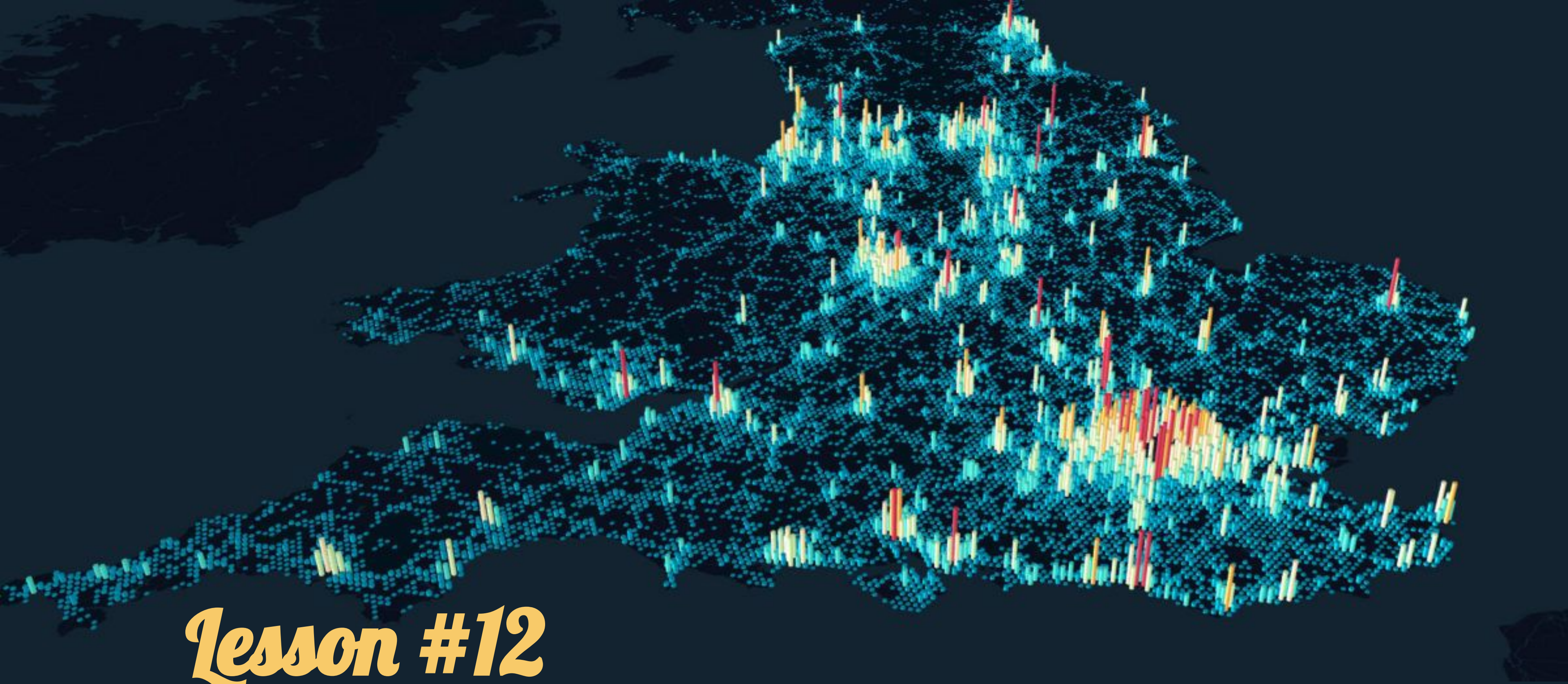


Lesson #12

Storytelling from geographic data

April 2019

Update from repository

```
git clone https://github.com/ivanovitchm/datascience_one_2019_1
```

Or ….

```
git pull
```

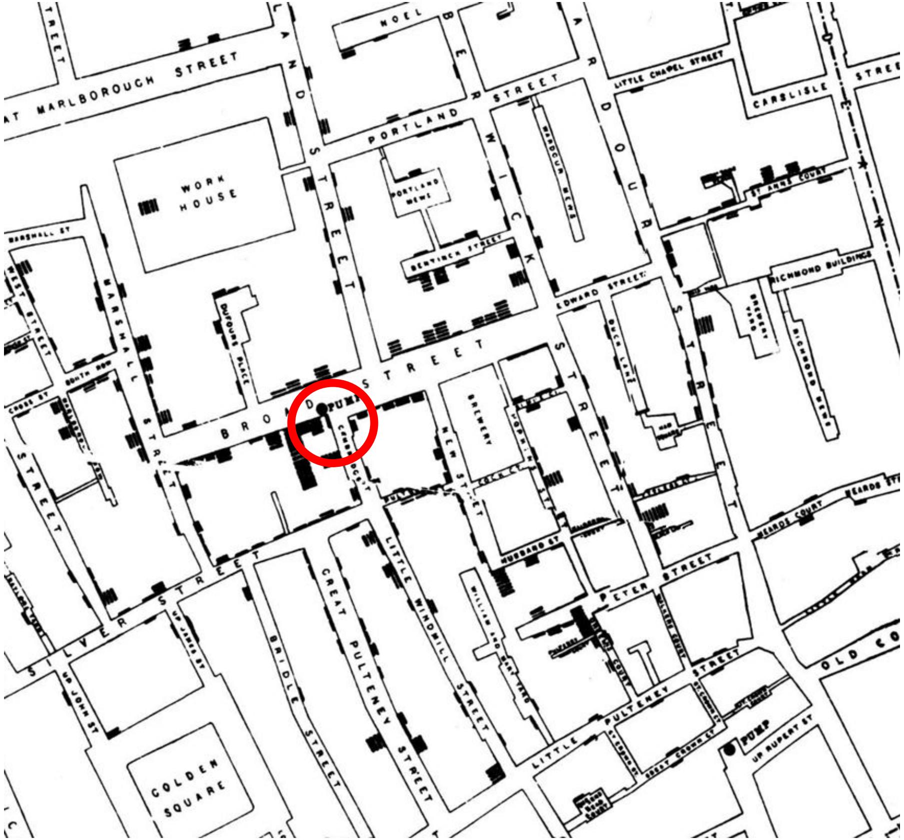
motivation

Cholera Outbreak 1854

Maps Save Lives

DEATH'S DISPENSARY.

OPEN TO THE POOR, GRATIS, BY PERMISSION OF THE PARISH.

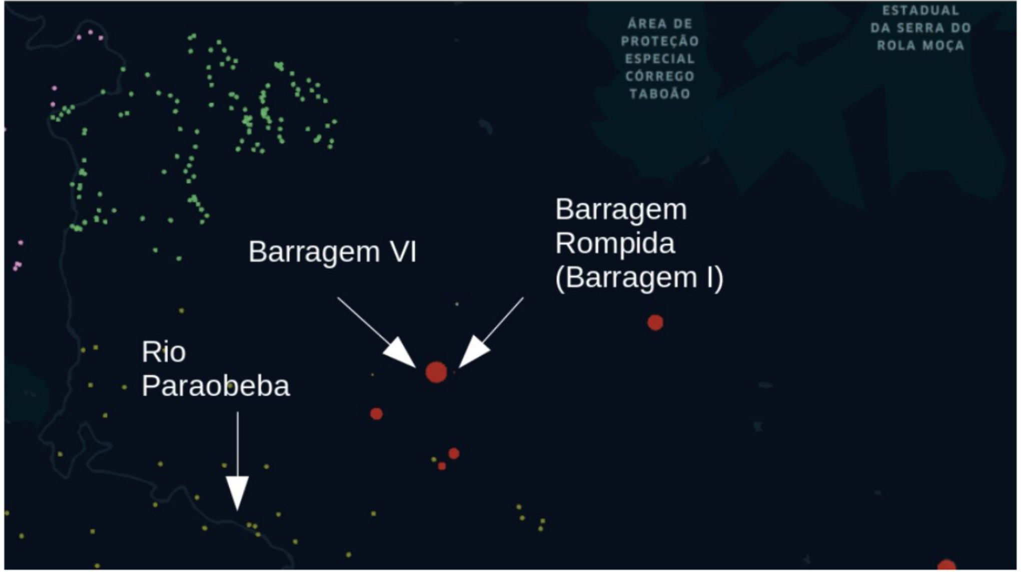


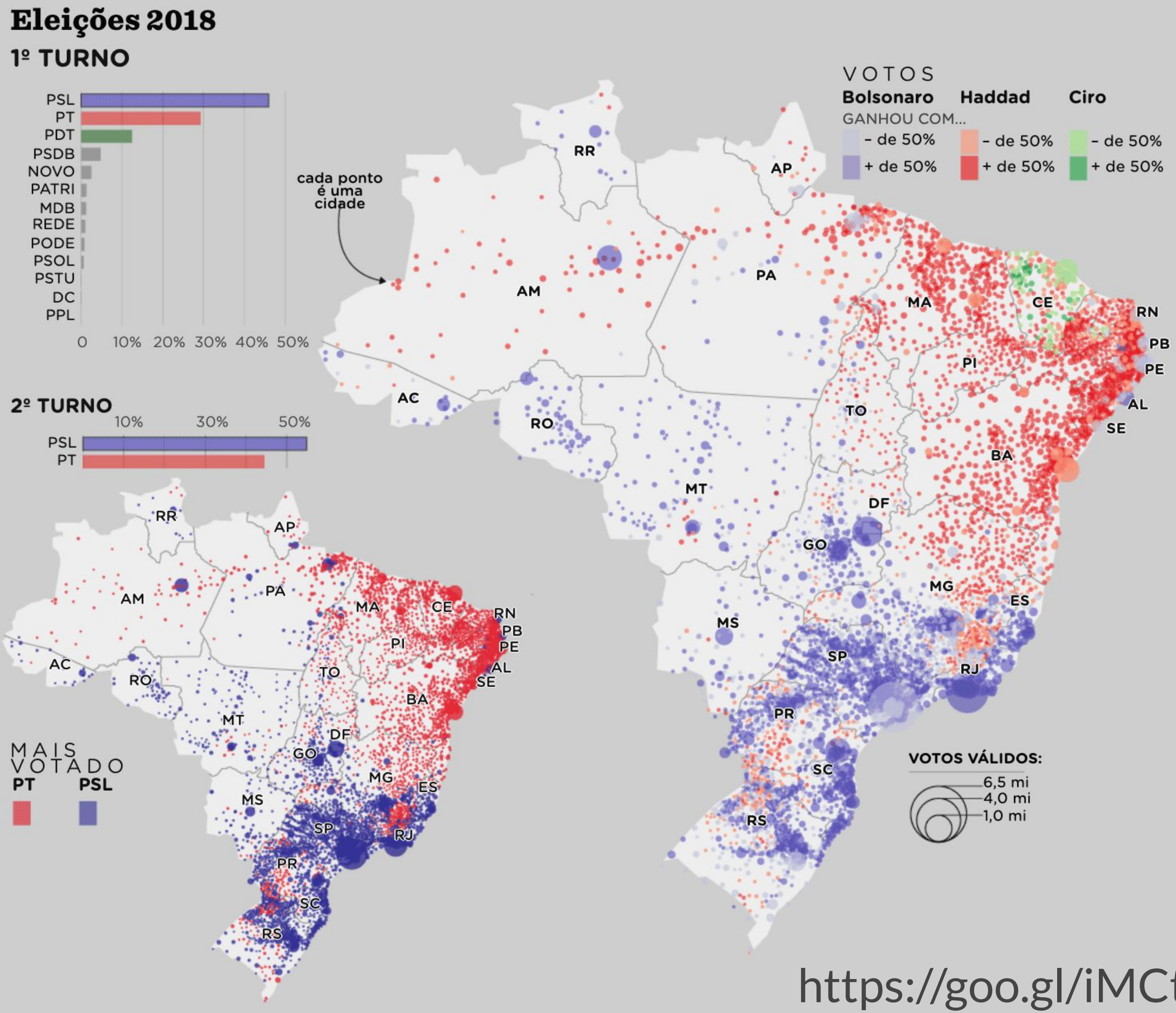
Tejuco

#data4good: Ciência de dados vs Riscos das barragens 🇧🇷

RICARDO CAPPRA

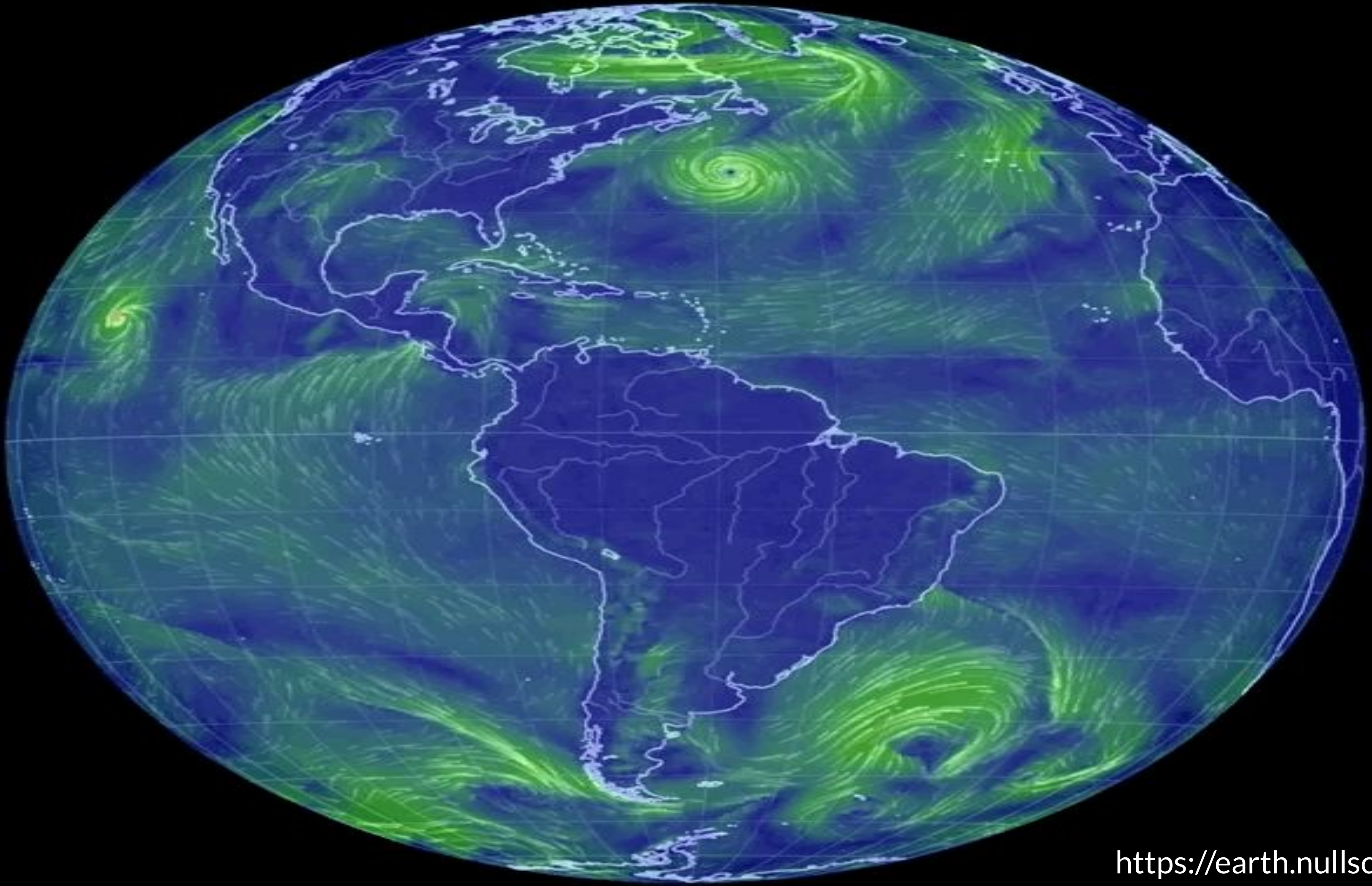
https://cappra.com.br/2019/02/19/ciencia-dados-barragens/


ÁREA DE PROTEÇÃO ESPECIAL CÓRREGO TABOÃO
ESTADUAL DA SERRA DO ROLA MOÇA
Barragem VI
Barragem Rompida (Barragem I)
Rio Paraobeba



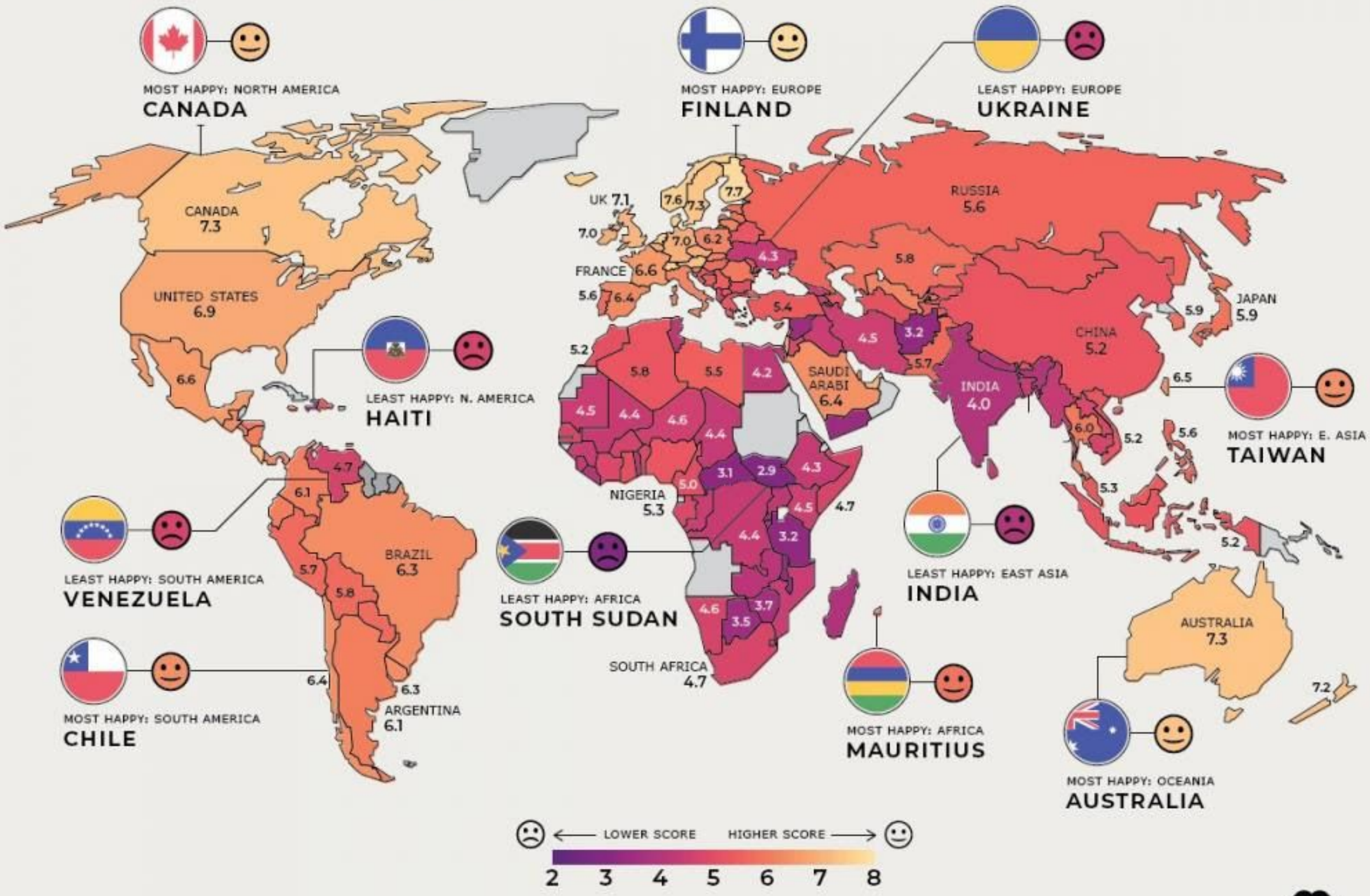
Geographic data is always present in our everyday lives

https://goo.gl/iMCthG



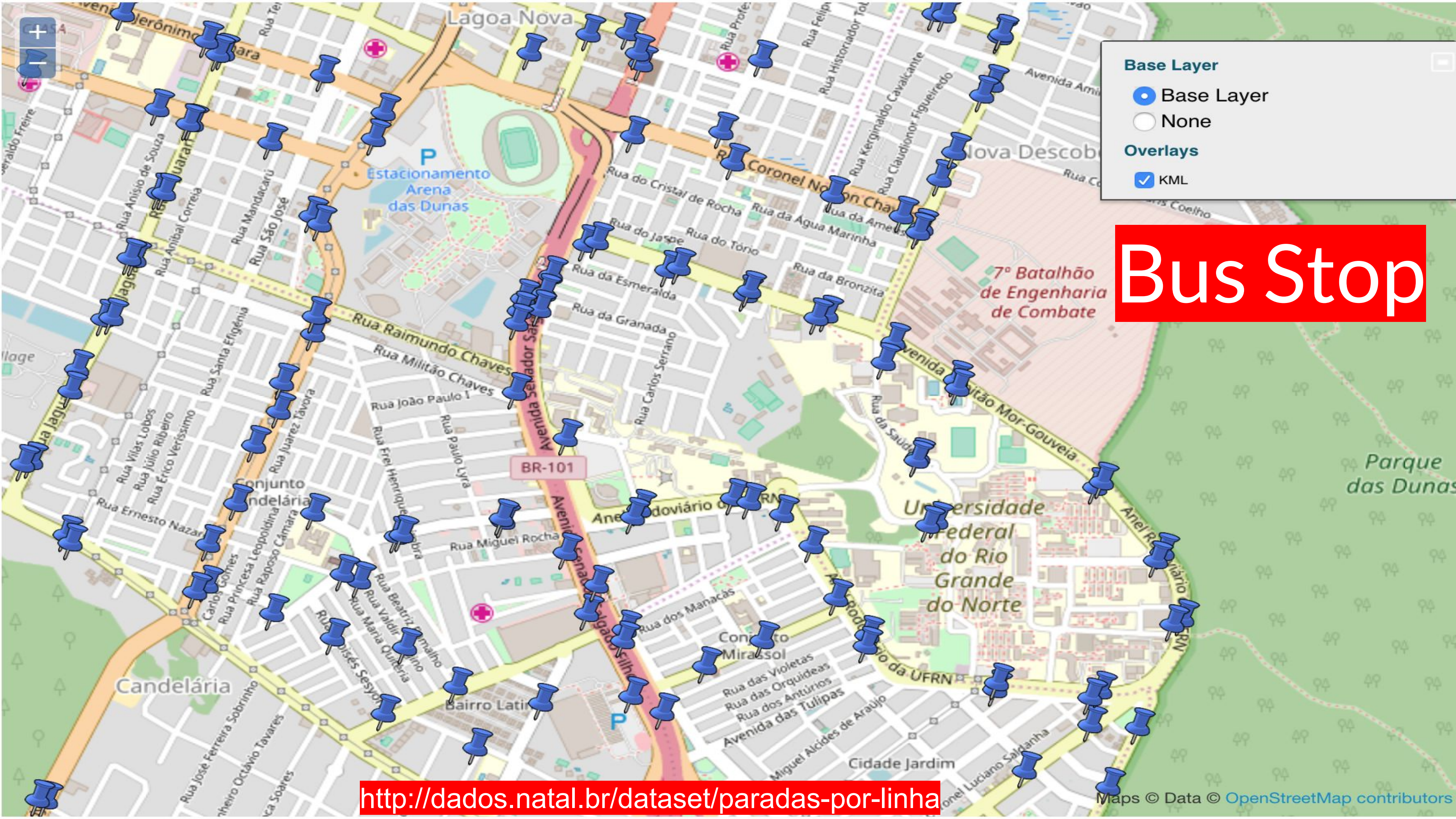
https://earth.nullschool.net/

THE MOST AND LEAST HAPPY COUNTRIES AROUND THE WORLD



SOURCE: World Happiness Report 2019

visualcapitalist.com



Base Layer

● Base Layer

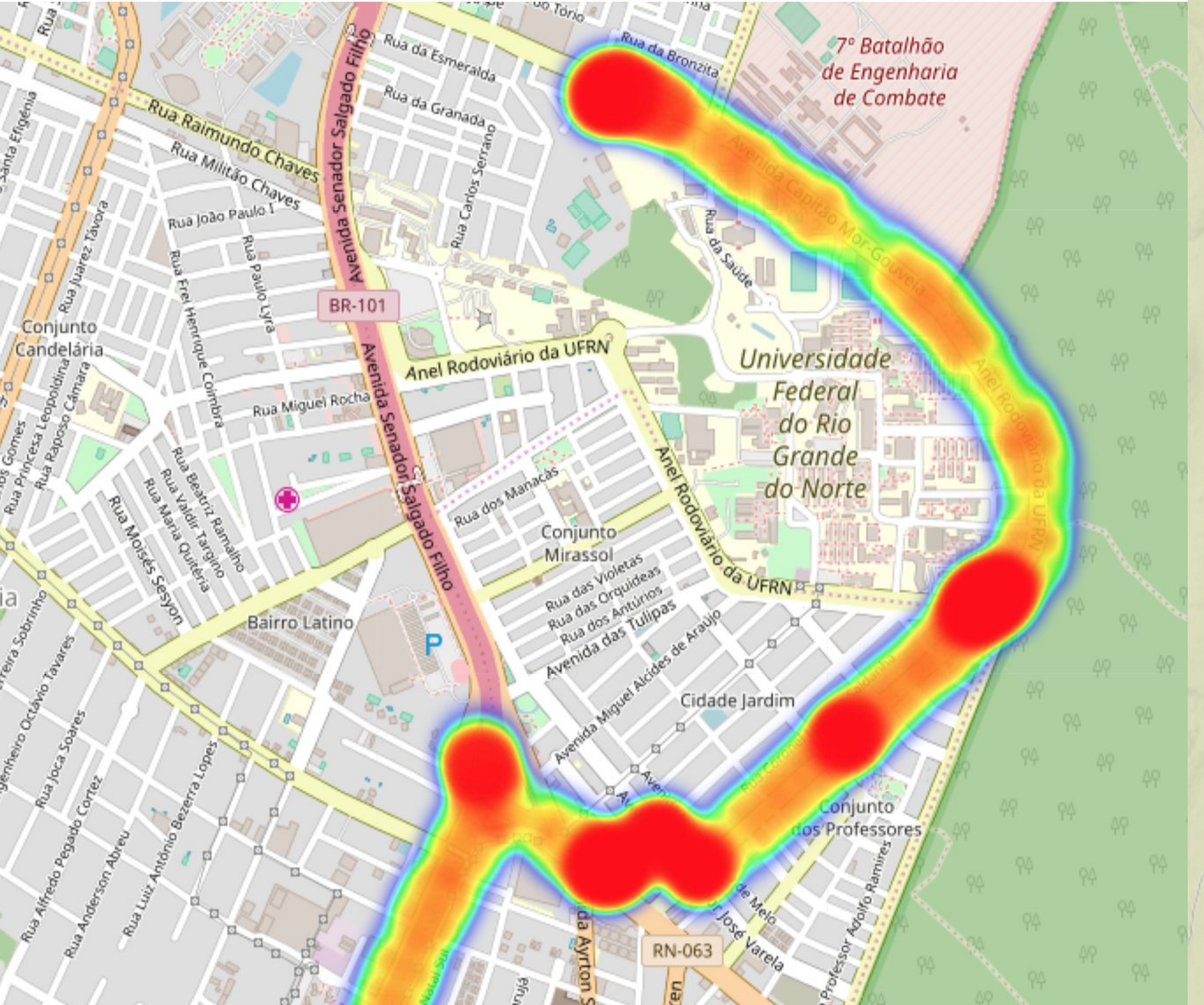
○ None

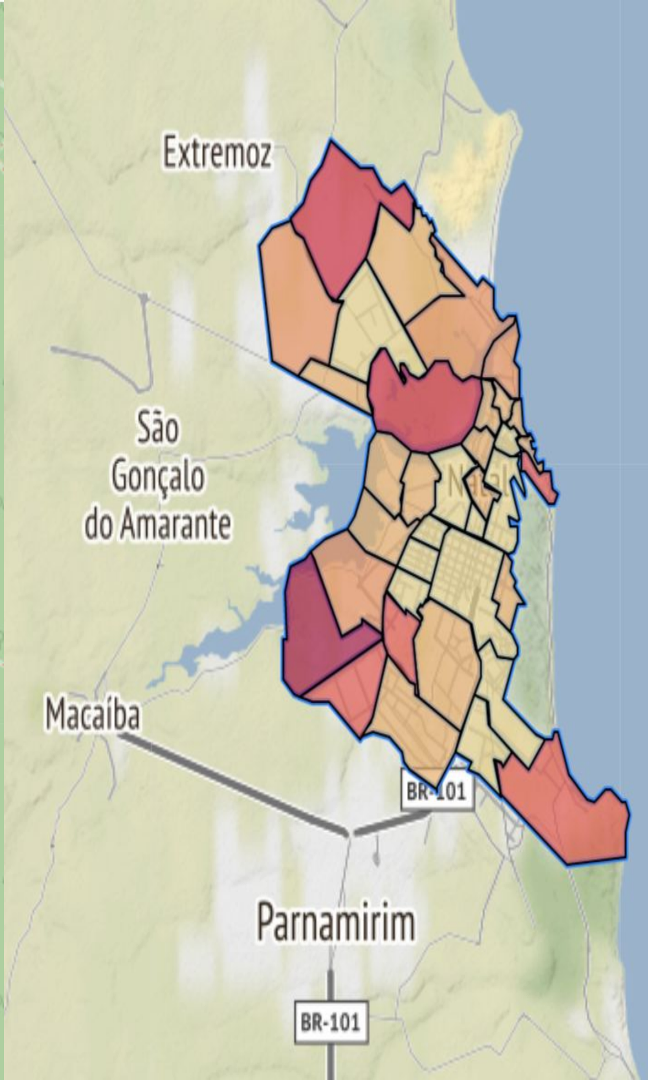
Overlays

☑ KML

Bus Stop

http://dados.natal.br/dataset/paradas-por-linha


7º Batalhão
de Engenharia
de Combate
Universidade
Federal
do Rio
Grande
do Norte
Rua da Bronzita
Rua da Esmeralda
Rua da Granada
Rua Carlos Serrano
Rua Raimundo Chaves
Rua Militão Chaves
Rua João Paulo I
Rua Paulo Lyra
Rua Frei Henrique Coimbra
Avenida Senador Salgado Filho
BR-101
Avenida Capitão Mor Gouveia
Rua da Saúde
Anel Rodoviário da UFRN
Conjunto
Candelária
Rua Miguel Rocha
Rua dos Manacás
Conjunto
Mirassol
Rua das Violetas
Rua das Orquídeas
Rua dos Antúrios
Avenida das Tulipas
Avenida Miguel Alcides de Araújo
Bairro Latino
Cidade Jardim
Conjunto
dos Professores
RN-063
Rua Professor Adolfo Ramires
Rua Beatriz Ramalho
Rua Valdir Targino
Rua Maria Quitéria
Rua Moisés Sesyon
Rua Juarez Távora
Rua Raposo Câmara
Rua Princesa Leopoldina
Rua Santa Efigênia
Rua Joca Soares
Rua Alfredo Pegado Cortez
Rua Anderson Abreu
Rua Luiz Antônio Bezerra Lopes
Engenheiro Octávio Tavares
P


Extremoz
São
Gonçalo
do Amarante
Natal
Macaíba
Parnamirim
BR-101
BR-101

Raw geographic data like latitudes and longitudes are difficult to understand using the data charts and plots we've discussed so far

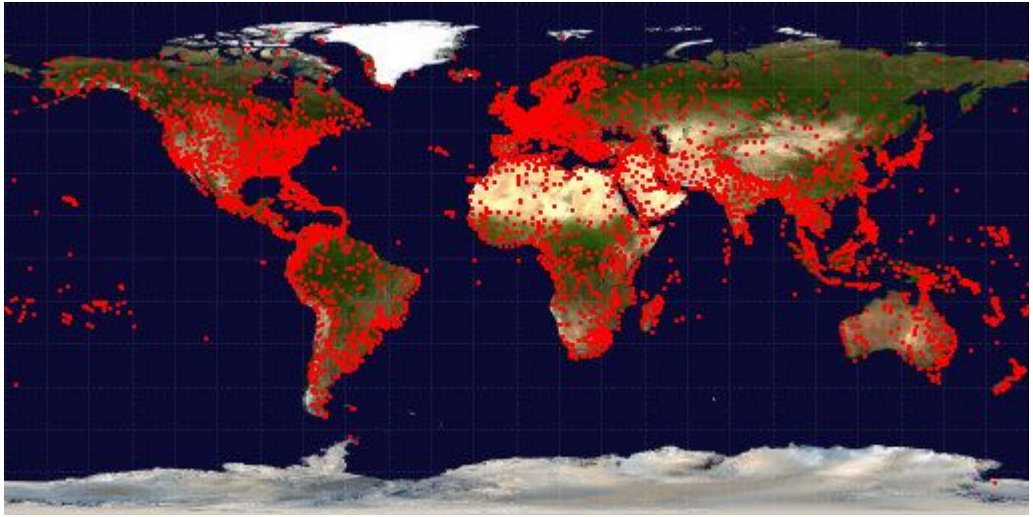
- Case study: open flights
- Geographic coordinate system: problems
- Basemap toolkit

LEARNING TODAY

Geographic dataset

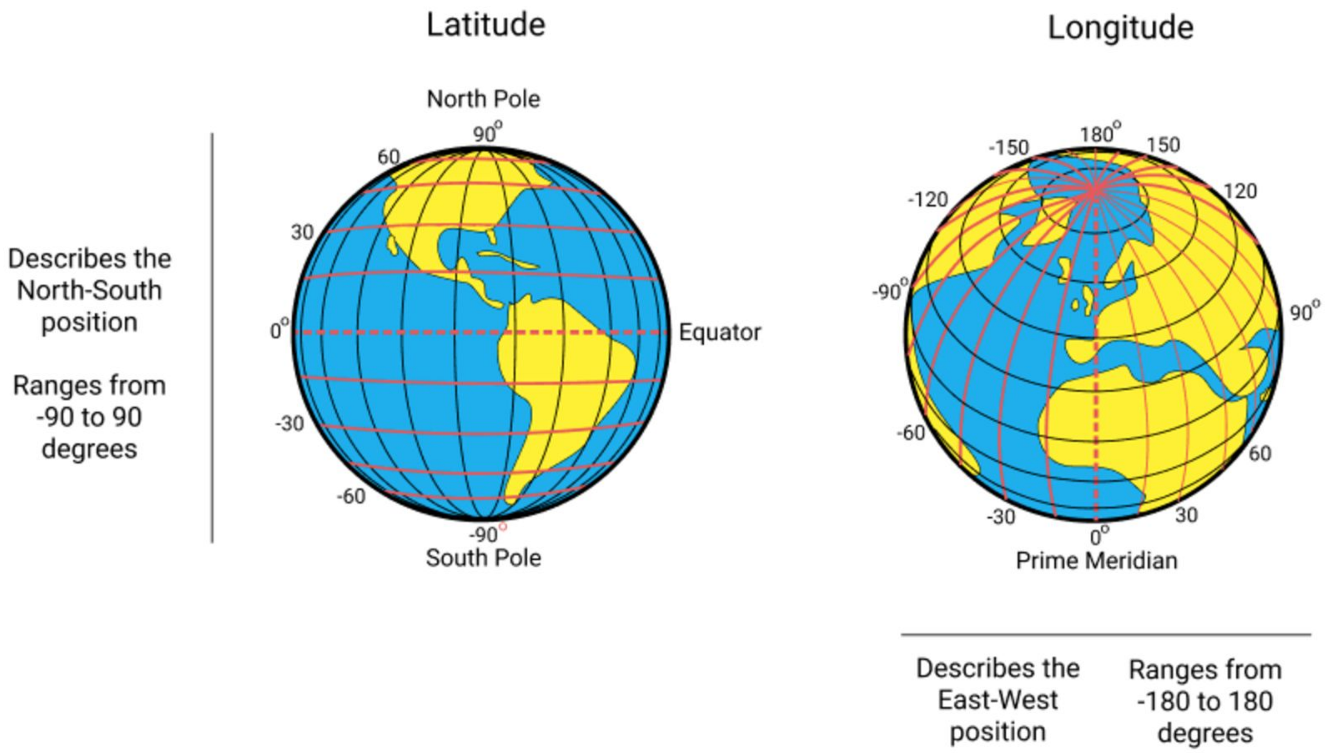
Airport, airline and route data

- airlines.csv - data on each airline.
  - country - where the airline is headquartered.
  - active - if the airline is still active.
- airports.csv - data on each airport.
  - name - name of the airport.
  - city - the airport is located.
  - country - country the airport is located.
  - code - unique airport code.
  - latitude - latitude value.
  - longitude - longitude value.
- routes.csv - data on each flight route.
  - airline - airline for the route.
  - source - starting city for the route.
  - dest - destination city for the route.



https://openflights.org/data.html

Geographic coordinate system



Latitude

Describes the North-South position

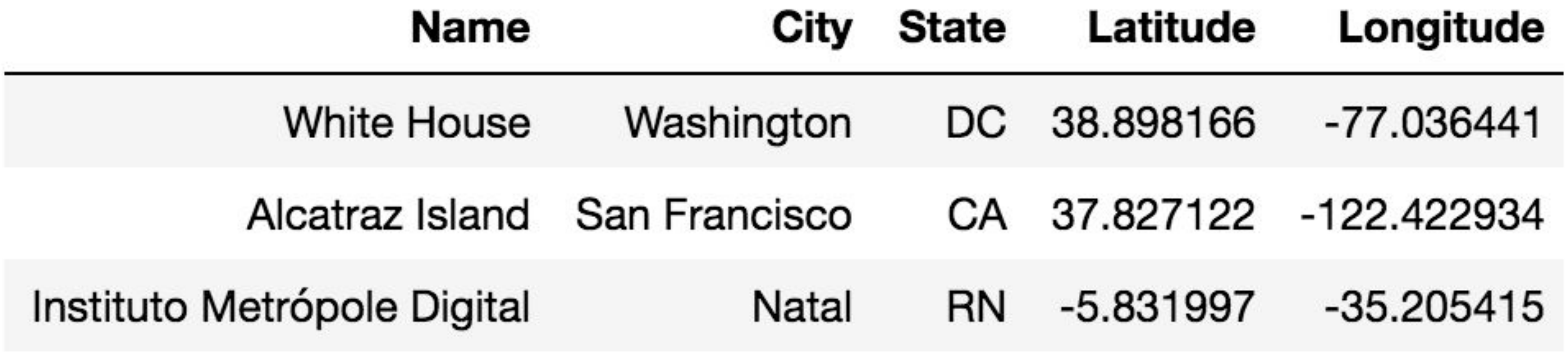
Ranges from -90 to 90 degrees

Longitude

Describes the East-West position

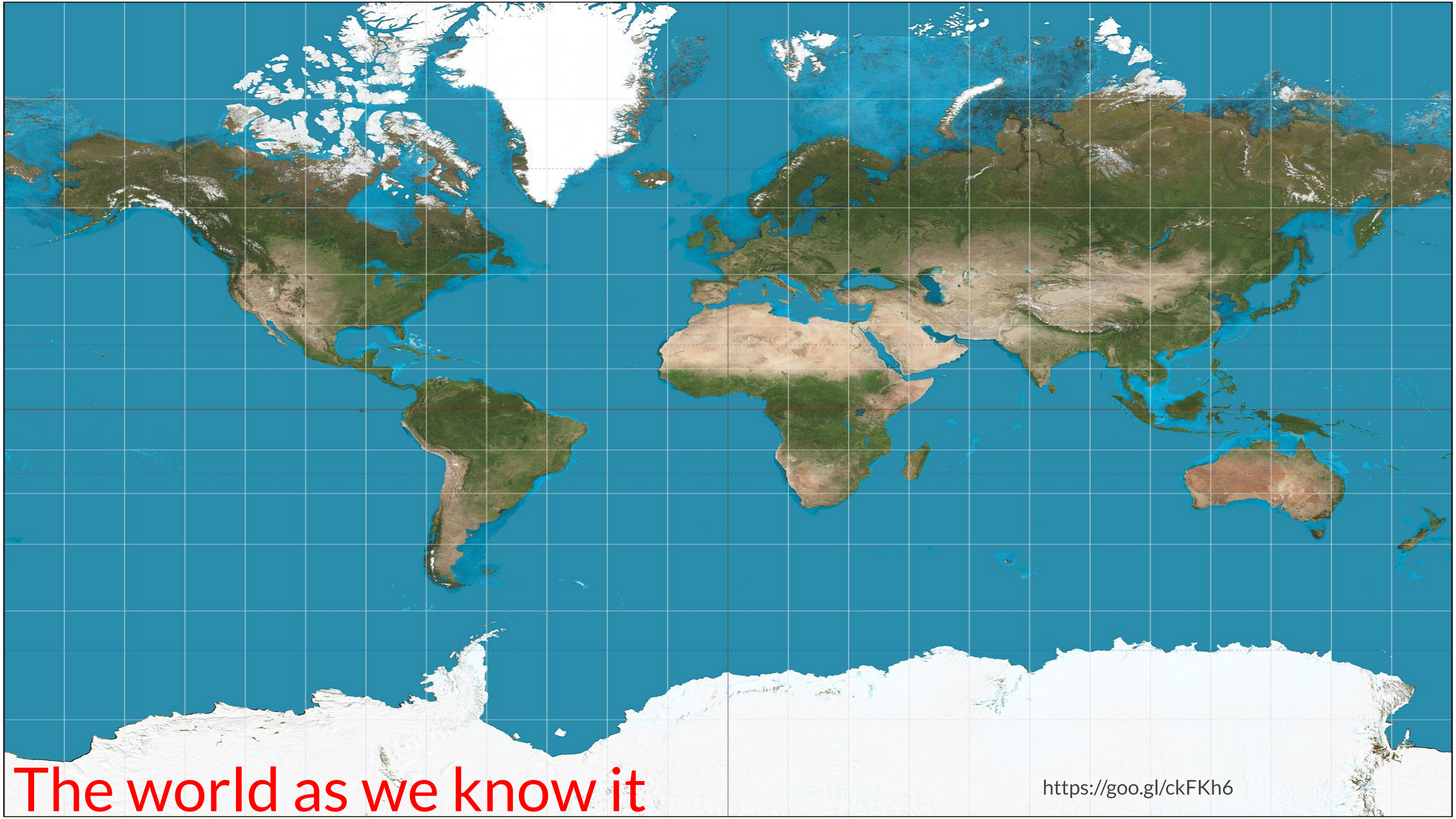
Ranges from -180 to 180 degrees

Geographic coordinate system



Name	City	State	Latitude	Longitude
White House	Washington	DC	38.898166	-77.036441
Alcatraz Island	San Francisco	CA	37.827122	-122.422934
Instituto Metrópole Digital	Natal	RN	-5.831997	-35.205415

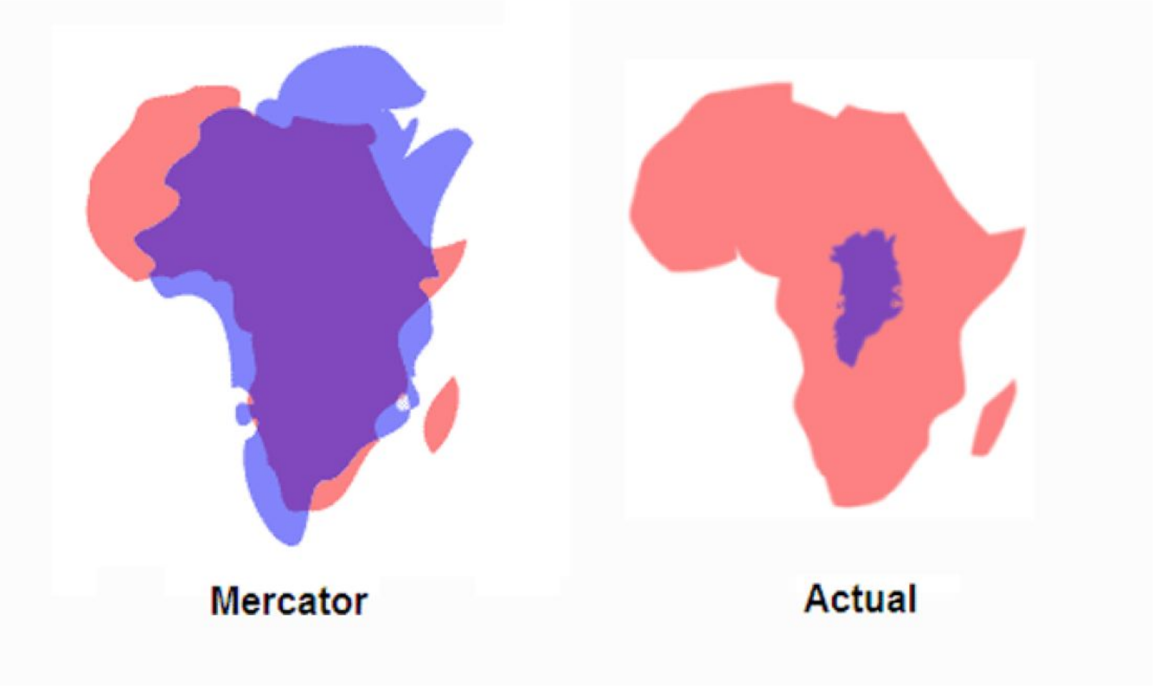
the problem with maps

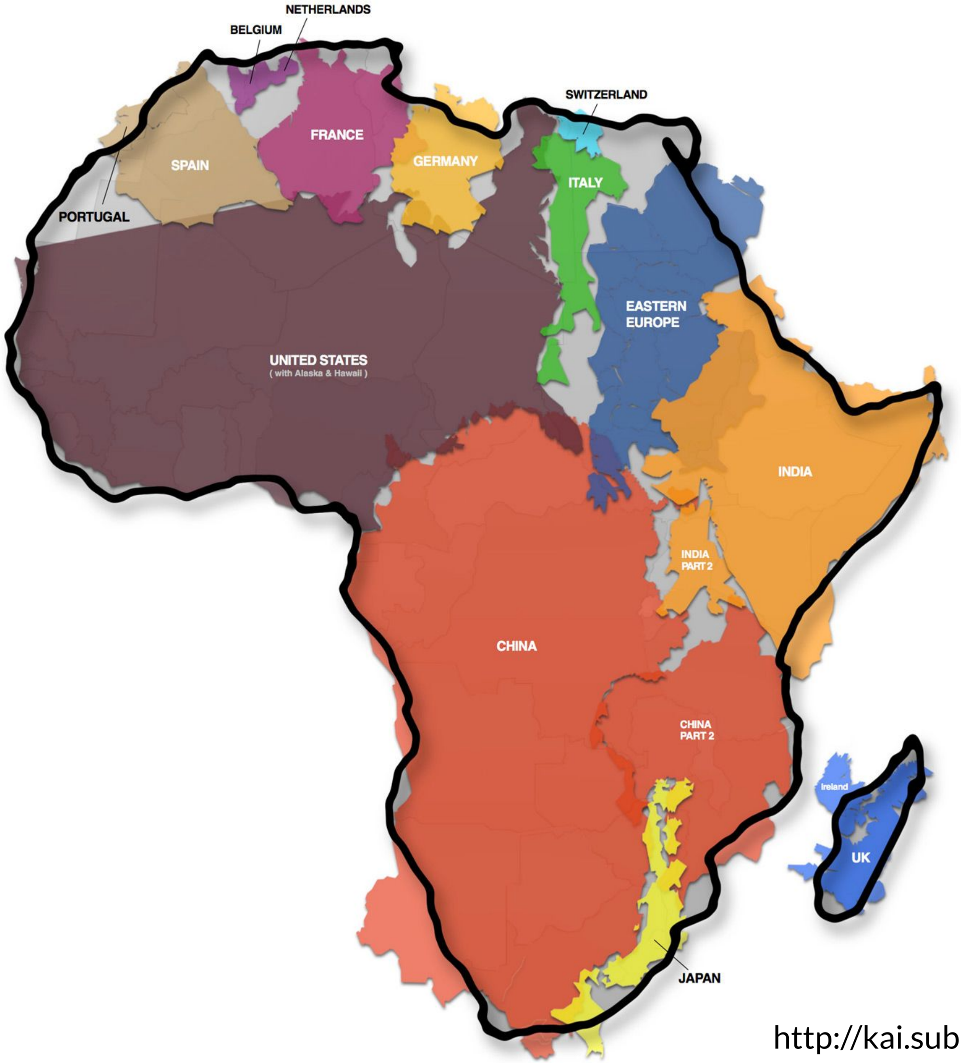


The world as we know it

https://goo.gl/ckFKh6

Greenland is no Africa





The true size of Africa

http://kai.sub.blue/en/africa.html

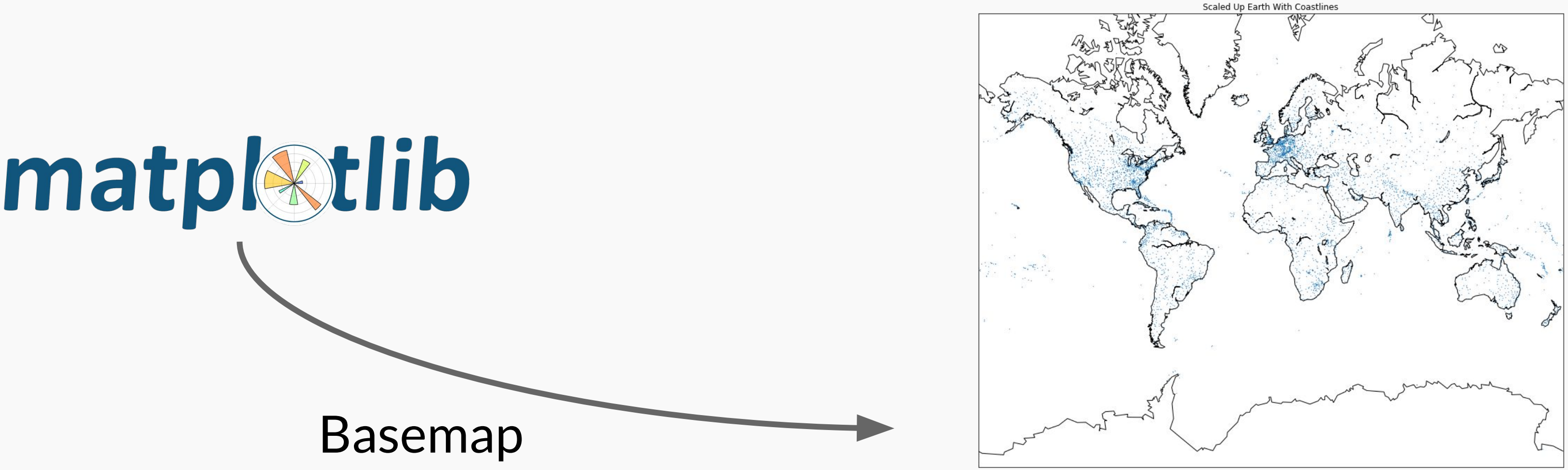
Map Projections

Two types of maps:

- Reference: accuracy is the most important
- Thematic: the data, i.e., getting the story right is the most important

Basemap Toolkit

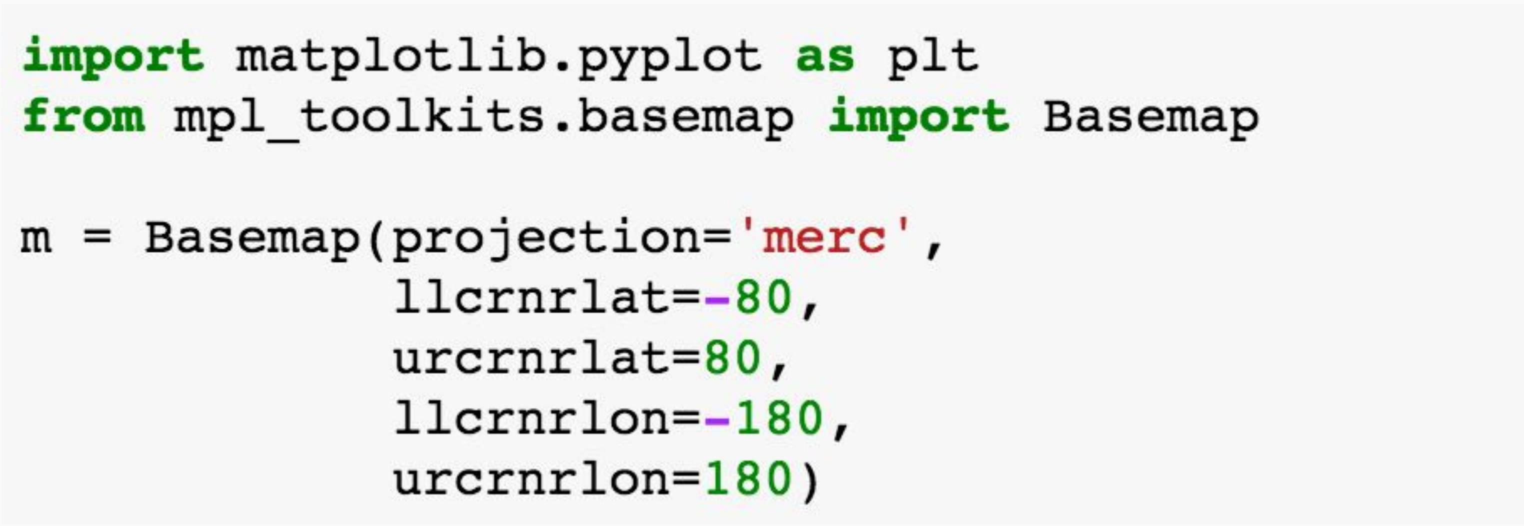
Basemap is an extension to Matplotlib that makes it easier to work with geographic data



matplotlib

Basemap

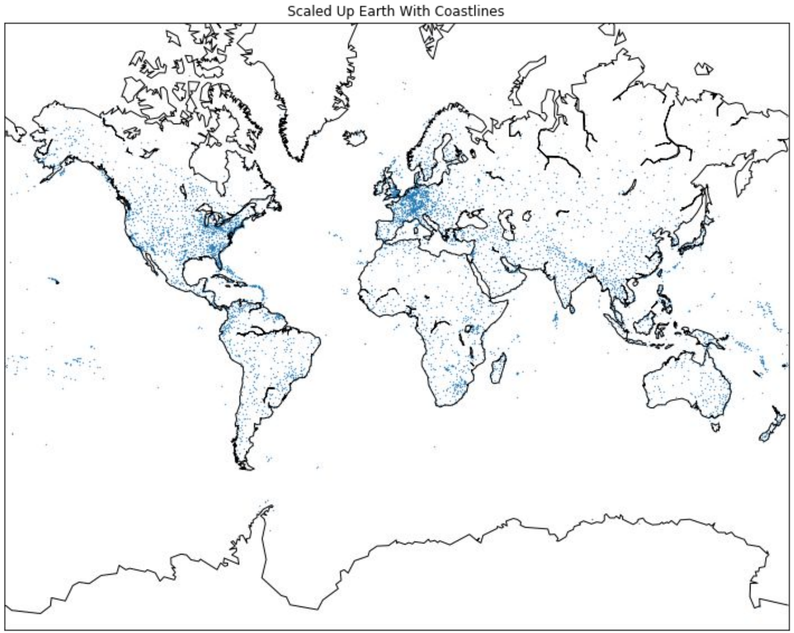
Workflow with basemap

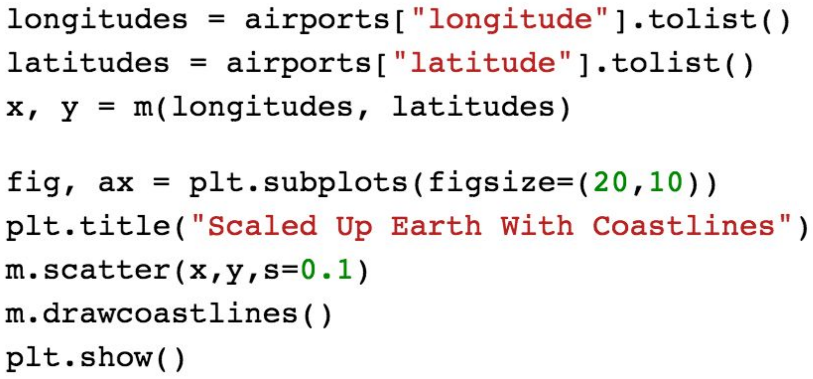


```python
import matplotlib.pyplot as plt
from mpl_toolkits.basemap import Basemap

m = Basemap(projection='merc',
            llcrnrlat=-80,
            urcrnrlat=80,
            llcrnrlon=-180,
            urcrnrlon=180)
```

Converting from spherical to cartesian coordinates

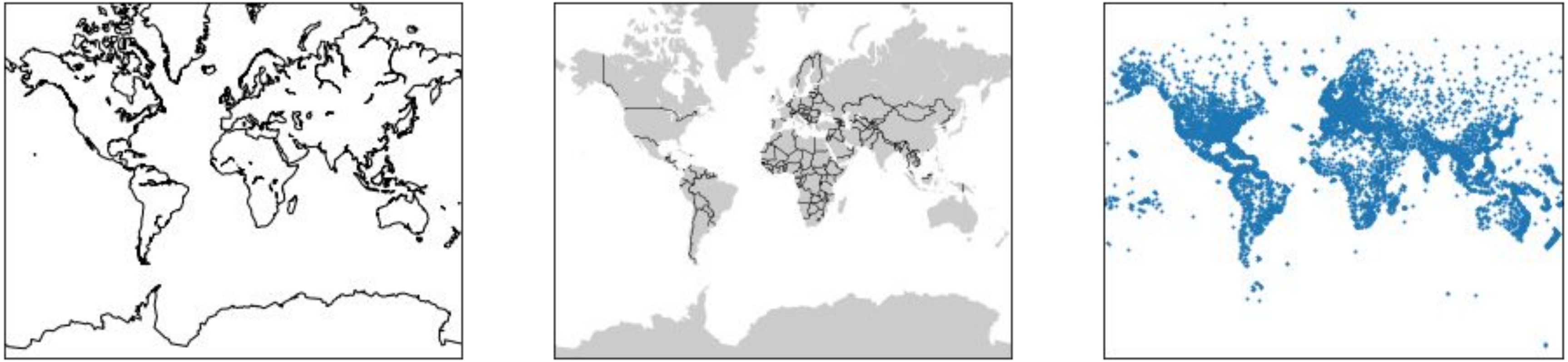


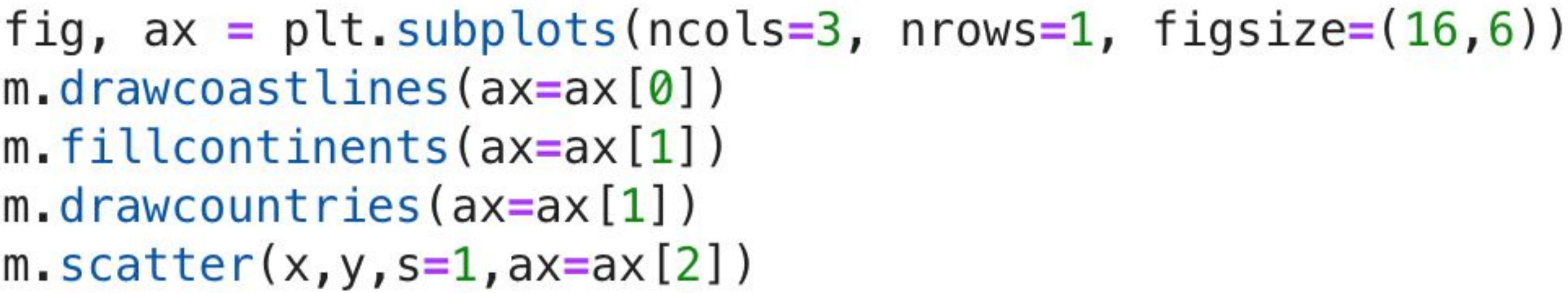


```
longitudes = airports["longitude"].tolist()
latitudes = airports["latitude"].tolist()
x, y = m(longitudes, latitudes)

fig, ax = plt.subplots(figsize=(20,10))
plt.title("Scaled Up Earth With Coastlines")
m.scatter(x,y,s=0.1)
m.drawcoastlines()
plt.show()
```

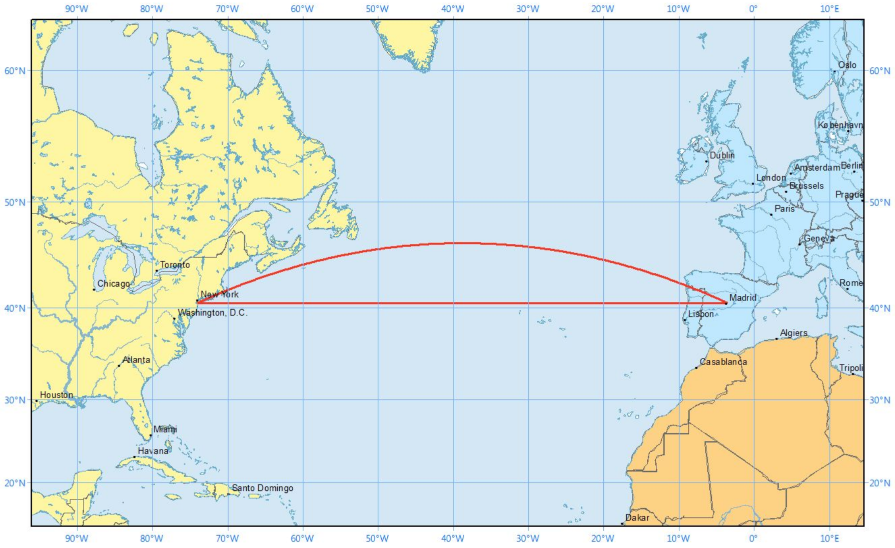
Customizing the plot using Basemap

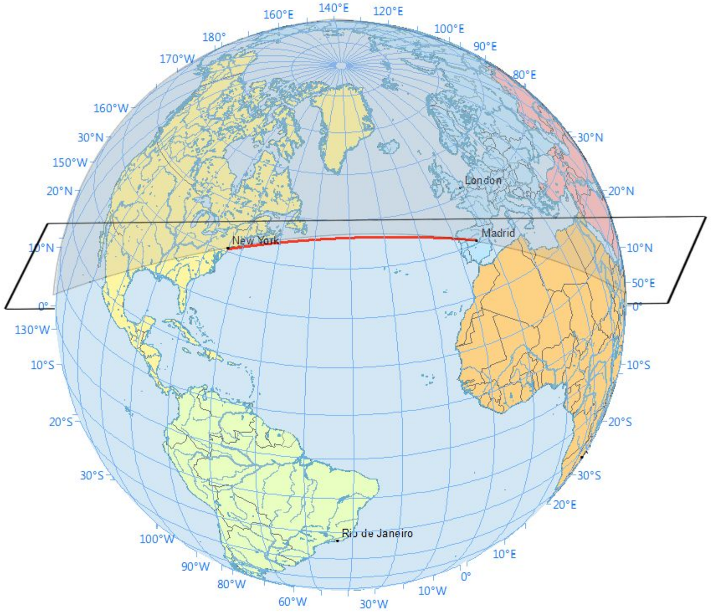




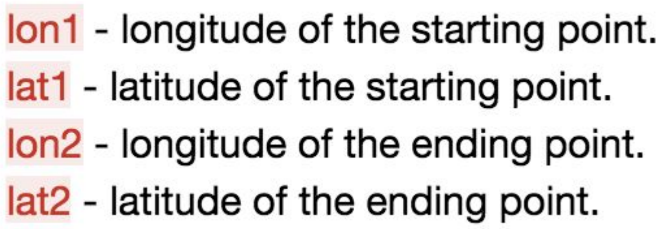
```
fig, ax = plt.subplots(ncols=3, nrows=1, figsize=(16,6))
m.drawcoastlines(ax=ax[0])
m.fillcontinents(ax=ax[1])
m.drawcountries(ax=ax[1])
m.scatter(x,y,s=1,ax=ax[2])
```

Introduction to great circles





Displaying great circles

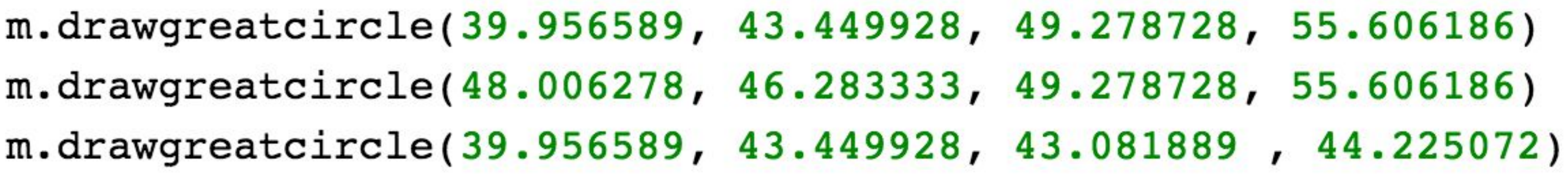


lon1 - longitude of the starting point.

lat1 - latitude of the starting point.

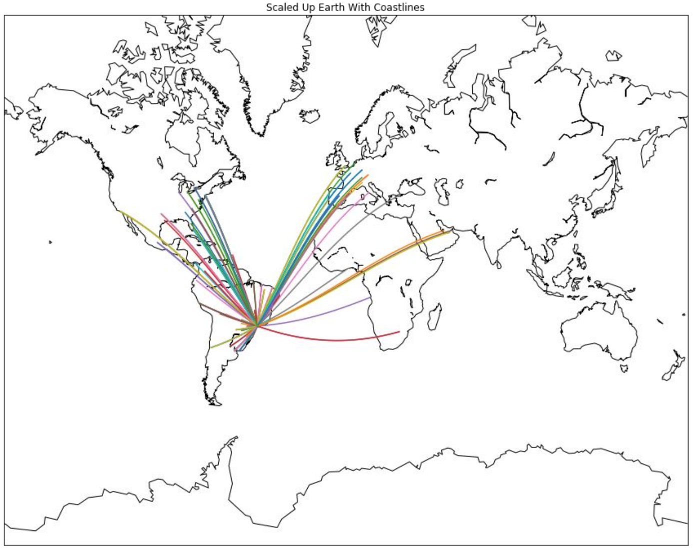
lon2 - longitude of the ending point.

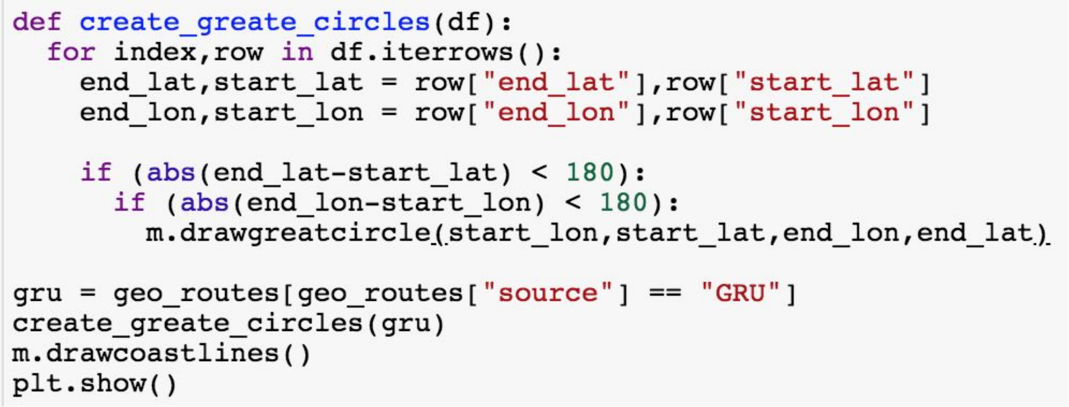
lat2 - latitude of the ending point.



```
m.drawgreatcircle(39.956589, 43.449928, 49.278728, 55.606186)
m.drawgreatcircle(48.006278, 46.283333, 49.278728, 55.606186)
m.drawgreatcircle(39.956589, 43.449928, 43.081889 , 44.225072)
```

Great circles: case study

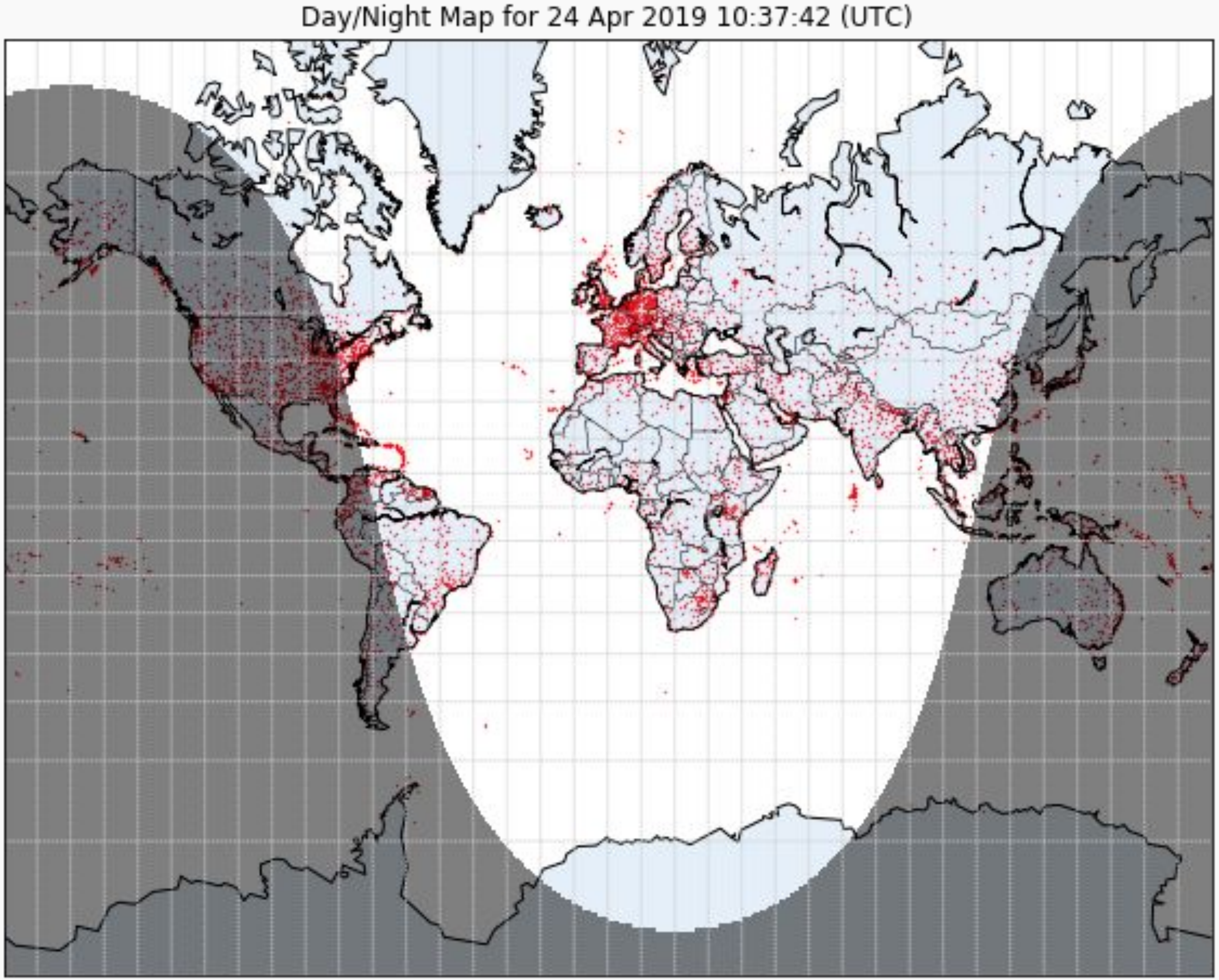




```
def create_greate_circles(df):
  for index,row in df.iterrows():
    end_lat,start_lat = row["end_lat"],row["start_lat"]
    end_lon,start_lon = row["end_lon"],row["start_lon"]

    if (abs(end_lat-start_lat) < 180):
      if (abs(end_lon-start_lon) < 180):
        m.drawgreatcircle(start_lon,start_lat,end_lon,end_lat)

gru = geo_routes[geo_routes["source"] == "GRU"]
create_greate_circles(gru)
m.drawcoastlines()
plt.show()
```



Day-night terminator on map

Lesson 12 - Visualizing Geographic Data.ipynb