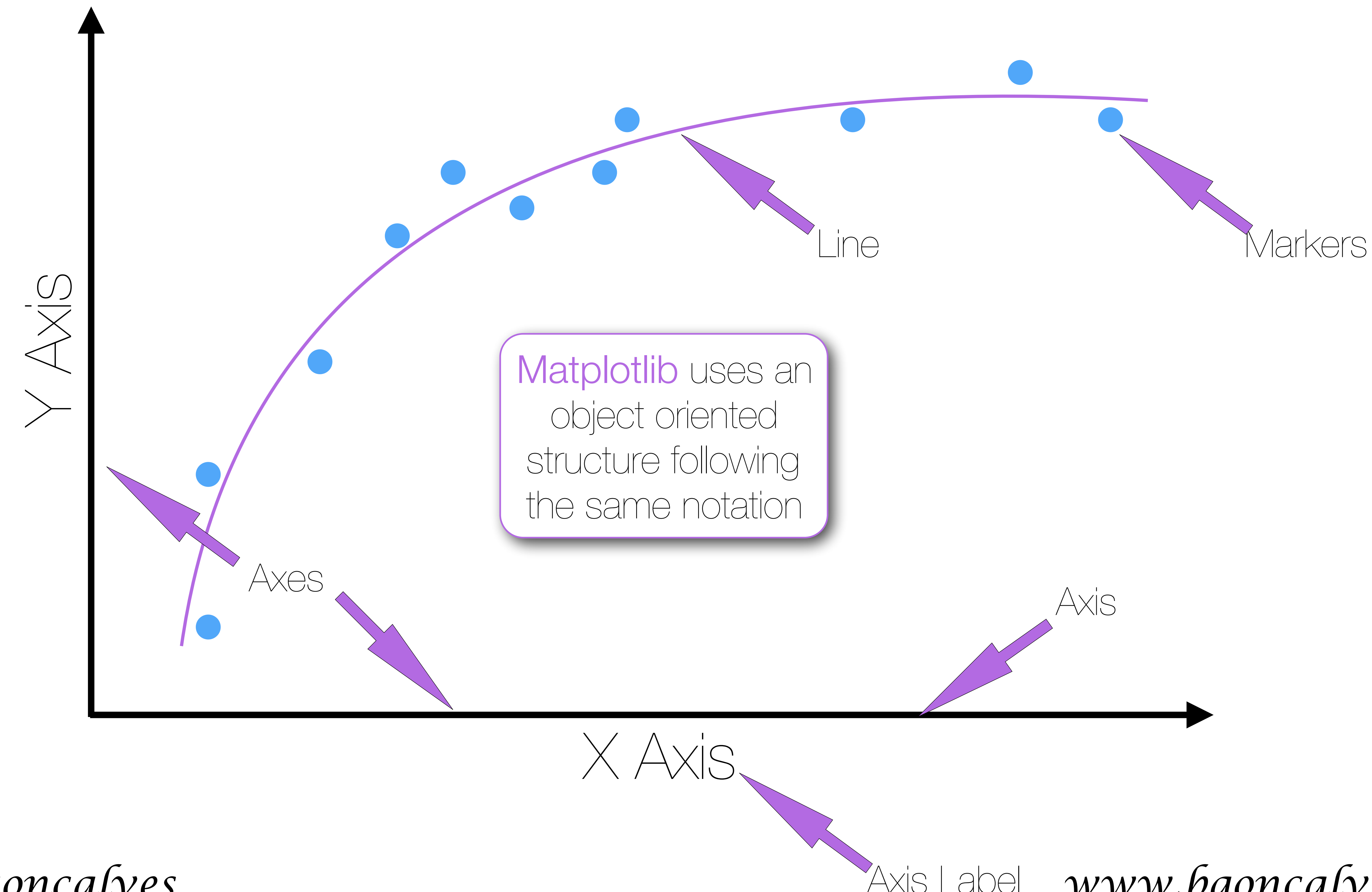
Lecture VI - Case study

Bruno Gonçalves
www.bgoncalves.com

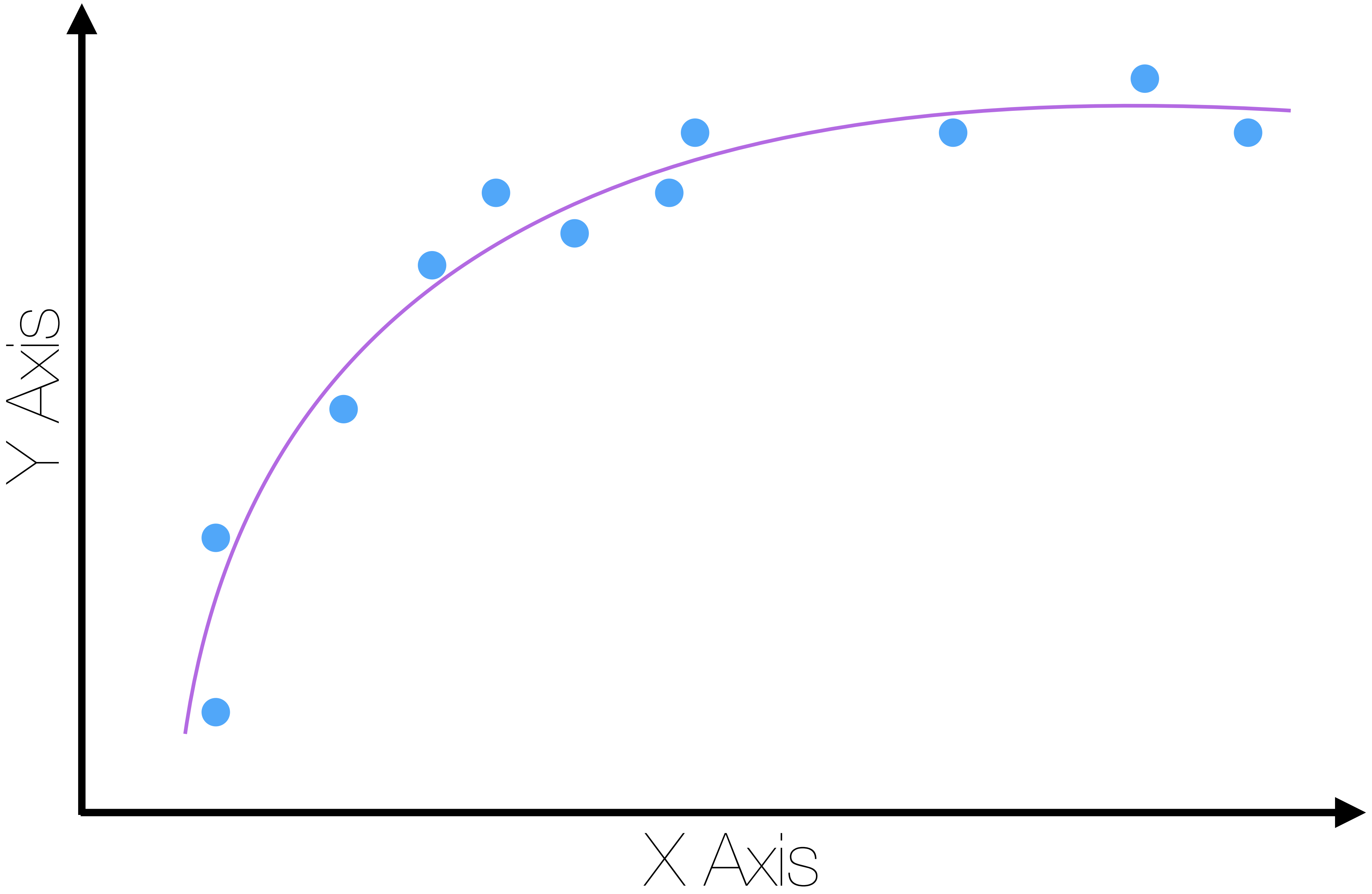
NYU DATA SCIENCE

Matplotlib

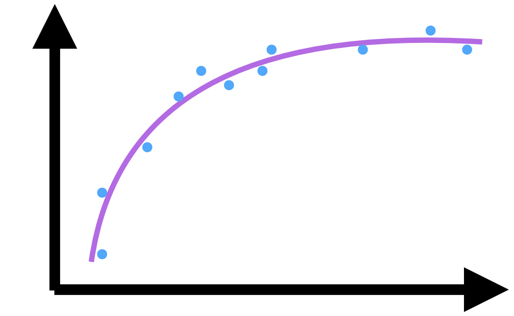
Basic Plotting



Basic Plotting

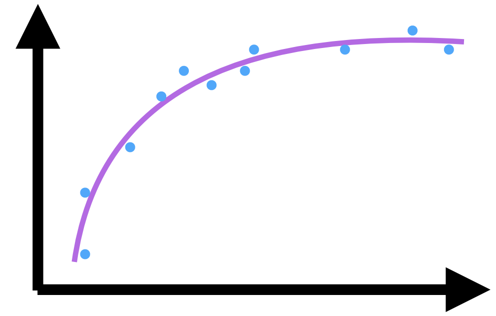


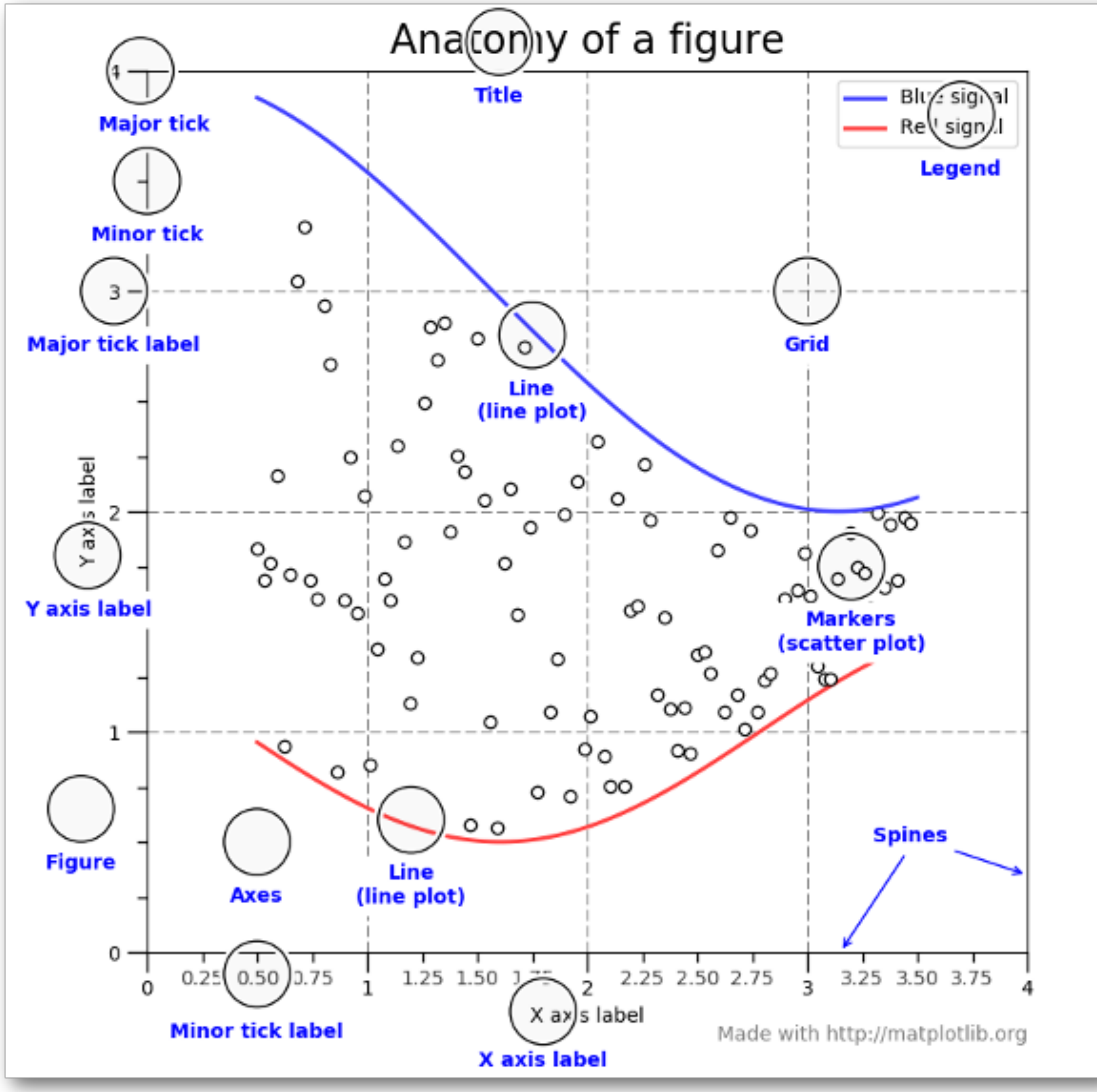
Basic Plotting



- Matplotlib uses an object oriented structure following an intuitive notation
- Each Axes object contains one or more Axis objects.
- A Figure is a set of one or more Axes.
- Each Axes is associated with exactly one Figure and each set of Markers is associated with exactly one Axes.
- In other words, Markers/Lines represent a dataset that is plotted against one or more Axis. An Axes object is (effectively) a subplot of a Figure.

Basic Plotting - Programmatically!

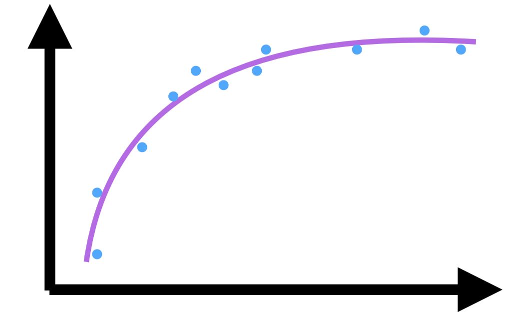




@bgoncalves

www.bgoncalves.com

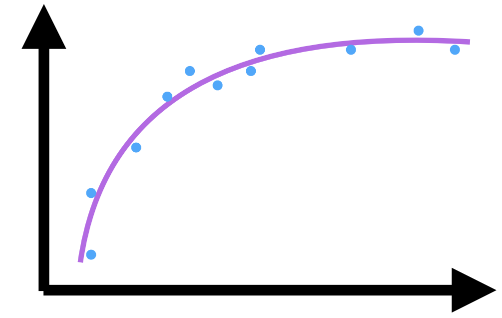
Basic Plotting - Programmatically!



https://matplotlib.org/2.0.0/

- While the Figure object (plt) controls the way in which the figure is displayed.
    - .gca() - Get the current Axes, creating one if necessary
    - .show() - Show the final figure
    - .savefig("filename.ext") - Save the figure to "filename.ext" where ".ext" defines the format the saved image ()

```
filetypes = {'ps': 'Postscript', 'eps': 'Encapsulated Postscript', 'pdf': 'Portable Document Format',
'pgf': 'PGF code for LaTeX', 'png': 'Portable Network Graphics', 'raw': 'Raw RGBA bitmap', 'rgba': 'Raw
RGBA bitmap', 'svg': 'Scalable Vector Graphics', 'svgz': 'Scalable Vector Graphics', 'jpg': 'Joint
Photographic Experts Group', 'jpeg': 'Joint Photographic Experts Group', 'tif': 'Tagged Image File
Format', 'tiff': 'Tagged Image File Format'}
```



Basic Plotting - Programmatically!

- The first step is to import the pyplot module from matplotlib and instanciating a Figure object:

```
import matplotlib.pyplot as plt
fig = plt.figure()
```

- The convention is to import pyplot as plt
- To create subplots (Axes) you use .subplots(nrows, ncols, sharex=False, sharey=False) instead of .figure(). set sharex and/or sharey to True to keep the same scale in both cases.
- .subplots - returns a (fig, ax_lst) tuple where ax_lst is a list of Axes and fig is the Figure.
- Axes have several methods of interest:
    - .plot(x, y) - Make a scatter or line plot from a list of x, y coordinates.
    - .imshow(mat) - Plot a matrix as if it were an image. Element 0,0 is plotted in the top right corner.
    - .bar(x, y) - Make a bar plot where x is a list of the lower left coordinates of each bar and y is the respective height.
    - .pie(values, labels=labels) - Produce a pie plot out of a list of values list and labeled with labels
    - .savefig(filename) - Write the current figure as an static image

Challenge - Matplotlib

- Plot the function:

$$f\left(x\right) = x^2$$

- and

$$g\left(x\right) = \left(x-3\right)^3$$

- side by side (separate Axes objects) for:

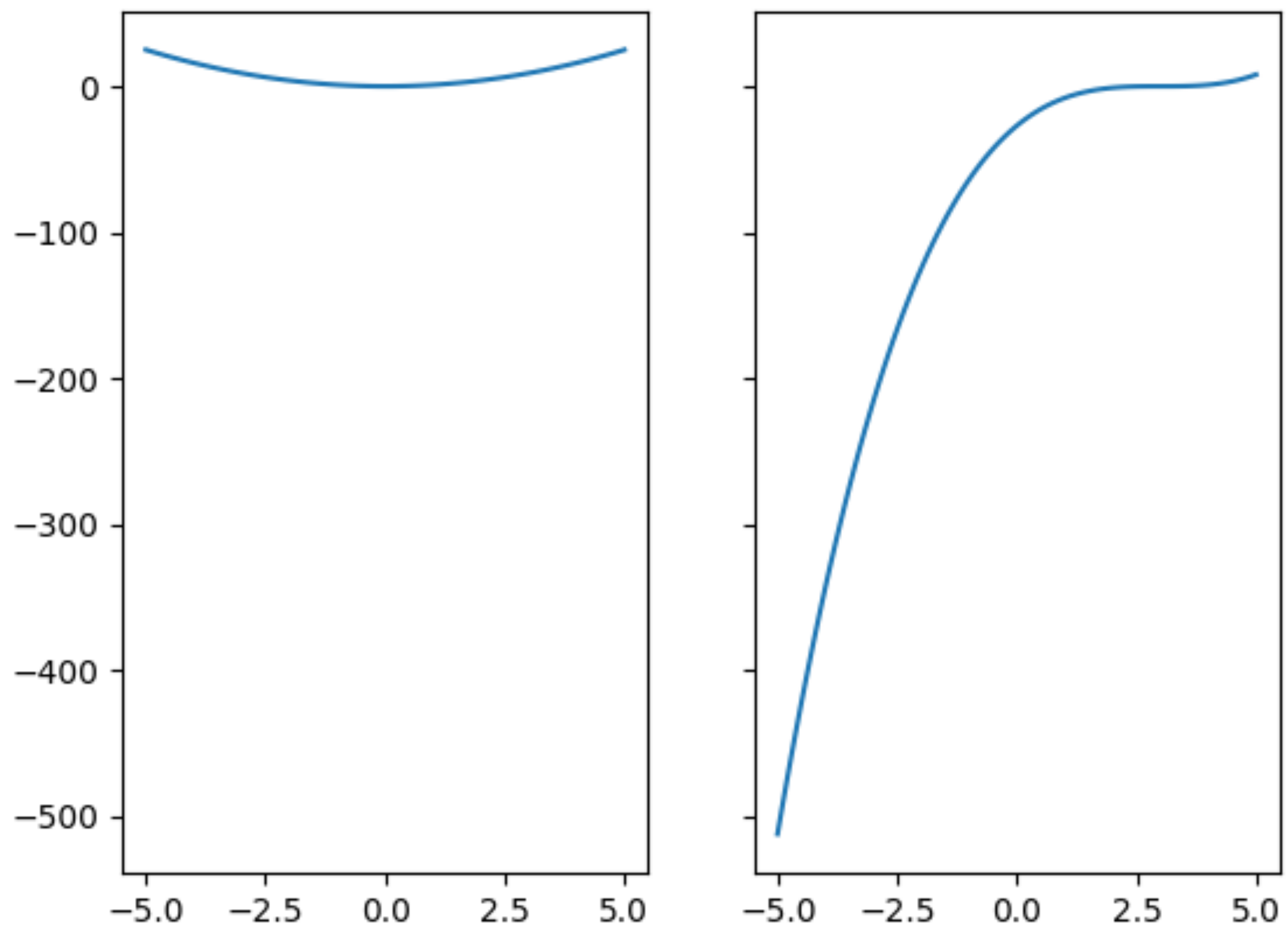
$$x \in \left[-5, 5\right]$$

Challenge - Matplotlib

```
import matplotlib.pyplot as plt
import numpy as np

fig, ax_lst = plt.subplots(1, 2, sharex=True, sharey=True)

x = np.linspace(-5, 5, 100)
y1 = np.power(x, 2.)
y2 = np.power(x-3, 3.)

ax_lst[0].plot(x, y1)
ax_lst[1].plot(x, y2)
fig.savefig('demo.png')
```


matplotlib_demo.py

Challenge - Matplotlib



Challenge - Matplotlib

- Plot the function:

$$f\left(x\right) = x^2$$

- and

$$g\left(x\right) = \left(x - 3\right)^3$$

- side by side (separate Axes objects) for:

$$x = 1, 2, 3, 4, 5$$

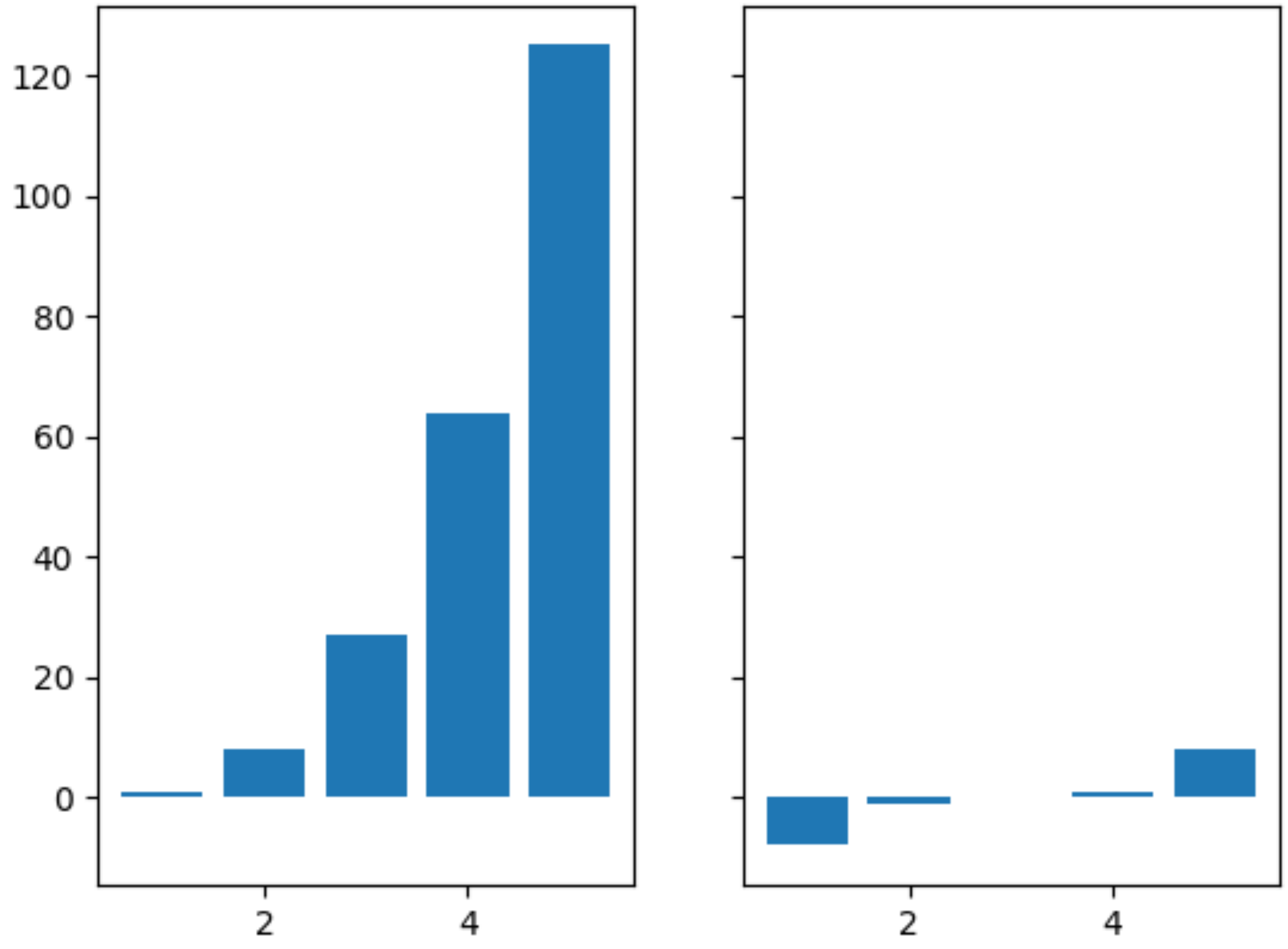
- As a bar chart

Challenge - Matplotlib

```
import matplotlib.pyplot as plt
import numpy as np

fig, ax_lst = plt.subplots(1, 2, sharex=True,
sharey=True)

x = np.arange(1, 6)
y1 = np.power(x, 3.)
y2 = np.power(x-3, 3.)

ax_lst[0].bar(x, y1)
ax_lst[1].bar(x, y2)
fig.savefig('demo_bar.png')
```

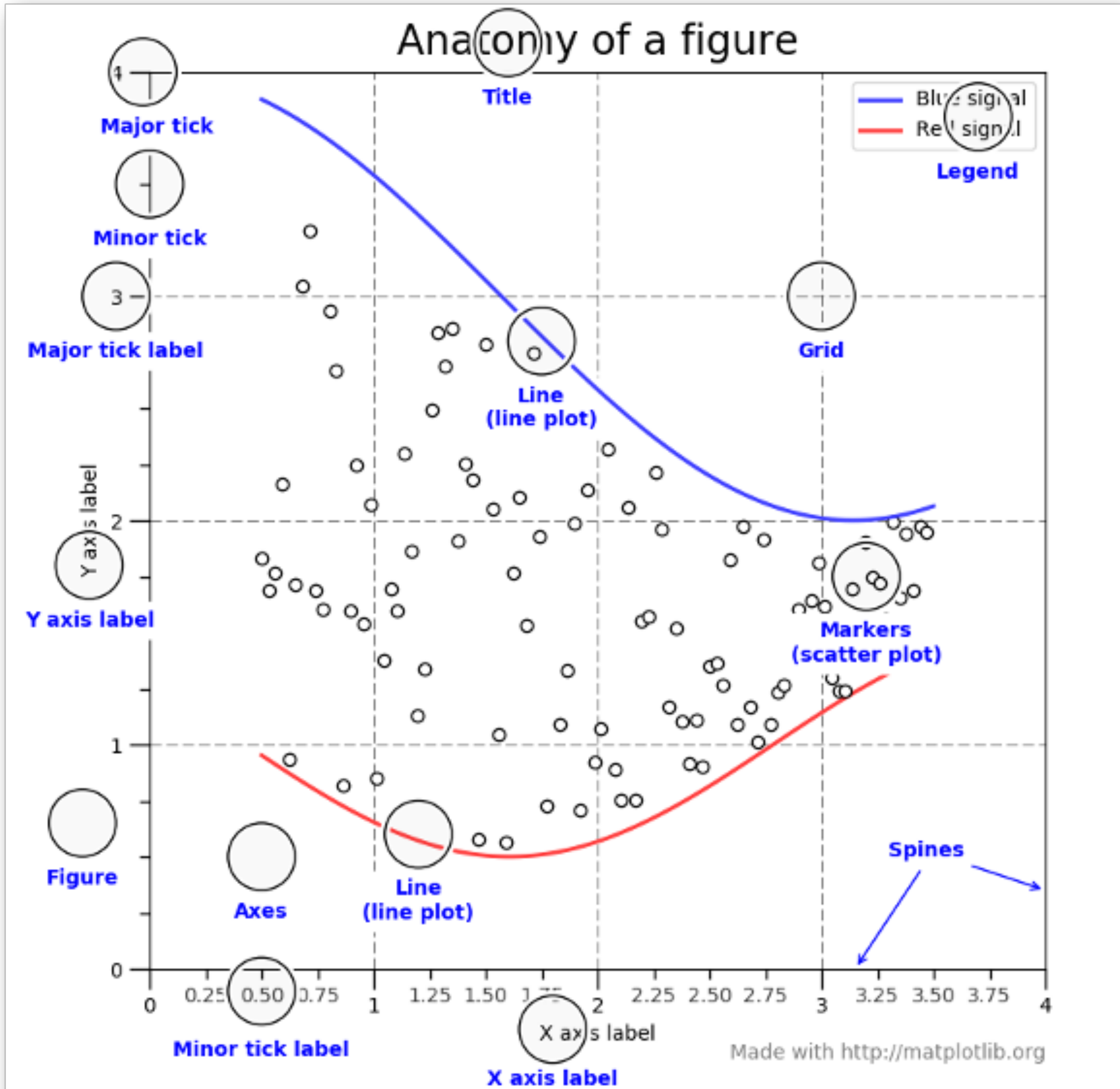

matplotlib_demo_bar.py

Challenge - Matplotlib



Matplotlib - decorations

https://matplotlib.org/2.0.0/



- The respective functions are named in an intuitive wa Every Axes object has as methods:
  - .set_xlabel(label)
  - .set_ylabel(label)
  - .set_title(title)
- And axis limits can be set using:
  - .set_xlim(xmin, xmax)
  - .set_ylim(ymin, ymax)
- Tick marks and labels are set using:
  - .set_xticks(ticks)/.set_yticks(ticks)
  - .set_xticklabels(labels)/.set_yticklabels(labels)

Matplotlib - Images

https://matplotlib.org/2.0.0/

- .imshow(fig) - Display an image on a set of axes.
- fig can be any matrix of numbers.
- Further plotting can occur by simply using the functions described above

Challenge - imshow

- Plot the population distribution of the US (similarly to last time) from

../Lecture V/geofiles/US_pop.asc

- Add a line connecting two points:

(39.163355, -86.523435)

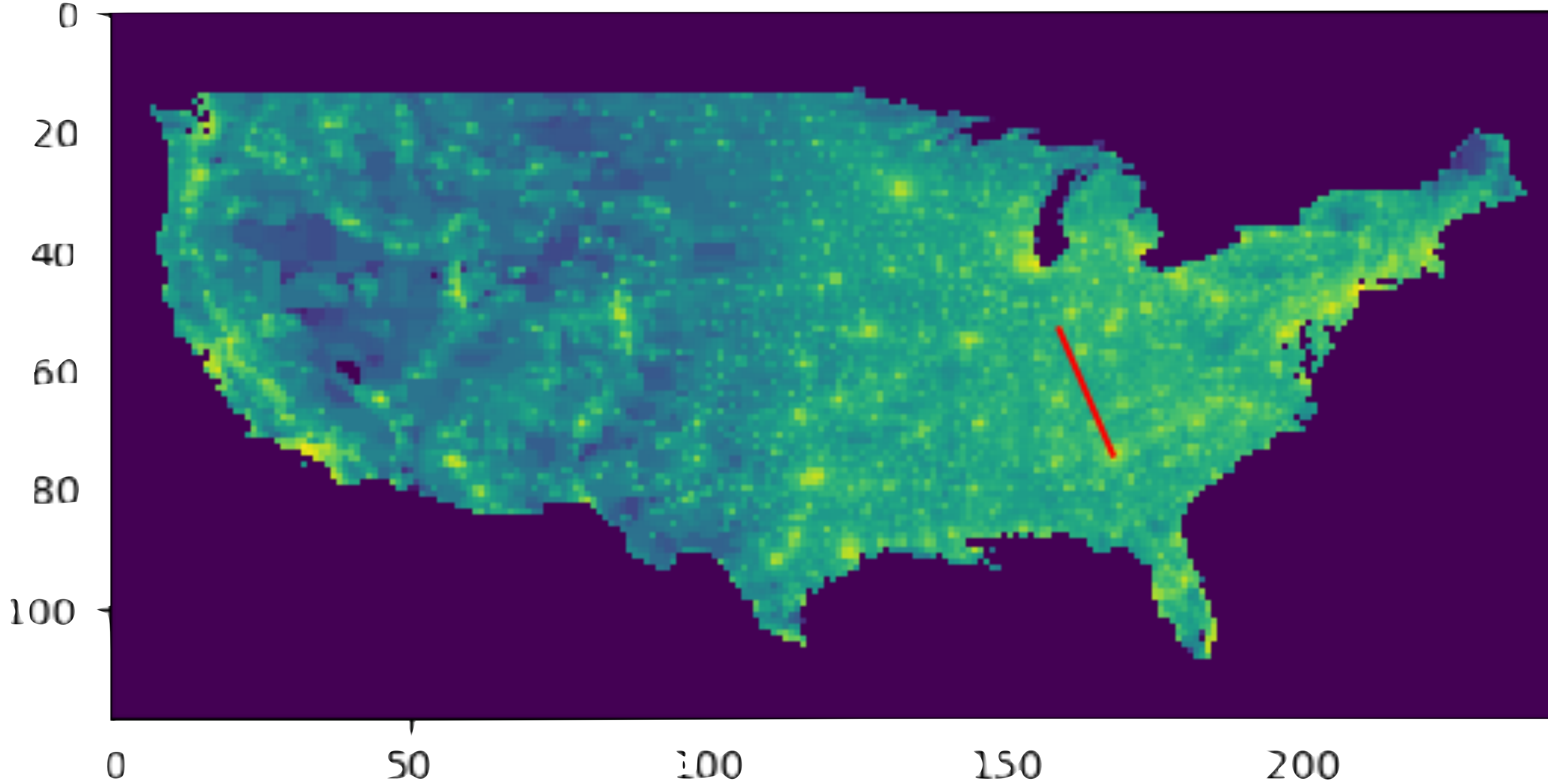
(33.761926, -84.404820)

Challenge - imshow

```
import numpy as np
import matplotlib.pyplot as plt

(…)

fig, ax = plt.subplots(1,1)
data, xllcorner, yllcorner, cellsize  = load_asc('../Lecture V/geofiles/US_pop.asc')
ax.imshow(np.log(data+1))

x1, y1 = map_points(xllcorner, yllcorner, cellsize, data.shape[0], -86.523435, 39.163355)
x2, y2 = map_points(xllcorner, yllcorner, cellsize, data.shape[0], -84.404820, 33.761926)

ax.plot([x1, x2], [y1, y2], 'r-')
fig.savefig('Manhattan_ASC.png')
```


matplotlib_imshow.py

Challenge - imshow



Challenge - Overlap a raster and a shapefile

https://www.census.gov/geo/maps-data/data/cbf/cbf_state.html

- Replot the US population file:

../Lecture V/geofiles/US_pop.asc

- And overlap the shape file

geofiles/48States/48States.shp

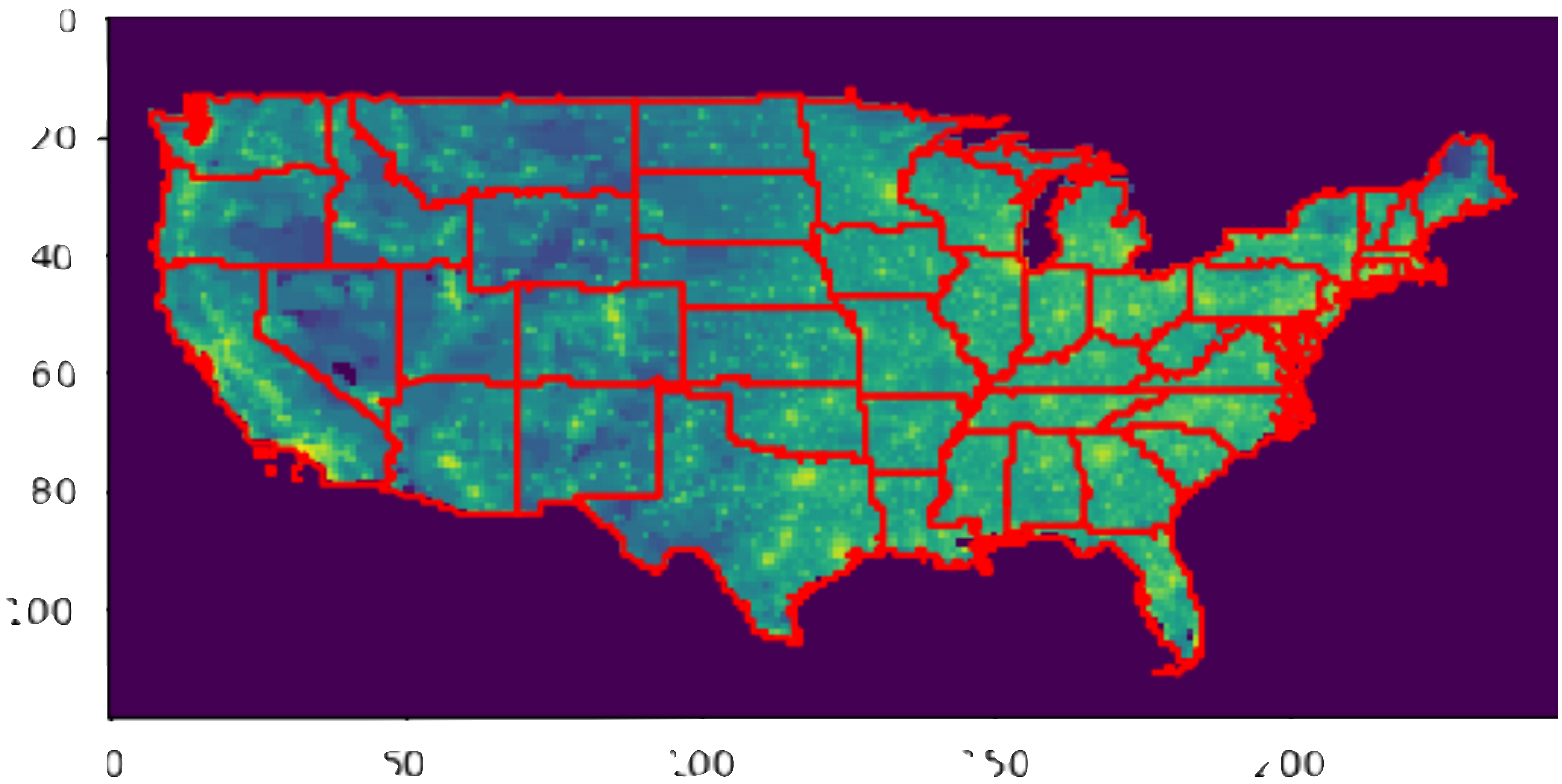
- On top of it. Don't forget that you have to convert the lat and lon values to matrix cell coordinates.

Challenge - Overlap a raster and a shapefile

```
import numpy as np
import matplotlib.pyplot as plt
import shapefile

(…)

fig, ax = plt.subplots(1,1)
data, xllcorner, yllcorner, cellsize  = load_asc('../Lecture V/geofiles/US_pop.asc')
ax.imshow(np.log(data+1))

shp = shapefile.Reader('geofiles/48States/48States.shp')

pos = None
count = 0
for shape in shp.iterShapes():
        points = np.array(shape.points)
        parts = shape.parts
        parts.append(len(shape.points))

        for i in range(len(parts)-1):
            positions = []

            for j in range(parts[i+1]-parts[i]):
                x_orig = points.T[0][parts[i]+j]
                y_orig = points.T[1][parts[i]+j]
                x, y = map_points(xllcorner, yllcorner, cellsize, data.shape[0], x_orig, y_orig)
                positions.append([x, y])

            positions = np.array(positions)

            ax.plot(positions.T[0], positions.T[1], 'r-')

fig.savefig('US overlap.png')
```

matplotlib_overlap.py

Challenge - Overlap a raster and a shapefile



@bgoncalves

Matplotlib Basemap

Basemap

- The Basemap module is the workhorse and returns a Basemap object when instantiated.
- The Basemap object has many useful methods to assist in drawing a map:
    - .drawcoastlines() - To draw the coastlines of continents
    - .drawmapboundary() - To draw the boundary of the map
    - .fillcontinents() - add color to the continents
    - etc…
- The constructor for Basemap can take many different arguments to be able to handle different projections, but it defaults to the Plate Carrée projection centered at (0, 0)
- The minimal map is simply:

```
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt

map = Basemap()
map.drawcoastlines()
plt.savefig('basemap_demo.png')
```

- Please note that with the .drawcoastlines() call nothing is plotted as our map has no content.

Basemap

- We can also visualize just specific regions by setting the bbox and center coordinates by setting

llcrnrlon, llcrnrlat, urcrnrlon, urcrnrlat

- And

lat_0, lon_0

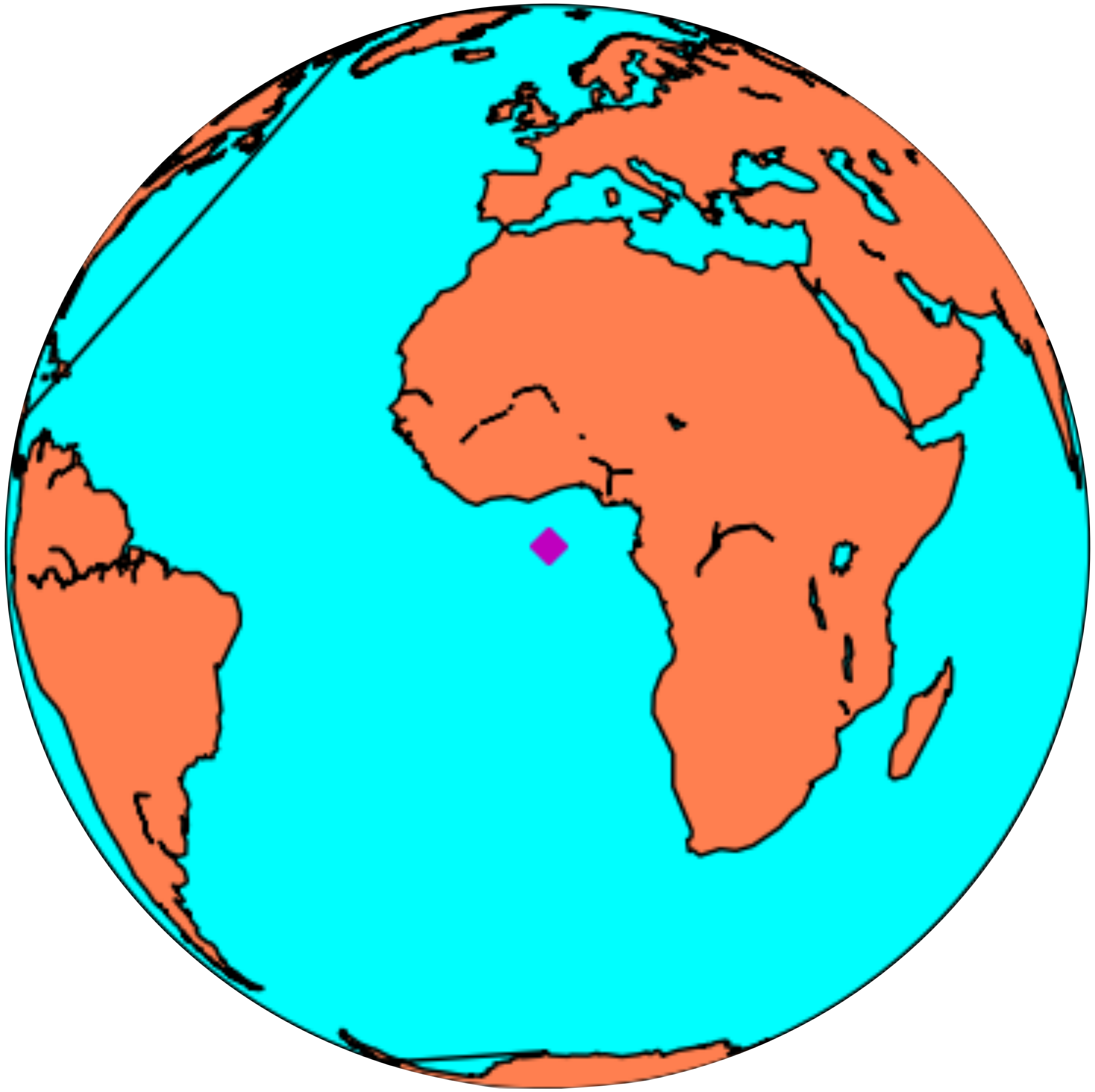
- Respectively.
- We can convert arbitrary **lat, lon** values to map coordinates by calling the map() object directly.
- After we obtain the map coordinates we can add them to the map by calling the .plot(x, y) method of the map object.

Basemap Example

```
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt

map = Basemap(projection='ortho', lat_0=0, lon_0=0)

map.drawmapboundary(fill_color='aqua')
map.fillcontinents(color='coral',lake_color='aqua')
map.drawcoastlines()

x, y = map(0, 0)

map.plot(x, y, marker='D',color='m')

plt.savefig('globe.png')
```


basemap_example.py

Basemap Example



smopy

smopy

https://github.com/rossant/smopy

- smopy lets you retrieve OpenStreetMap map tiles for an arbitrary bbox
- .Map(bbox, z=zoom) - Returns an image for the the box defined by lllat, lllon, urlat, urlon and zoom is the zoom level, from 1 to 9.
- The .Map() function return a map object.
- The map object provides a method .to_pixels(lat, lon) to convert arbitrary lat, lon pairs to the internal coordinates of the figure.
- A reference to a matplotlib Axes object can be obtained by calling .show_mpl() on the map object.
- A reference to the current figure can be obtained using plt.gcf() for further manipulations.
- map

Challenge - smopy

- Use smopy to get a map of the entire US, using the bbox coordinates:

(24.396308, -124.848974, 49.384358, -66.885444)

- And add a line between coordinates:

(39.163355, -86.523435)

(33.761926, -84.404820)

Challenge - smopy

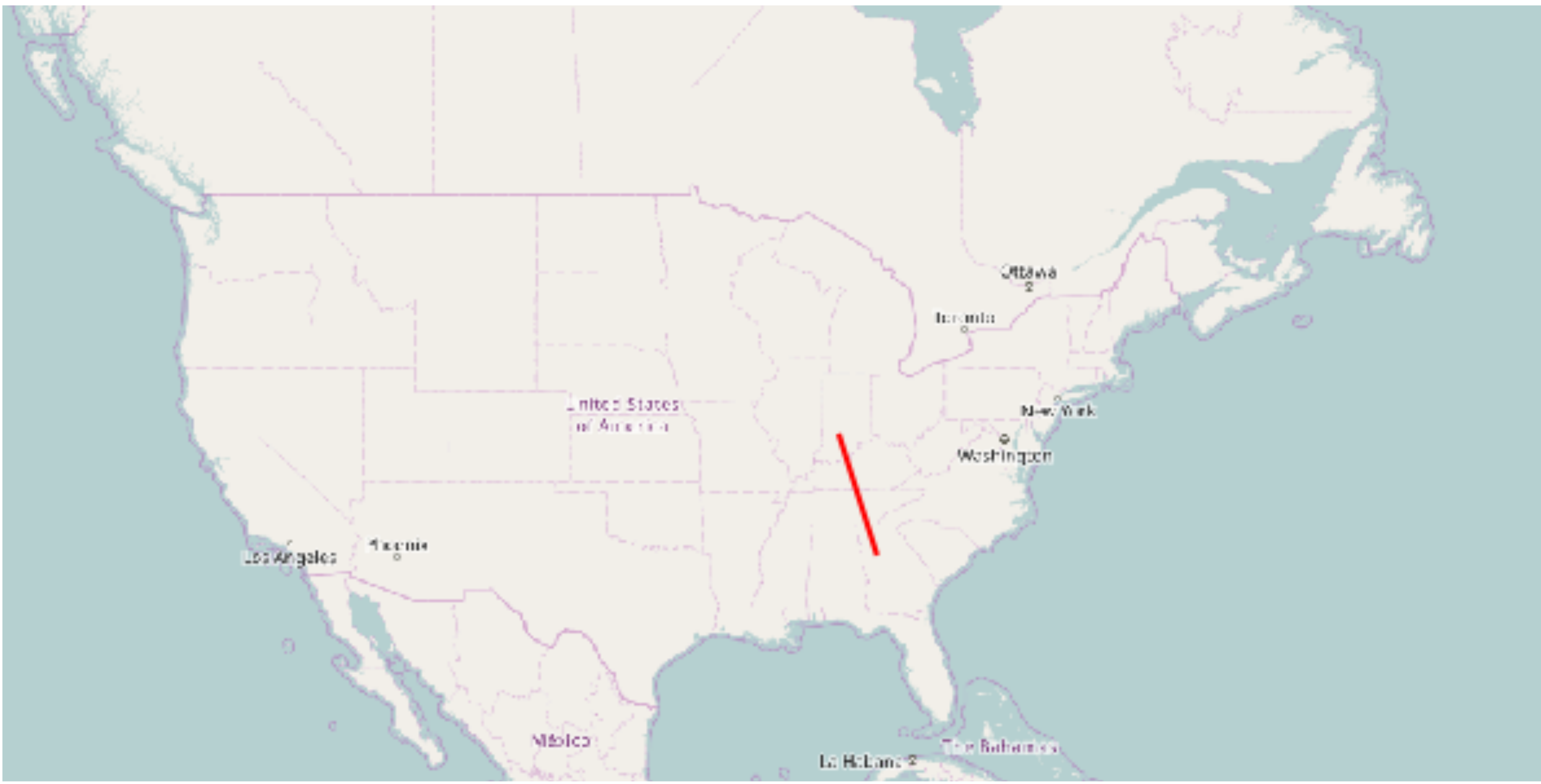
https://github.com/rossant/smopy

```
import smopy
import matplotlib.pyplot as plt

map = smopy.Map(24.396308, -124.848974, 49.384358, -66.885444)
ax = map.show_mpl()
fig = plt.gcf()

x0, y0 = map.to_pixels(39.163355, -86.523435)
x1, y1 = map.to_pixels(33.761926, -84.404820)

ax.plot([x0, x1], [y0, y1], 'r-')
fig.savefig('US_OSM.png')
```


smopy_demo.py

Challenge - smopy

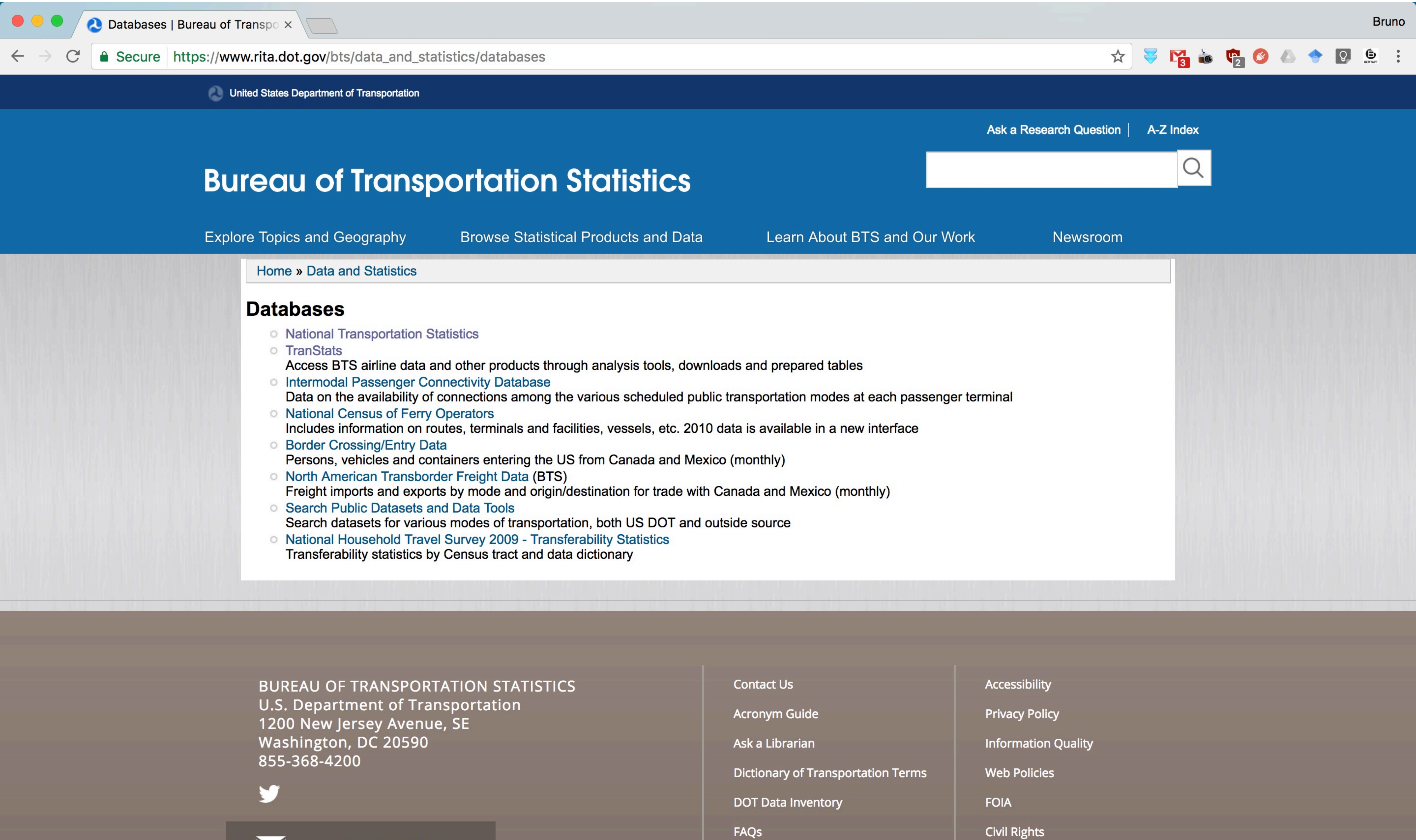
https://github.com/rossant/smopy



Case Study - GLEaM

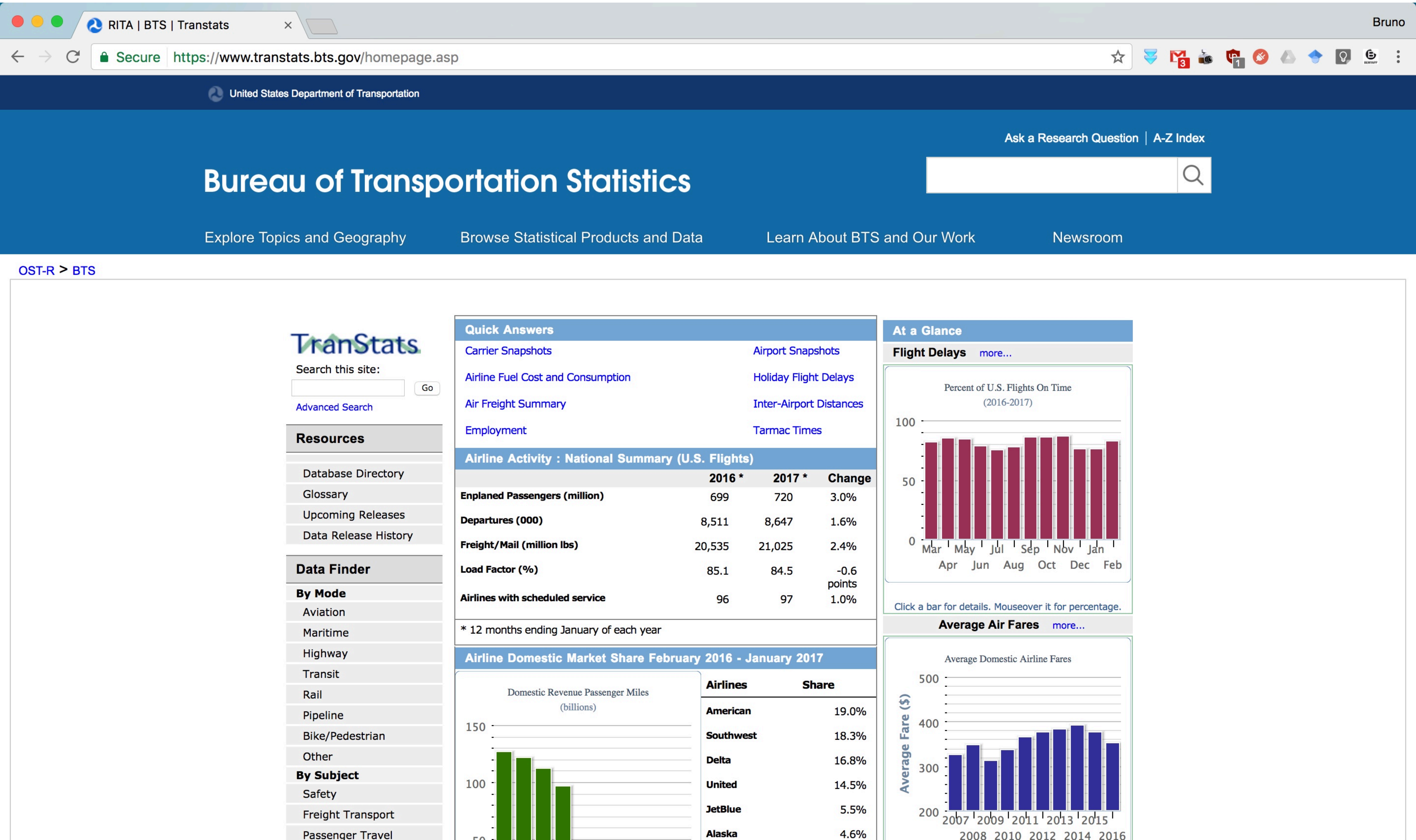
US Flight dataset

https://www.rita.dot.gov/bts/data_and_statistics/databases



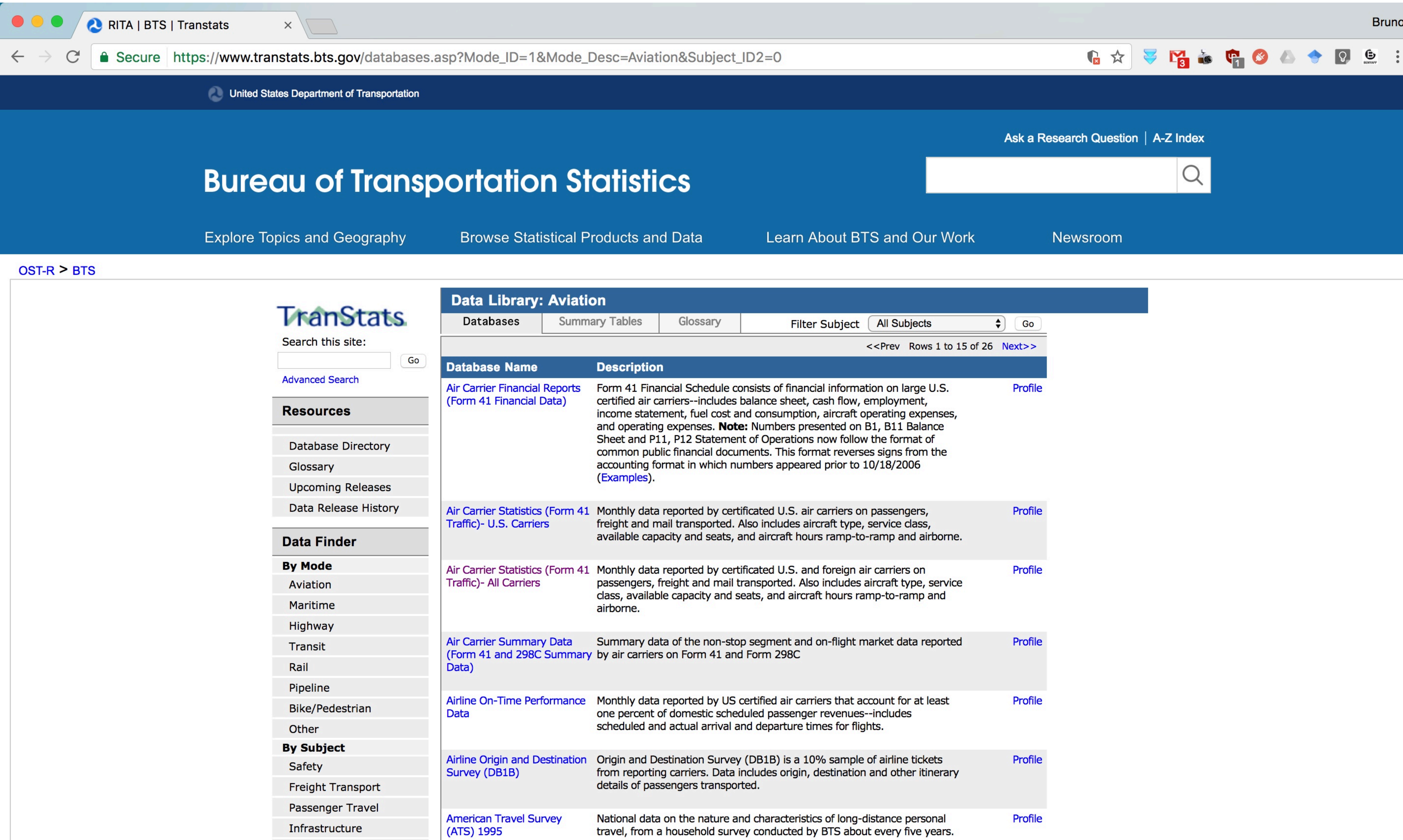
US Flight dataset

https://www.transtats.bts.gov/homepage.asp

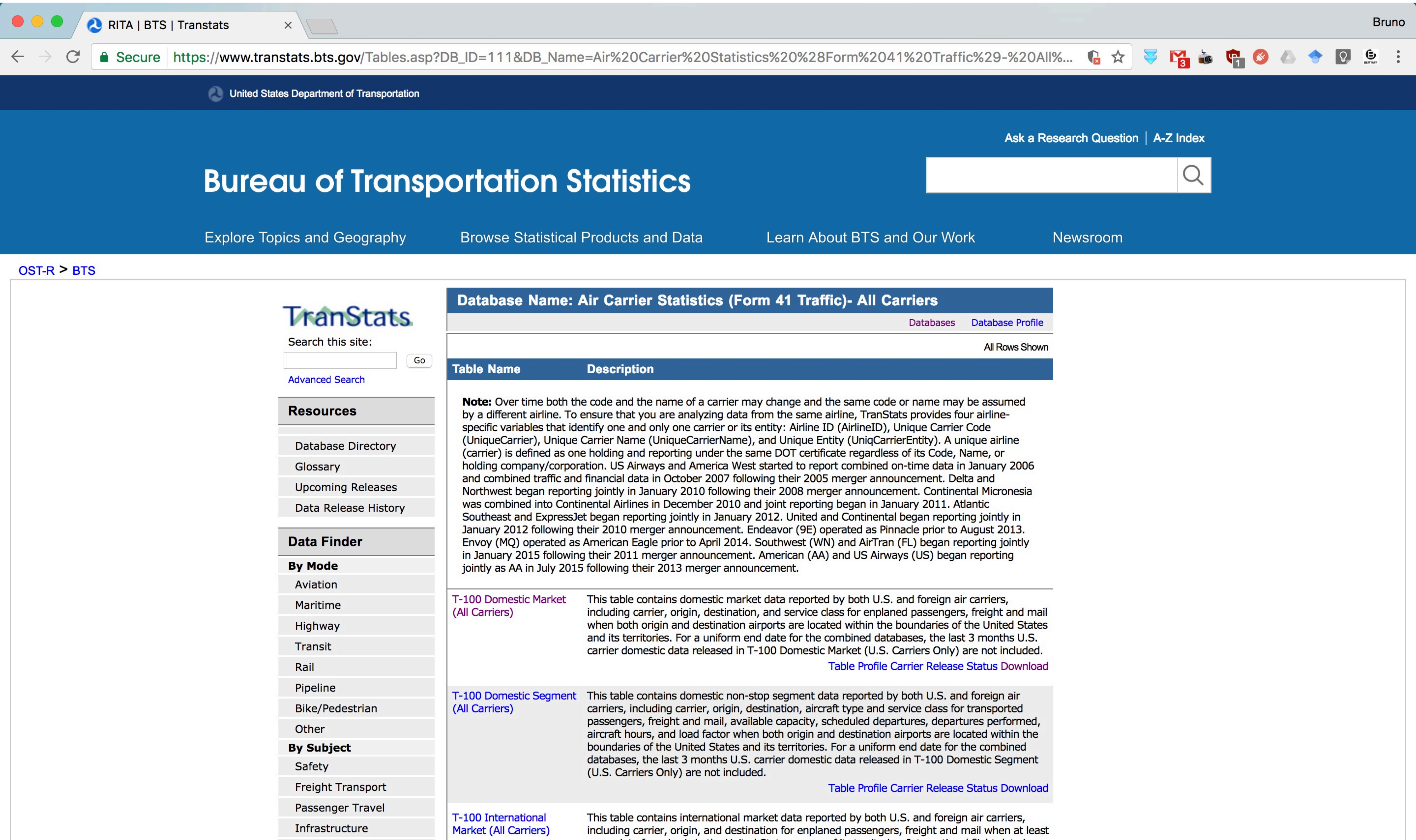


US Flight dataset

https://www.transtats.bts.gov/databases.asp?Mode_ID=1&Mode_Desc=Aviation&Subject_ID2=0

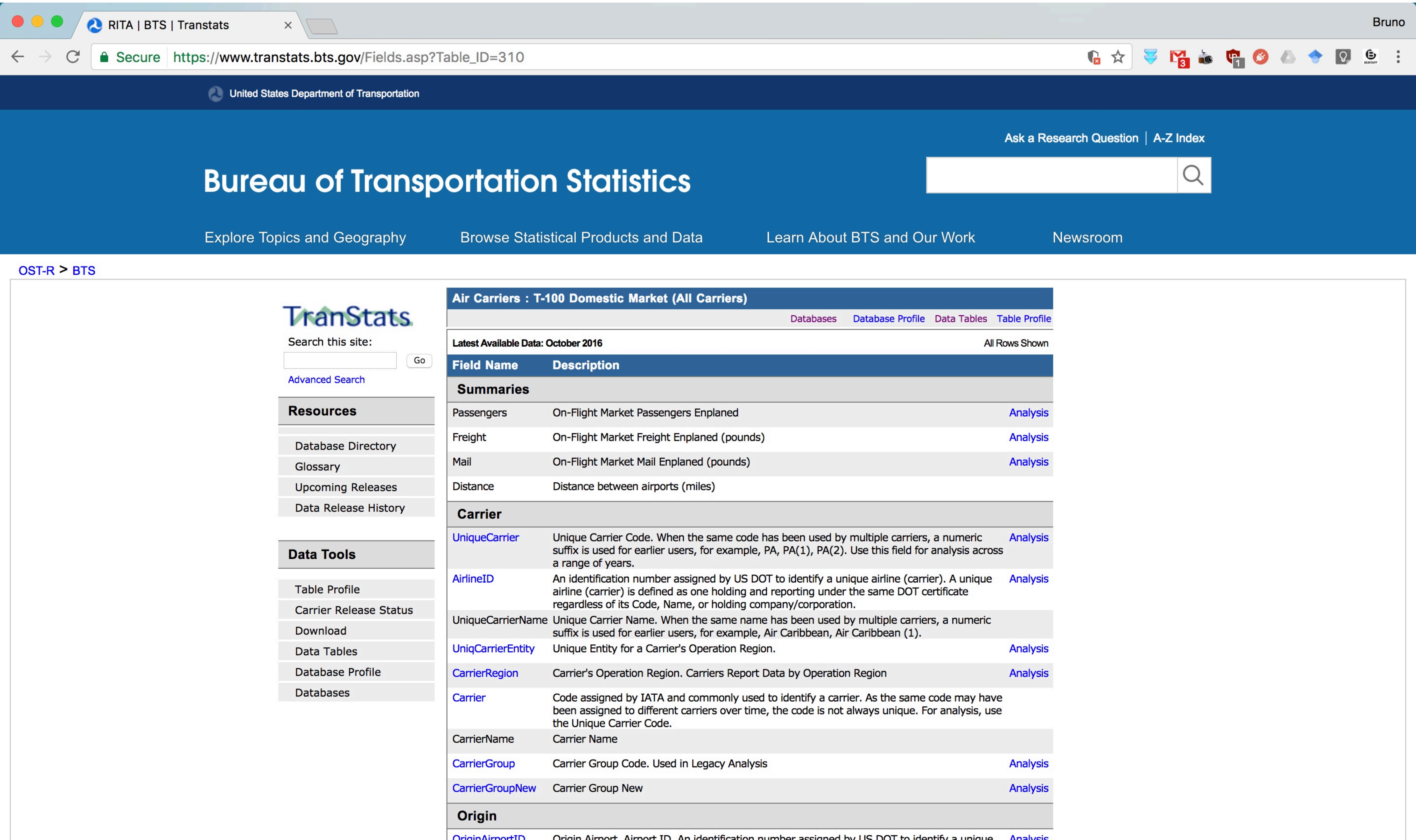


US Flight dataset



US Flight dataset

https://www.transtats.bts.gov/Fields.asp?Table_ID=310



RITA | BTS | Transtats

Bruno

Secure https://www.transtats.bts.gov/Fields.asp?Table_ID=310

United States Department of Transportation

Ask a Research Question | A-Z Index

Bureau of Transportation Statistics

Explore Topics and Geography · Browse Statistical Products and Data · Learn About BTS and Our Work · Newsroom

OST-R > BTS

TranStats

Search this site: Go

Advanced Search

Resources

- Database Directory
- Glossary
- Upcoming Releases
- Data Release History

Data Tools

- Table Profile
- Carrier Release Status
- Download
- Data Tables
- Database Profile
- Databases

Air Carriers : T-100 Domestic Market (All Carriers)

Databases · Database Profile · Data Tables · Table Profile

Latest Available Data: October 2016 — All Rows Shown

Field Name	Description	
Summaries		
Passengers	On-Flight Market Passengers Enplaned	Analysis
Freight	On-Flight Market Freight Enplaned (pounds)	Analysis
Mail	On-Flight Market Mail Enplaned (pounds)	Analysis
Distance	Distance between airports (miles)	
Carrier		
UniqueCarrier	Unique Carrier Code. When the same code has been used by multiple carriers, a numeric suffix is used for earlier users, for example, PA, PA(1), PA(2). Use this field for analysis across a range of years.	Analysis
AirlineID	An identification number assigned by US DOT to identify a unique airline (carrier). A unique airline (carrier) is defined as one holding and reporting under the same DOT certificate regardless of its Code, Name, or holding company/corporation.	Analysis
UniqueCarrierName	Unique Carrier Name. When the same name has been used by multiple carriers, a numeric suffix is used for earlier users, for example, Air Caribbean, Air Caribbean (1).	
UniqCarrierEntity	Unique Entity for a Carrier's Operation Region.	Analysis
CarrierRegion	Carrier's Operation Region. Carriers Report Data by Operation Region	Analysis
Carrier	Code assigned by IATA and commonly used to identify a carrier. As the same code may have been assigned to different carriers over time, the code is not always unique. For analysis, use the Unique Carrier Code.	
CarrierName	Carrier Name	
CarrierGroup	Carrier Group Code. Used in Legacy Analysis	Analysis
CarrierGroupNew	Carrier Group New	Analysis
Origin		
OriginAirportID	Origin Airport, Airport ID. An identification number assigned by US DOT to identify a unique	Analysis

US Flight dataset

https://www.transtats.bts.gov/TableInfo.asp?Table_ID=310

RITA | BTS | Transtats

Bruno

Secure https://www.transtats.bts.gov/TableInfo.asp?Table_ID=310

United States Department of Transportation

Ask a Research Question | A-Z Index

Bureau of Transportation Statistics

Explore Topics and Geography | Browse Statistical Products and Data | Learn About BTS and Our Work | Newsroom

OST-R > BTS

TranStats

Search this site:

Go

Advanced Search

Resources

- Database Directory
- Glossary
- Upcoming Releases
- Data Release History

Data Tools

- Analysis
- Table Contents
- Download

: T-100 Domestic Market (All Carriers)

Databases | Database Profile | Data Tables | Table Contents

Property	Description
Name	T-100 Domestic Market (All Carriers)
Description	This table contains domestic market data reported by both U.S. and foreign air carriers, including carrier, origin, destination, and service class for enplaned passengers, freight and mail when both origin and destination airports are located within the boundaries of the United States and its territories. For a uniform end date for the combined databases, the last 3 months U.S. carrier domestic data released in T-100 Domestic Market (U.S. Carriers Only) are not included.
Records	5,570,059
Fields	36
First Year	1990
Last Year	2016
Frequency	Monthly
Latest Available Data	October, 2016

Terms	Definitions
Air Freight	Property, other than express and passenger baggage transported by air.
Airline ID	An identification number assigned by US DOT to identify a unique airline (carrier). A unique airline (carrier) is defined as one holding and reporting under the same DOT certificate regardless of its Code, Name, or holding company/corporation. Use this field for analysis across a range of years.
Airport Code	A three character alpha-numeric code issued by the U.S. Department of Transportation which is the official designation of the airport. The airport code is not always unique to a specific airport because airport codes can change or can be reused.
Airport ID	An identification number assigned by US DOT to identify a unique airport. Use this field for airport analysis across a range of years because an airport can change its airport code and airport codes can be reused.
Carrier Code	Code assigned by IATA and commonly used to identify a carrier. As the same code may have been assigned to different carriers over time, the code is not always unique.
City Market ID	An identification number assigned by US DOT to identify a city market. Use this field to consolidate airports serving the same city market.

US Flight dataset

https://www.transtats.bts.gov/TableInfo.asp?Table_ID=310

RITA | BTS | Transtats

Bruno

Secure https://www.transtats.bts.gov/TableInfo.asp?Table_ID=310

United States Department of Transportation

Ask a Research Question | A-Z Index

Bureau of Transportation Statistics

Explore Topics and Geography | Browse Statistical Products and Data | Learn About BTS and Our Work | Newsroom

OST-R > BTS

TranStats

Search this site:

Go

Advanced Search

Resources

- Database Directory
- Glossary
- Upcoming Releases
- Data Release History

Data Tools

- Analysis
- Table Contents
- Download

: T-100 Domestic Market (All Carriers)

Databases | Database Profile | Data Tables | Table Contents

Property	Description
Name	T-100 Domestic Market (All Carriers)
Description	This table contains domestic market data reported by both U.S. and foreign air carriers, including carrier, origin, destination, and service class for enplaned passengers, freight and mail when both origin and destination airports are located within the boundaries of the United States and its territories. For a uniform end date for the combined databases, the last 3 months U.S. carrier domestic data released in T-100 Domestic Market (U.S. Carriers Only) are not included.
Records	5,570,059
Fields	36
First Year	1990
Last Year	2016
Frequency	Monthly
Latest Available Data	October, 2016

Terms	Definitions
Air Freight	Property, other than express and passenger baggage transported by air.
Airline ID	An identification number assigned by US DOT to identify a unique airline (carrier). A unique airline (carrier) is defined as one holding and reporting under the same DOT certificate regardless of its Code, Name, or holding company/corporation. Use this field for analysis across a range of years.
Airport Code	A three character alpha-numeric code issued by the U.S. Department of Transportation which is the official designation of the airport. The airport code is not always unique to a specific airport because airport codes can change or can be reused.
Airport ID	An identification number assigned by US DOT to identify a unique airport. Use this field for airport analysis across a range of years because an airport can change its airport code and airport codes can be reused.
Carrier Code	Code assigned by IATA and commonly used to identify a carrier. As the same code may have been assigned to different carriers over time, the code is not always unique.
City Market ID	An identification number assigned by US DOT to identify a city market. Use this field to consolidate airports serving the same city market.

US Flight dataset

bit.ly/BTS_Airports

RITA | BTS | Transtats

Bruno

Secure https://www.transtats.bts.gov/FieldInfo.asp?Field_Desc=Origin%20Airport%2C%20Airport%20ID.%20An%20identification%20number%20as...

United States Department of Transportation

Ask a Research Question | A-Z Index

Bureau of Transportation Statistics

Explore Topics and Geography | Browse Statistical Products and Data | Learn About BTS and Our Work | Newsroom

OST-R > BTS

TranStats

Search this site: Go

Advanced Search

Resources

- Database Directory
- Glossary
- Upcoming Releases
- Data Release History

Data Finder

By Mode

- Aviation
- Maritime
- Highway
- Transit
- Rail
- Pipeline
- Bike/Pedestrian
- Other

By Subject

- Safety
- Freight Transport
- Passenger Travel
- Infrastructure

Air Carriers : T-100 Domestic Market (All Carriers)

Field: Origin Airport, Airport ID. An identification number assigned by US DOT to identify a unique airport. Use this field for airport analysis across a range of years because an airport can change its airport code and airport codes can be reused.

Format results for printing | Download Lookup Table | Databases | Data Tables | Table Contents

<<Prev Rows : 1 - 100 of 1974 Next>>

Code	Description
10001	Afognak Lake, AK: Afognak Lake Airport
10003	Granite Mountain, AK: Bear Creek Mining Strip
10004	Lik, AK: Lik Mining Camp
10005	Little Squaw, AK: Little Squaw Airport
10006	Kizhuyak, AK: Kizhuyak Bay
10008	Elizabeth Island, AK: Elizabeth Island Airport
10009	Homer, AK: Augustin Island
10010	Hudson, NY: Columbia County
10011	Peach Springs, AZ: Grand Canyon West
10012	Blairstown, NJ: Blairstown Airport
10013	Crosbyton, TX: Crosbyton Municipal
10014	Fairbanks/Ft. Wainwright, AK: Blair Lake
10015	Deadmans Bay, AK: Deadmans Bay Airport
10016	Hallo Bay, AK: Hallo Bay Airport
10017	Red Lake, AK: Red Lake Airport
10020	Selawik, AK: Roland Norton Memorial

Challenge - US Airports

- Geocode the airport names from

flights/L_AIRPORT_ID.csv

- Keep in mind that, unfortunately, not all airports will be found.

Challenge - US Airports

```python
import googlemaps
from google_accounts import accounts

app = accounts["torino"]
gmaps = googlemaps.Client(key=app["api_key"])

line_count = 0
for line in open("flights/L_AIRPORT_ID.csv"):
    fields = line.strip()[:-1].split(',')

    line_count += 1

    if line_count > 1:
        try:
            airport_id = fields[0]
            airport_name = ", ".join(fields[2:])

            geocode_result = gmaps.geocode(airport_name)

            print(airport_id[1:-1], geocode_result[0]["geometry"]["location"]["lat"],\
                  geocode_result[0]["geometry"]["location"]["lng"])
        except:
            pass
```

@bgoncalves

google_airports.py

Challenge - Airport Network

- Build the airport network. Each airport is a node and the weight of each edge is given by the number of flights between airports.

- Plot the edge weight distribution

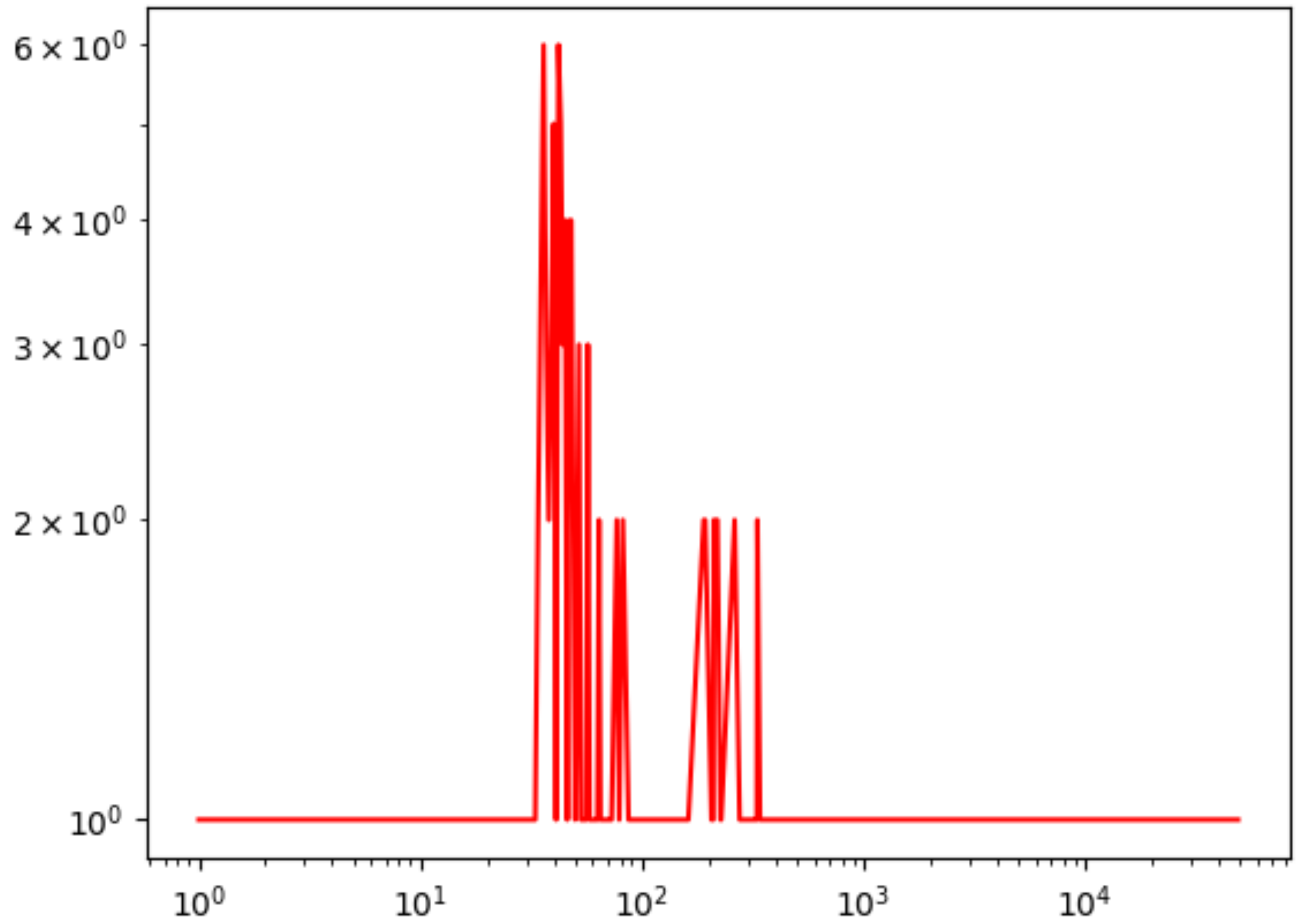
Challenge - Airport Network

```python
from collections import Counter
import matplotlib.pyplot as plt
import numpy as np

line_count = 0
edges = {}

for line in open("flights/4567008_T_T100D_MARKET_ALL_CARRIER.csv"):
    fields = line.strip().split(',')

    line_count += 1

    if line_count == 1:
        header = dict(zip([field[1:-1] for field in fields], range(len(fields))))
        continue

    node_i = line[header["ORIGIN_AIRPORT_ID"]]
    node_j = line[header["DEST_AIRPORT_ID"]]
    edge = (node_i, node_j)

    edges[edge] = edges.get(edge, 0) + 1

Pw = Counter(edges.values())
Pw = list(Pw.items())
Pw.sort(key=lambda x:x[0])

Pw = np.array(Pw)

fig, ax = plt.subplots()
ax.loglog(Pw.T[0], Pw.T[1], 'r-')
plt.savefig('Pw.png')
```

airport_network.py

@bgoncalves

Challenge - Airport Network



@bgoncalves

airport_network.py

Challenge - Distance Matrix

- Calculate the distance between every pair of airports listed in

airport_gps.dat

- using the vincenty distance and print out every pair that is less than 20km apart.

Challenge - Distance Matrix

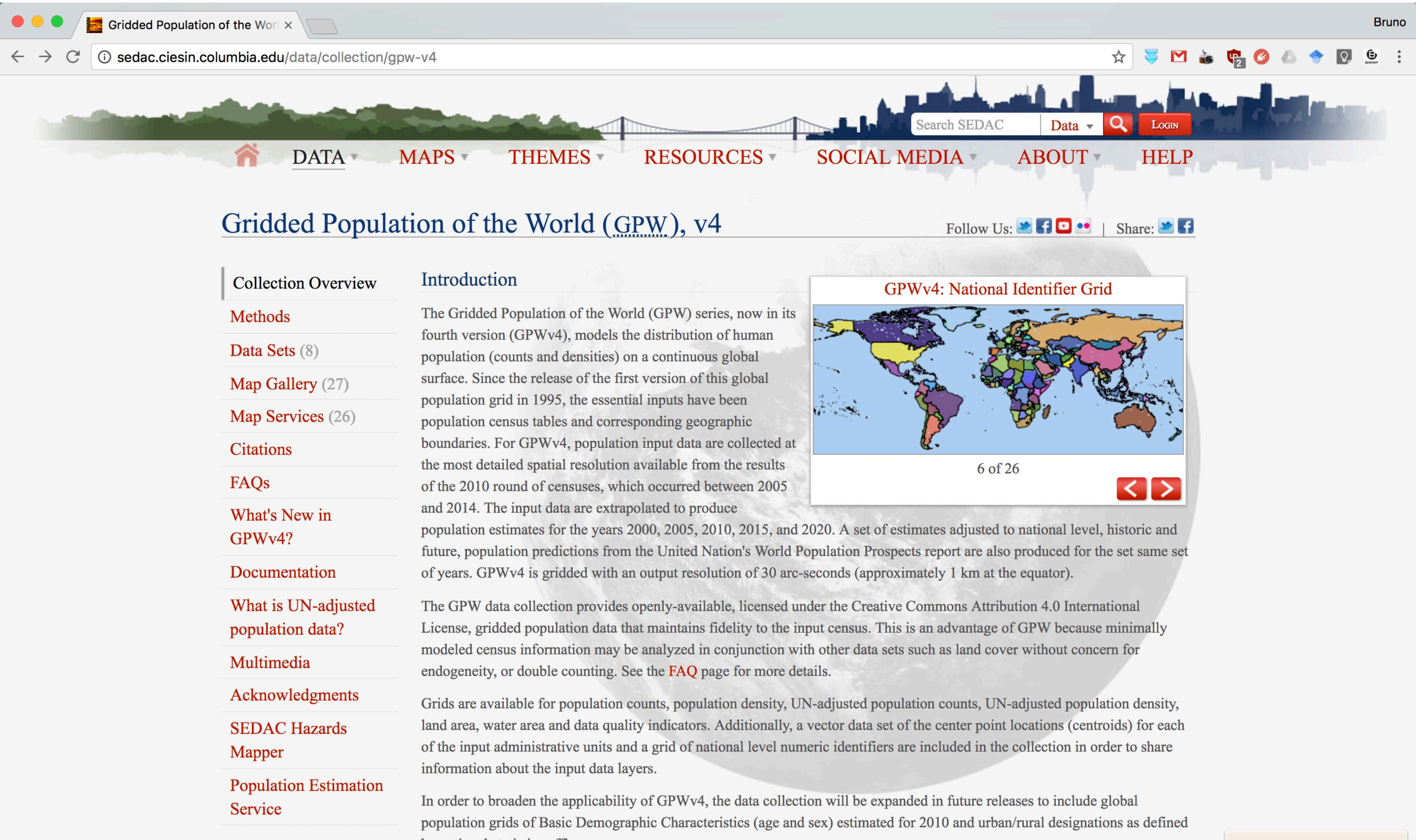
```
from geopy import distance

airports = {}

for line in open("airport_gps.dat"):
    fields = line.strip().split()

    airport_id = int(fields[0])
    lat = float(fields[1])
    lon = float(fields[2])

    airports[airport_id] = (lat, lon)

airport_list = airports.keys()

for airport_i in airport_list:
    for airport_j in airport_list:
        try:
            if airport_i != airport_j:
                dist = distance.distance(airports[airport_i], airports[airport_j]).km

                if dist < 20:
                    print(airport_i, airport_j, dist)
        except:
            pass
```

@bgoncalves

airport_matrix.py

GPW

http://sedac.ciesin.columbia.edu/data/collection/gpw-v4



Gridded Population of the Wor ×

sedac.ciesin.columbia.edu/data/collection/gpw-v4

Search SEDAC | Data | LOGIN

DATA | MAPS | THEMES | RESOURCES | SOCIAL MEDIA | ABOUT | HELP

Gridded Population of the World (GPW), v4

Follow Us: | Share:

- Collection Overview
- Methods
- Data Sets (8)
- Map Gallery (27)
- Map Services (26)
- Citations
- FAQs
- What's New in GPWv4?
- Documentation
- What is UN-adjusted population data?
- Multimedia
- Acknowledgments
- SEDAC Hazards Mapper
- Population Estimation Service

Introduction

GPWv4: National Identifier Grid

6 of 26

The Gridded Population of the World (GPW) series, now in its fourth version (GPWv4), models the distribution of human population (counts and densities) on a continuous global surface. Since the release of the first version of this global population grid in 1995, the essential inputs have been population census tables and corresponding geographic boundaries. For GPWv4, population input data are collected at the most detailed spatial resolution available from the results of the 2010 round of censuses, which occurred between 2005 and 2014. The input data are extrapolated to produce population estimates for the years 2000, 2005, 2010, 2015, and 2020. A set of estimates adjusted to national level, historic and future, population predictions from the United Nation's World Population Prospects report are also produced for the set same set of years. GPWv4 is gridded with an output resolution of 30 arc-seconds (approximately 1 km at the equator).

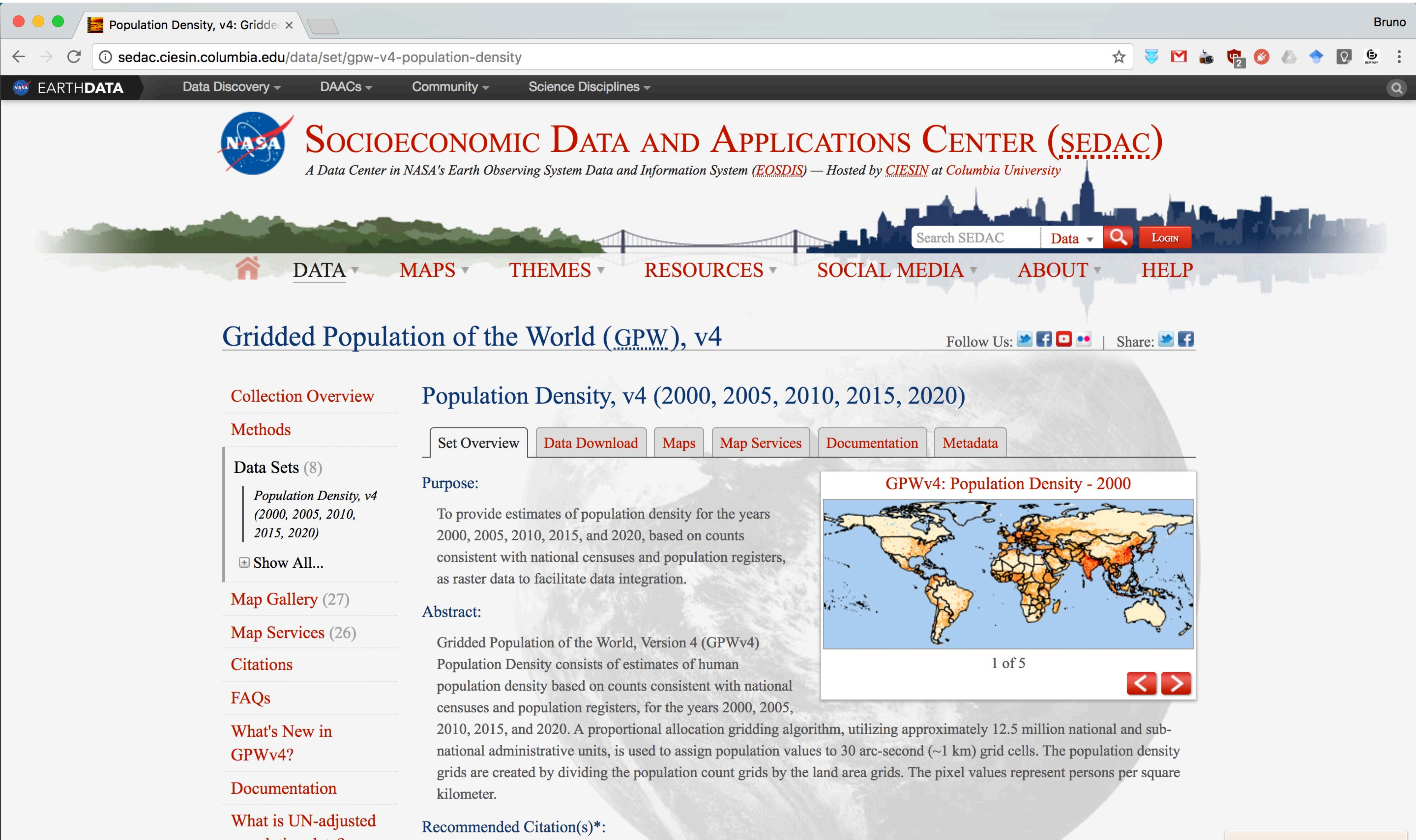
The GPW data collection provides openly-available, licensed under the Creative Commons Attribution 4.0 International License, gridded population data that maintains fidelity to the input census. This is an advantage of GPW because minimally modeled census information may be analyzed in conjunction with other data sets such as land cover without concern for endogeneity, or double counting. See the FAQ page for more details.

Grids are available for population counts, population density, UN-adjusted population counts, UN-adjusted population density, land area, water area and data quality indicators. Additionally, a vector data set of the center point locations (centroids) for each of the input administrative units and a grid of national level numeric identifiers are included in the collection in order to share information about the input data layers.

In order to broaden the applicability of GPWv4, the data collection will be expanded in future releases to include global population grids of Basic Demographic Characteristics (age and sex) estimated for 2010 and urban/rural designations as defined

GPW

http://sedac.ciesin.columbia.edu/data/collection/gpw-v4


Population Density, v4: Gridde ×

Bruno

sedac.ciesin.columbia.edu/data/set/gpw-v4-population-density

EARTHDATA Data Discovery DAACs Community Science Disciplines

Socioeconomic Data and Applications Center (SEDAC)

A Data Center in NASA's Earth Observing System Data and Information System (EOSDIS) — Hosted by CIESIN at Columbia University

Search SEDAC | Data | Login

DATA MAPS THEMES RESOURCES SOCIAL MEDIA ABOUT HELP

Gridded Population of the World (GPW), v4

Follow Us: | Share:

- Collection Overview
- Methods
- Data Sets (8)
  - *Population Density, v4 (2000, 2005, 2010, 2015, 2020)*
  - Show All...
- Map Gallery (27)
- Map Services (26)
- Citations
- FAQs
- What's New in GPWv4?
- Documentation
- What is UN-adjusted

Population Density, v4 (2000, 2005, 2010, 2015, 2020)

Set Overview | Data Download | Maps | Map Services | Documentation | Metadata

GPWv4: Population Density - 2000

1 of 5

Purpose:

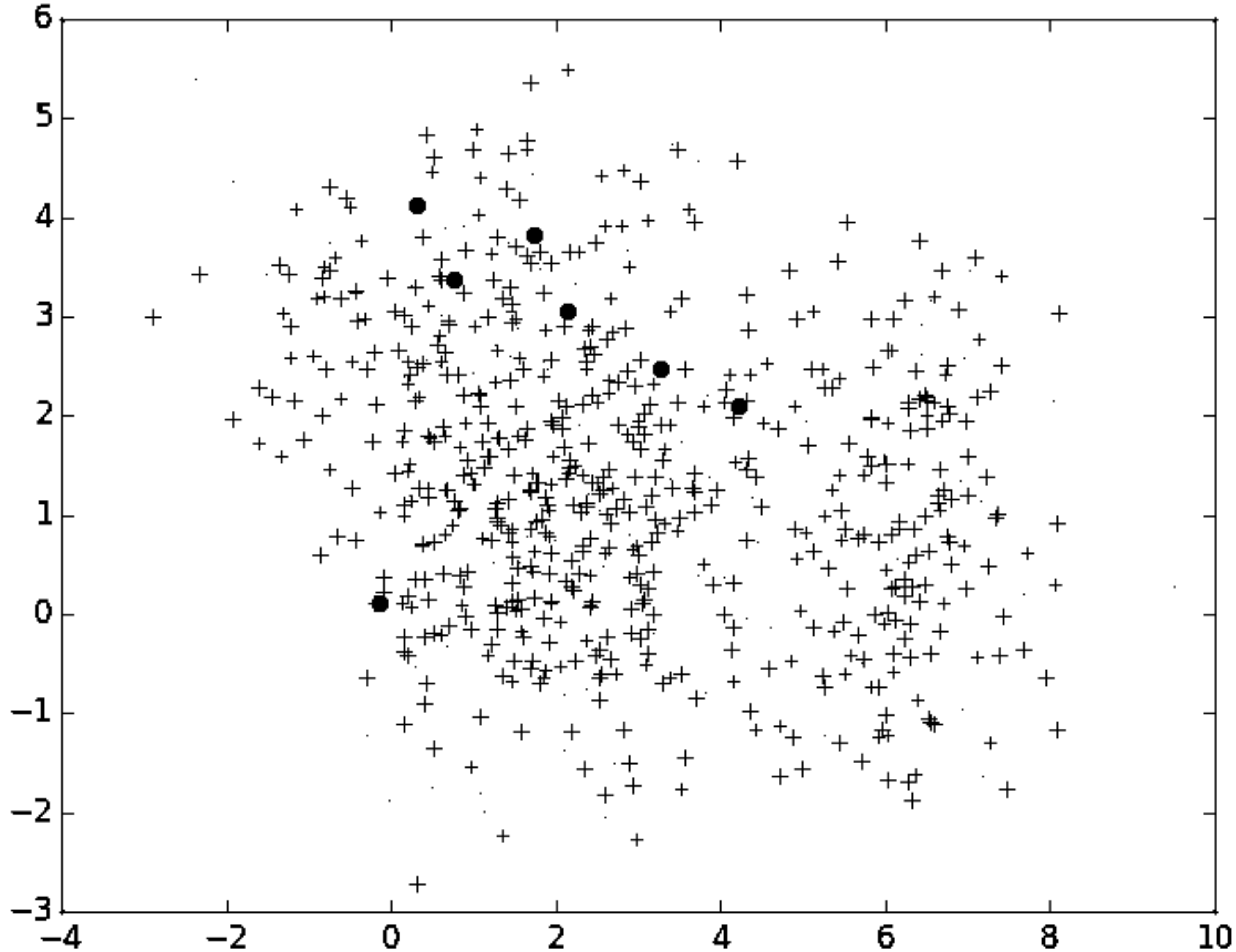
To provide estimates of population density for the years 2000, 2005, 2010, 2015, and 2020, based on counts consistent with national censuses and population registers, as raster data to facilitate data integration.

Abstract:

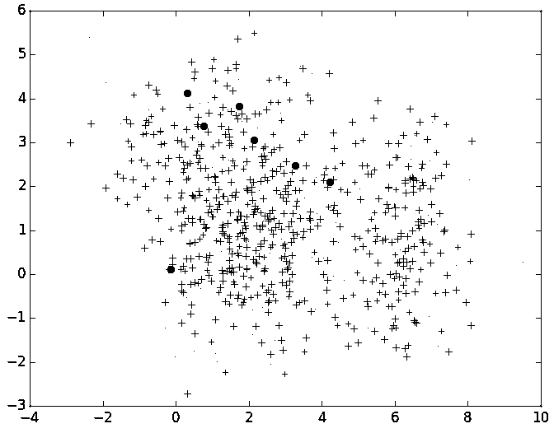
Gridded Population of the World, Version 4 (GPWv4) Population Density consists of estimates of human population density based on counts consistent with national censuses and population registers, for the years 2000, 2005, 2010, 2015, and 2020. A proportional allocation gridding algorithm, utilizing approximately 12.5 million national and sub-national administrative units, is used to assign population values to 30 arc-second (~1 km) grid cells. The population density grids are created by dividing the population count grids by the land area grids. The pixel values represent persons per square kilometer.

Recommended Citation(s)*:

Clustering



K-Means



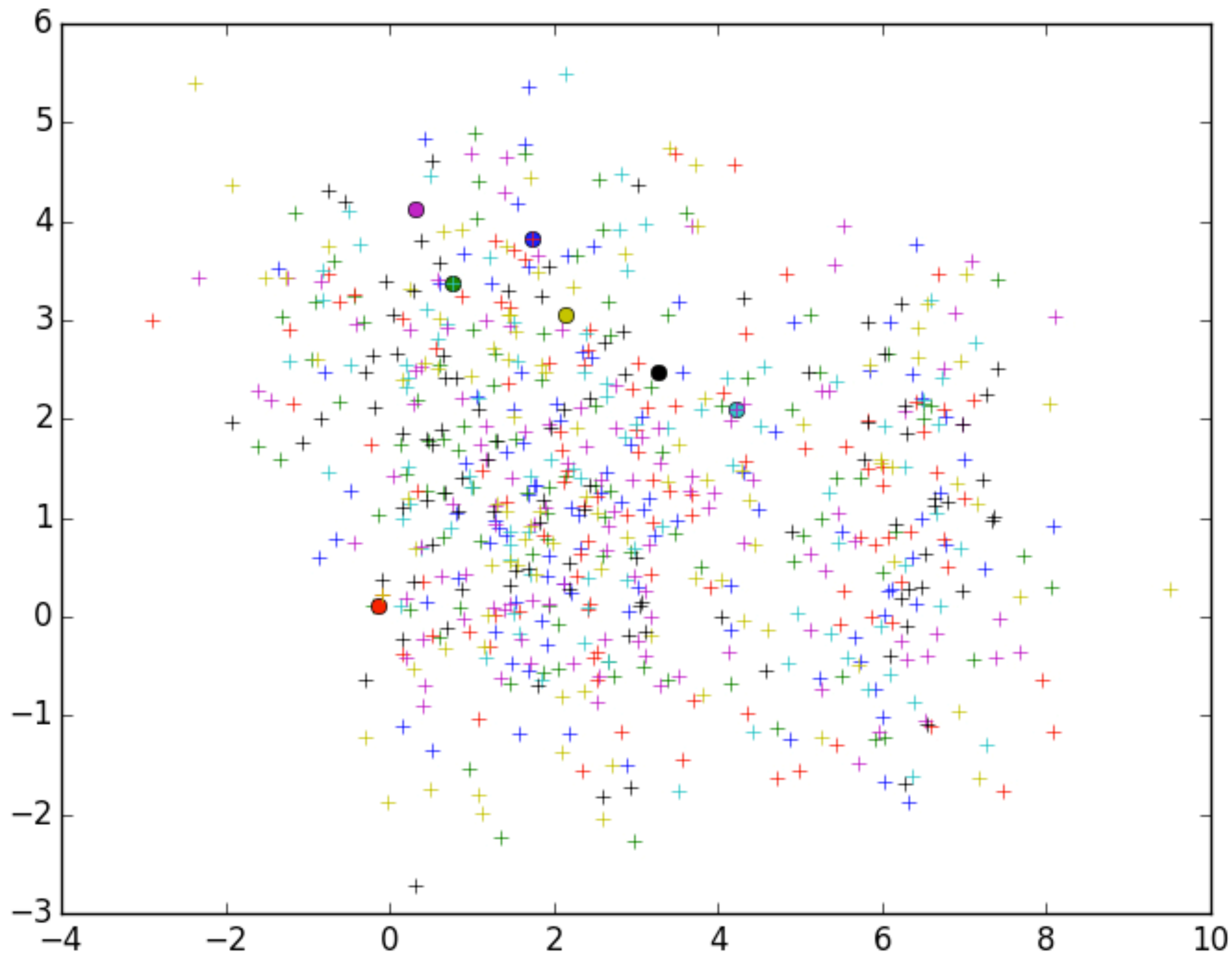
- Choose k randomly chosen points to be the centroid of each cluster

- Assign each point to belong the cluster whose centroid is closest

- Recompute the centroid positions (mean cluster position)

- Repeat until convergence

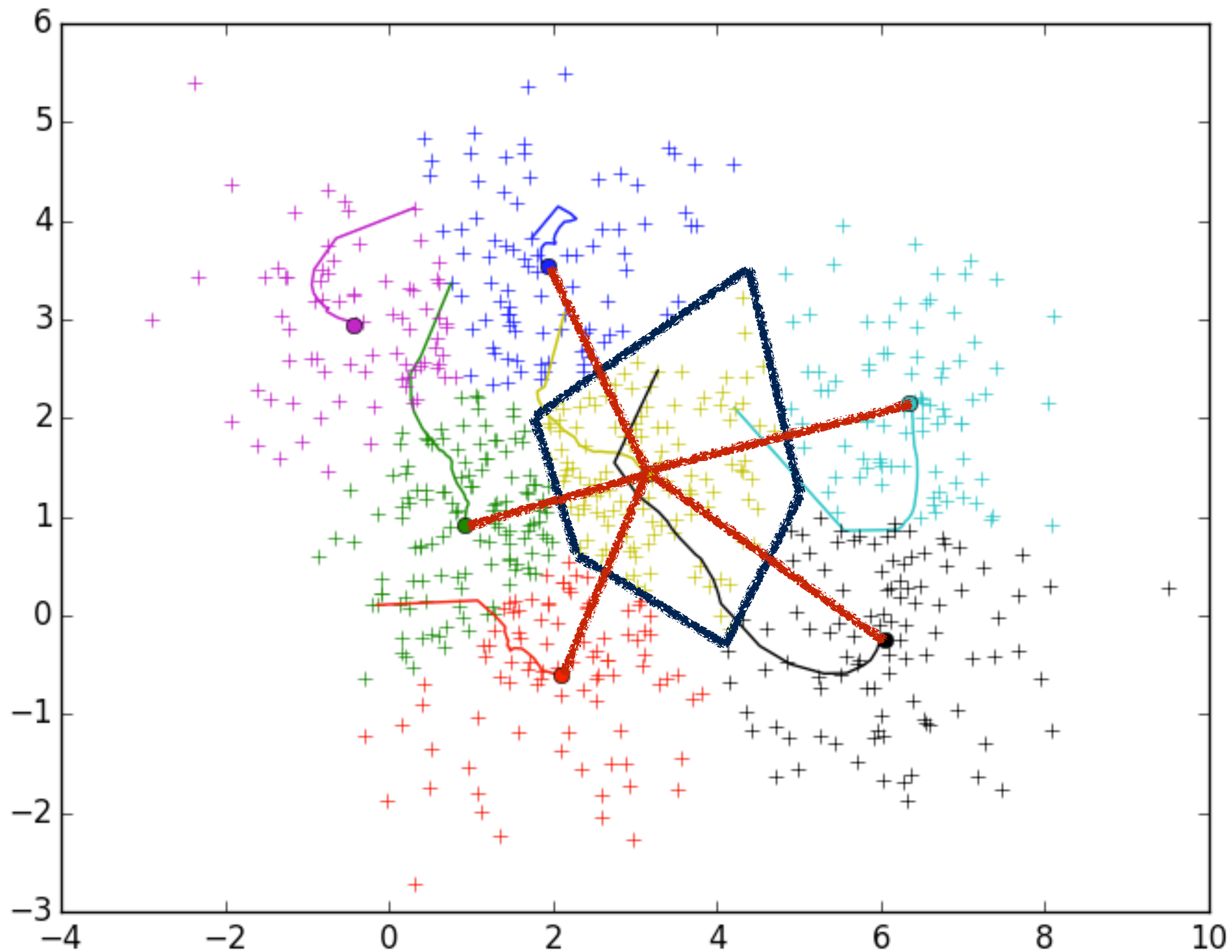
K-Means



@bgoncalves http://youtu.be/M0sCjfRP32k www.bgoncalves.com

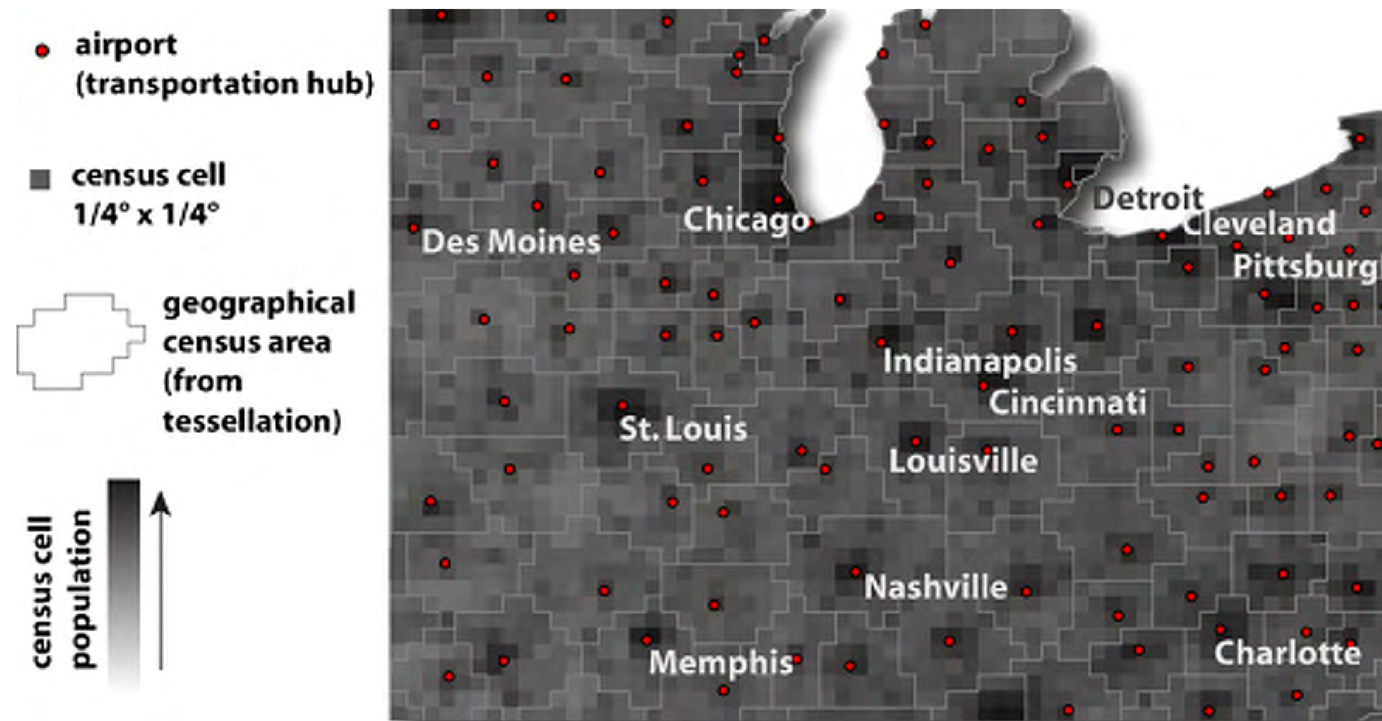
K-Means: Structure

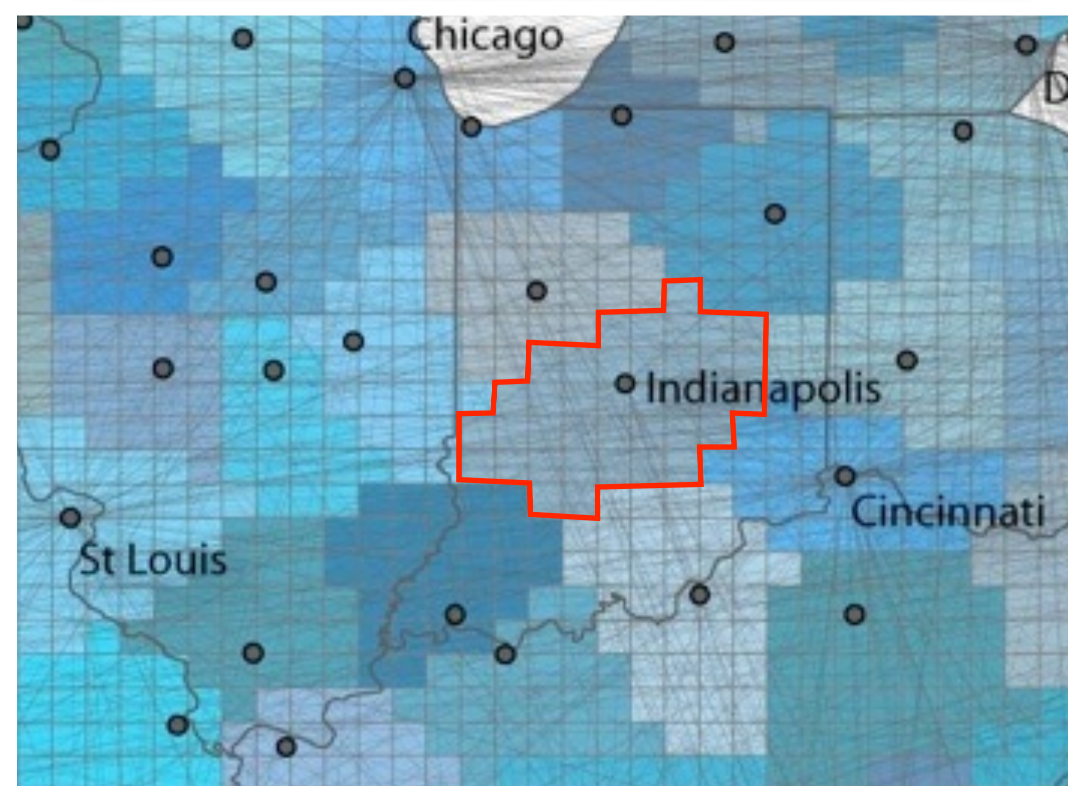
Voronoi Tesselation



Population Distribution

PNAS 106, 21484 (2009)



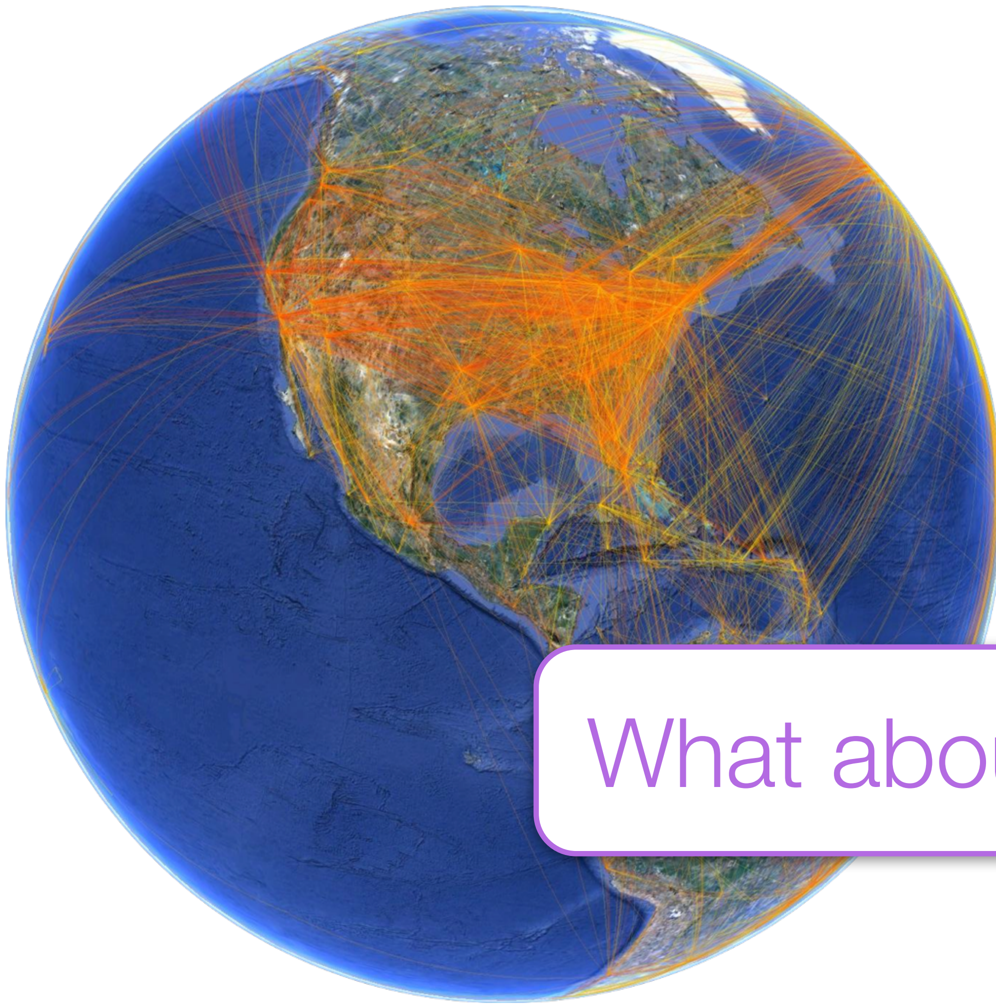


Complete IATA and OAG databases:

3362 airports worldwide
220 countries

Long Range Mobility

PNAS 106, 21484 (2009)



$$p_{jl} = \frac{w_{jl}}{N_j} \Delta t$$

- Probability that any individual in class X travel from j→l

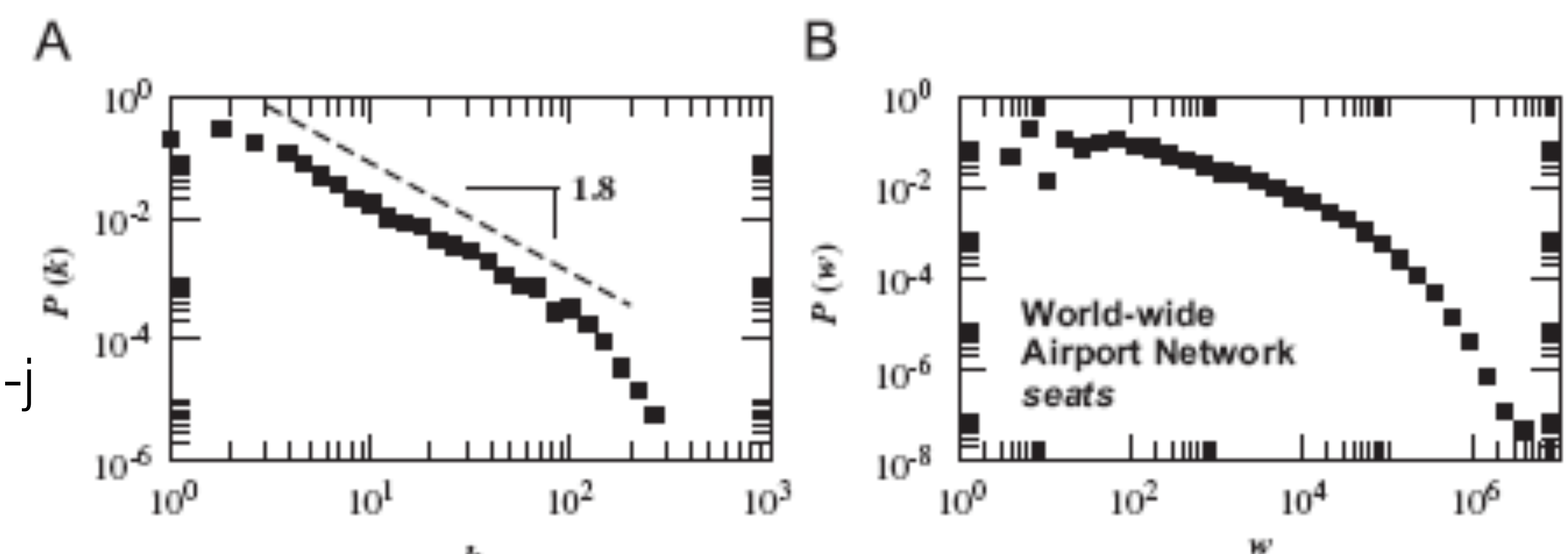
w

population

What about Short Range Mobility?

Complete IATA and OAG databases:

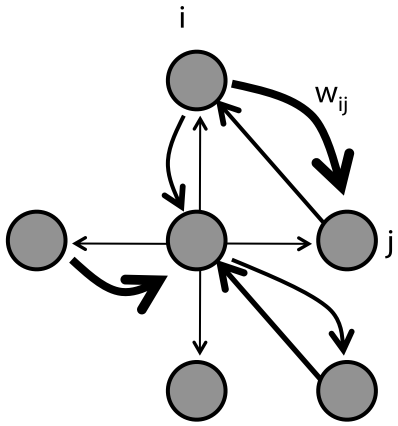
3362 airports worldwide
220 countries
> 20,000 connections
w_{ij} #passengers on connection i-j
>99% total traffic



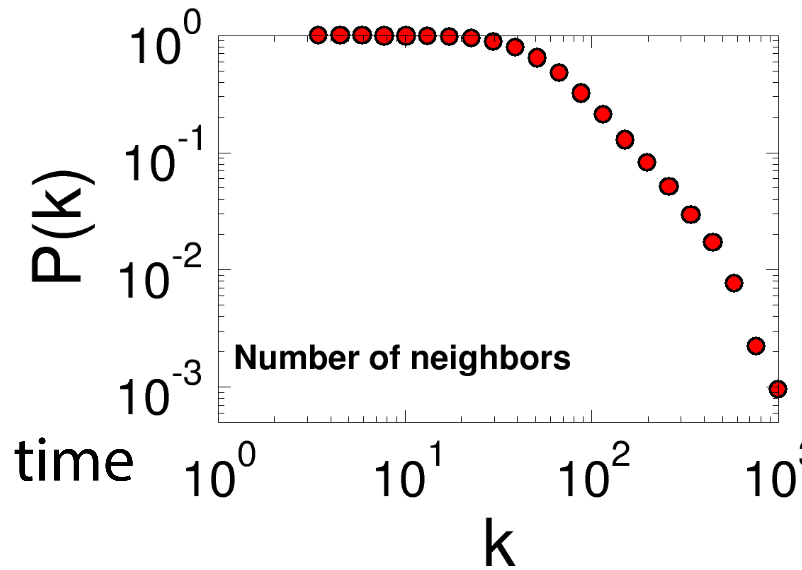
@bgoncalves

Commuting Network

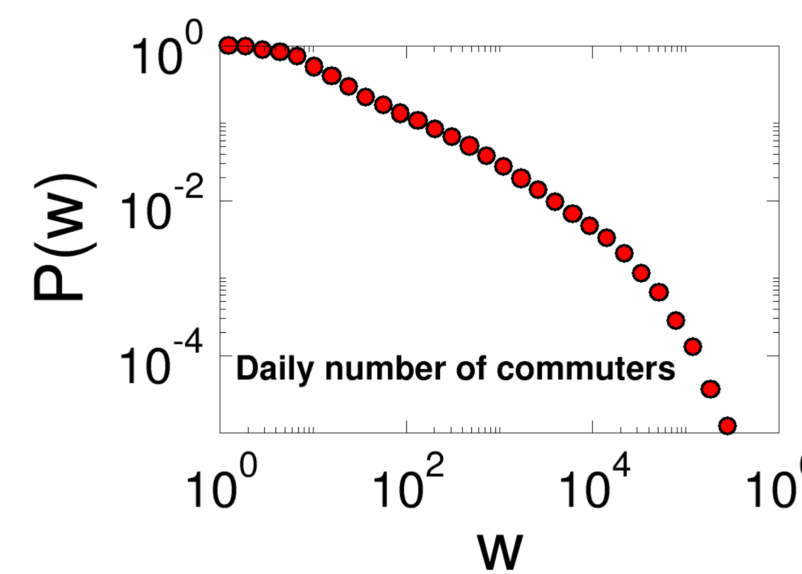
PNAS 106, 21484 (2009)



each node i : subpopulation (census area)
each link (ij) : interaction between subpopulations i and j
weight w_{ij} : number of people commuting from i to j per unit time



US county commuting network



Gravity-Law of Commuting

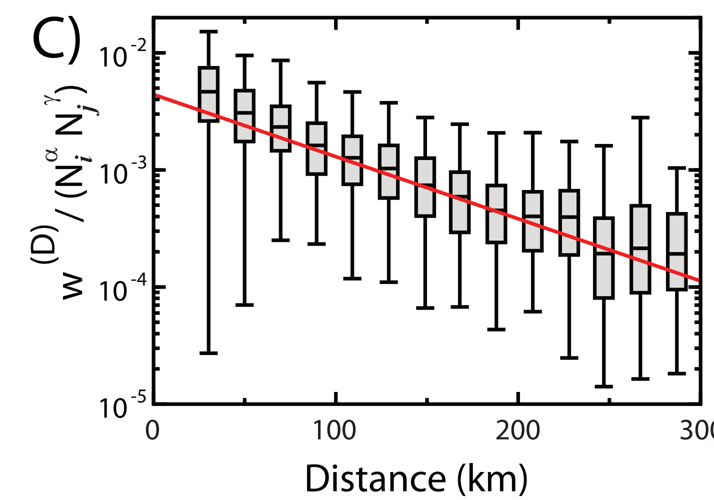
PNAS 106, 21484 (2009)

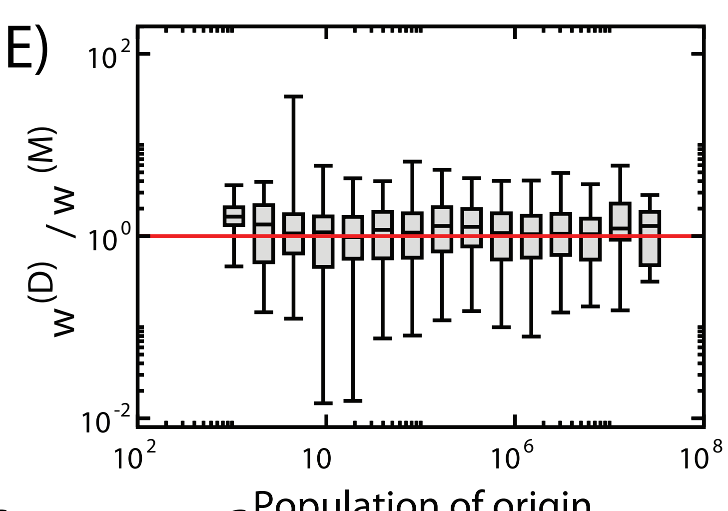
Table 1
Commuting networks in each continent. Number of countries (*N*), number of administrative units (*V*) and inter-links between them (*E*) are summarized.

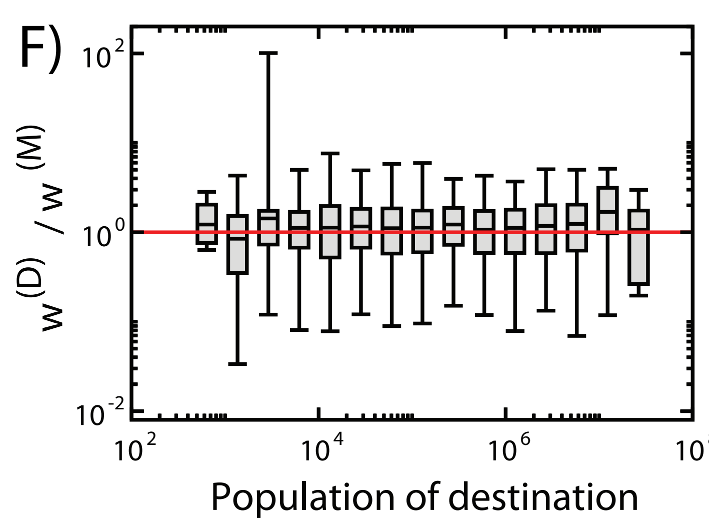
Continent	N	V	E
Europe	17	65,880	4,490,650
North America	2	6986	182,255
Latin America	5	4301	102,117
Asia	4	4355	380,385
Oceania	2	746	30,679
Total	30	82,268	5,186,186

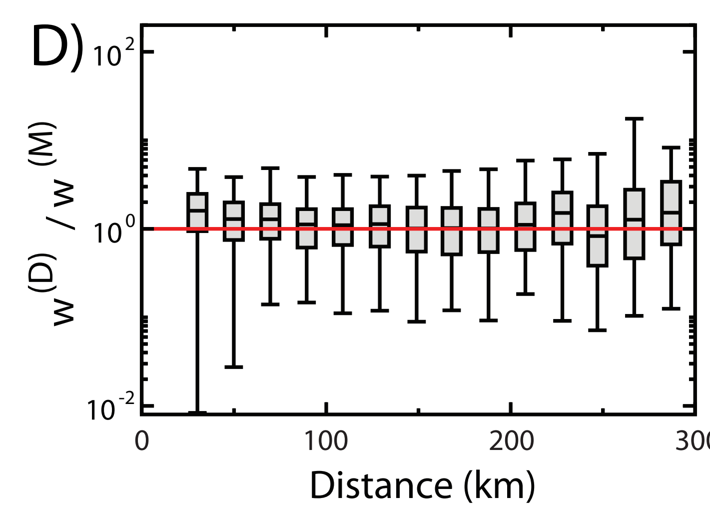
$$w_{ij} = C\, N_i^{\alpha}\, N_j^{\gamma}\, e^{-\beta\, d_{ij}}$$

d (km)	Parameter	Estimate	Standard Error	p-value	R^2
≤ 300	α	0.46	0.01	$< 2E-16$	0.7972
	γ	0.64	0.01	$< 2E-16$	
	β	0.0122	0.0002	$< 2E-16$	
> 300	α	0.35	0.06	$6.91E-09$	0.5369
	γ	0.37	0.06	$2.12E-09$	









@bgoncalves

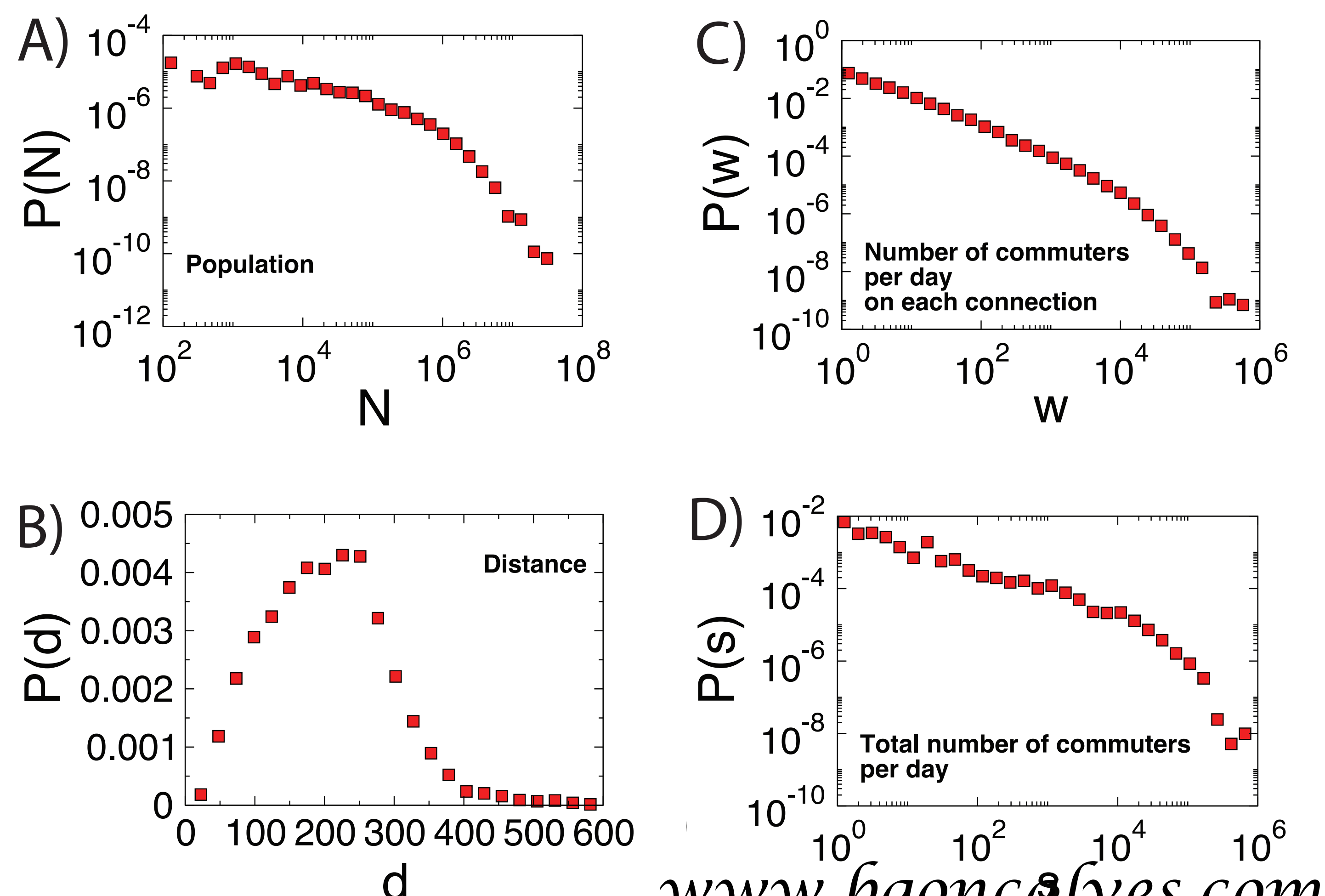
www.bgoncalves.com

Short Range Mobility

PNAS 106, 21484 (2009)

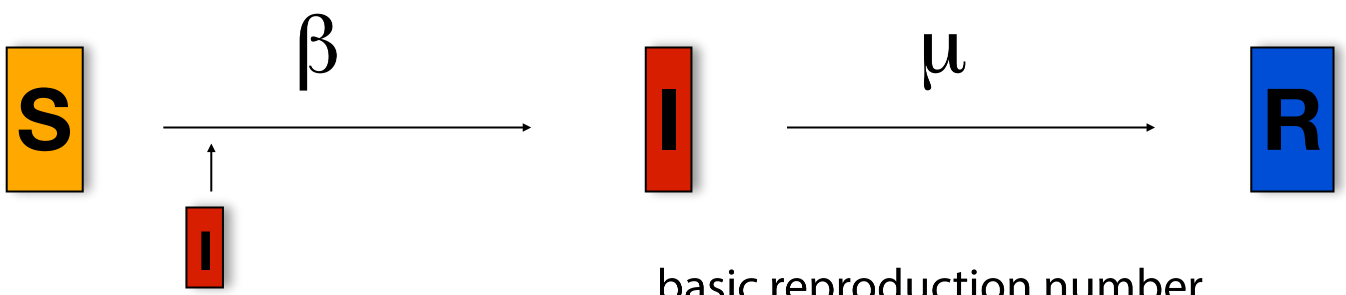
Table 1
Commuting networks in each continent. Number of countries (N), number of administrative units (V) and inter-links between them (E) are summarized.

Continent	N	V	E
Europe	17	65,880	4,490,650
North America	2	6986	182,255
Latin America	5	4301	102,117
Asia	4	4355	380,385
Oceania	2	746	30,679
Total	30	82,268	5,186,186



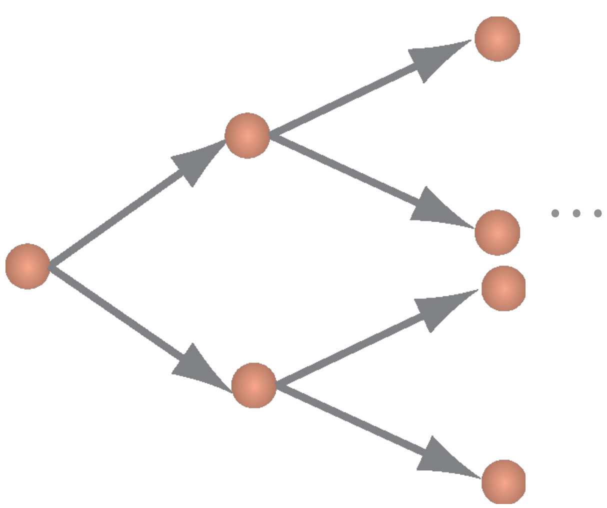
Intra-population: The SIR model

PNAS 106, 21484 (2009)



basic reproduction number

$R_0 > 1$ ⟶ exponential growth

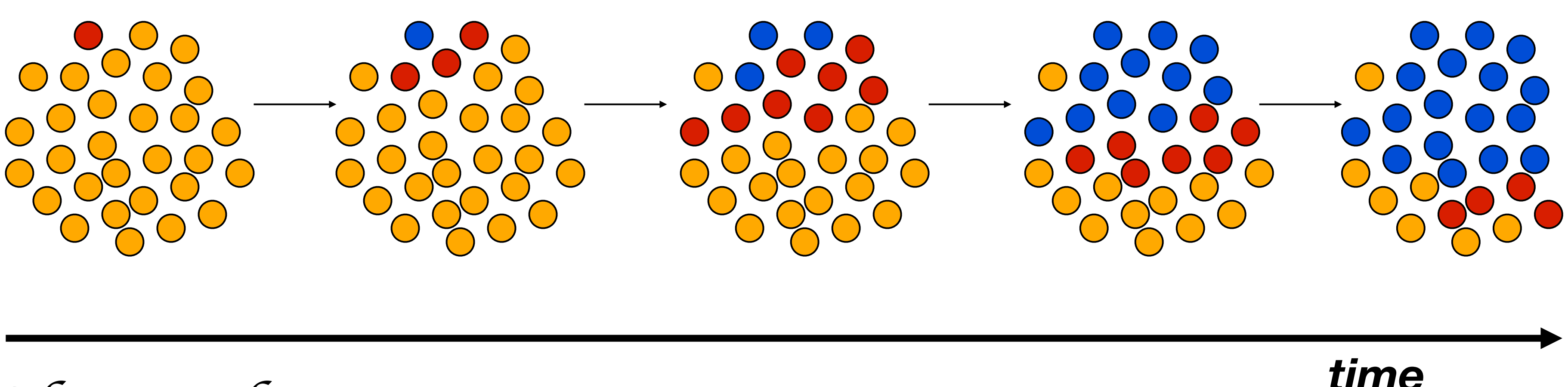


Stochastic coupling terms

$$S_{j,t+\Delta t} = S_{j,t} - Binom_j(S_{j,t}\,,\, \beta\Delta t\, I_{j,t}/N) + \Omega_j(S)$$

$$I_{j,t+\Delta t} = I_{j,t} + Binom_j(S_{j,t}\,,\, \beta\Delta t\, I_{j,t}/N) - Binom_j(I_{j,t}, \mu\Delta t) + \Omega_j(I)$$

$$R_{j,t+\Delta t} = R_{j,t} + Binom_j(I_{j,t}\,,\, \mu\Delta t) + \Omega_j(R)$$

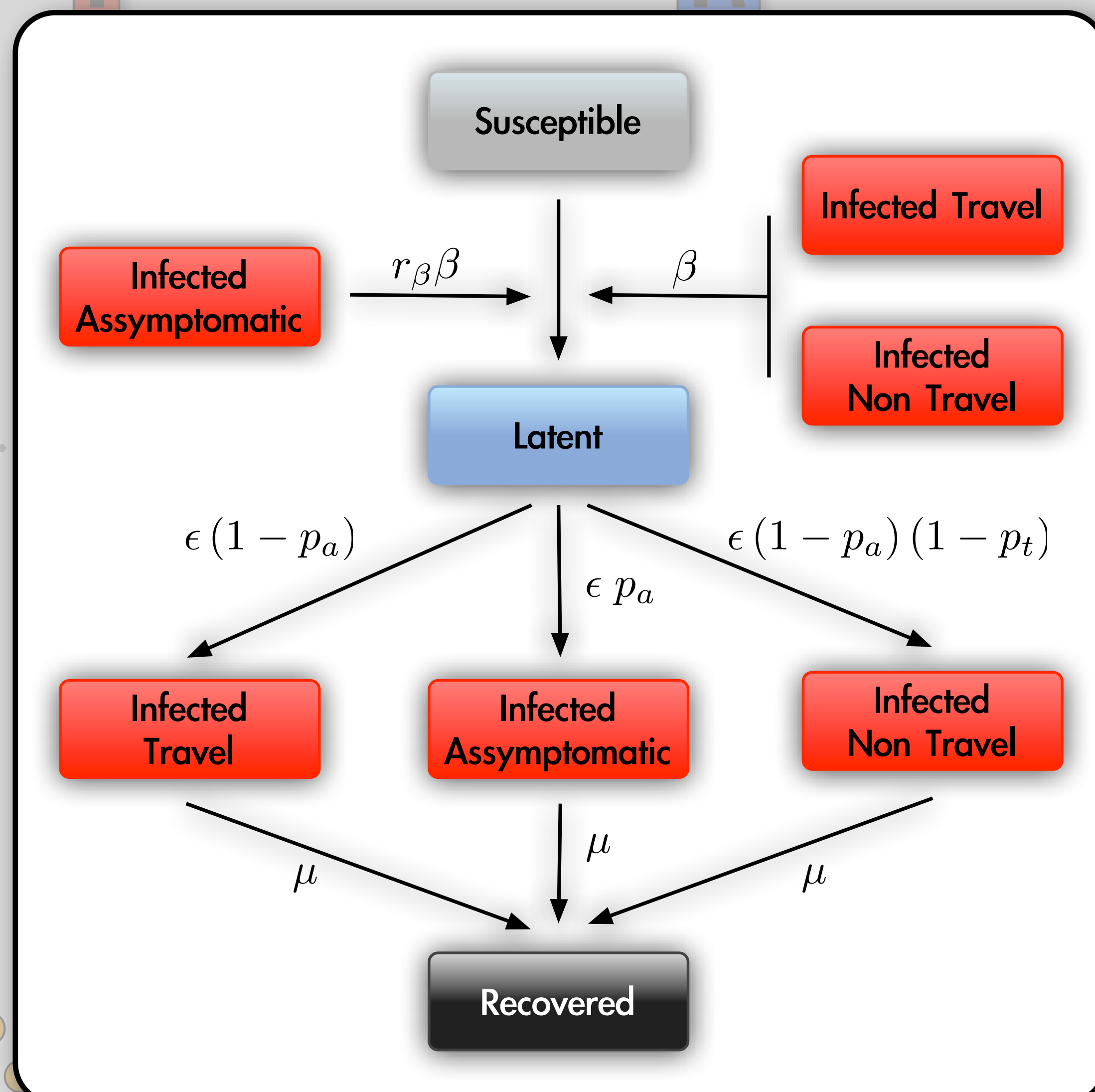


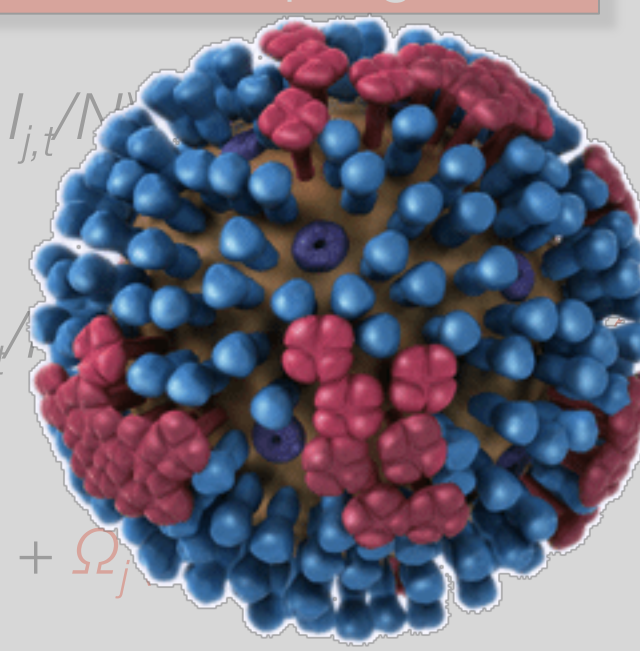
Intra-population: The SIR model

PNAS 106, 21484 (2009)

S $\xrightarrow{\beta}$ I $\xrightarrow{\mu}$ R

Stochastic coupling terms

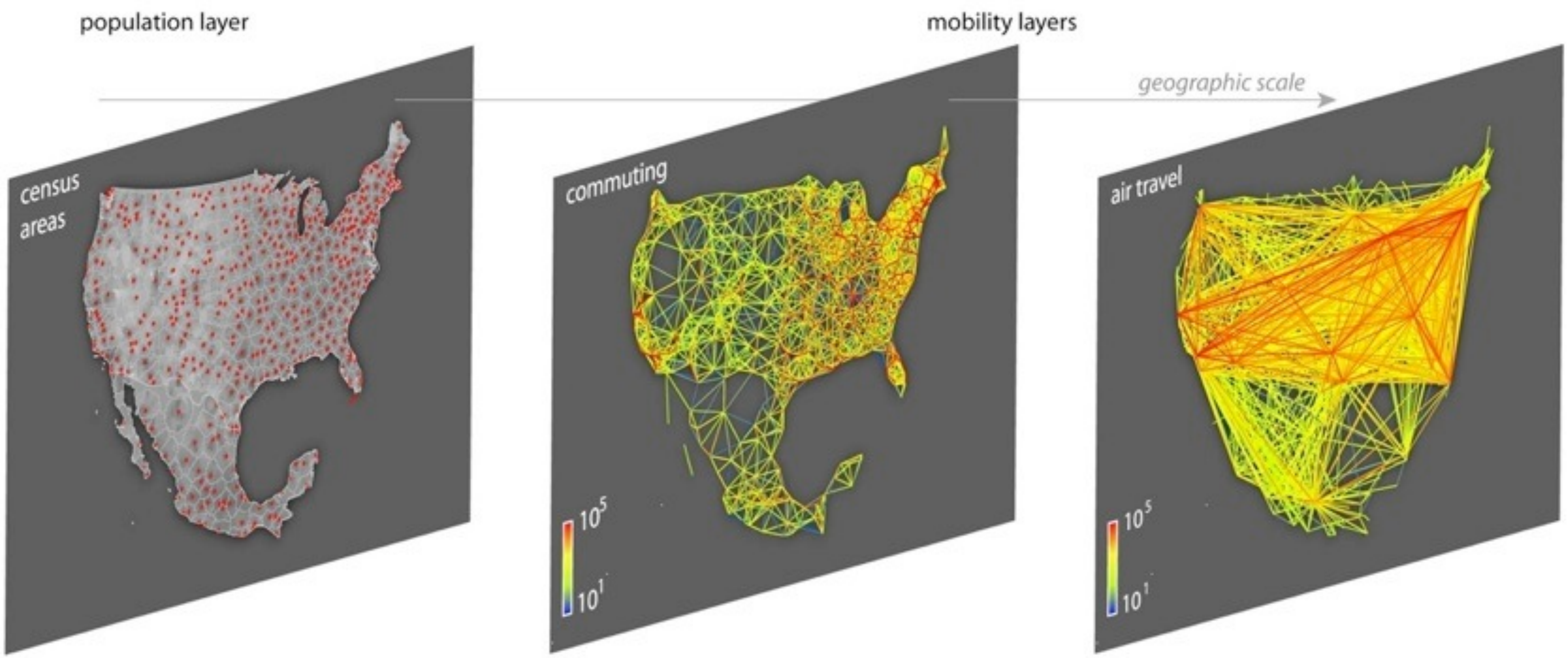


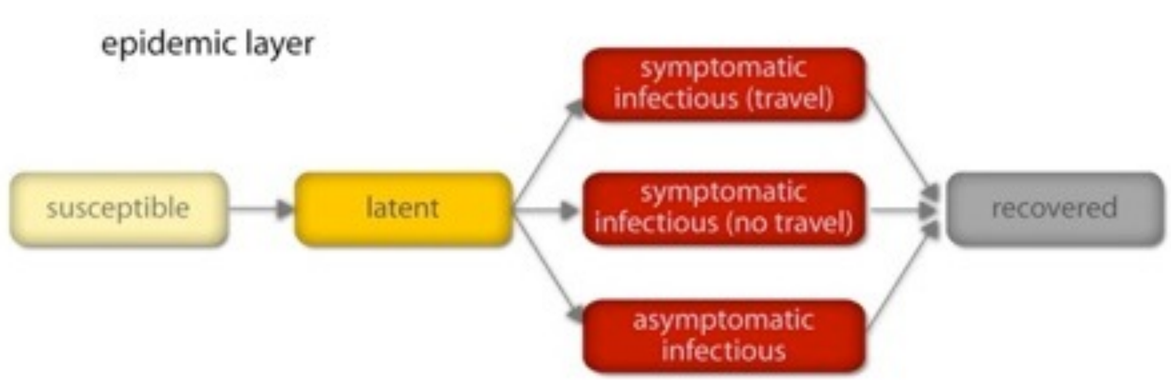


20,000 -120,000 coupled stochastic equations

time

Global Epidemic and Mobility Modeller Platform

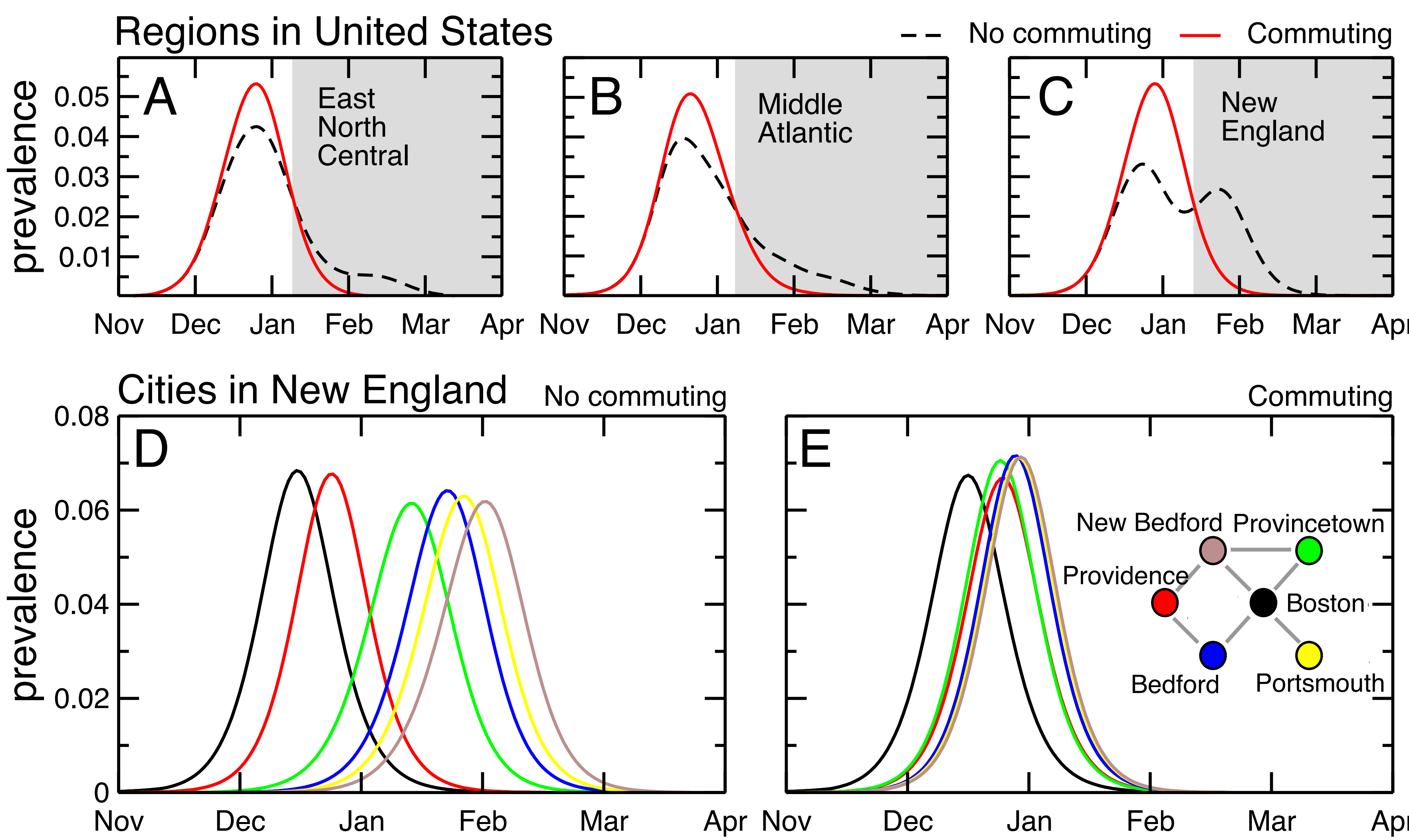




Parameter	Value	Description
β	from R_0	transmission probability
ε^{-1}	1.9 [1.1-2.5] d	average latency period
μ^{-1}	3 [3-5] d	average infectious period
p_t	50%	probability of traveling for infectious individuals
p_a	33%	probability of being asymptomatic
r_β	50%	relative infectiousness of asymptomatic infectious individuals

Effect of Commuting

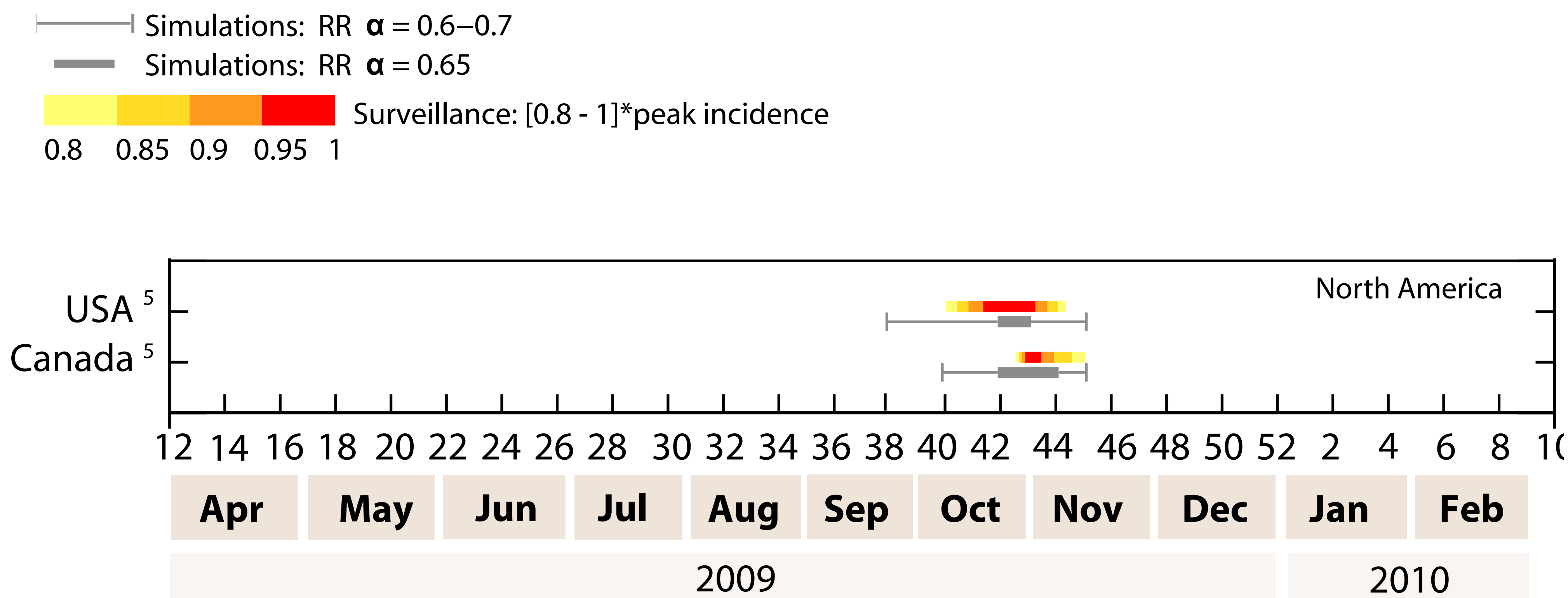
PNAS 106, 21484 (2009)



Invasion Tree

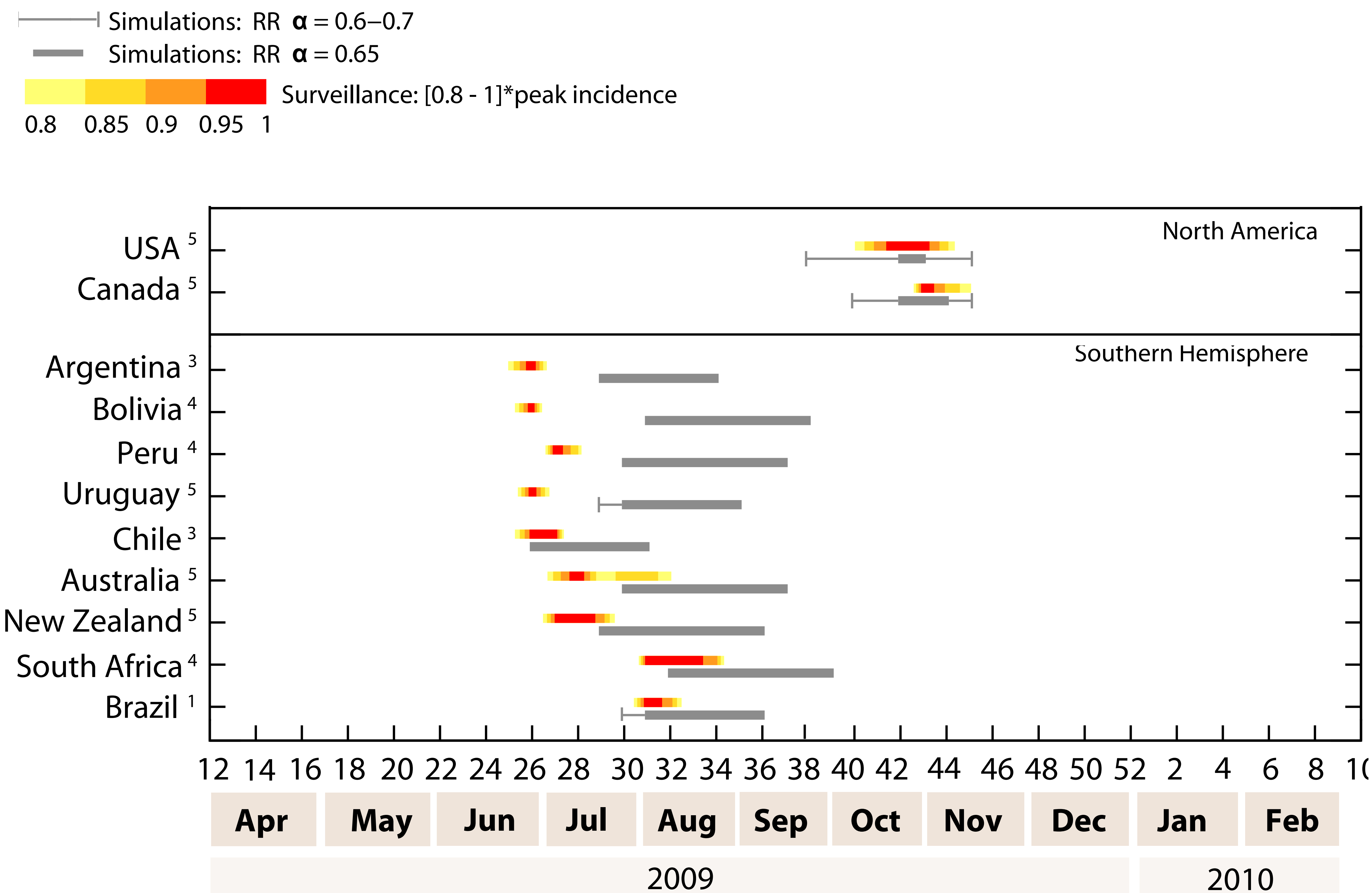
Temporal Evolution

BMC Medicine 10, 165 (2012)



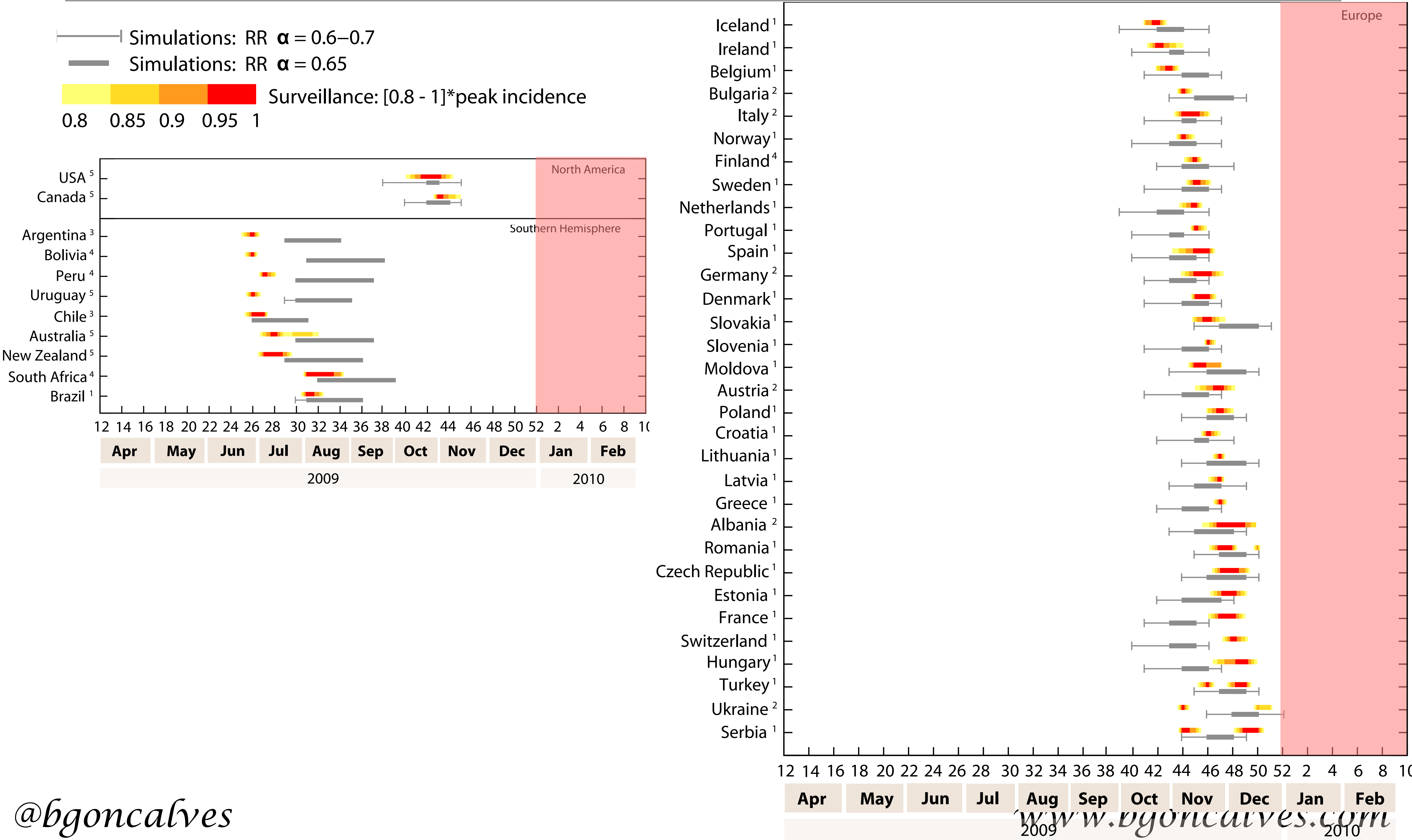
Temporal Evolution

BMC Medicine 10, 165 (2012)

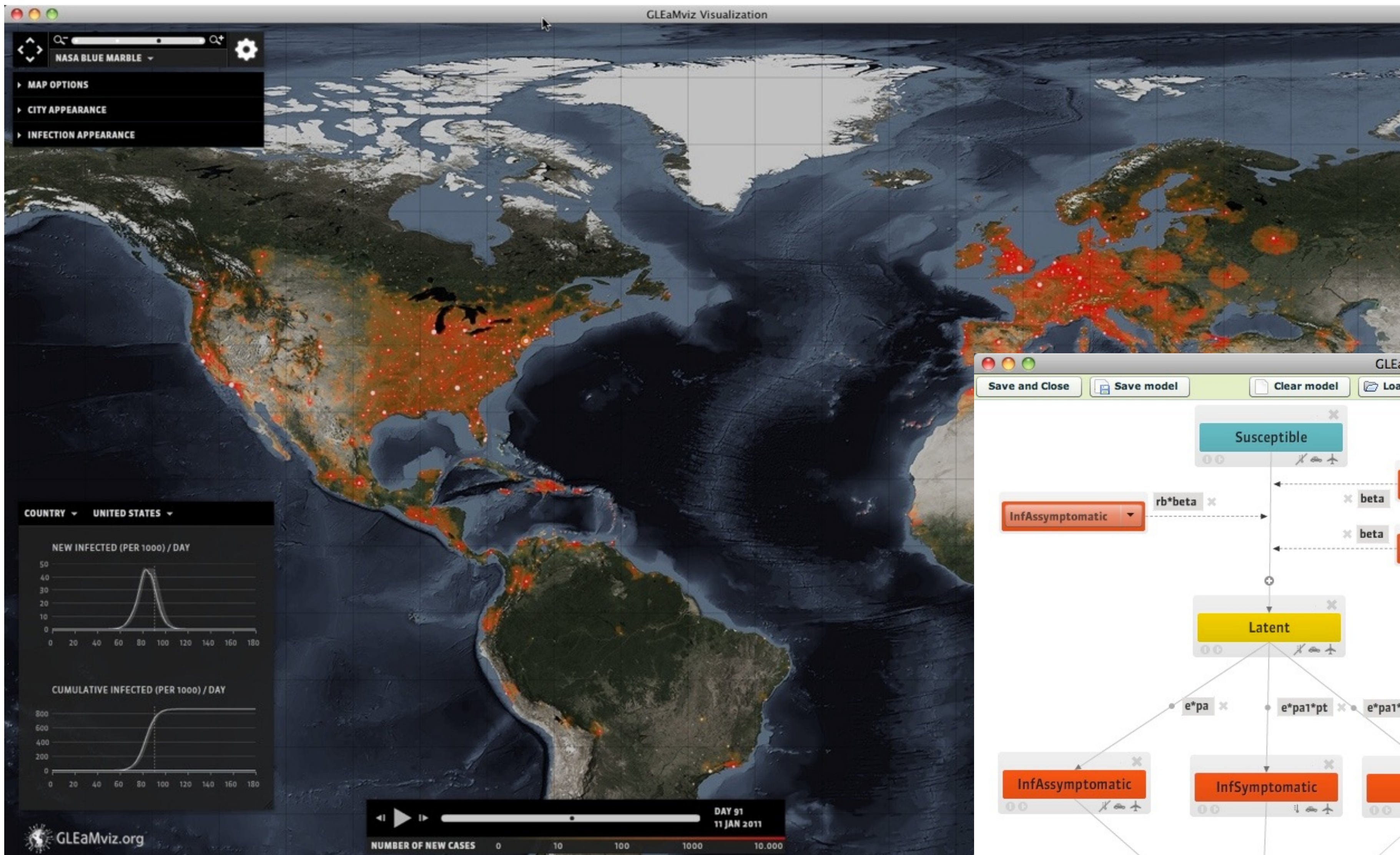


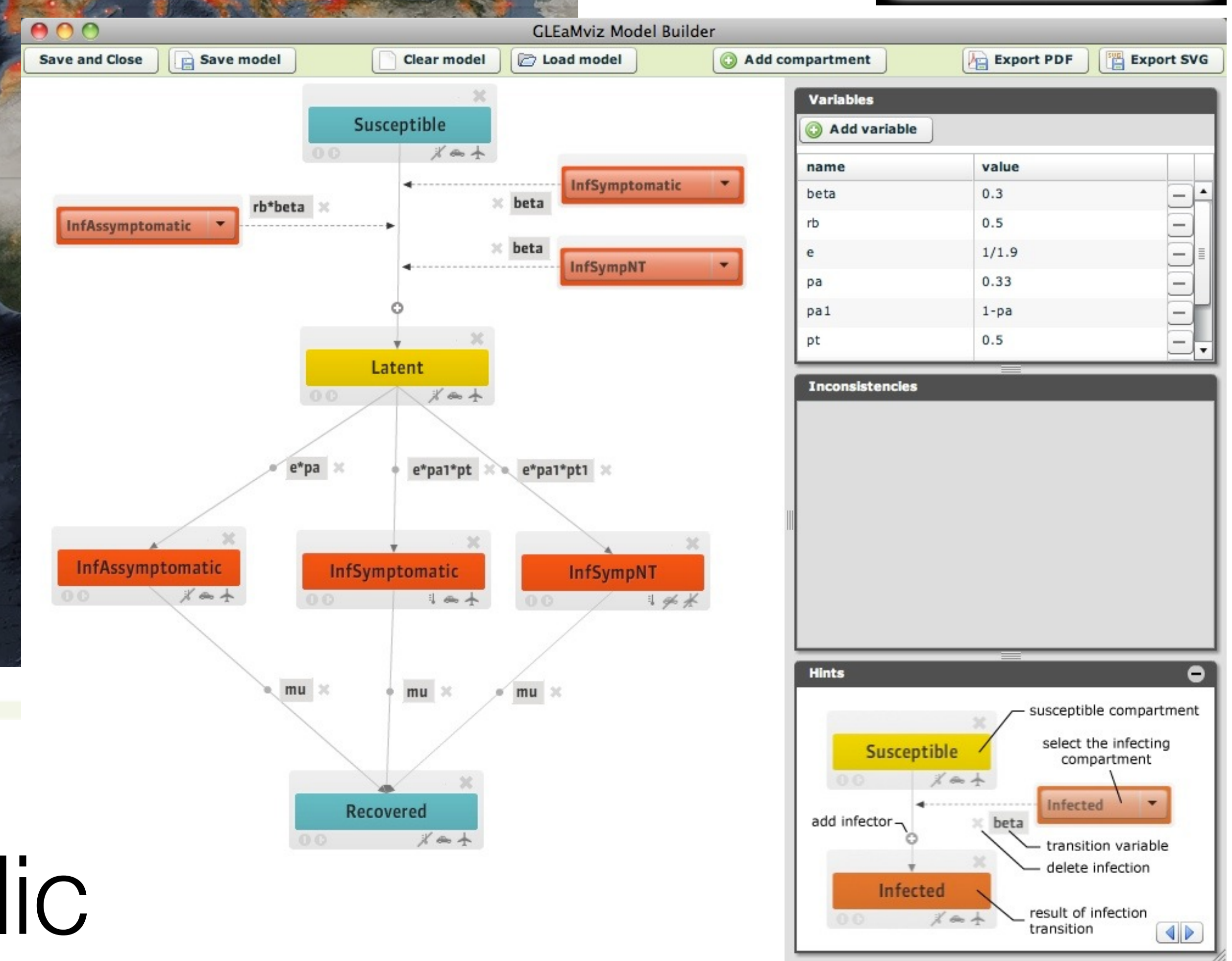
Temporal Evolution

BMC Medicine 10, 165 (2012)



GLEaMviz.org





Freely available to the public

Data (For) Science

by dataforscience

An irregular update on all things Data aimed at Scientists in other fields.

email@example.com

SUBSCRIBE

TinyLetter's Privacy Policy

http://tinyletter.com/dataforscience