Coordinate Reference Systems

WORKING WITH GEOSPATIAL DATA IN PYTHON

Joris Van den Bossche

Open source software developer and teacher, GeoPandas maintainer

datacamp

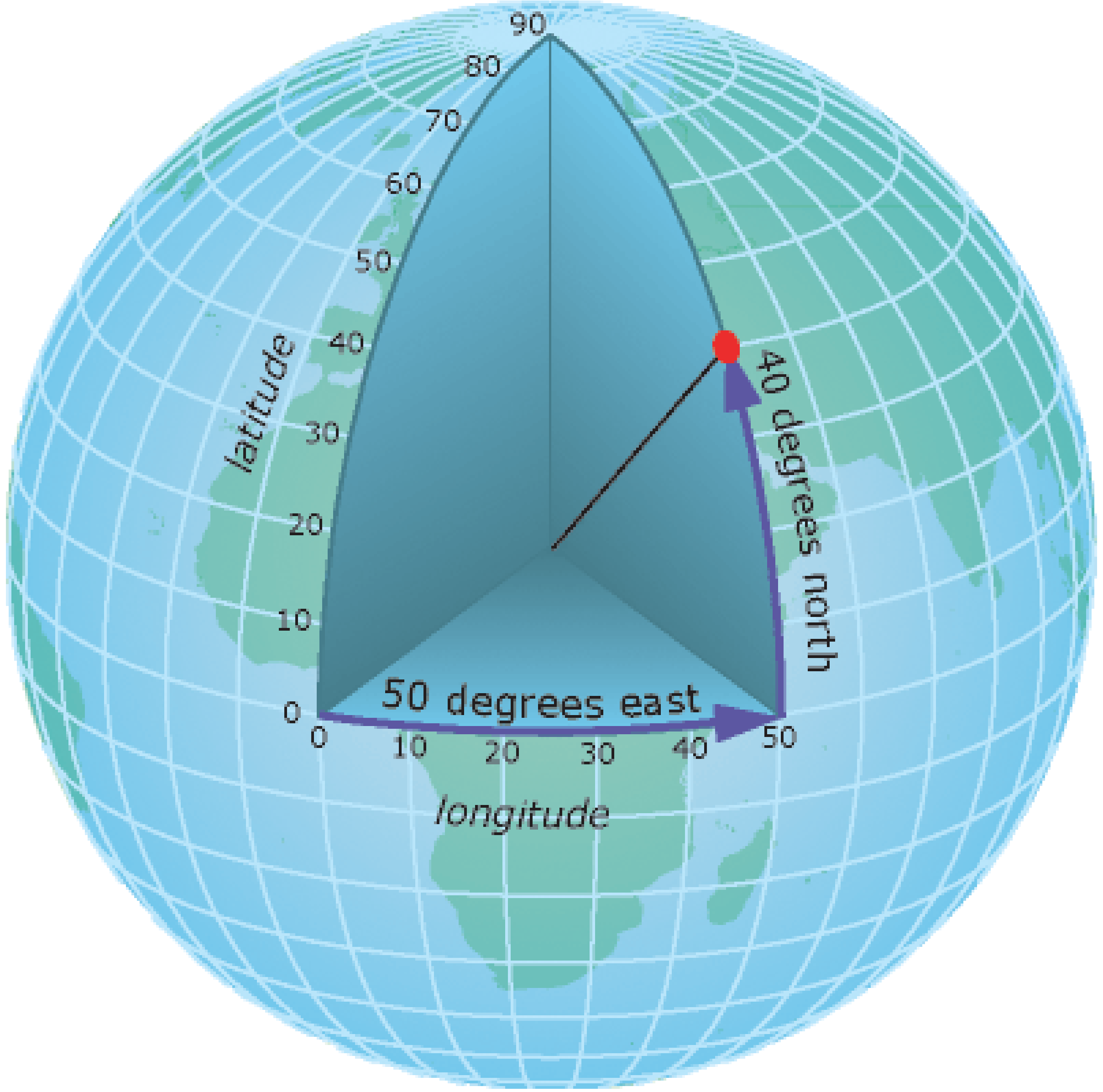
Coordinate Reference System (CRS)

Location of the Eiffel Tower:

```
POINT (2.2945 48.8584)
```

> → The **Coordinate Reference System (CRS)** relates the coordinates to a specific location on earth.

Geographic coordinates



Degrees of latitude and longitude.

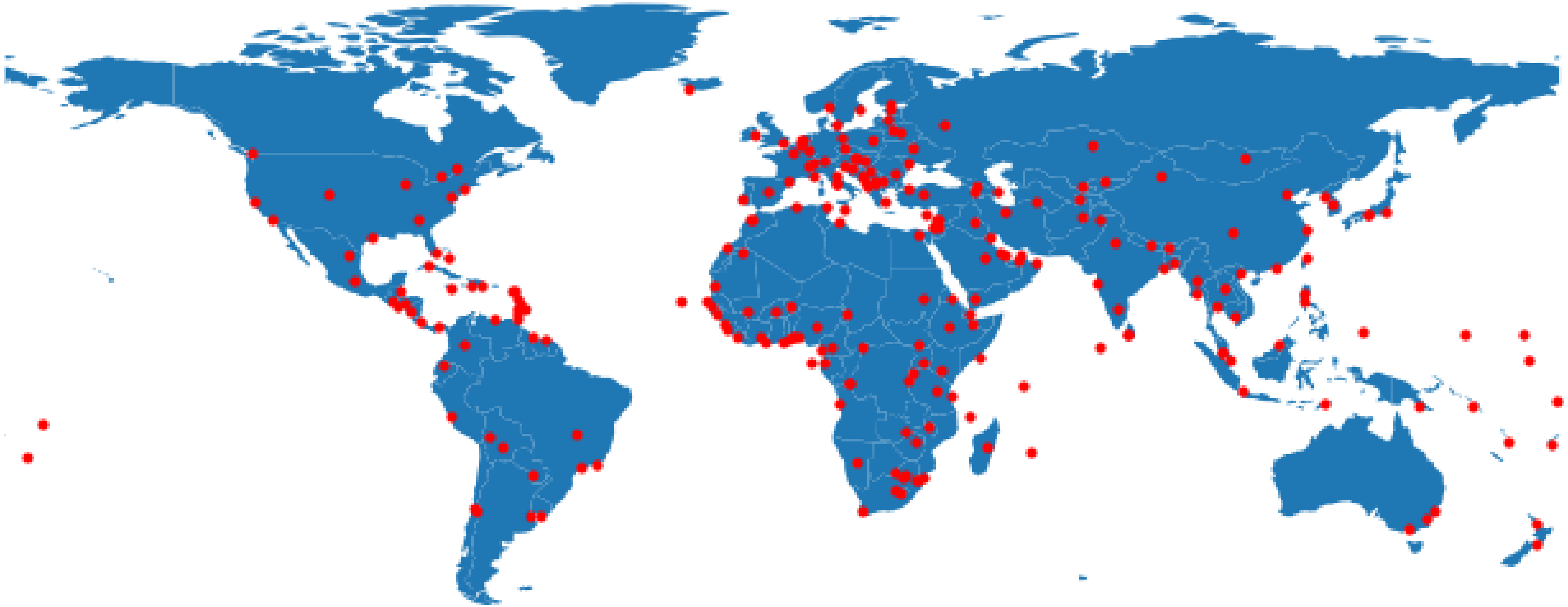
E.g. 48°51′N, 2°17′E

Used in GPS, web mapping applications...

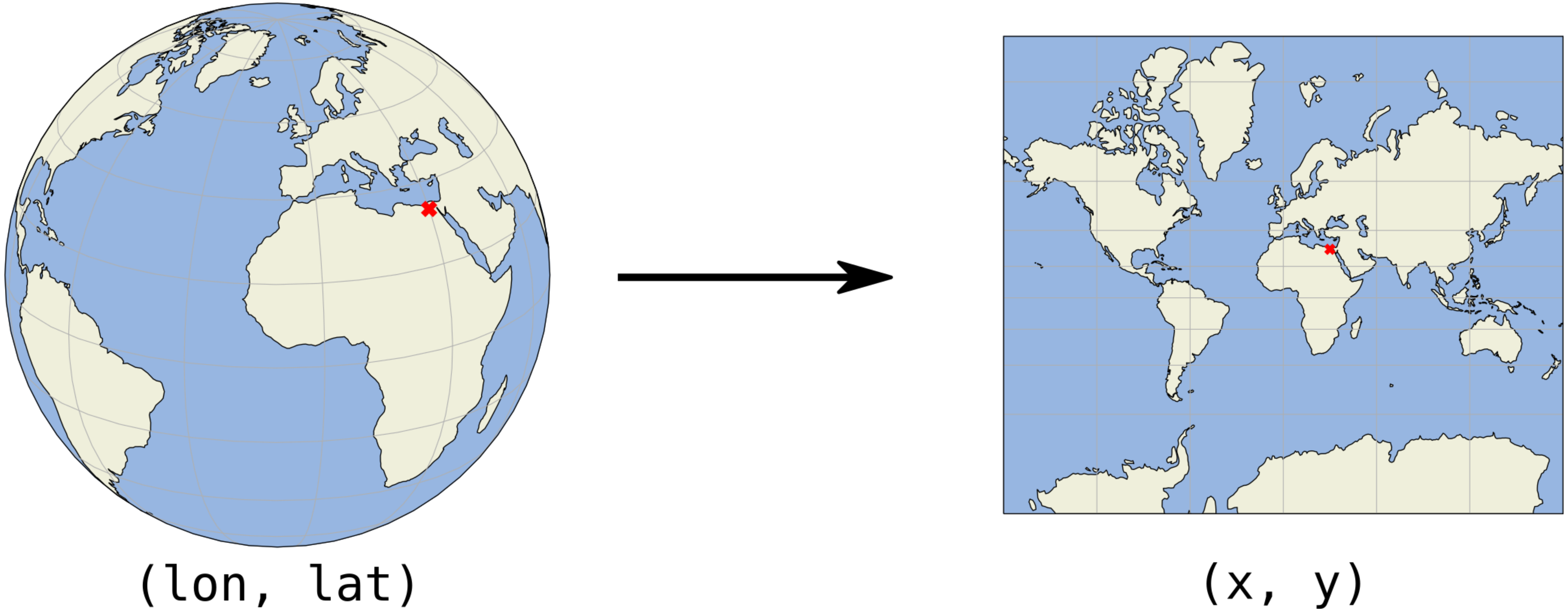
Attention!
in Python we use (lon, lat) and not (lat, long)

- Longitude: [-180, 180]
- Latitude: [-90, 90]

Maps are 2D



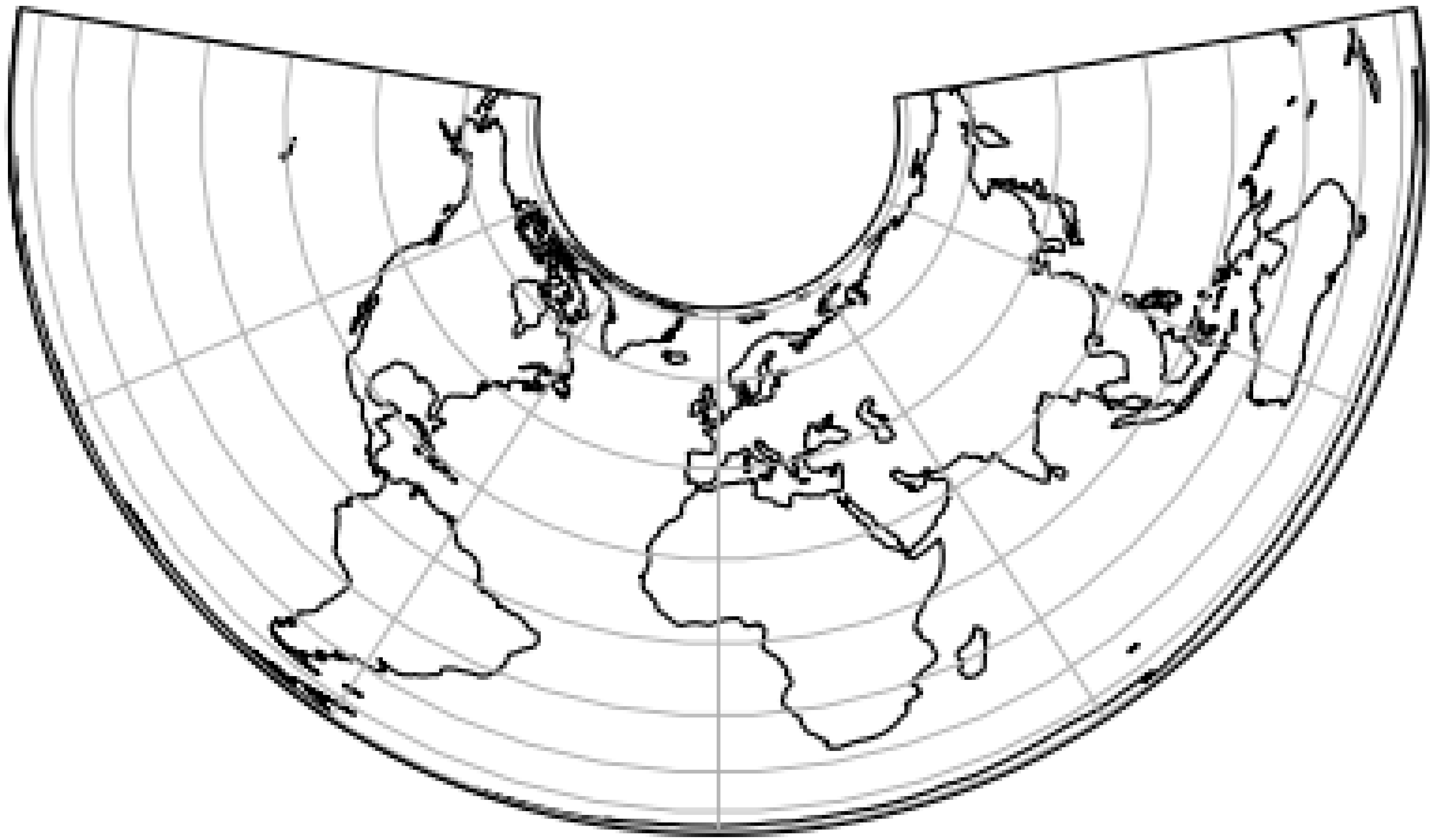
Projected coordinates



`(x, y)` coordinates are usually in meters or feet

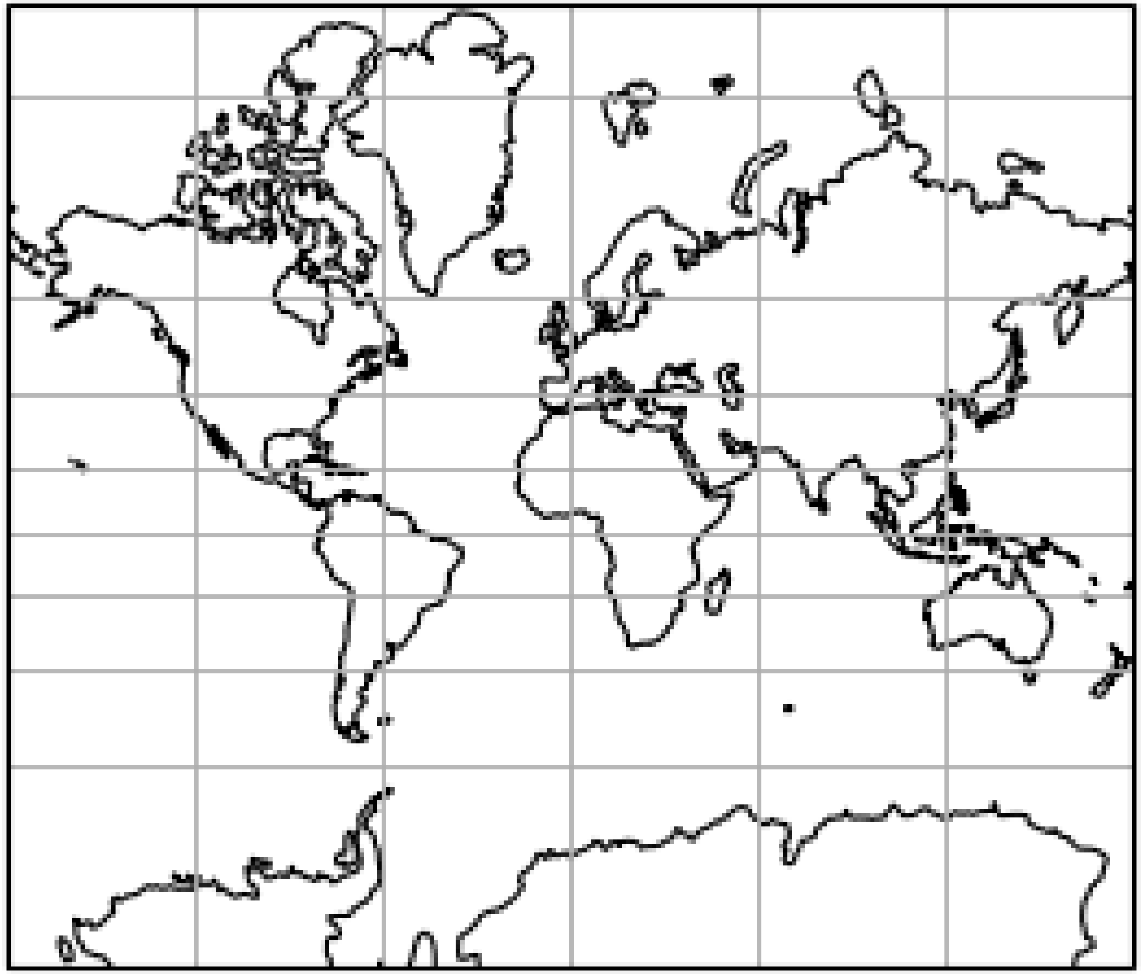
Projected coordinates - Examples

Albers Equal Area projection



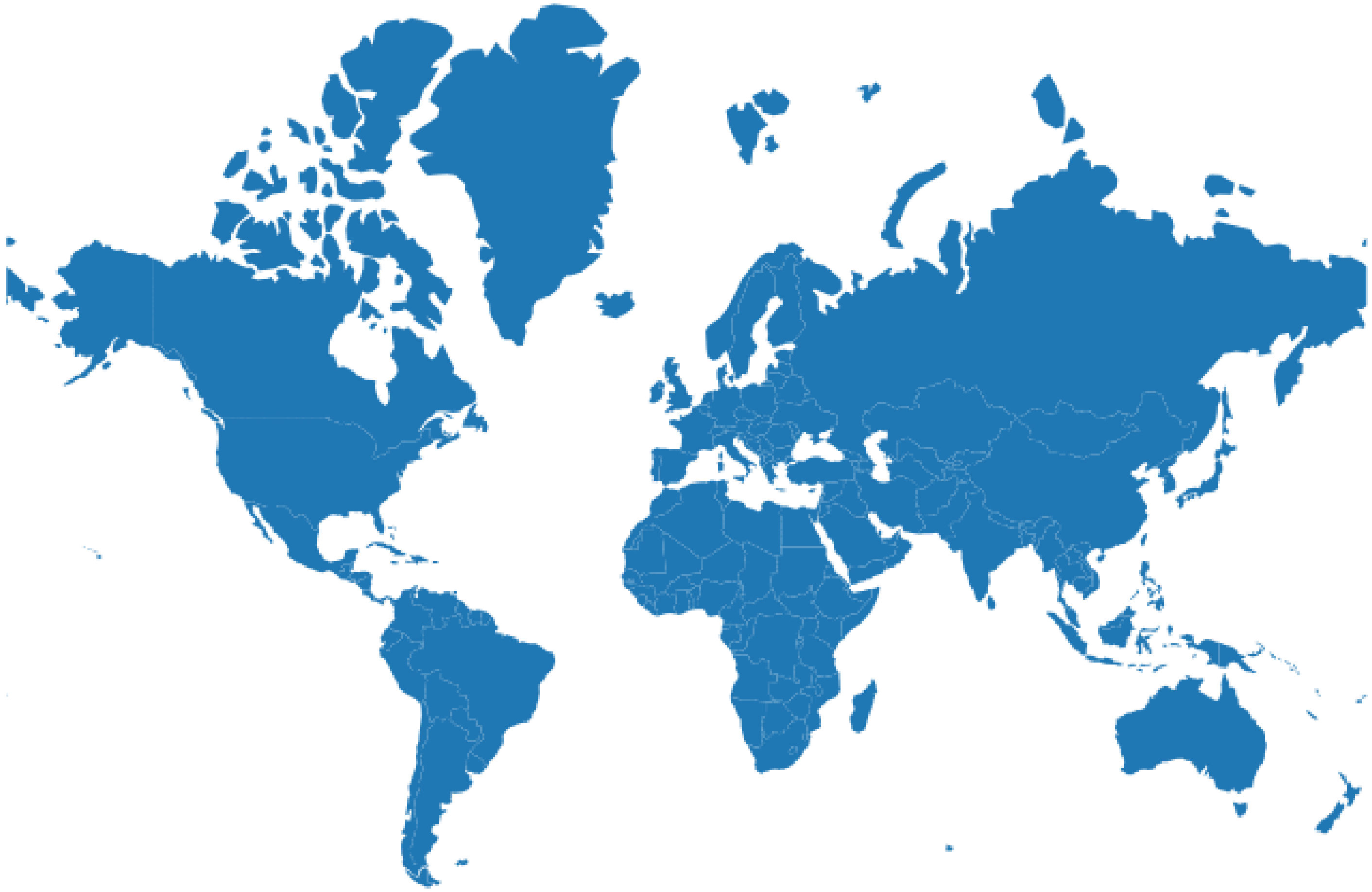
Projected coordinates - Examples

Mercator projection



Projected coordinates - Examples

Projected size vs actual size (Mercator projection)



Specifying a CRS

`proj4` string

Example: `+proj=longlat +datum=WGS84 +no_defs`

Dict representation:

```
{'proj': 'longlat', 'datum': 'WGS84', 'no_defs': True}
```

EPSG code

Example:

`EPSG:4326` = WGS84 geographic CRS (longitude, latitude)

CRS in GeoPandas

The `.crs` attribute of a GeoDataFrame/GeoSeries:

```python
import geopandas
gdf = geopandas.read_file("countries.shp")
print(gdf.crs)
```

```
{'init': 'epsg:4326'}
```

Summary

- "geographic" (long, lat) versus "projected" (x, y) coordinates
- Coordinates Reference System (CRS) in GeoPandas: `.crs` attribute
- Most used geographic CRS: WGS84 or EPSG:4326

Let's practice

WORKING WITH GEOSPATIAL DATA IN PYTHON

Working with coordinate systems in GeoPandas

WORKING WITH GEOSPATIAL DATA IN PYTHON

Joris Van den Bossche

Open source software developer and teacher, GeoPandas maintainer

CRS information in GeoPandas

The `.crs` attribute of a GeoDataFrame/GeoSeries:

```python
import geopandas
gdf = geopandas.read_file("countries.shp")
print(gdf.crs)
```

```
{'init': 'epsg:4326'}
```

Setting a CRS manually

```
gdf_noCRS = geopandas.read_file("countries_noCRS.shp")
print(gdf_noCRS.crs)
```

```
{}
```

Add CRS information to `crs` :

```
# Option 1
gdf.crs = {'init': 'epsg:4326'}

# Option 2
gdf.crs = {'proj': 'longlat', 'datum': 'WGS84', 'no_defs': True}
```

Transforming to another CRS

```python
import geopandas
gdf = geopandas.read_file("countries_web_mercator.shp")
print(gdf.crs)
```

```
{'init': 'epsg:3857', 'no_defs': True}
```

The `to_crs()` method:

```python
# Option 1
gdf2 = gdf.to_crs({'proj': 'longlat', 'datum': 'WGS84', 'no_defs': True})
# Option 2
gdf2 = gdf.to_crs(epsg=4326)
```

Why converting the CRS?

1) Sources with a different CRS

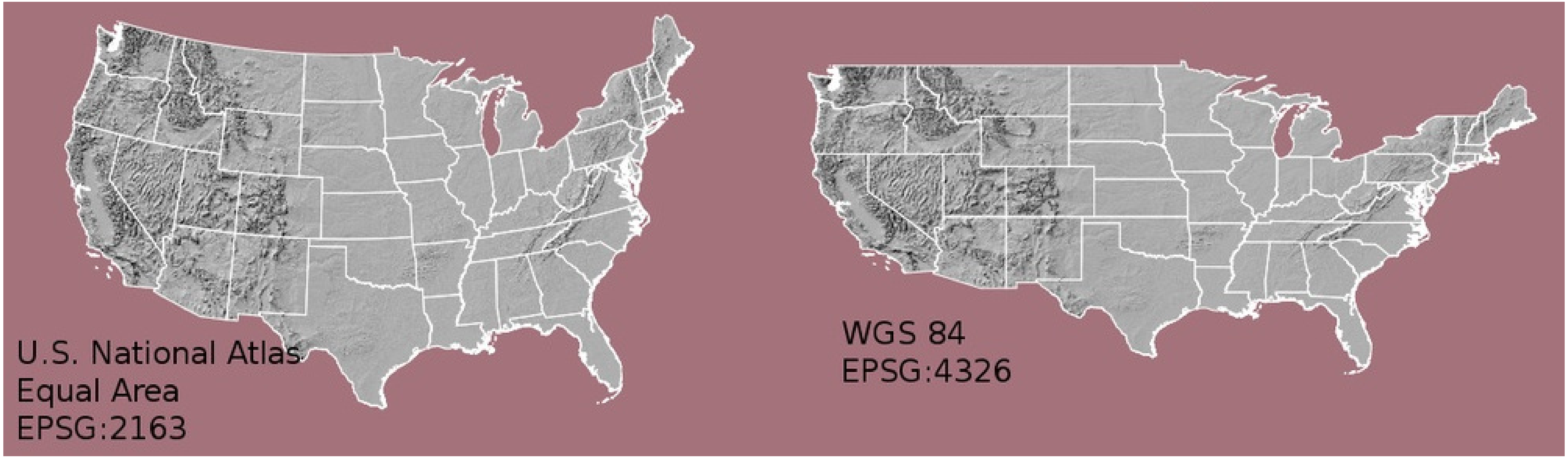
```
df1 = geopandas.read_file(...)
df2 = geopandas.read_file(...)


df2 = df2.to_crs(df1.crs)
```

Why converting the CRS?

1) Sources with a different CRS

2) Mapping (distortion of shape and distances)



Why converting the CRS?

1) Sources with a different CRS

2) Mapping (distortion of shape and distances)

3) Distance / area based calculations

How to choose which CRS to use?

Tips:

- Use projection specific to the area of your data
- Most countries have a standard CRS

Useful sites:

- http://spatialreference.org/
- https://epsg.io/

Summary

- To convert to another CRS: the `to_crs()` method
- Make sure different datasets have the same CRS
- When calculating distance, area, ... -> use a **projected CRS**

Useful sites:

- **http://spatialreference.org/**
- **https://epsg.io/**

Let's practice!

WORKING WITH GEOSPATIAL DATA IN PYTHON

Spatial operations: creating new geometries

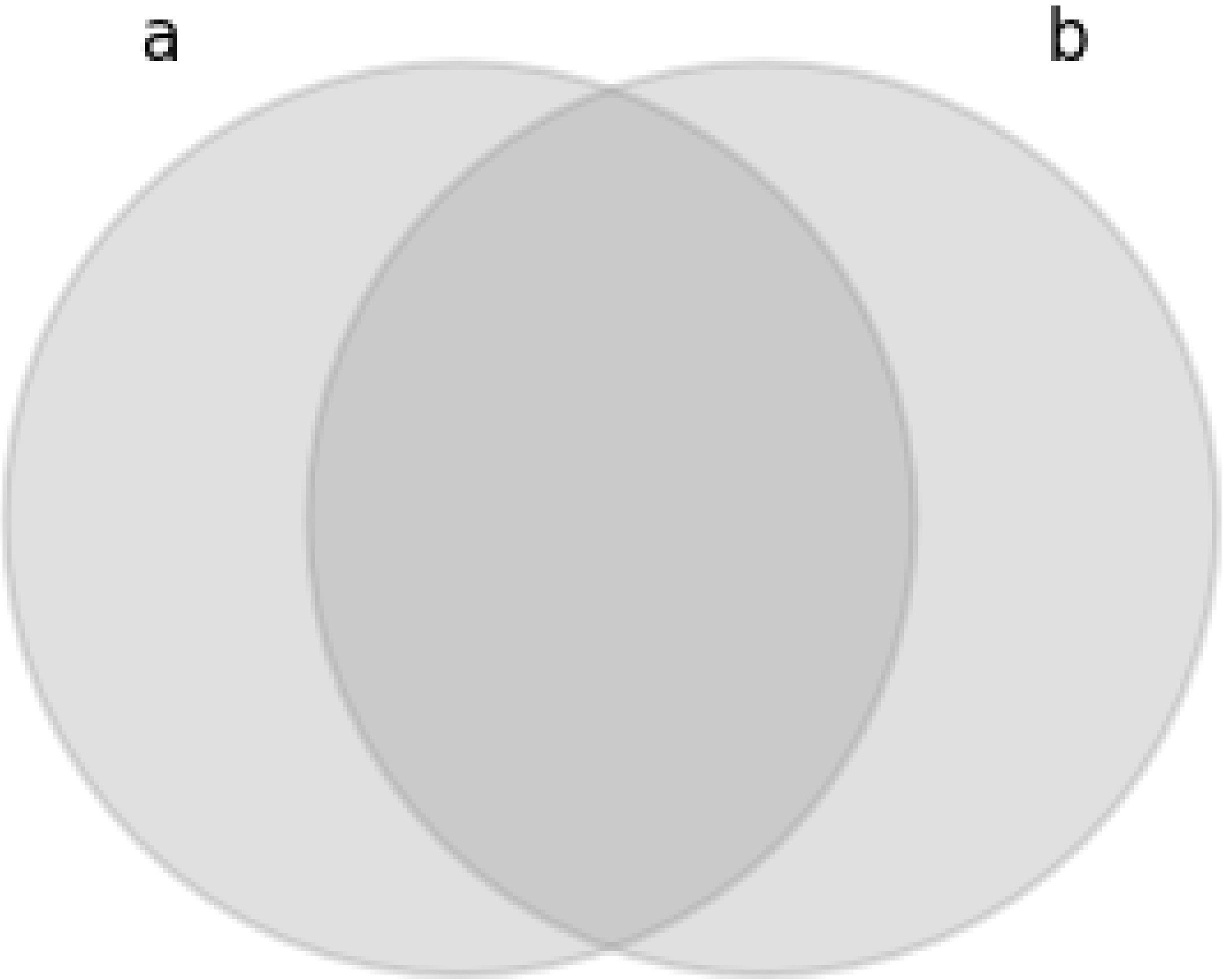
WORKING WITH GEOSPATIAL DATA IN PYTHON

Joris Van den Bossche

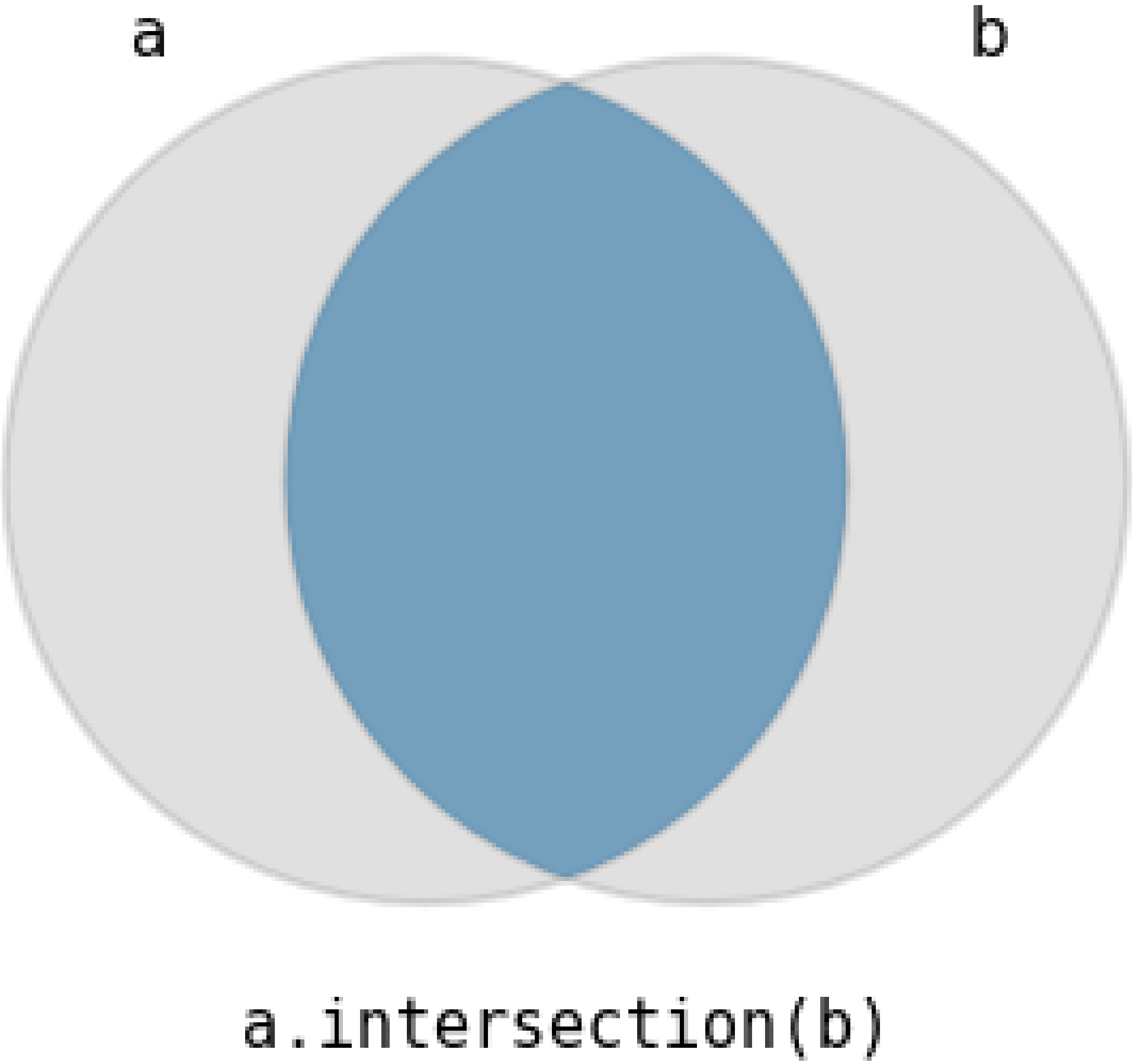
Open source software developer and teacher, GeoPandas maintainer

datacamp

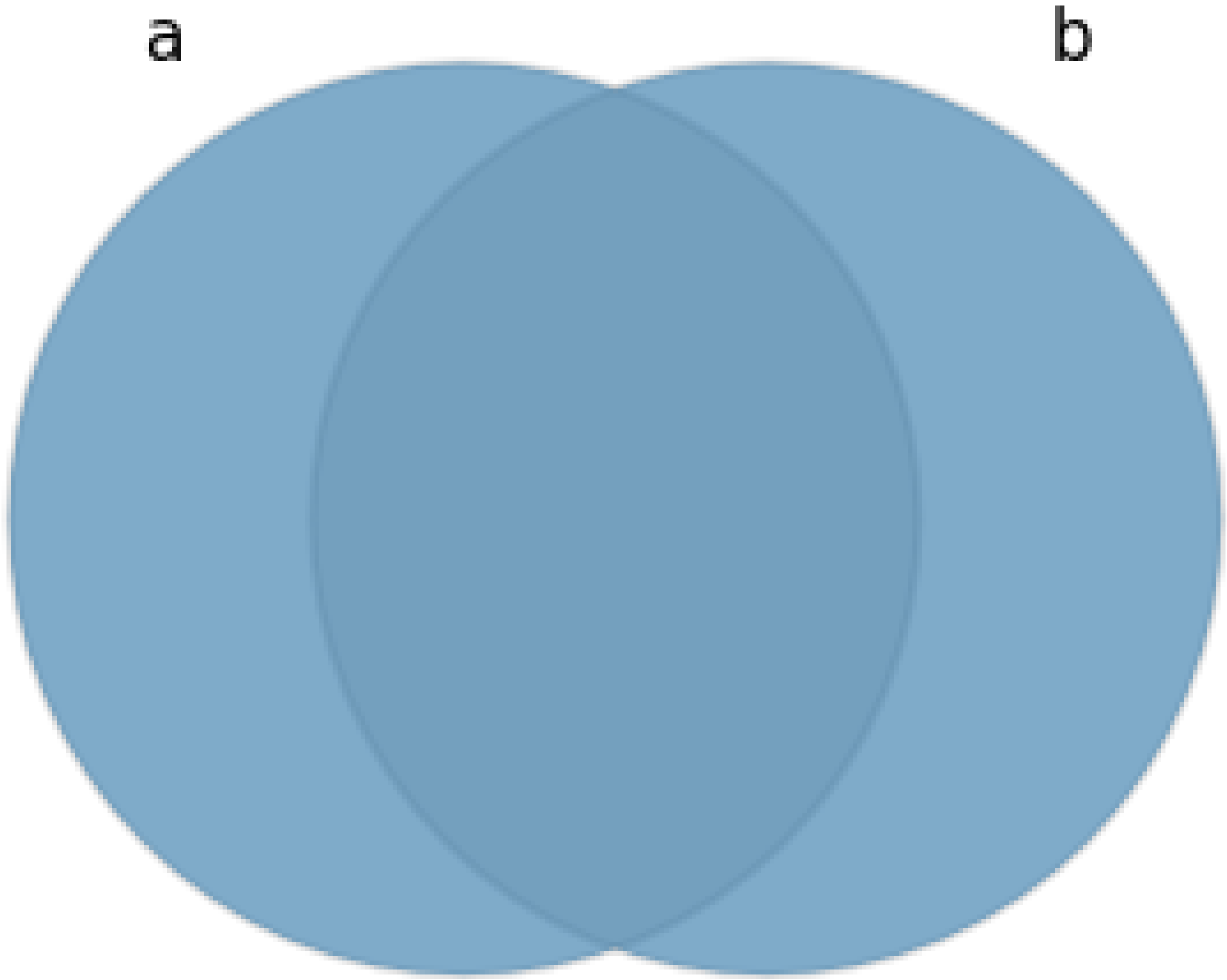
Spatial operations



Spatial operations: intersection

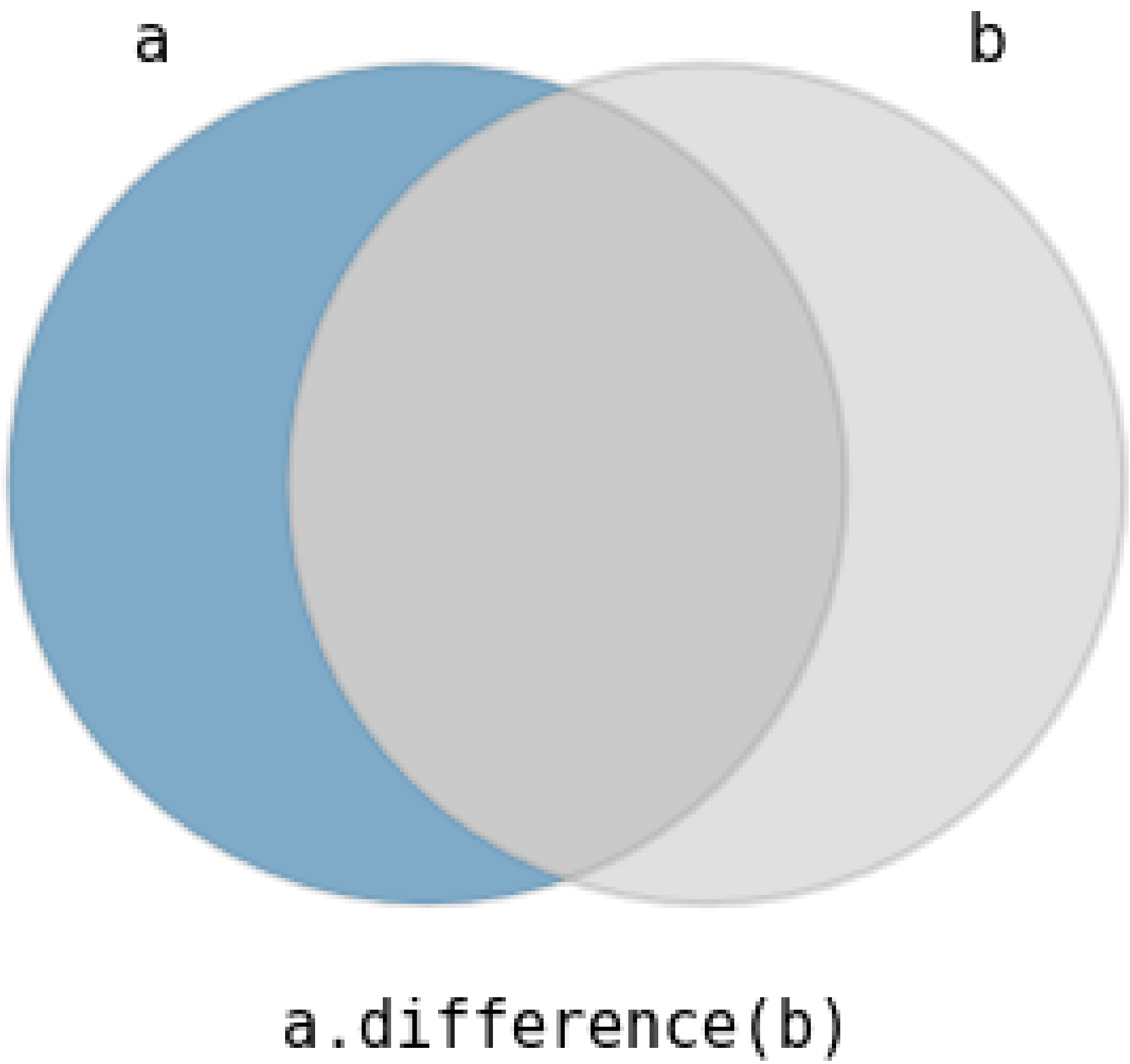


Spatial operations: union



a.union(b)

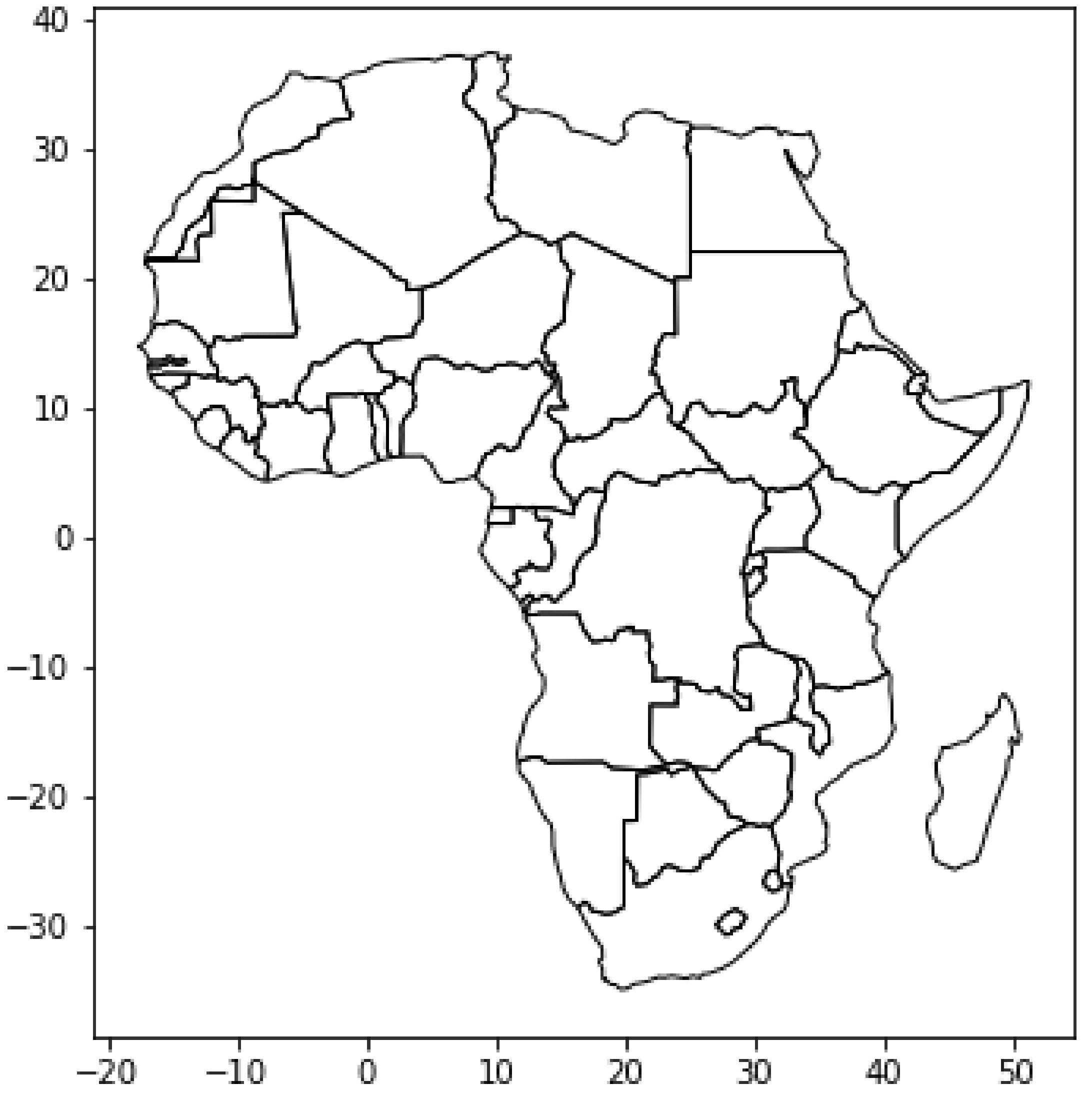
Spatial operations: difference



Spatial operations with GeoPandas

```
africa.head()
```

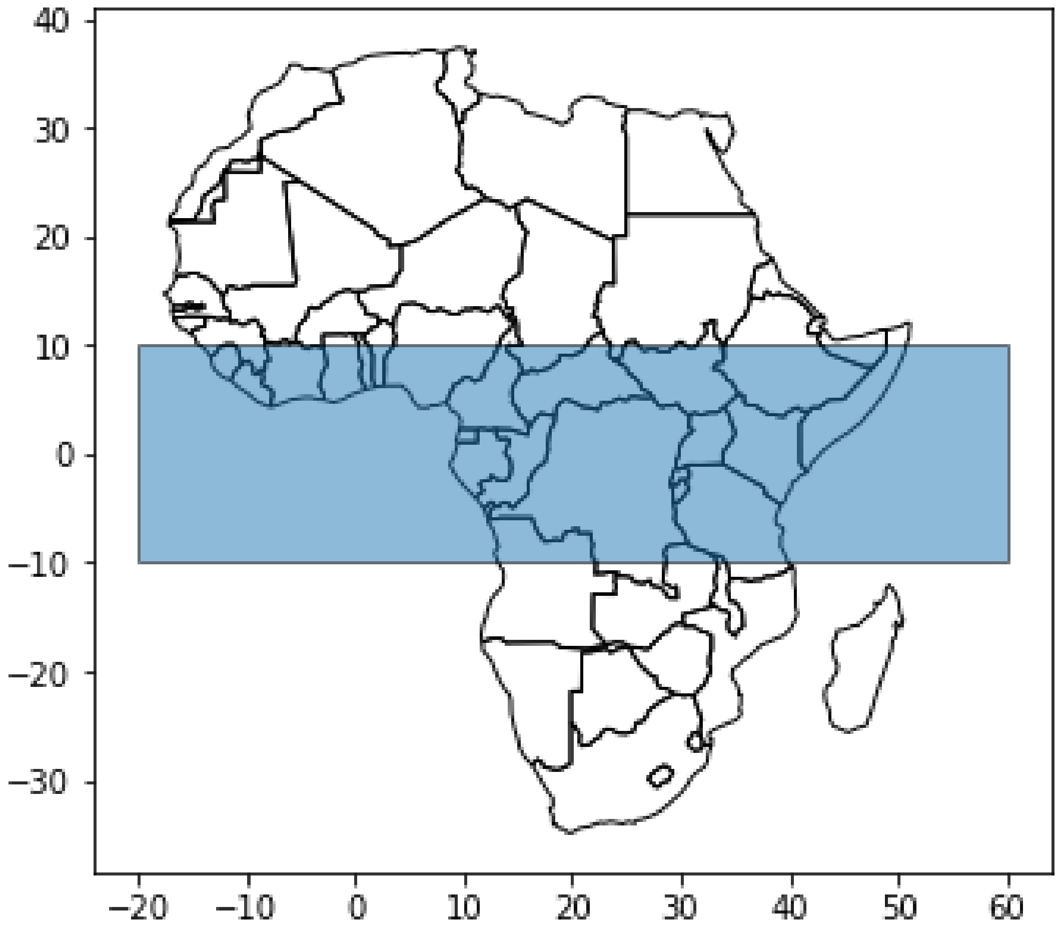
```
           name                 geometry
0        Angola  (POLYGON ((23.90...
1       Burundi  POLYGON ((29.339...
2         Benin  POLYGON ((2.6917...
3  Burkina Faso  POLYGON ((2.1544...
4      Botswana  POLYGON ((29.432...
```



Spatial operations with GeoPandas

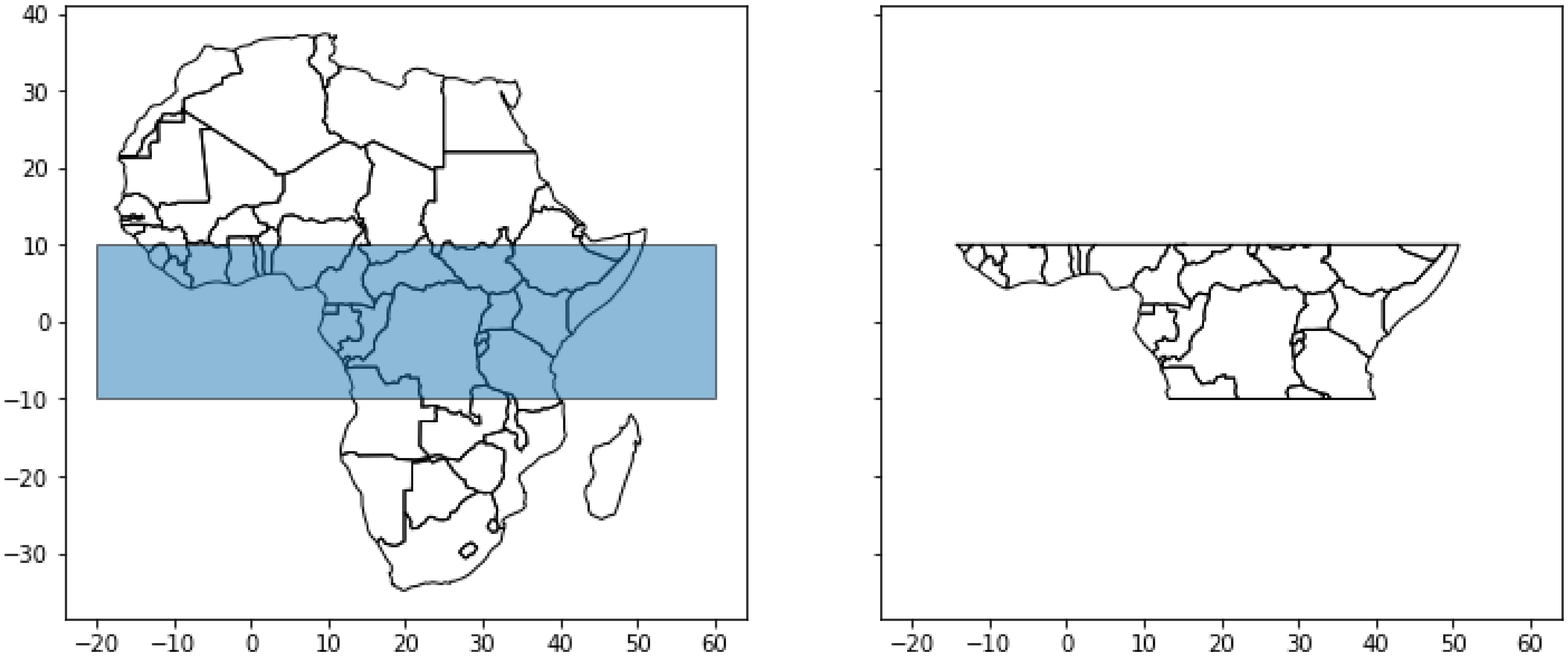
```
print(box)
```

```
POLYGON ((60 10, 60 -10, -20 -10, -20 10)
```



Spatial operations with GeoPandas

```
africa.intersection(box)
```



Spatial operations with GeoPandas

```
africa.head()
```

```
          name                                            geometry
0       Angola  (POLYGON ((23.90415368011818 -11.7222815894063...
1      Burundi  POLYGON ((29.33999759290035 -4.499983412294092...
2     Botswana  POLYGON ((29.43218834810904 -22.09131275806759...
...
```

```
africa.intersection(box)
```

```
0    (POLYGON ((13.22332255001795 -10, 13.120987583...
1    POLYGON ((29.33999759290035 -4.499983412294092...
2                                                   ()
...
dtype: object
```

Let's practice!

WORKING WITH GEOSPATIAL DATA IN PYTHON

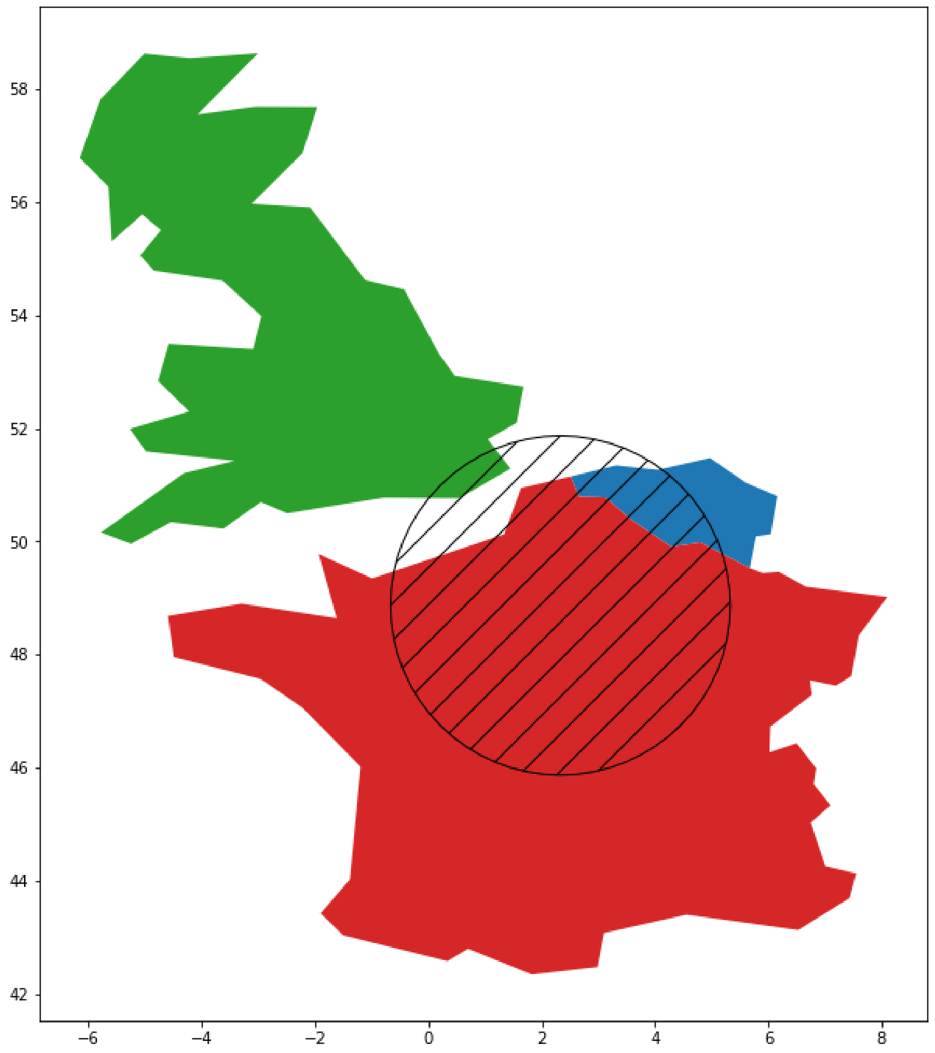
Overlaying spatial datasets

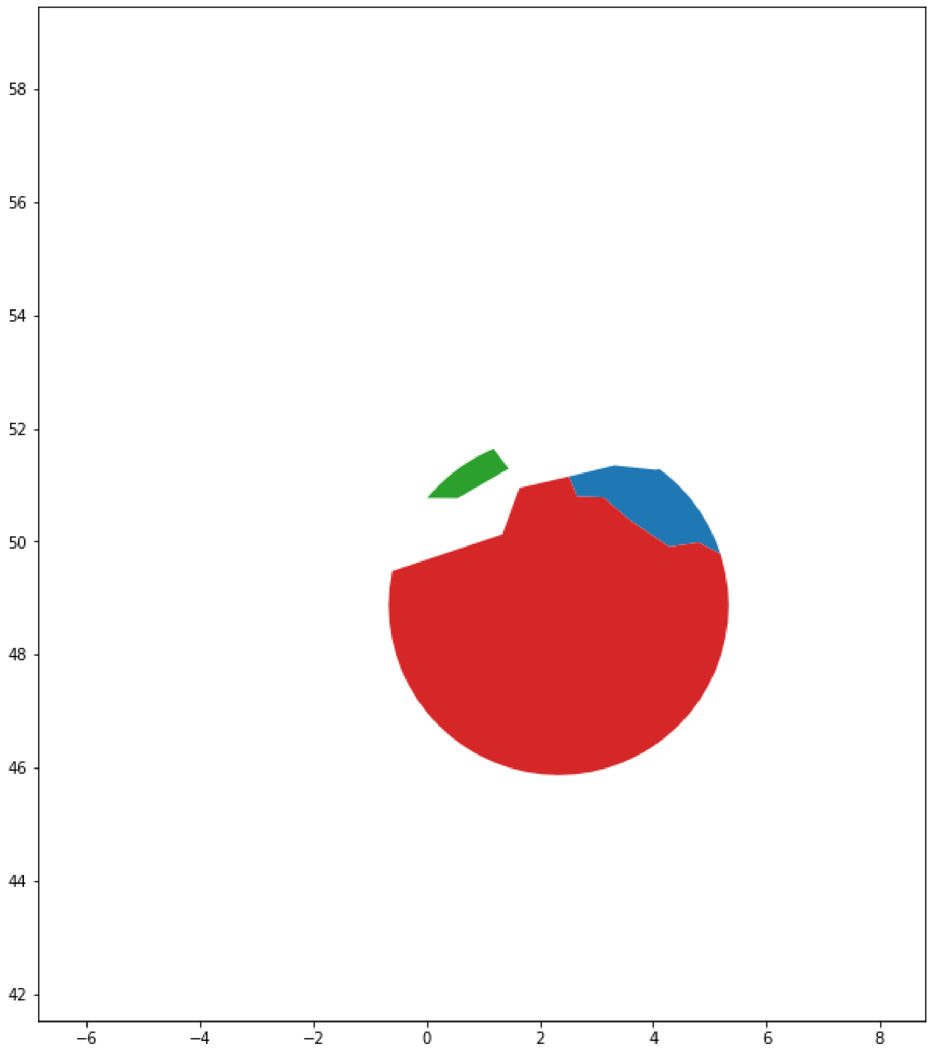
WORKING WITH GEOSPATIAL DATA IN PYTHON

Joris Van den Bossche

Open source software developer and teacher, GeoPandas maintainer

Intersection with a polygon





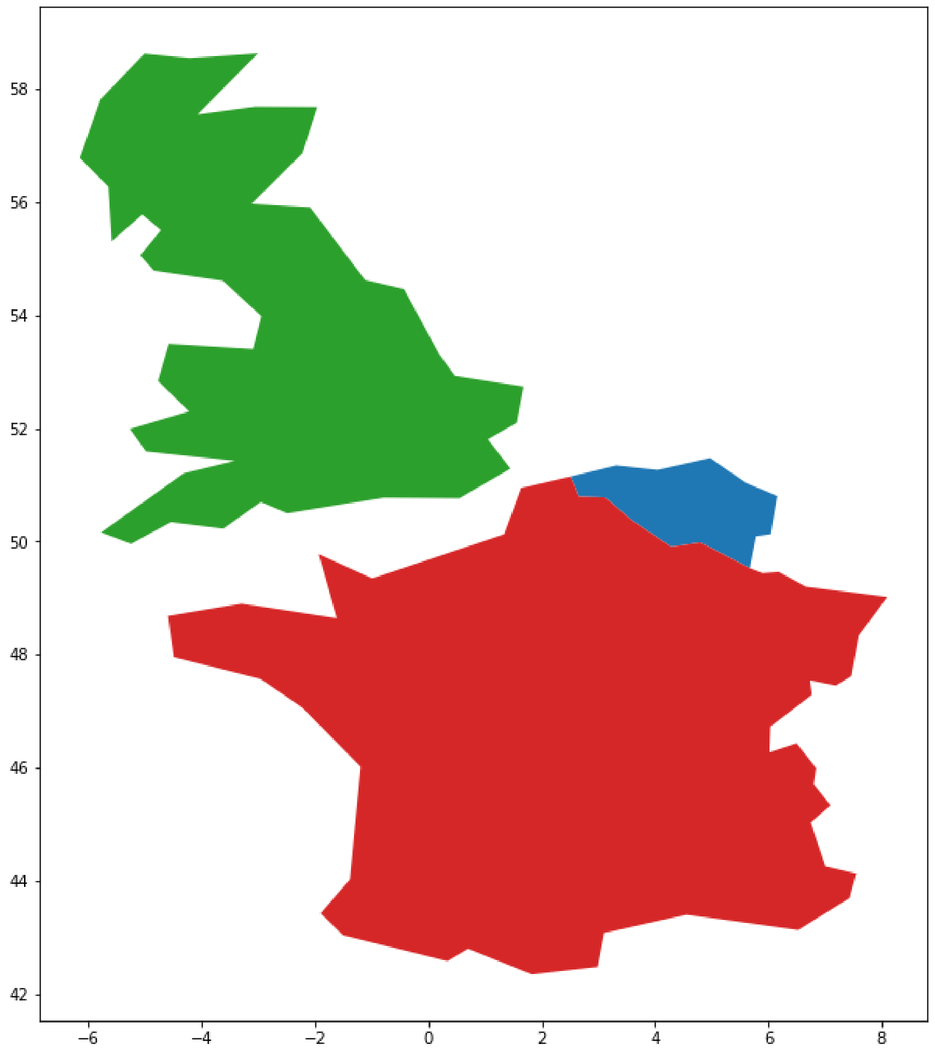
```
countries.intersection(circle)
```

Intersection with a polygon

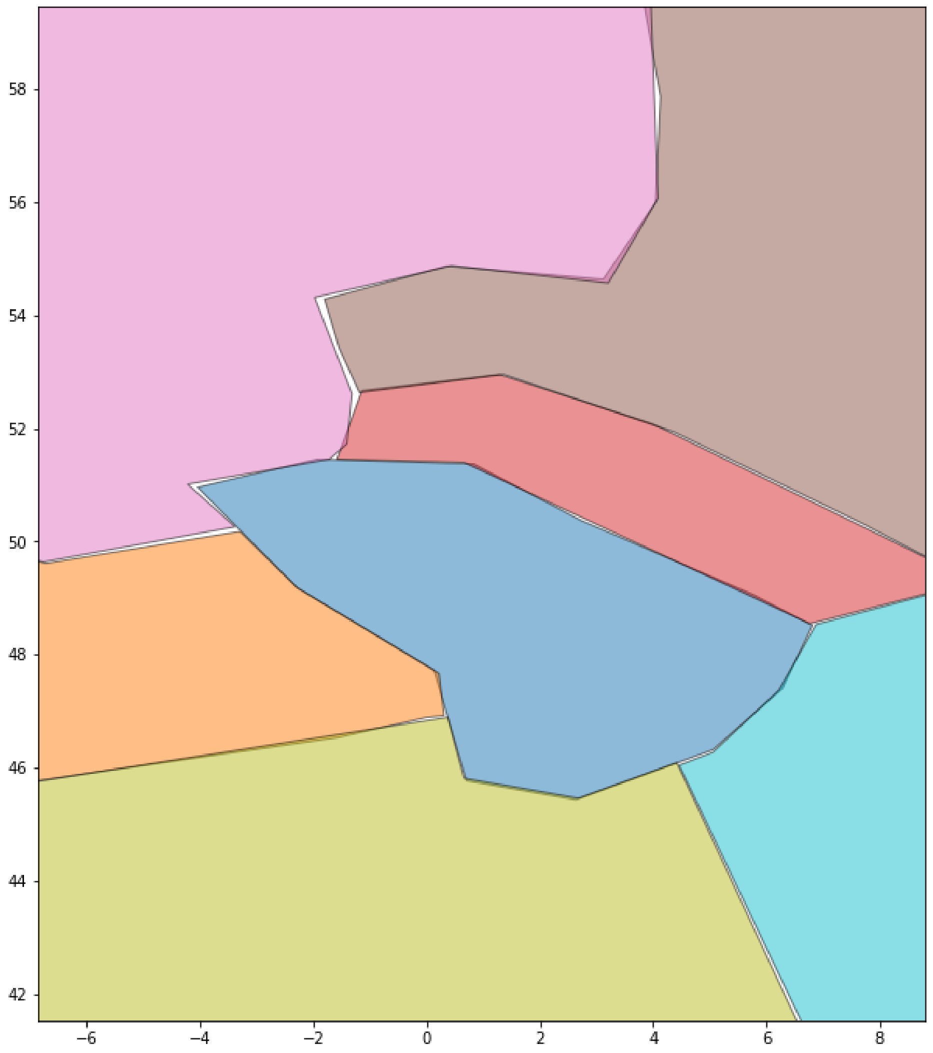
Limitations of `countries.intersection(circle)` :

- Only intersecting a GeoSeries with a single polygon
- Does not preserve attribute information

Overlaying two datasets

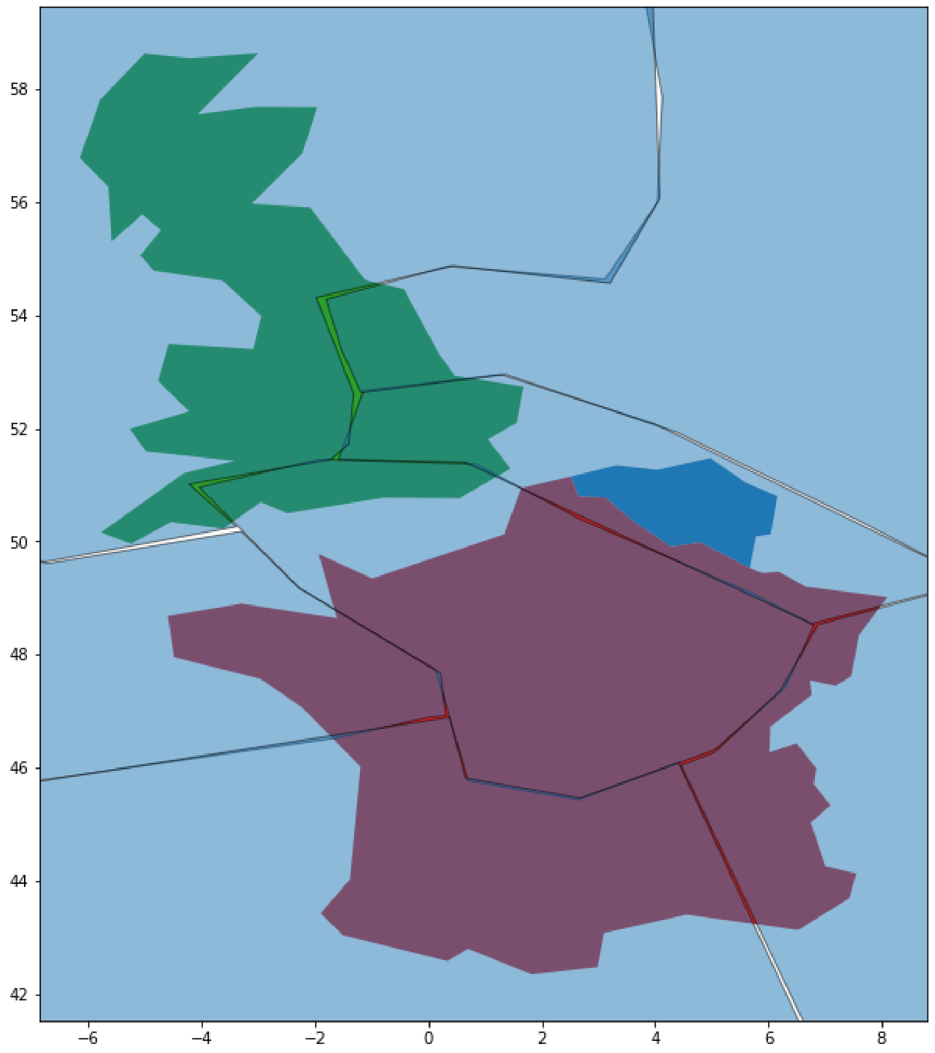


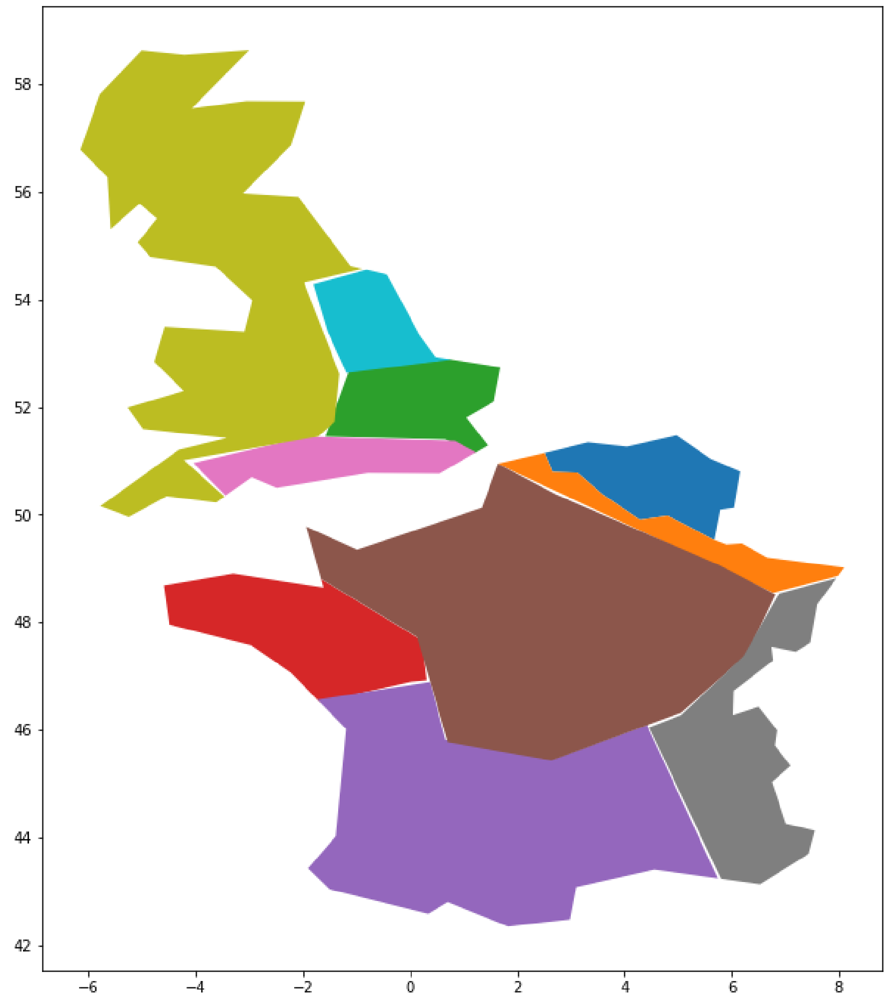
```
countries.plot()
```



```
geologic_regions.plot()
```

Overlaying two datasets





```
geopandas.overlay(countries, geologic_regions,
                  how='intersection')
```

Overlay vs intersection

Intersection method (with single polygon)

```
countries.intersection(geologic_region_A)
```

```
0                            ()
1    POLYGON ((-1.661 48.803...
2    POLYGON ((1.201 51.145,...
dtype: object
```

Overlay method

```
geopandas.overlay(countries, geologic_regions,
                  how='intersection')
```

```
     name geologic_region              geometry
1  France               C  POLYGON ((2.5 51....
2      UK               C  POLYGON ((0.7 52 ...
3  France               B  POLYGON ((-1.7 46...
..    ...             ...                   ...
```

Let's practice!

WORKING WITH GEOSPATIAL DATA IN PYTHON