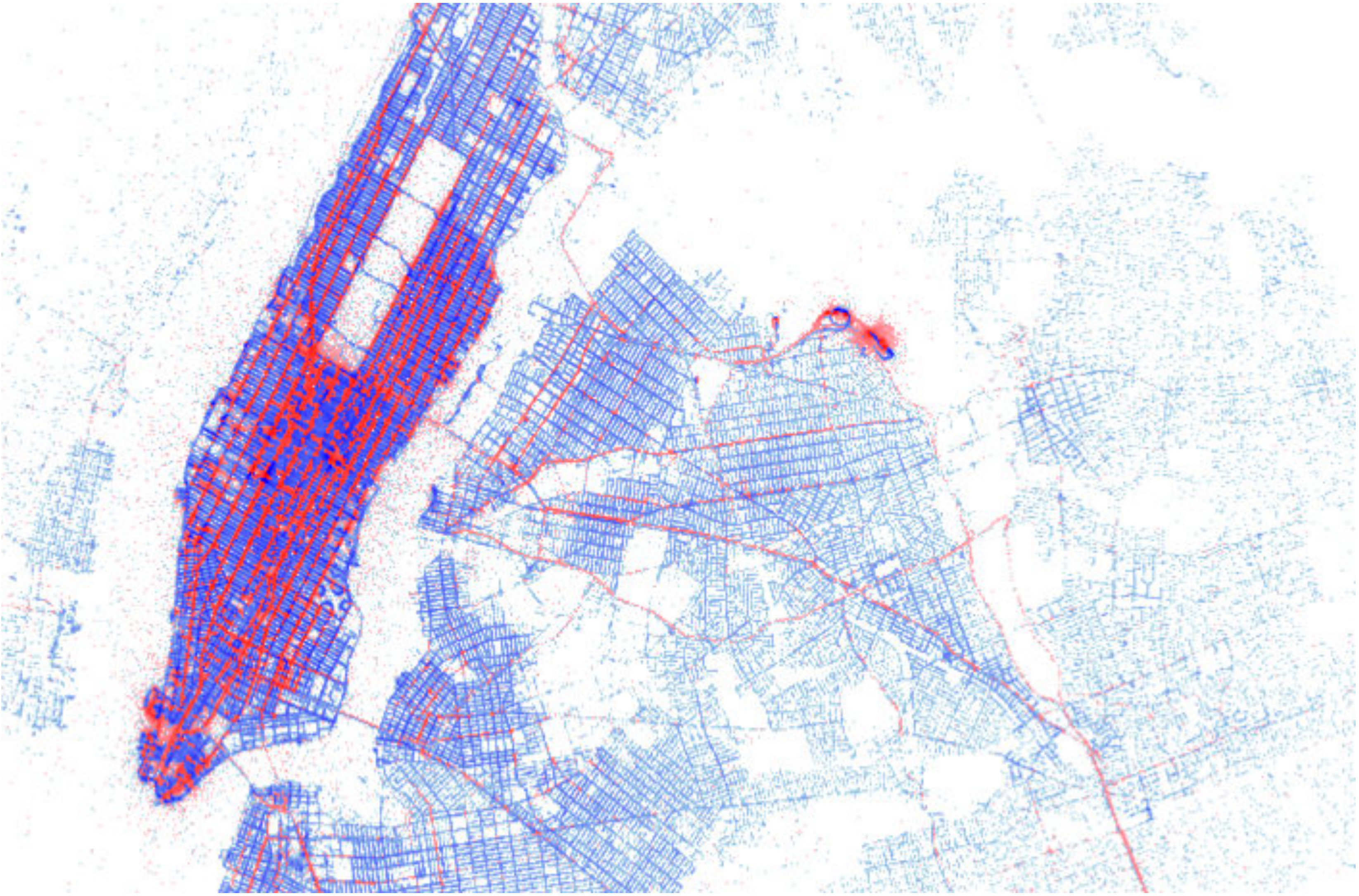
# NY Taxi cab data

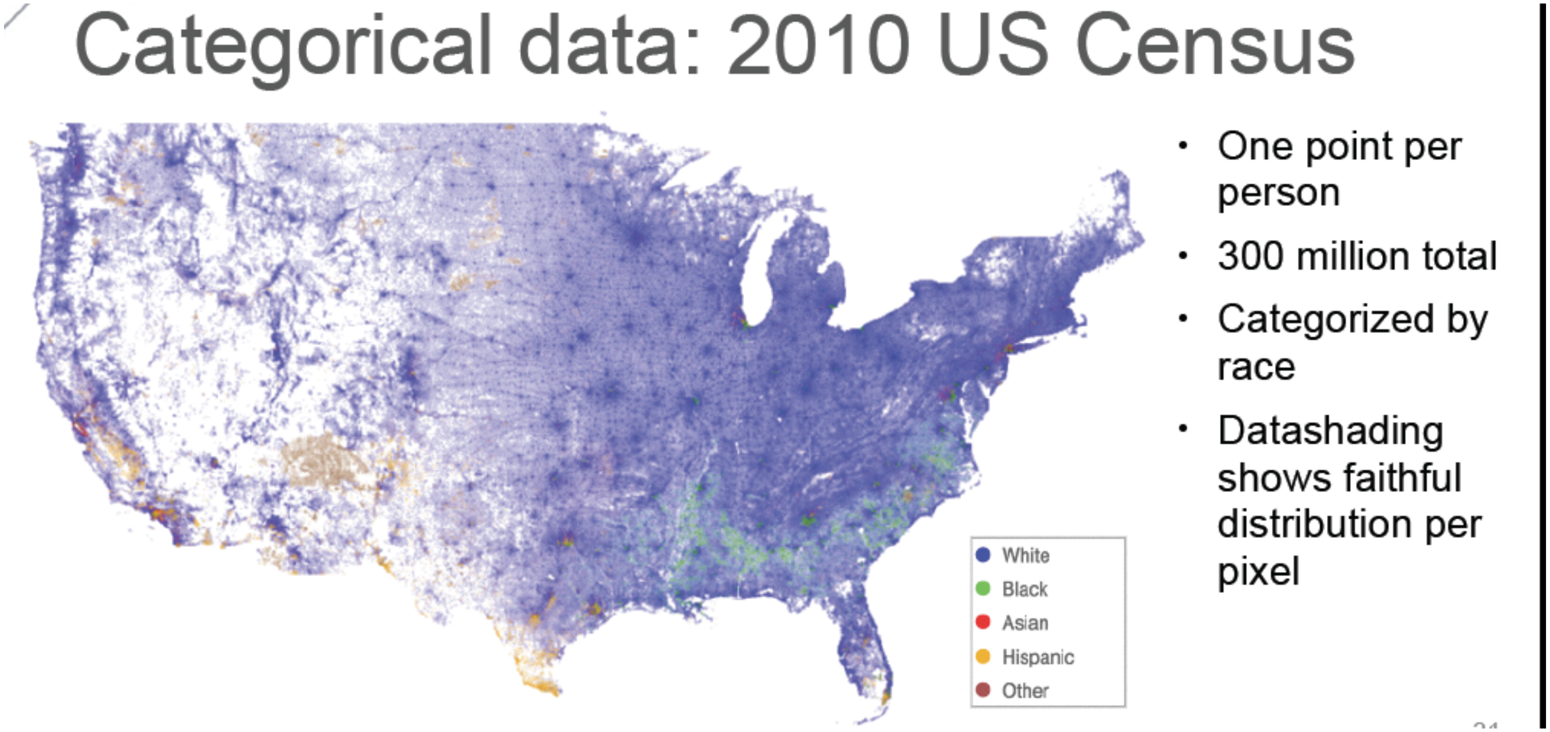
This demo shows how traditional plotting tools break down for large datasets, and how to use datashading to make even large datasets practical interactively.



- 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

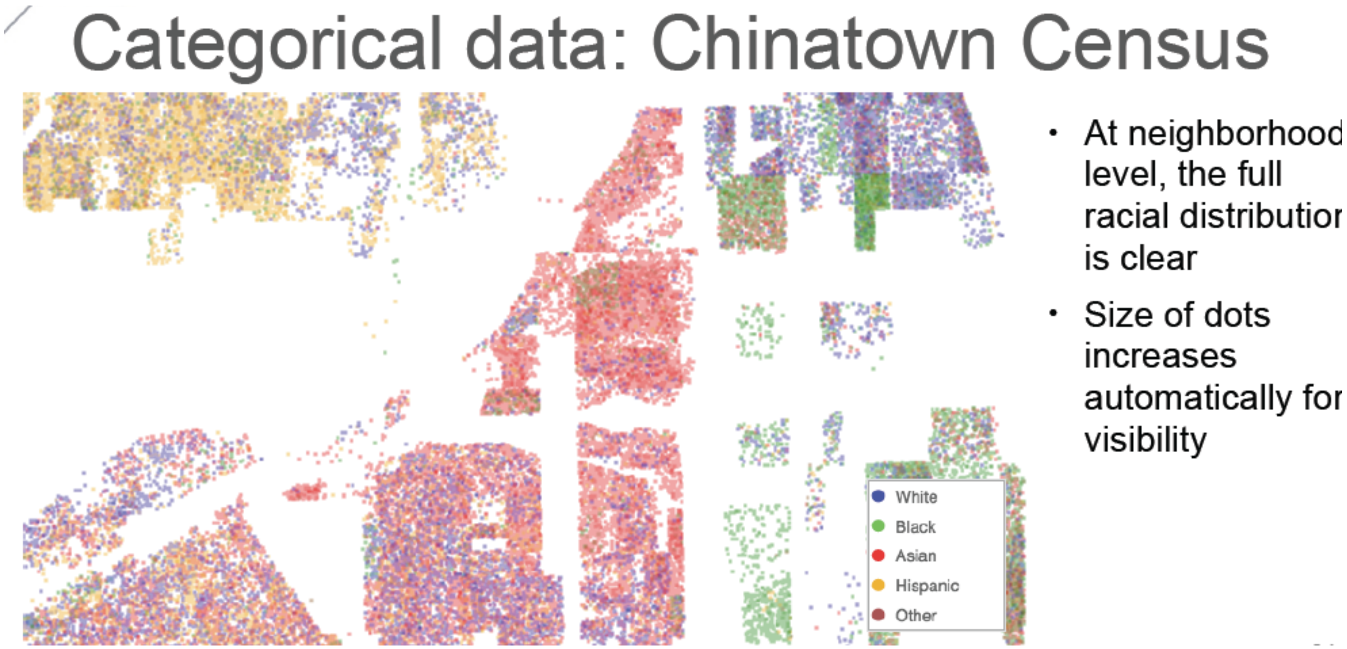
**Datashading automatically reveals the shape of the dataset, including outliers, hot spots, and missing data**





# Categorical data: 2010 US Census

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



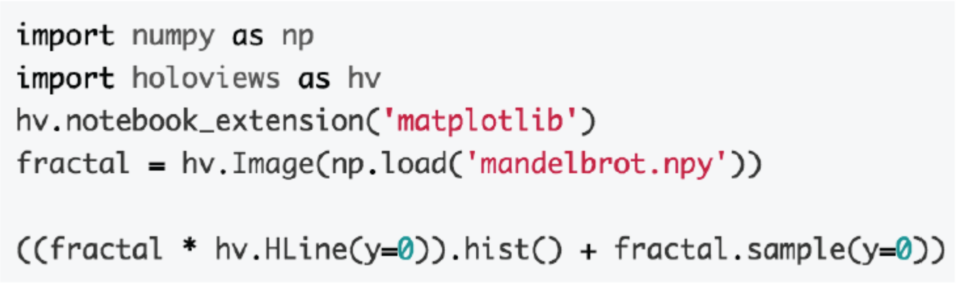
# Categorical data: Chinatown Census

- At neighborhooc level, the full racial distributior is clear
- Size of dots increases automatically for visibility

# HoloViews: Stop plotting your data

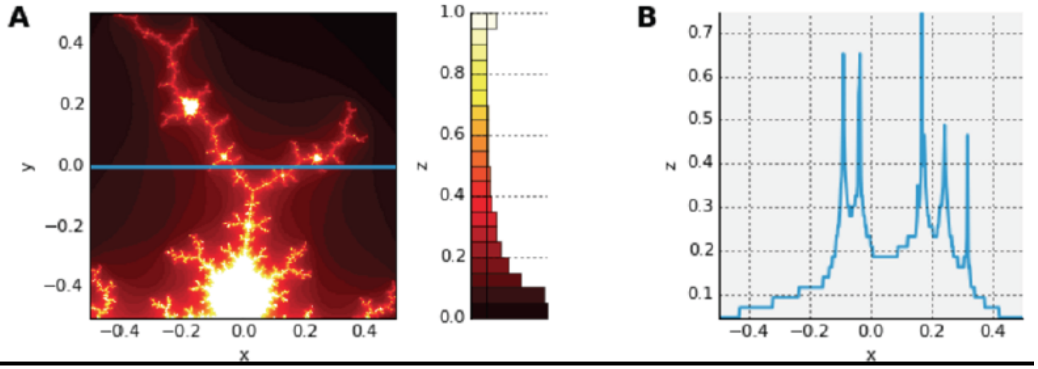
- Jupyter notebooks are great for capturing the steps involved in data analysis
- Yet visualizing anything fills up the notebook with reams of unreadable plotting code

  — detail after detail that obscures the flow of your analysis
- Instead of plotting your data, just annotate it
- Then just select, combine, and overlay the data you want, and HoloViews will plot it



```
import numpy as np
import holoviews as hv
hv.notebook_extension('matplotlib')
fractal = hv.Image(np.load('mandelbrot.npy'))

((fractal * hv.HLine(y=0)).hist() + fractal.sample(y=0))
```



# Summary

- Modern Visualization is
  - Interactive
  - Browser based
  - Provides a wide variety of ways to “see” your data
  - Fast response to feedback or change of settings (towards “real time”)
  - Can work on very large datasets